Implementación y Puesta en Marcha de una plataforma de prácticas de control con arquitectura Lego

Ing. David Herrera Alfonso
280230

Trabajo de grado presentado para optar al título de Magister en Ingeniería - Automatización Industrial

Director: Ing Leonardo Bermeo

Universidad Nacional de Colombia
Facultad de Ingeniería
Posgrado en Automatización Industrial
Departamento de Ingeniería Eléctrica y Electrónica
Bogotá, 2010

Aprobada por la Facultad de Ingeniería en cumplimiento de los requisitos exigidos para otorgar el título de: **Maestría en Ingeniería - Automatización Industrial**

LEONARDO BERMEO CLAVIJO.
Director de la Tesis

Jurado

Jurado

Universidad Nacional de Colombia
Bogotá D.C. Noviembre de 2010

Agradecimientos

Este trabajo contó con el apoyo del Departamento de Ingeniería Eléctrica y Electrónica de la Universidad Nacional de Colombia, sede Bogotá, a quienes se agradece la oportunidad de trabajar con los elementos de Lego dentro de los cursos de Control ofrecidos por el mismo.

Dedicado quienes trabajan en el Posgrado de Automatización Industrial de la Universidad Nacional de Colombia, sede Bogotá, por su constante ayuda y colaboración.

Al Ing. Leonardo Bermeo Clavijo por su dirección y sus consejos, sin los cuales no hubiese sido posible la culminación de este trabajo.

A Jenny, Sergio, Juan Pablo y Camilo, por su invaluable compañerismo y amistad.

A mi familia, por su constante apoyo y comprensión.

Contenido

Índice de figuras

Parte I

Preliminares

1 Introducción

1.1 Justificación

En la industria se presentan de manera recurrente todo tipo de procesos que implican el control de un conjunto de variables en dominios físicos múltiples, como flujos de caudal, temperaturas, presiones, niveles de líquido y voltajes, entre muchas otras, lo cual exige entonces un profundo tratamiento de los temas relacionados con la teoría de control y sus posibles aplicaciones prácticas, al seno de los cursos que las universidades y otros centros tengan destinados para la enseñanza de estos temas.

Durante mucho tiempo, la educación en control se ha centrado en el aleccionamiento de los conceptos teóricos, acompañado en épocas recientes de simulaciones con software especializado. Sin embargo la teoría básica requiere un esfuerzo intelectual considerable, que en caso de no verse reflejado en ejercicios prácticos y/o de simulación, puede mostrar problemas de apropiación del conocimiento por parte de los estudiantes. De acuerdo a [9], se nota que la mayoría de los estudiantes (de control) encuentran la teoría de control conceptualmente difícil, la perciben como periférica y tienen problemas integrándola con otro material, sugiriendo además que en la instrucción de control se está muy preocupado con la “parte analítica”del proceso de control porque no se sabe como enseñar otros problemas implicados en la síntesis del proceso de control. Con estos problemas se refiere a la falta de claridad a la hora de establecer la conexión entre la teoría y la práctica, es decir la contextualización del problema de control.

Históricamente, la forma de afrontar los asuntos de control del mundo real dentro del ambiente académico es la de realizar análisis sobre problemas de control clásicos muy comunes dentro del ambiente académico, como son el péndulo invertido, la bola y la viga, sobre los cuales ya existen grandes avances teóricos y prácticos tanto en modelado como en control, provocando avances significativos en la enseñanza y el progreso del control.

Los mencionados vacios de contextualización, se han venido supliendo mediante la implementación de las plantas clásicas en laboratorios dedicados. La razón primordial radica en que la enseñanza

de la ingeniería de control se beneficia mediante el uso de laboratorios de control, con experimentos en los que se utiliza hardware real como parte integral de los cursos [12], [20]. Además, algunas empresas han creado diversos módulos que recrean las mismas, productos de muy buena calidad y fiabilidad. Sin embargo, no son muchas las compañías que trabajan en este campo y sus soluciones aunque de gran nivel, son costosas y limitan la creatividad del estudiante a la hora de intentar nuevos retos, dado que se tratan en su mayoría, de plantas de arquitectura cerrada o con pocas posibilidades expansión sobre piezas prediseñadas.

Los detalles observados hacen que la implementación de un buen banco de pruebas sea más un problema que una solución, ya que los costos pueden ser prohibitivos para muchas instituciones. Para paliar en alguna forma tal inconveniente, algunos investigadores han propuesto algunas soluciones, en las que se describen formas económicas y muy completas de realizar prácticas de control basados en los retos clásicos. Algunos ejemplos de propuestas se pueden ver en [9], [12] y [14] entre otros. En las soluciones halladas se ve la clara intención de utilizar materiales de bajo costo para la implementación de la prácticas, acompañados de modelos matemáticos que describen adecuadamente los montajes realizados. En general, son soluciones muy aceptables con buenos resultados.

Entre algunos de los dilemas fundamentales de tales esquemas radica en que por lo general se trata de enfoques en los que se involucra una sola planta clásica, que puede haber sido desarrollada con piezas de arquitectura abierta o no, aunque en general a bajo costo. El problema de los trabajos realizados con piezas hechas a la medida es que muy a pesar de ser fáciles de implementar, no tienen la facilidad de ser expansibles o modificadas en alguna forma para que sobre ellas sea posible desarrollar nuevas prácticas; pese a esto representan un buen modelo a seguir.

En cuanto a los trabajos en los que se contruyeron plantas sobre medios abiertos, como Lego®, se ve una mayor libertad en cuanto a las posibles prácticas que con las mismas, pero el defecto fundamental que las rodea es la carencia, en muchos de los casos, de un modelado apropiado, a partir del cual se pueda iniciar una tarea de control con fines educativos. En dichas ocasiones, las plantas son realizadas más con fines demostrativos o competitivos (competencia robóticas), y las estrategias de control utilizadas no son mostradas con claridad, e incluso, no corresponden a formas sofisticadas (desde un punto de vista teórico y académico) sino a soluciones de programación por software y corrección de tipo On-Off.

Desde hace varias décadas, la necesidad de obtener medios para poner en práctica la teoría de control a bajo costo es cada vez más fuerte. En [15], ya se ve un ejemplo de como combinar la enseñanza de control con pocos recursos, específicamente, sobre sistemas eléctricos de fácil montaje. Por otra parte, ya se ha visto que los desarrollos en Lego son muy útiles en el fortalecimiento de cursos de control [21], robótica y dinámica, además de ser unos excelentes impulsores de la mecatrónica [16]. En [13] es notable la intención de proponer un sistema de enseñanza para control embebido a bajo costo con Lego.

Se hace entonces muy importante crear un conjunto básico, organizado y detallado de prácticas de control, basadas en la implementación de retos clásicos de control. El ideal de un paquete esta naturaleza, es el de contar con características como la flexibildad de construcción, la facilidad en la implementación de estrategias de control y el bajo costo de sus componentes. Su objetivo fundamental sería el de permitir la enseñanza y aplicación de los conceptos básicos de implementación de controladores.

La motivación principal de este trabajo es la de desarrollar el conjunto de prácticas bajo la filosofía descrita con anterioridad, que colabore de forma activa en la educación en control, y en la aplicación y asimilación de conocimientos por parte de los estudiantes

1.2 Alcances

Con este trabajo no se pretende implementar y dar solución a todos los problemas de control clásico que se pueden hallar en la teoría. El objetivo buscado es el de desarrollar una serie de plantas basados en los retos clásicos de control (servo-mecanismo, bola y la viga, sistema de balance, servo-mecanismo con acoples flexibles múltiples), con diferentes grados de dificultad, todo esto con el fin de crear una guía básica para la realización de experimentos de control, con posibilidad de extensión a plantas más complejas e investigación.

La arquitectura a utilizar en la parte del hardware es la provista por el paquete Lego®Mindstorms®. Éste cuenta con los elementos básicos para conformar sistemas de control sencillos (sensores, actuadores, sistemas con la capacidad de implementar estrategias de control) pero muy ilustrativos de los conceptos básicos del control y los sistemas retroalimentados. La plataforma de implementación de los controladores es la unión entre lenguaje C estándar montado sobre sistemas de emulación de Linux (Cygwin sobre Microsoft Windows) y Python. Con la utilización del software y plataformas de programación libres, se pretende dar un valor agregado a las plantas y sistemas montados, ya que estos no implicarían un costo económico adicional para cualquier usuario final.

Para mantener la similitud entre los sistemas reales y sus pares simulados, se desarrollarán prácticas de control con una dificultad en implementación no muy superior a la que resulta al trabajar con las simulaciones. Dado que se trata de sistemas con una finalidad pedagógica, no tiene sentido que la complejidad en la implementación del control de los sistemas que se vayan a construir se convierta en un impedimento para su utilización.

1.3 Plantas Seleccionadas

Para el desarrollo del proyecto fueron escogidos cuatro sistemas sobre los cuales se pueden llevar a cabo tareas de modelado y control de una complejidad básica y media. El propósito es ilustrar conceptos básicos de la tarea de control y los efectos de la realimentación sobre la dinámica de los sistemas. Algunos son sistemas de uso muy frecuente en la teoría de control para el estudio de estrategias de control tanto clásico como avanzado, o versiones de tales dispositivos. Son un vehículo autónomo, un sistema de Bola y Viga, un Segway y un servomecanismo con acoples flexibles o elásticos.

El sistema Ball & Beam, o la bola y la viga, consiste de una viga larga cuya inclinación puede ser modificada mediante el giro de un servomecanismo conectado a la misma, y sobre la cual rueda una esfera. El propósito es gobernar la posición de la bola sobre la viga. Es un ejemplo muy común dentro de la teoria de control.

El segway es un sistema de transporte personal eléctrico de dos llantas, con autobalanceo controlado mediante un computador o procesador. El usuario se inclina en la dirección en la que quiere que se mueva el dispositivo y los motores y el control interno mantienen el sistema en posición vertical todo el tiempo. Es una versión del péndulo invertido, un ejemplo muy conocido en la literatura y laboratorios control automático. Aquí se presenta un segway autónomo, donde no hay un pasajero, sino estrategias diferentes para el movimiento, siendo el interés principal mantener el sistema en posición vertical.

Un sistema AGV (Auto Guided Vehicle), de forma muy sintética, es un vehículo que transita de manera autónoma, sin la necesidad de un conductor. Son sistemas concebidos para el transporte de materiales o supervisión de eventos o procesos, usualmente repetitivos, que pueden resultar peligrosos, tediosos o difíciles de realizar para un operador humano. Su popularidad dentro de la industria es creciente, dada su capacidad de realizar tareas de transporte de forma repetitiva y casi sin interrupciones. Es un sistema muy interesante de controlar y que se puede usar dentro de contextos científicos o industriales, de ahí su inclusión en el proyecto.

Los servomecanismos son uno de los ejemplos aplicados más comunes del control automático. Se trata de sistemas que deben manejar una variable de salida de una forma muy precisa y automática. Esa labor la realiza mediante la detección del error entre la variable de salida y el valor final esperado de la misma (referencia). Es una evidente ilustración de control en lazo cerado. El servomotor es la planta aquí propuesta, que en general requiere el control preciso de la velocidad y/o posición de su eje. Para otorgarle al ejercicio una complejidad y riqueza adicionales, se decidió incorporar al sistema un eje adicional conectado al motor mediante acoples flexibles, similar al que se encuentra en el Laboratoire d'Automatique de Grenoble, Francia [3]. Esto hace el modelo del sistema tenga un orden mayor e incluso modos resonantes, haciendo más interesante el ejercio de control.

1.4 Hardware y Software

El proyecto tiene dos componentes fudamentales, el módulo de software, que permite implementar las estrategias de control, y el módulo de hardware, con el que se construyen y diseñan las plantas. El componente de hardware escogido para la construcción es el producto conocido como Lego®Mindstorms®y para el software, la unión entre Python, para la implementación de las librerias que permitan el enlace con el hardware, a manera de interfaz, y GCC, para la implementación formal de las estrategias de control. En capítulos posteriores se hace una descripción más detallada de cada módulo.

1.5 Trabajos Similares

En las áreas de control y ciencias de la computación, es muy usual encontrar proyectos didácticos similares al planteado en este documento, enfocados principalmente en problemas de robótica móvil. En [18] por ejemplo, se puede ver un proyecto de construcción, diseño y enseñanza de robótica móvil en programas de posgrado con Lego. La conjunción entre control y robótica es aún más notoria en proyectos en los que técnicas descentralizadas de navegación son aplicadas [17] o en los que se utiliza la robótica dentro del ámbito del diseño mecatrónico [24]. En este último campo, es de resaltar el estudio de robots con comportamiento giroscópico que se hace en [16].

Desde un punto de vista del control aplicado y las plantas clásicas en educación, se pueden encontrar trabajos enfocados en la reproducción de retos aislados como el péndulo invertido [21] o el servomotor [13], en el cual se muestra una aplicación muy interesante de control embebido a bajo costo. Existen también ejemplos de aplicaciones industriales implementadas con Lego, en las que se utiliza el Brick de Lego como un regulador industrial [11].

En cuanto a trabajos con pretensiones similares al aquí mostrado, se puede hallar el desarrollo de kits de procesos de control [9], donde se analizan y discuten los resultados de construir diferentes plantas y utilizarlas en ambientes académicos. Este proyecto mezcla elementos Lego con materiales de fácil acceso y bajo costo. Es uno de los ejemplos más cercanos a los propósitos del proyecto presentado en este documento.

Dentro del aspecto de software, podemos encontrar librerías desarrolladas en Labview^{TM}y Matlab que realizan la compilación y descarga de archivos de control para el Brick. No obstante, las dos plataformas de desarrollo son comerciales y de costo elevado.

1.6 Estructura del documento

El documento consta de dos partes principales. En la primera, aparecen los capítulos correspondientes a la descripción y justificación del proyecto (cápitulo 1), los componentes de hardware y software utilizados en su realización (capítulo 2), la metodología de diseño utilizada (capítulo 3), y una descripción de proyectos similares (capítulo 4).

La segunda parte está conformada por la descripción y las pruebas realizadas con cada una de las plantas seleccionadas (capítulos 5 al 8) y la exposición del software creado para acompañar los sistemas y permitir la implementación de controladores (capítulo 9). Finalmente aparece el capítulo correspondiente a las conclusiones y trabajo futuro, además de los anexos, que incluyen algunas indicaciones de uso de cada sistema y los códigos fuente creados.

2
Elementos de la Plataforma de Prácticas para Control

2.1 Hardware

Para la realización del proyecto, se decidió utilizar el producto Lego® Mindstorms® NXT. Se trata de un artículo perteneciente a la familia de kits desarrollados por The Lego Group basados en la filosofía de ladrillos. Este tipo de sistemas permiten al usuario (el público infantil es su mayor cliente) construir innumerables prototipos de máquinas, mecanismos y robots, o simplemente recrear situaciones del mundo real. En particular, la línea Mindstorms fue pensada para la construcción de robots, por lo que es usual encontrar ejemplos en la literatura relacionada con el producto o en Internet sobre todo tipo de criaturas robóticas construidas con el mismo, desde caminadores bípedos hasta insectos. No obstante su público objetivo, niños de 8 a 16 años, el Lego Mindstorms se ha convertido en un punto de referencia para los aficionados a la robótica de todas las edades y nacionalidades, haciéndolo muy famoso a nivel mundial.

De entre las razones para seleccionar Lego Mindstorms como la plataforma de hardware a utilizar dentro de este proyecto, además de su capacidad de reproducir casi cualquier mecanismo real, sin las implicaciones técnicas y económica que conlleva la contrucción de éstos por medios tradicionales, se encuentran la filosofía educativa original del producto y los componentes electrónicos del sistema, asimilables a muchos de los elementos de un sistema de control realimentado, como se verá en la siguiente sección.

2.1.1 Ventajas y Desventajas de Lego

Dentro de las ventajas de usar el sistema Lego Midstorms, podemos encontrar [34]:

- Gran facilidad de montaje y desarmado, no es necesario usar soldadura, tornillos, u otros métodos de sujeción o unión. Todo sistema que es armado se puede desarmar rápidamente. Además, eso permite usar las piezas en múltiples diseños diferentes.

- Se encuentra muy extendido a nivel mundial, lo que permite encontrar gran cantidad de información e ideas por Internet, diseños, soluciones, participar en foros y competiciones. Además se pueden hallar libros dedicados al tema e incluso publicaciones científicas y académicas que hacen referencia a su utilización.
- No se trata de un paquete cerrado, es decir, se puede comprar más ampliaciones de lego, adquirir piezas deterioradas o perdidas, o añadir piezas construidas manualmente, como por ejemplo, sensores o motores, e incluso circuitos neumáticos.
- Múltiples posibilidades y lenguajes de programación, desde el nivel más básico e intuitivo, como el NXT-G, hasta el uso de lenguajes conocidos como C o Java, entre otros.
- Que sea escalable, es decir, que a partir de un material básico haya opciones de ampliación.
- Muy indicado para entornos educativos, desde colegios hasta universidades, pues se puede aprender de forma fácil tanto mecánica como electrónica.

Las desventajas del sistema son [34]:

- La principal desventaja de LEGO es su estructura. Está formada por bloques de LEGO, que se unen por simple presión. Si bien se pueden añadir elementos de refuerzo y sujeción, pero para diseños exigentes, no es recomendable. Golpes, caídas, pueden debilitar rápidamente la estructura, llegando a desarmar el robot. No obstante, las aplicaciónes que se proponen en el presente trabajo, no implican que los sistemas construidos vayan a ser sometidos a regimenes de trabajo peligrosos para su integridad estructural.
- No se pueden construir estructuras circulares, pues todas las piezas y ladridos de LEGO son rectangulares.
- Relación masa-volumen. Las piezas LEGO no son útiles en diseños donde la relación masa-volumen se hace crítica. Por ejemplo, para construir un robot de competencia (lucha o SUMO), no sería eficiente, pues la estructura LEGO es demasiado liviana, y se deberían añadir pesos para hacer el robot más robusto, o el caso contrario, para construir robot pequeños, ligeros, y resistentes, las piezas LEGO, si bien tienen una gran calidad constructiva, no alcanzan la calidad de materiales cómo la fibra de carbono.

En cuanto al precio, las opiniones que se pueden hallar son diversas. Para los aficionados a la robótica, Lego resulta más costoso para las construcciones de tales sistemas, ya que es más barato construir un robot con materiales caseros que usar piezas “prefabricadas”. Sin embargo, desde el punto de vista de control, la utilización de Lego para la construcción de plantas en la que se puedan aplicar sus conceptos, no conlleva las elevadas implicaciones económicas y técnicas de contar con soluciones ya fabricadas como los servomotores y péndulos invertidos existentes en el mercado. El costo del Lego es muy inferior al de estos sistemas.

Durante el año 2006, la Universidad Nacional de Colombia adquirió 8 equipos Lego Mindstorms NXT equipados con respectivos sensores giroscópicos, de luz temperatura, ángulo y tacto, a la compañía Adtech LTDA. por un valor de 15 millones de pesos colombianos[1]. Cada uno de estos permite la cosntrucción de varias de las plantas desarrolladas para la plataforma de prácticas aquí mostrada, además de una infinidad de dispositivos robóticos con los que fácilmente pueden realizarse prácticas de control, robótica y otras ciencias. En el mismo periodo fue comprado a ICL Didáctica LTDA un helicóptero Quanser con sus accesorios (fuentes, tarjetas de desarrollo, software, etc.), que permite la realización de diversos experimentos de control, tanto a nivel de pregrado como de posgrado. El costo fue de 62 millones de pesos colombianos. La diferencia de precios es abrumadora.

De acuerdo a los requerimientos generales de este proyecto, las ventajas ofrecidas por la plataforma Lego la hacen ideal para el mismo, y compensan las desventajas, que no representan mayor problema a la hora de construir e implementar las prácticas propuestas.

El Lego tiene su cerebro en un dispositivo llamado "*Ladrillo programable*". Estos dispositivos son ladrillos plásticos de un tamaño similar a un mazo de cartas con microcomputadores portátiles embebidos. Al ser usados con piezas de Lego, o de otros materiales, permiten modelar y contruir robots para ser operados mediante motores y sensores. Claramente, introducen a la persona en el uso intuitivo de elementos propios del control automático. De acuerdo al National Centre for Technology in Education de Irlanda [27], los ladrillos programables pueden ser usados en el campo educativo, por ejemplo, para:

- Motivar a estudiantes de todas las edades y habilidades para crear y desarrollar sus propios modelos basados en un campo de temas dentro de un currículo.
- Ofrecer al estudiante un entendimiento más profundo de muchos conceptos científicos y su habilidad para conducir experimentos basados en comportamiento, realimentación y control.
- Mejorar la comprensión de conceptos de ingeniería, como el funcionamiento de mecanismos.
- Introducir a los estudiantes a la programación básica y la robótica.

Estas razones convierten al Lego Mindstorms en el producto ideal para desarrollar una plataforma de prácticas educativas, no solo a nivel de control automático, sino puede cubrir un amplio espectro de niveles educativos, desde escolar básico hasta universitario. Esta es la razón para incluir al Lego Mindstorms como el hardware de desarrollo de las plantas.

2.1.2 Reseña Histórica del Lego Mindstorms

Lego Mindstorms NXT es el resultado de un largo proceso de integración e investigación entre el Massachusetts Institute of Technology (MIT) y Lego, iniciado en el año 1986. El acuerdo original

[1] Tomado de la web de la Facultad de Ingeniería, Universidad Nacional de Colombia

estipulaba entre otras cosas, la financiación de las investigaciones realizadas por el grupo de epistemología y aprendizaje del MIT por parte de Lego, y a cambio, obtendría ideas innovadoras para sus productos, lo cuales podría lanzar al mercado sin pagar regalías [2].

El convenio llevó al desarrollo del dispositivo conocido como "*Ladrillo Programable*" o "*Brick*". La idea del MIT era la de crear un dispositivo basado en microcontroladores que fuese muy fácil de programar. El resultado era un pequeño computador embebido dentro de una estructura plástica de Lego, que era capaz de interactuar de forma física con el mundo en una gran variedad de formas. Originalmente diseñado para niños, el Brick tenía la premisa de ofrecer al niño la posibilidad de introducir y controlar computadores en el mundo, en lugar de controlar y manipular mundos dentro del computador. Este enfoque, le otorga al usuario del Brick el poder no solo de crear y controlar objetos reales, sino también le permite crear conexiones entre diversos conceptos científicos y el mundo que lo rodea [22].

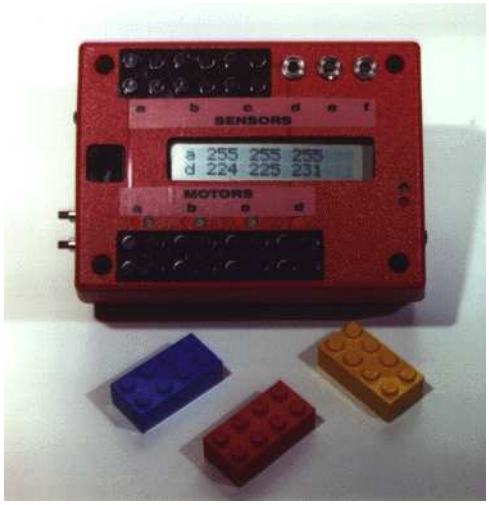


Figura 2.1: Versión experimental: MIT Programmable Brick[a]

[a]Tomada de http://el.media.mit.edu/logo-foundation/pubs/logoupdate/v7n1/v7n1-pbrick.html

El Ladrillo Programable fue probado por el MIT en diversas aplicaciones, tales como los ambientes activos, en los que dos o más niños construían mecanismos que les permitieran manipular los objetos de su entorno y su comportamiento, o las criaturas autónomas, en las que se contruían máquinas con apariencia de animales, se les otorgaba un cierto comportamiento para que interactuaran con el entorno. Todo se desarrollaba en forma de retos, y obligaba a los niños a plantear soluciones. Los experimentos tuvieron un éxito tal, que impulsó la creación de una versión comercializable por parte de Lego [22].

[2]Disponible en http://es.wikipedia.org/wiki/Lego_Mindstorms y [31]

El primer resultado de uso comercial del acuerdo, fue la primera generación de Mindstorms, el RCX. La primera versión de tres que saldrían al mercado, apareció en el mercado en 1998 con un precio cercano a los 200 dolares americanos. Los resultados en ventas fueron superiores a cualquier expectativa, vendiendo más de 80.000 unidades en tres meses, lo que permitió su increible popularidad dentro del campo de la robótica a nivel internacional [31].

Las características principales del RCX son en términos de potencia, inferiores a la de la versión actual del sistema, NXT. Entre éstas se puede destacar una CPU Hitachi H8/3292 a 16MHz, 16 KB de memoria ROM y 32 KB de RAM. Adicionalmente contaba con tres entradas para sensores, tres salidas para actuadores y un puerto de infrarojos para comunicación. Su apariencia es muy cercana a la de los bloques tradicionales de Lego; de hecho, los sensores y el ladrillo programable conservan la forma de ladrillos caracteristica de la fábrica [31] (Fig. 2.2).

Figura 2.2: Primera versión comercial: RCX[a]

[a] Tomada de [31]

Una desventaja adicional del sistema RCX frente a la versión NXT la representan sus actuadores (motores eléctricos). Desprovistos de encoder y con un reductor de velocidad pequeño, no ofrecen un gran torque, aunque si una gran eficiencia. Su principal ventaja es su tamaño reducido, pero al carecer de encoder, requiere de sensores externos para la estimación de variables como su posición angular.

Durante el año 2004, debido a una crisis de resultados financieros en Lego, con pérdidas cercanas a los 190 millones de euros, se rumoró que la compañía abandonaría la línea Mindstorms y regresaría a su mercado tradicional. Sin embargo, a comienzos de 2006, Lego anunció el NXT, que

Figura 2.3: Motor Eléctrico del RCX[a]

[a]Tomada de http://www.philohome.com/motors/motorcomp.htm

salió al mercado en junio de ese mismo año [31][3].

El paquete de desarrollo que viene con el sistema NXT está enfocado principalmente hacia el campo de la robótica, por lo que permite construir sistema mucho más elaborados e interesantes desde el punto de vista de la construcción mecánica que sus versiones anteriores. Es por esta razón que se prefiere el uso del NXC en este trabajo sobre sus predecesores. Incluso la estética del sistema actual es más atractiva y sus piezas son similares a la perfileria con la que se construyen máquinas sencillas hoy día.

[3]http://es.wikipedia.org/wiki/Lego_Mindstorms

Figura 2.4: Versión actual: NXT[a]

[a]Tomada de http://shop.lego.com/ByTheme/Product.aspx?p=9841&cn=17&d=70

La línea de productos Mindstorms no ha sido bien vista por los aficionados a Lego, que la acusaban de perder la orientación tradicional de la empresa, ni por los miembros del nombrado grupo de epistemología del MIT, dado que considran que algunas de las decisiones comerciales de Lego contradecían los resultados de las investigaciones conjunta [31]. Sin importar estas razones, Lego Mindstorms se ha convertido en uno de las herramientas de desarrollo favoritas de los aficionados a la robótica de cualquier edad o nacionalidad.

La enorme popularidad de Lego Mindstorms, hace que el producto llame la atención de cualquier usuario, disminuyendo la aprensión hacia el mismo, e invita a descubrir la características del sistema. Esta es una razón muy importante a la hora de escoger al Lego como el material base de la plataforma. Como punto adicional, la popularidad trae consigo un caudal enorme de información y documentación, como ya se habia mencionado, que facilita en gran medida el trabajo con este producto.

2.1.3 Componentes del Lego Mindstorms

El sistema Lego Mindstorms cuenta con diferentes elementos constructivos, de sensórica y procesamiento que son comunes a cualquier sistema de control en lazo cerrado simple, haciéndolo ideal para la cosntrucción y diseño de las plantas que conforman la plataforma de prácticas de control. A continuación, se mostrará una descripción de los componentes más importantes del Lego Mindstorms.

Procesador: Brick Lego Mindstorms

El Brick o Ladrillo Programable es la parte funcional más importante del sistema. Es donde se ubica el cerebro del mismo y se determinan las señales de control necesarias para estabilizar, modificar la dinámica o cambiar el comportamiento de cualquier sistema construido. En pocas palabras, es el lugar donde se implementan los controladores. Dichas señales se sintetizan a partir de la información recogida por los sensores. La lectura también se realiza en el Brick. Éste cuenta con una serie de dispositivos montados en una tarjeta, que le otorgan las mencionadas capacidades. La tarjeta cuenta con[4]:

- Un microprocesador principal: ARM7TDMI, de 32 Bits a 48 MHz; Memoria Flash de 256KB y RAM de 64KB.
- Un microcontrolador ATMEGA48 de 8 Bits a 4MHz; Memoria Flash de 4KB y RAM de 512 Bytes.

[4] http://mindstorms.lego.com/en-us/whatisnxt/default.aspx

- Controlador USB 1.1, Controlador Bluetooth BlueCore 4.
- 4 puertos de entrada para sensores, 3 puertos de salida para actuadores.
- Display LCD.

El microprocesador es el "cerebro" del Brick, donde se descargan y corren los algoritmos programados. El microcontrolador es el encargado de leer los puertos de entrada (sensores) y de generar señales de tipo PWM hacia los puertos de salida (actuadores). El microprocesador y el microcontrolador utilizan el protocolo I^2C para comunicación, mientras la comunicación del Brick con el PC es una labor del microprocesador realizada usando USB o Bluetooth.

Los puertos de entrada/salida del Brick NXT usan una interfaz de 6 cables con un conector similar al tipo RJ-12 (común en telefonía); cada uno de estos cables tiene un color y una función asociada que dependen de la función del puerto (entrada o salida), y del tipo de sensor se conecte al puerto (solo entrada). La tabla 2.1 muestra la configuración de cables en los puertos de *entrada* [23].

No. de cable	Color	Nombre
1	Blanco	AN
2	Negro	GND
3	Rojo	GND
4	Verde	VCC
5	Amarillo	DIGI0
6	Azul	DIGI1

Tabla 2.1: Configuración de cables los puertos de entrada del Brick NXT

Cada uno de los cables tiene una función definida dentro del puerto, como se muestra a continuación:

- **AN**: tiene dos fines, como entrada analógica, o como alimentación de 9 *V* para usar sensores del tipo RCX (antecesor del NXT). Cuando se usa como entrada analógica, esta pasa a un ADC de 10 bits, con un rango de entrada de 0 a 5 *V* y un tiempo de muestreo de 3ms.
- **GND**: corresponde a los pines de tierra. Todas las señales en el Brick son referidas a esta tierra.
- **VCC**: es la alimentación para los sensores del tipo NXT, y corresponde a 4,3 *V*.
- **DIGI0**, **DIGI1**: corresponden a señales digitales de 3,3 *V* conectadas directamente al microprocesador del Brick. Son usados principalmente para comunicaciones I^2C.

En la tabla 2.2 se observa la configuración de cables en los puertos de *salida* [23]. Vale la pena anotar, que aún cuando los puertos están destinados como salidas, existen 2 cables (2 líneas) en cada uno de ellos, que sirven como *realimentación* para realizar lecturas de los actuadores del tipo NXT.

No. de cable	Color	Nombre
1	Blanco	M1
2	Negro	M2
3	Rojo	GND
4	Verde	VCC
5	Amarillo	TACHO0
6	Azul	TACHO1

Tabla 2.2: Configuración de cables en los puertos de salida del Brick NXT

Las funciones de cada cable son:

- **M1**, **M2**: manejan los servomotores NXT. Los servomotores NXT son controlados a través de un puente H y por modulación de ancho de pulso (PWM). La corriente sobre los puertos es de aproximadamente 700 *mA* y puede llegar a valores pico de 1 *A*.
- **GND**: es la tierra de los puertos de salida, a diferencia de las tierras de los puertos de entrada que corresponden a 2 pines.
- **VCC**: alimentación de 4,3 *V* para sensores del tipo NXT.
- **TACHO0**, **TACHO1**: son 2 *entradas* para leer el encoder óptico que poseen los servomotores NXT. La lectura se hace con señales en cuadratura.

Actuadores: Motores

Los motores que vienen incluidos con el Kit de Lego Mindstorms, son dispositivos electromecánicos de corriente continua e imán permanente. Las variaciones vienen por el lado de la potencia disponible y de la transmisión. El torque y la velocidad de rotación de cada motor está determinado por la existencia y el tipo de engranajes reductores que contenga, además de su potencia nominal [35].

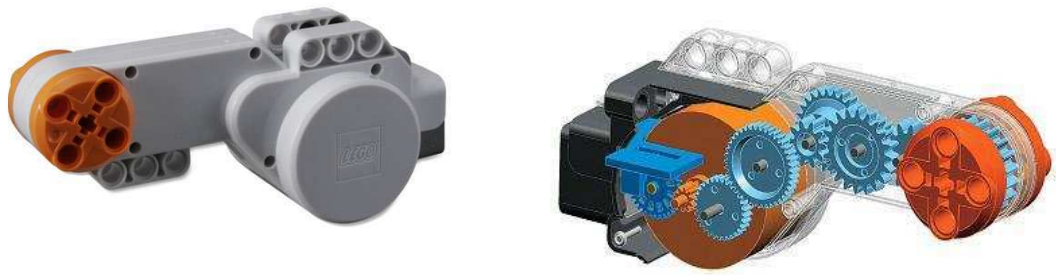
Todos los motores de la línea RCX disponen de un encapsulado compatible con los ladrillos LEGO. Los motores NXT en cambio adoptan el nuevo sistema de ensamble de piezas largas, igual que el que tiene el Brick. Se destaca que tanto los sistemas RCX como los NXT disponen de un gran surtido de accesorios para el mejor uso de estos motores. Entre otras piezas se tienen trenes de engranajes, juntas universales (cardan), diferenciales, poleas, cremalleras, etc. Con esto se puede

dotar a los sistemas con características especiales como por ejemplo bloquear totalmente el motor en ausencia de tensión usando una combinación de piñón y tornillo sin fin (Tomado de [35]).

Usualmente, el control de estos motores se hace mediante la técnica de modulación de ancho de pulso (PWM) Algunos motores cuentan con protección interna contra sobrecargas (termistores) y también contra sobretensiones. Esto implica que no siempre es posible sostener velocidades máximas por tiempo indefinido [35].

El motor de la línea RCX (Fig. 2.3) tiene un peso cercano a los 30 *gr* y alcanza una velocidad nominal de 340 RPM (libre). Su consumo de corriente es bajo (9 *mA* sin carga y 340 *mA* frenado) pero el torque que entrega es bajo comparado con muchos motores DC de características similares (6 $N \cdot cm$)[5]. No obstante, puede ser conectado al NXT con un cable conector especial, por lo que se puede considerar para aplicaciones donde se involucre la velocidad en lugar de la fuerza.

El motor de la línea NXT (servo motor), que se observa en la figura 2.5(a), es más pesado (80 *gr*) y lento (170 RPM libre), lo que se debe a la incorporación de un tren reductor (Fig. 2.5(b)). La caja reductora le permite al motor generar torques más grandes (50 $N \cdot cm$) con un consumo mayor (60 *mA* libre y 2000 *mA* frenado). Es el más indicado para labores que impliquen más fuerza como la construcción de robots y dispositivos mecánicos móviles.



(a) Aspecto Físico (b) Caja Reductora

Figura 2.5: Motor NXT[a]

[a]Tomada de http://www.philohome.com/nxtmotor/nxtmotor.htm

El servomotor tiene un sensor de rotación incorporado que mide la posición angular. Además, es capáz de reportar dicha información al Brick. Esto permite un control completo del motor y la ejecución de pasos muy precisos (con exactitud de hasta un grado). Los sistemas operativos operativos embebidos en el Brick (firmware) permiten manejar varios motores a la vez, e incluso sincronizarlos

[5]Tomado de http://www.philohome.com/motors/motorcomp.htm

para ser manejados a la misma velocidad.

Sensores

Los sensores permiten a cualquier sistema conocer información sobre su entorno. A partir de los datos recogidos se pueden tomar decisiones correctivas para cambiar la dinámica y el comportamiento de los sistemas. Esa es la base de disciplinas como el control en lazo cerrado, la robótica y la inteligencia artificial, entre otras. El Lego Mindstorms cuenta con sensores de diferentes tipos que le permiten medir variables de naturaleza diversa, por lo que es muy útil para realizar tareas de control sobre muchos tipos de sistemas, especialmente de tipo mecánico.

Estos dispositivos son una razón adicional para usar Lego Mindstorms dentro de este proyecto, ya que se encuentran especialmente adaptados para el uso con el Brick, por lo que (casi siempre) se evita la tarea de caracterización y linealización de los sensores, haciendo que el usuario se preocupe solo por la tarea de control propuesta, y aproveche la medida dada por el sensor tan pronto como conecta el mismo. Otorgan simplicidad, practicidad y facilidad de uso. A continuación se muestran algunos de los sensores que vienen con el producto y que serán usados en el proyecto.

Sensor de Distancia: Ultrasonido

El Sensor Ultrasónico viene incluido en el kit de Lego Mindstorms, y su principal función es detectar las distancias y el movimiento de un objeto que se interponga en el camino del robot, mediante el principio de detección ultrasónica. Este sensor es capaz de detectar desde 0 a 255 *cms*, con una precisión relativa del ± 3 *cms*[6].

Mediante el principio del eco, el sensor tiene la capacidad de recibir la información de los distintos objetos que se encuentren dentro de su rango detección, teniendo un mejor reflejo del eco los elementos planos que los cuerpos curvos, como esferas u otros elementos similares. Es interesante advertir que si se realizan conexiones múltiples de este sensor, se puede detener la ejecución de los programas y/o lectura de los distintos elementos dado que es un sensor de baja velocidad, que maneja el protocolo de comunicación I^2C. El sensor está configurado por defecto para medir la distancia a un objeto, pero también puede ser capaz de medir las distancias hasta un máximo de 8 objetos en un único periodo de medida[7].

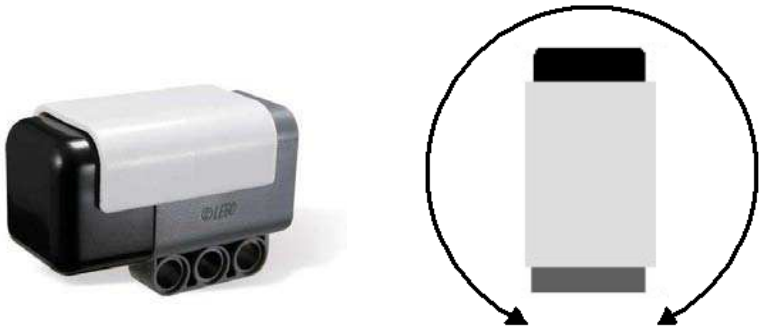
[6] http://es.wikipedia.org/wiki/Lego_Mindstorms
[7] http://ladrillikos.wikidot.com/transductores-y-sensores

Figura 2.6: Sensor de Distancia[a]

[a]Tomada de http://shop.lego.com/product/?p=9846&LangId=2057&ShipTo=US

Giroscopio o Giróscopo

El giroscopio es fabricado por la compañia Hi-Technic, y es compatible con los puertos de entrada del Brick NXT (Fig. 2.4). Es un giróscopio de un solo eje que mide la velocidad angular en grados por segundo (puede medir hasta ±360°). El sensor utiliza el pin de entrada analógica del puerto del Brick, por lo que la lectura es un valor entre 0 y 1023. La lectura puede hacerse hasta 300 veces por segundo (≈ cada 3.3 *ms*). El eje de medición está definido en el plano vertical al posicionar el giróscopo con la tapa negra hacia arriba como lo muestra la Fig. 2.7(b).



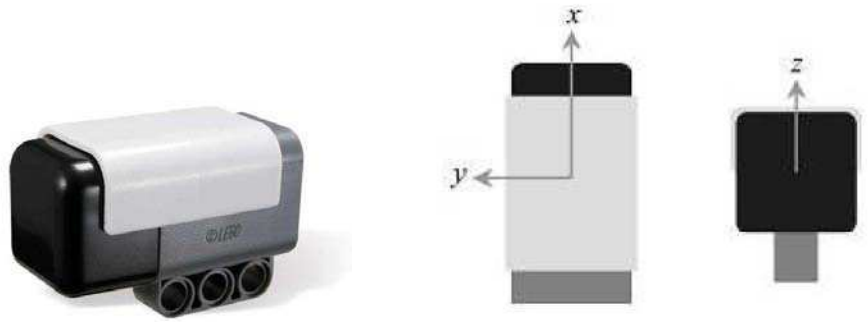
(a) Aspecto Físico (b) Eje de medición

Figura 2.7: Giroscopio NXT[a]

[a]Tomada de http://www.teamhassenplug.org/NXT/NXTSensors.html

Acelerómetro

El acelerómetro también es fabricado por la compañía Hi-Technic, y es compatible con los puertos de entrada del Brick NXT. Es un acelerómetro digital de 3 ejes (x, y, z) cuya medida esta en el rango de $-2g$ a $+2g$, con 200 muestras por cada g (Fig. 2.8(a)).



(a) Aspecto Físico (b) Ejes de medición

Figura 2.8: Acelerómetro NXT[a]

[a]Tomada de http://www.teamhassenplug.org/NXT/NXTSensors.html

El acelerómetro transmite el vector de aceleraciones en x, y, z, vía I^2C usando los pines digitales del puerto de entrada del Brick. La lectura puede hacerse hasta 100 veces por segundo (cada 10 ms). Los ejes x, y, z, de medición se muestran en la figura 2.8(b).

Brújula Magnética: Compás

La brújula o compás, es capaz de medir el campo mágnético terrestre y calcula el ángulo con respecto al norte magnético. Puede ser usado como un medio de orientación geográfico para un dispositivo móvil. Su utilización no presenta ningún problema, salvo que hay que tratarla como un sensor de baja velocidad (se comunica mediante el protocolo I^2C). Al trabajar con ella, devuelve directamente un entero de 0 a 359, el valor en grados de diferencia al norte magnético. Según las instrucciones del mismo fabricante, es conveniente montarla alejada de los motores y el ladrillo inteligente (al menos 10 cm de este último y 15 de aquellos), para evitar interferencias, y mantenerlo sujeto firme y horizontalmente[8]. El compás puede verse en la figura 2.9.

[8]Disponible en http://www.hitechnic.com/cgi-bin/commerce.cgi?preadd=action&key=NMC1034

Figura 2.9: Brújula Magnética[a]

[a]Tomada de http://www.hitechnic.com/cgi-bin/commerce.cgi?preadd=action&key=NMC1034

EOPD

El sensor EOPD es un detector de proximidad electro-óptico. Es un dispositivo que permite detectar objetos y medir distancias, empleando el principio de reflexión de luz visible para su operación. Es similar al sensor de luminosidad estándar, pero al emitir pulsos de luz, elimina la interferencia producida por el ambiente o superficies de fondo. Esta emisión, hace que el brillo sea relativamente bajo pero seguro para la lectura, dado que la potencia de salida es baja.

El dispositivo obtiene la diferencia entre la luz medida en el detector antes que el pulso sea emitido y la luz emitida durante el pulso. Entonces realiza la resta, por lo que la lectura es una medida directa de la energía de la luz reflejada, eliminando cualquier interferencia provocada por la luz ambiente. Puede detectar las piezas de Lego® en su modo de funcionamiento x1 o la bola roja del kit Mindstorms® en el modo x4, en un rango de hasta 18 cms[9].

Figura 2.10: EOPD[a]

[a]Tomada de http://www.hitechnic.com/cgi-bin/commerce.cgi?preadd=action&key=NEO1048

[9]http://www.hitechnic.com/cgi-bin/commerce.cgi?preadd=action&key=NEO1048

2.2 Software

El software que acompaña a las plataformas de práctica de control desarrolladas con Lego consta de dos partes principales: una parte de programación directa sobre el Brick, y otra de interfaz de usuario. A continuación se muestran las alternativas existentes para la programación directa sobre el Brick. Más adelante, se ilustra el software usado para la interfaz de usuario.

2.2.1 Herramientas de Programación de Lego

La versión comercial de Lego Mindstorms NXT incluye un software de desarrollo de tipo gráfico, basado en NI Labview, muy fácil de usar, funcional en Windows y Mac. Sin embargo, dado que se busca una mayor profundidad al programar y un acceso a nivel más bajo, se hace necesaria la escogencia de un lenguaje de programación por entrada de texto (aún de alto nivel), más funcional y orientado a un uso más serio que el propuesto por LEGO (que, no obstante, es bastante completo y útil para una introducción a la programación).

Lenguajes Existentes

Next Byte Codes NBC y Not eXactly C NXC

El Next Byte Codes (NBC) es un lenguaje de programación simple, con sintaxis de lenguaje ensamblador (assembler) que puede ser usado para programar el Brick NXT. Not eXactly C (NXC) es un lenguaje de alto nivel, similar a C, construido sobre el compilador del NBC. Al igual que su lenguaje base, es ampliamente usado en la programación del Lego Mindstorms[10].

El NXT tiene un intérprete de código de bytes (provisto por Lego), que puede ser usado para ejecutar programas. El compilador del NXC traduce el código fuente a código de bytes NXT, que puede ser ejecutado en el Brick. Aunque el preprocesador y las estructuras de control son muy similares a las de C, el NXC no es un lenguaje de propósito general, lo que se ve representado en ciertas limitaciones heredadas del intérprete de código de bytes [36]. El NXC posee una interfaz muy fácil de usar, y que permite tener acceso a las principales características del Brick, de ahí su popularidad.

Lejos NXJ

LeJOS NXJ es un proyecto de programación consistente en el desarrollo de un firmware (sistema operativo) de reemplazo para el Brick NXT. El lenguaje base utilizado es Java, por lo que se le considera una Máquina Virtual (Java Virtual Machine) en miniatura para el Brick. Inclute un API con las

[10] http://bricxcc.sourceforge.net/nbc/

clases necesarias para trabajar con el Brick, además de las herramientas necesarias para descargar el código en el mismo[11].

Con LeJOS NXJ se puede trabajar programación orientada a objetos (Java), interrupciones, manejo de arreglos, recursión, sincronización, excepciones, manejo de variables de Java (enteros, cadenas de texto, flotantes), clases de java.lang, java.util y java.io, y un API muy completo. Es un proyecto muy interesante y bien documentado, del cual se desprende el proyecto nombrado a continuación, el NXT Osek.

NXT Osek

Se trata de un sistema operativo de tiempo real (RTOS) diseñado para el Brick NXT, derivado del proyecto Lejos NXJ y basado en el estándar OSEK[12], de gran popularidad al interior de la industria automotriz. Este sistema permite la utilización de un ambiente de programación fundamentado en C/C++, que usa el GNU ARM Toolchain, una interfaz de programación de aplicaciones (API) de uso libre, desarrollada para los procesadores RISC de Acorn (posteriormente manufacturados por Advanced RISC Machines Ltd.).

La interfaz proporciona acceso a los sensores y motores NXT así como para otros dispositivos (mediante funciones específicamente desarrolladas), soporte para operaciones en punto flotante, capacidad de ejecutar múltiples funciones (tasks) en tiempo real, una rápida ejecución y un mínimo consumo de memoria. Una gran ventaja de la interfaz es la utilización del compilador de GNU, de buen desempeño y muy popular entre los programadores de software libre[13].

Cualquier programa escrito en C/C++ para el procesador ARM7 del Brick NXT es compilado con GCC (GNU ARM) a través de Cygwin. El archivo binario creado para ser ejecutado en el Brick es un archivo con extensión *.rxe*, el cuál es descargado al Brick usando la herramienta NeXTTool.

A continuación describiremos algunas de las herramientas adicionales que son necesarias para la utilización del NXT Osek:

- **Cygwin:** se trata de un ambiente desarrollado por Cygnus Solutions para proporcionar una plataforma similar al sistema operativo Linux sobre Windows. Consiste en dos partes fundamentales, un DLL que actua como una capa emuladora del API de Linux proporcionando parte de su funcionalidad, y una colección de herramientas que le dan al usuario la sensación de estar usando Linux.

[11] http://lejos.sourceforge.net/nxt/nxj/tutorial/Preliminaries/Intro.htm
[12] OSEK: Offene Systeme und deren Schnittstellen für die Elektronik im Kraftfahrzeug (Sistemas abiertos y las correspondientes interfaces para electrónica automotriz)
[13] http://lejos-osek.sourceforge.net/whatislejososek.htm

Con Cygwin es posible acceder y usar diferentes utilidades de Linux como GCC (compiladores de carácter libre para C/C++ y Fortran), el cual es fundamental para crear los archivos ejecutables que irán en el Brick.

- **GNU ARM Toolchain:** GNU Toolchain Es un término general para nombrar a una colección de herramientas de programación producidas por el proyecto GNU. Estas herramientas permiten realizar todo el proceso de compilación, enlazado (link) y depuración de una aplicación. De forma más específica, GNU ARM Toolchain es el uso de tales herramientas para generar código ARM.
- **Fantom NXT Driver:** Es el paquete de software que le permite al PC reconocer el Brick y actúa como interfaz con el mismo. Maneja la comunicación USB con el dispositivo.
- **Mindstorms NXT Driver:** es el paquete que permite actualizar los drivers para el dispositivo y usualmente es usado para descargar un nuevo firmware en el dispositivo.
- **NextTool:** Es una aplicación realizada por John Hansen para comunicación con el Brick. Funciona bajo línea de comandos y opera sobre los drivers mencionados en los numerales anteriores.

Entornos Integrados de Desarrollo

- **NXT-G:** es el IDE original[31], basado en Labview (National Instruments) y viene incluido con el producto NXT. Contiene tutoriales, herramientas de actualización de firmware y otras características. Si bien tiene un gran valor educativo o de iniciación, no es de gran utilidad en ambientes donde se requiera mayor profundidad en los aspectos de programación.
- **Robolab:** se trata de una versión anterior al NXT-G, compatible con el RCX. Se encuentra descontinuado.
- **BricxCC:** de libre descarga y código abierto, es probablemente el más completo de todos. BricxCC es un entorno para NQC y NBC/NXC con múltiples opciones, desde herramientas de actualización de firmware, explorador de archivos, control del NXT desde el PC, etc. Es uno de los más usados alrededor del mundo, por parte de los entusiastas de Lego Mindstorms. Es posible hallarlo para libre descarga e instalación en la página web del proyecto Bricx Command Center http://bricxcc.sourceforge.net/.
- **RobotC:** se trata de un entorno que permite la utilización del lenguaje *RobotC*, de licencia comercial (se adquiere pagando por el mismo) y muy popular entre aficionados de alto nivel, especialmente dentro de competiciones. Se consigue en la página de su proyecto http://www.robotc.net/.
- **Eclipse:** es un IDE muy familiar para los programadores de lenguajes como C y Java. Si bien no se trata de un entorno desarrollado específicamente para Lego, puede ser ajustado muy fácilmente para ser usado con Lejos o NXT-Osek.

La opción escogida para trabajar en este proyecto fue el NXT-Osek, ya que ofrece soporte para variables de tipo flotante (muy importante en la implementación de controladores), permite programar en un lenguaje de uso extendido como C, al tratarse de un sistema operativo en tiempo real, es ideal para sistemas de control embebido [25], y porque ya se cuenta con experiencia en el uso del mismo (ver capítulo 4). En la tabla 2.3 (tomada de [31]) aparece un resumen de los lenguajes y herramientas de programación más comunes dentro de la comunidad de desarrolladores de Lego®Mindstorms®.

2.2.2 Interfaz con el Usuario

Python

Se trata de un lenguaje de programación orientado a objetos, que puede ser utilizado con múltiples propósitos de desarrollo de software. Esta versatilidad y la capacidad de integrarse con otros lenguajes y herramientas, lo hace ideal para la propuesta aquí mostrada. Es la interfaz entre el usuario y el sistema construido, teniendo por funciones la recepción del controlador diseñado (función de transferencia, por ejemplo), la implementación del mismo en forma de algoritmo en GCC para NXT-OSEK, su compilación y descarga al Brick (Procesador embebido de Lego), todo en un mismo programa.

Python cuenta con un gran desarrollo de paquetes especializados en algebra lineal, análisis de sistemas dinámicos y control. Uno de las más importantes es SciPy[14], un software de código abierto para ciencia, matemática e ingeniería. La librería permite utilizar funciones y comandos similares a los de programas dedicados (i.e Matlab, Scilab), dentro de Python, haciéndola ideal para el tratamiento de entidades matemáticas como funciones de transferencia y representaciones en variables de estado, fundamentales en la implementación de controladores. En este proyecto fue utilizada una librería adicional, que permite aplicar SciPy directamente sobre problemas de control, sistemas dinámicos y Mecatrónica [15]. Fue desarrollada en la Southern Illinois University at Edwardsville, y contiene aplicaciones para control y análisis de sistemas dinámicos, de utilidad cercana a la de sus similares en lenguajes técnicos, como Matlab.

[14] http://www.scipy.org/SciPy
[15] Creada por el Dr. Ryan Krauss http://www.siue.edu/~rkrauss/python_intro.html

Nombre	Tipo	Firmware	IDE	SO	Libre
NXT-G Incluido al adquirir el NXT	Gráfico	Original	Sí (basada en NI Labview)	Win/Mac	No
RoboLAB Orientado al uso educativo (descontinuado, se usaba en las anteriores versiones de Lego Mindstorms)	Gráfico	Original	Sí	Win/Mac	No
NQC (Not Quite C) (se usaba en anteriores versiones de Lego Mindstorms)	C/C++	Original	Sí	Win/Mac Linux	Sí (MPL[a])
NBC (Next Byte Codes)	Ensamblador	Original[b]	Sí	Win/Mac Linux	Sí (MPL)
NXC (Not eXactly C) Es el más usado a nivel mundial y el más documentado	C/C++ (Custom)	Original	Sí	Win/Mac Linux	Sí (MPL)
RobotC	C	Original	Sí	Win/Mac	No
NI Labview Toolkit	Gráfico	Original	No (Usa NI Labview)	Win/Mac	No
leJOS NXJ	Java	Custom	No (Se puede usar Eclipse o Netbeans)	Cualquiera[c]	Sí (MPL)
pbLua	Lua	Custom	No[d]	Win/Mac Linux/BSD	Freeware
NXT-OSEK	ANSI C/C++	Custom	No (Usa Eclipse)	Win/Mac Linux	Sí (MPL)

Tabla 2.3: Resumen de Lenguajes para programación del NXT[31]

[a] Mozilla Public License (ver http://www.mozilla.org/MPL/MPL-1.1.html)
[b] Puede ser actualizado a un firmware mejorado
[c] Requiere la Java Virtual Machine
[d] Los programas se compilan en el mismo NXT

2. Elementos de la Plataforma de Prácticas para Control

3
Estrategia de Diseño Propuesta

La realización de cualquier esquema de prácticas de control implica el desarrollo de una metodología completa que permita de forma concreta, la realización del diseño inicial del reto de control, conocer y modelar adecuadamente la planta construida, calcular e implementar las estrategias de control apropiadas, y finalmente, probar y validar el trabajo realizado con ensayos apropiados. En el marco del presente trabajo, se propone la puesta en marcha de un conjunto de plantas basadas en los retos de control, lo que hace necesario poner los anteriores postulados dentro de una metodología completa de desarrollo para cada una de las plantas que se proponga construir. Dichos pasos se resumen a continuación [30]:

1. *Implementación de la planta.* Se realiza la integración de los componentes constitutivos (hidráulicos, mecánicos, electrónicos, etc) del sistema a ser controlado, sensores y actuadores.

2. *Modelamiento y/o Identificación.* En este paso se obtiene un modelo lineal continuo o discreto del proceso a controlar.

3. *Diseño preliminar.* Con el modelo obtenido y especificaciones dadas (en el tiempo o en la frecuencia) para el sistema de lazo cerrado, se hace un diseño preliminar del controlador en el dominio continuo o discreto [6].

4. *Discretización.* Si el controlador se diseñó en el dominio análogo se usa genéricamente la transformación bilineal. El objetivo unico de este paso es seleccionar cuidadósamente el tiempo de muestreo T, ponderando el tiempo de cálculo del procesador con la pérdida de margen de fase debida al retardo indroducido por el controlador digital.

5. *Implementación digital en un lenguage de alto nivel.* Normalmente se usa C y un sistema operativo de tiempo real (RTOS) [5].

6. *Pruebas de Hardware.* En esta parte se realizan directamente las pruebas del controlador diseñado para todas las condiciones de operación previsibles del sistema. Las pruebas realizadas

determinaran un ciclo de iteraciones en la síntesis del controlador desde el paso 2. Eventualmente, si existe mucha divergencia entre las predicciones del paso 3 y los resultados experimentales, será necesario rehacer el modelo del sistema desde el paso 1.

El proceso comienza con la selección del reto de control y su respectivo modelo matemático genérico. Esto puede corresponder a la revisión de la literatura existente o a la obtención de un modelo específico de acuerdo a las necesidades o modificaciones que impliquen las características de la plataforma de desarrollo. La siguiente fase corresponde al diseño mecánico de la planta o sistema a controlar. En esta parte, se plantea el reto de control, la disposición física de los componentes y se eligen los actuadores y sensores más apropiados para la tarea propuesta. Dada la flexibilidad de la plataforma Lego, esta parte se puede llevar a cabo de forma recursiva y a la par de otras partes del proceso, a medida que este avanza, es decir, se desarrolla una estrategia de diseño iterativo, lo que va de la mano con un modelo matemático de los sistemas construidos.

El siguiente paso es la identificación del sistema. El modelo analítico previo contiene parámetros que por lo general son desconocidos, y que pueden ser estimados mediante este tipo de procedimientos de identificación. Otra motivación para llevar a cabo un procedimiento de esta índole es la necesidad de contar con modelos precisos con los que sea posible diseñar estrategias de control más exactas. El modelado e identificación son tareas que en general deben ir de la mano, dado que la primera indica el orden del sistema y los parámetros que deben ser tenidos en cuenta permitiendo la obtención de modelos adecuados. Un buen modelo garantiza que el siguiente paso, el diseño e implementación del controlador, se lleve a cabo de forma más confiable.

El diseño de la estrategia de control es la parte fundamental del proceso. En esta etapa se ponen a prueba los conocimientos adquiridos en sesiones teóricas y se otorga funcionalidad al prototipo de acuerdo a ciertos parámetros o requerimientos de funcionamiento. La implementación del controlador es un problema adicional que involucra conocimientos sobre la plataforma usada para poner en funcionamiento la estrategia diseñada. En el caso discreto, implica la utilización de conceptos de programación sobre dispositivos digitales, lo que enriquece el ejercicio.

Por último, se validan los resultados mediante un conjunto de pruebas que dependen del tipo de planta y los requerimientos iniciales de diseño. Estas pruebas sirven para evaluar y determinar si los modelos y controladores obtenidos se ajustan al desempeño deseado. Es fácil notar que esta estrategia se puede aplicar al diseño de cualquier máquina, prototipo o proceso. La capacidad de llevarlas a cabo durante el periodo de entrenamiento del ingeniero, de manera sencilla y económica, le otorga a este tipo de iniciativas un gran valor agregado. La figura 3.1 resume el proceso descrito.

El propósito básico de la enseñanza de control experimental es que un estudiante recorra los pasos metodológicos de síntesis de un controlador de manera ágil y sin tener que abordar problemas secundarios (calibración de sensores, diseño de acoples eléctricos o mecánicos, etc) que desvían la atención del problema central. La componente experimental de un curso de control debe perseguir dos objetivos claros[40]:

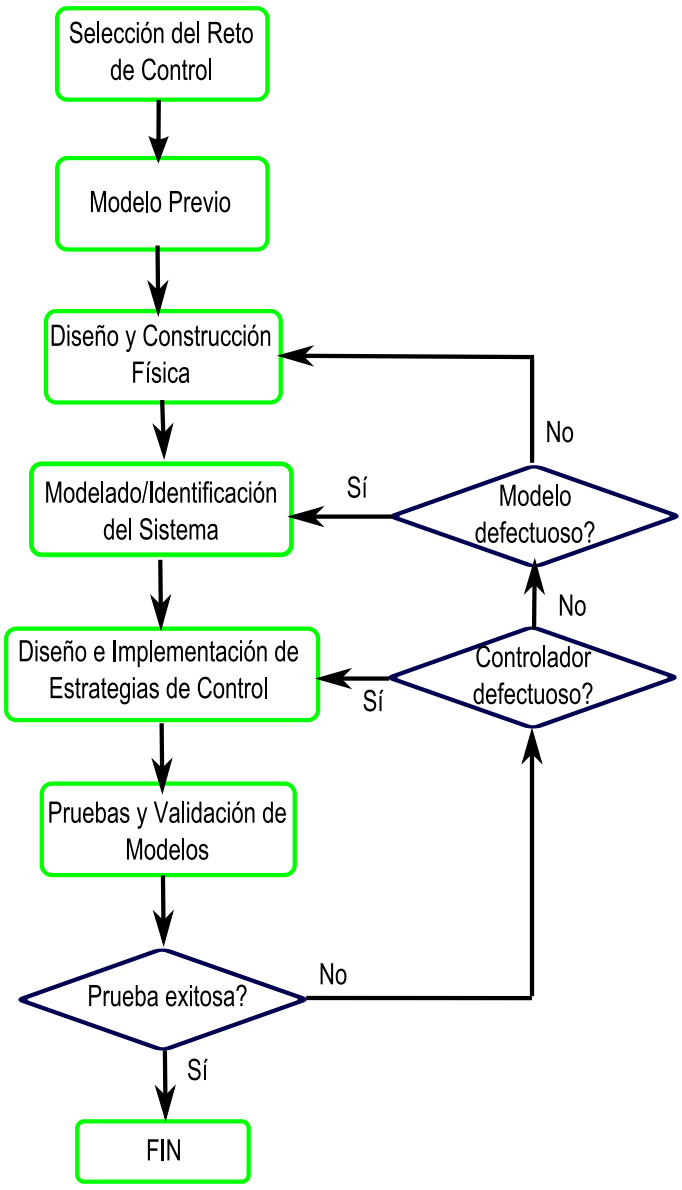


Figura 3.1: Esquema Propuesto

- Aplicar sobre un mismo sistema varias estrategias de control para establecer comparativamente límites y ventajas de cada una.

- Realizar controladores para sistemas de diferente dinámica y dominio físico. Al menos se deben incluir experimentos sobre sistemas con polos estables, resonantes, en el origen (integradores), en el semiplano derecho, retardos y/o ceros de fase no mínima.

3. Estrategia de Diseño Propuesta

Si bien la estrategia aquí mostrada recorre pasos de diseño de sistema adicionales, la finalidad del proyecto es facilitar al estudiante la realización, diseño e implementación de estrategias de control. Esto contribuye al aprendizaje y desmitifica el carácter poco práctico del control automático dentro de las aulas de clase.

4
Estado del Arte

Alrededor del mundo se han desarrollado aplicaciones educativas en ingeniería y educación, basadas en el Lego Mindstorms. Éstas son en términos generales, la propuesta de construcción de una planta con las piezas de Lego sobre la cual se implementan las estrategias de control en diferentes pruebas, o el desarrollo de software que permita conectar o controlar cualquier dispositivo construido con Lego Mindstorms con el PC, para aplicaciones educativas. A continuación se describen algunos de dichos proyectos.

4.1 Proyectos Existentes

4.1.1 LEGO MINDSTORMS - based mobile robots team

Se trata de una tesis doctoral en el área de Informática realizada por Daniele Benedettelli en la Universitá degli Studi di Siena, sobre robotica en grupo cooperativa. Ha sido desarrollada usando la version RCX de Mindstorms. Inicialmente, no tiene una relación con el propósito educativo de este proyecto, sino que se encuentra enfocada en la aplicación de la informatica al campo de la robotica. No obstante, el autor es uno de los principales desarrolladores de Lego Mindstorms en el mundo (es el creador del NXC) y sus aportes han sido fundamentales dentro de muchos proyectos educativos hechos con Lego. El informe original está disponible (en italiano) gratuitamente en su web personal, http://daniele.benedettelli.com.

4.1.2 APRIL - A PID Robot Implemented with LEGO

Es un proyecto de Kevin McLeod llevado a cabo en la University of British Columbia, sobre la implementación de un sistema de control con un regulador PID (Proporcional-Integral-Derivativo) en un Mindstorms NXT. Es básicamente un vehículo equipado con sensores ultrasónicos en la parte anterior, que le permiten leer la distancia a un obstáculo y mantenerse a una distancia fija del mismo. Está realizado en lenguaje NXC. El objetivo es modificar las constantes K_p, K_i y K_d del controlador,

y evaluar la capacidad del sistema para mantener la distancia al obstáculo. Tengamos en cuenta que la ley de control para un controlador PID es de la forma:

$$U(t) = U_p(t) + U_i(t) + U_d(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{d}{dt}e(t) \tag{4.1}$$

donde $e(t)$ es la señal de error, la diferencia entre la salida del sistema y la referencia deseada. El vehículo es el mostrado en la figura 4.1:

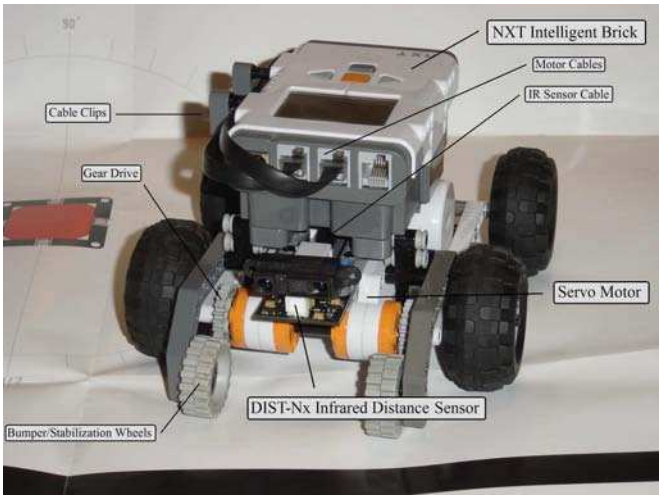


Figura 4.1: El vehículo April (Tomado de [10])

El informe se encuentra disponible para descarga gratuita en la página web personal del autor, http://www.physics.ubc.ca/~kevinmcl/projects/lego/APRIL/.

4.1.3 Matlab meets Mindstorms

Es el proyecto del Instituto de Imagen y Visión por Ordenador, RWTH Aachen, en Alemania, sobre el uso de Matlab en el control directo del NXT. Permite enlazar aplicaciones realizadas en Matlab con el sistema operativo del Brick. Es quizá uno de los proyectos más importantes sobre la materia, incluyendo cursos con asistencia de más de 300 alumnos y el uso de 100 robots. Es responsable también del toolbox de comunicación vía bluetooth para Matlab RWTH - Mindstorms NXT Toolbox, disponible gratuitamente en la web del proyecto[1].

[1] http://www.lfb.rwth-aachen.de/en/education/ws07/mindstorms.html

4.1.4 Estudio de las posibilidades didácticas en ingeniería de control del LEGO Mindstorms NXT

Es un estudio llevado a cabo en la Universidad Politécnica de Cartagena, España, por parte de Guillermo Nieves. El objetivo del estudio era recabar toda la información posible, estudiar y evaluar las capacidades de Lego Mindstorms para su uso docente en ingeniería de control [31]. En dicho trabajo, se pueden encontrar pruebas de funcionamiento de gran parte de los sensores aquí descritos, descripción de trabajos similares (principalmente en robótica con Lego) y aplicaciones del producto en ejercicios de control.

El trabajo muestra el marcado interés que existe al interior de las comunidades de aficionados a la robótica alrededor del mundo por utilizar el Lego Mindstorms en sus creaciones. La presencia de acutadores, sensores, procesador y muchas piezas móviles lo convierten en una herramienta ideal para la construcción de robots. Si bien, recalca que el trabajo serio en ingeniería de control es limitado, afirma que se trata una herramienta completa, madura y muy versátil, de una constitución física sólida y resistente (avalada por la gran calidad característica de LEGO), y capaz de comunicarse inalámbricamente con un computador, registrar datos, controlar sistemas, y del que se sabe absolutamente todo porque el propio fabricante se ha encargado de ello. La impresión no podría ser mejor.

4.1.5 Lego en la Universidad Nacional de Colombia

Es el proyecto base para esta propuesta. Se trata de dos prácticas en las que se hace una introducción al control digital y PID, y más adelante un par ejercicios más profundo de caracterización de sistemas e implementación de diversas estrategias de control de mayor complejidad. En cuanto a los ejercicos introductorios, se decidió controlar dos variables de interés del sistema, posición angular para el motor en funcionamiento normal, y velocidad angular para el motor en una aplicación específica, un vehículo sencillo.

Servomotor

Para introducir al estudiante a las acciones básicas de control (PID y ON-OFF), se propone controlar la posición/velocidad angular de un motor eléctrico y analizar el efecto de tales acciones sobre la dinámica y el comportamiento del sistema. Vale la pena recordar que la ley de control ON-OFF es de la forma:

$$U(t) = \begin{cases} u_{max} & \text{si } e(t) > 0 \\ u_{min} & \text{si } e(t) < 0 \end{cases} \tag{4.2}$$

Esta práctica no requiere un conocimiento detallado del sistema, ya que su propósito es el de otorgar un carácter cualitativo a estas estrategias de control, por lo que se trata de un proceso más de

sintonización que de cálculo de controladores. No obstante, es obtenido un modelo matemático simple que servirá de guía cualitativa para el resto del procedimiento. El sistema propuesto se muestra en la figura 4.2.

Figura 4.2: Servomotor Propuesto

Dada su sencillez, la plataforma escogida para la realización de las prácticas introductorias fue la NXC. El programa desarrollado para la práctica contiene las operaciones, corrimiento de bits y escalamientos de variable necesarios, para que el estudiante únicamente tenga que modificar las constantes K_p, K_i y K_d del controlador (similar al APRIL), y solo tenga que observar el efecto de cada constante sobre la dinámica, funcionamiento y desempeño del sistema. La implementación del control ON-OFF es directa, dada la sencillez de la ecuación que lo define.

Si bien el lenguaje NXC permite llevar a cabo tareas de una complejidad alta, como se puede observar más adelante en el desarrollo del vehículo autónomo, requiere de operaciones adicionales (escalamiento de variables, corrimiento de bits) que tendrían que ser implementadas por el usuario. Esto puede llevar a pérdida de datos por efecto de la aproximación y cuantificación de datos. Para los ejercicios más profundos se utilizó NXT-Osek, que permite trabajar con punto flotante, por lo que la implementación de los controladores es más directa.

Vehículo Simple

En un motor eléctrico común, las variables de mayor interés para un desarrollo en control en el campo industrial son la velocidad y la posición angular. Para aplicar la tarea de control a la variable de velocidad, se prefirió modificar la planta y convertirla en un vehículo como el mostrado en la figura 4.4.

Figura 4.3: Vehículo para control de Velocidad

El sistema NXC no cuenta con una función que permita la medición exacta de la velocidad angular, por lo que es necesario llevar a cabo un proceso de estimación de la misma a partir de medidas de posición, haciendo el uso de la relación:

$$\omega(t) = \frac{d}{dt}\theta(t) \tag{4.3}$$

la cual se puede aproximar a la función:

$$\omega(t) \approx \frac{\theta_{act} - \theta_{ant}}{T_s} \tag{4.4}$$

con θ_{act} como la posición angular actual y θ_{ant} la posición en un tiempo de muestreo anterior. Esta función permite estimar la velocidad del sistema. No obstante, puede llevar a ligeros errores numéricos, por lo que es de esperar que los sistemas tiendan a tener una ligera oscilación en estado estable. La salida será aproximadamente grados por segundo, multiplicado por factor de escala que dependerá del tiempo de muestreo.

Identificación y Control de Motor DC

Para diseñar controladores más avanzados que un PID, es necesario obtener un buen modelo de la planta. El método ARX de identificación de sistemas puede ser utilizado. Los fundamentos para usar esta herramienta son los conceptos de control digital y la aproximación de mínimos cuadrados de un sistema de ecuaciones lineales sobredeterminado [1]. Con estos conceptos se puede abordar la

identificación de sistemas para obtener un modelo discreto de buena calidad $G(z)$.

La entrada $u(t)$ de la planta es el porcentaje de voltaje aplicado ($u(t) \in [0, 100]$) y la salida $y(t)$ es la velocidad del robot, estimada mediante la posición detectada por el encoder del motor. Es parte de este trabajo la programación de rutinas que entregan una señal binaria pseudoaleatoria o una onda "chirp" al robot, detectan la salida y envian los datos a un computador vía USB o Bluetooth, para su posterior procesamiento.

Una vez identificado el sistema, se procede a diseñar los controladores e implementarlos en ecuaciones de diferencias. Para el diseño se puede utilizar un software especializado como Matlab, Octave, o Scilab. Un ejemplo de controlador puede ser:

$$C(s) = \frac{1{,}1224(s + 0{,}4294)(s + 7{,}117)}{s(s + 2{,}591)} \tag{4.5}$$

Dado que la plataforma de implementación es digital, es recomendable utilizar la transformación bilineal para convertir el controlador a una expresión discreta. En el ejemplo se obtuvo:

$$C(z) = \frac{1{,}1224(z - 0{,}9914)(z - 0{,}8609)}{(z - 1)(z - 0{,}9495)} \tag{4.6}$$

Cuando se cuenta con la expresión para el controlador digital, se expresa en forma de ecuaciónes de diferencia (variables de estado) y de esta forma poderlo escribir como ecuación en un lenguaje de alto nivel. Para el ejemplo anterior, dichas ecuaciones son:

$$\begin{aligned} x_1[k+1] &= x_1[k] + 1{,}4\, e[k] \\ x_2[k+1] &= 0{,}9495\, x_2[k] - 0{,}2874\, e[k] \\ u[k+1] &= 0{,}001891\, x_1[k] - 0{,}2874\, x_2[k] + 1{,}122\, e[k] \end{aligned} \tag{4.7}$$

Para la implementación del controlador se usa un lenguaje de alto nivel (GCC). Si se desea evitar errores de cuantificación de coeficientes es mejor que la realización digital del controlador sea hecha en punto flotante. Para ello se usa el sistema operativo de tiempo real NXT-OSEK que incluye tipos de datos float de 32 bits. Para el ejemplo, la versión implementada es:

Algoritmo 1: Controlador Implementado en C

```
#include "kernel.h"
#include "kernel_id.h"
#include "ecrobot_interface.h"
#include "math.h"

DeclareCounter(SysTimerCnt);
DeclareTask(Task1);

void user_1ms_isr_type2(void)
{
StatusType ercd;
ercd = SignalCounter(SysTimerCnt);
if (ercd != E_OK)
{
ShutdownOS(ercd);
}
}

int y=0;
int ref=90;
double e,x1,x2,x1ant,x2ant,u;

TASK(Task1)
{
y=nxt_motor_get_count(NXT_PORT_A);
e=(double)ref-(double)y;
x1=x1ant + 1.4*e;
x2=0.9495*x2ant - 0.2874*e;
u=0.0019*x1ant - 0.2874*x2ant + 1.122*e;
nxt_motor_set_speed(NXT_PORT_A,(int)u,0);
x1ant = x1;
x2ant = x2;
TerminateTask();
}
```

Tal como se observa, el código es muy sencillo y es equivalente para cualquier controlador implementado con esta herramienta. El estudiante puede modificar facilmente las lineas que definen el controlador y el tiempo de muestreo para probar muchos otros controladores. Esto toma apenas unos minutos, no implica un conocimiento profundo de programación y deja claros los problemas y las

herramientas necesarias que se deben tener en cuenta al enfrentar un problema de control.

Vehículo Tele-Controlado

La planta propuesta corresponde a un vehículo robótico tele-controlado o auto-guiado de cuatro llantas en dos dimensiones. Este consiste en un carro robot, con dos motores, uno de dirección y otro de tracción. El propósito es conseguir que el móvil sea capaz de moverse a través de un espacio dado usando una serie de sensores, y donde el usuario deberá encontrar los controladores de posición y/o velocidad necesarios para cumplir con el objetivo mencionado. Sobre este sistema es posible implementar controladores proporcionales, PID y de dos parámetros entre otros. La motivación de este reto es la posibilidad de enfrentar al usuario a un problema muy usual en robótica móvil, que es el posicionamiento en el espacio para tareas específicas, como los vehículos auto-guiados (AGV) existentes en la industria.

Inicialmente, se construyó el modelo mostrado en la figura 4.4, el cual pasa por un proceso de obtención de un modelo aproximado, aplicando las técnicas recursivas de estimación de modelos (identificación) contenidas en [1], Cáps. 11 y 15. La parte de tele-control fue realizada utilizando las estrategias de control implementadas en lenguaje Bricx y activadas desde un PC usando Matlab. Fue elaborado como proyecto final en un curso de posgrado. El proceso detallado de identificación y control se encuentra en [32]

Figura 4.4: Vehículo Auto-Guiado

Parte II

Plataforma Desarrollada

5
Vehículo Autónomo

La planta propuesta corresponde a un vehículo robótico auto-guiado, de cuatro llantas en dos dimensiones. Este consiste en un robot en forma de carro, con dos motores, uno de dirección y otro de tracción. La idea es conseguir que el móvil sea capaz de moverse a través de un espacio dado usando una serie de sensores, y donde el usuario deberá encontrar los controladores de posición y/o velocidad necesarios para cumplir con el objetivo mencionado. La motivación de este reto es la posibilidad de enfrentar al usuario a un problema muy usual en robótica móvil, que es el posicionamiento en el espacio para tareas específicas, como los vehículos auto-guiados (AGV) existentes en la industria.

Un sistema AGV (Auto Guided Vehicle), es un vehículo que transita de manera autónoma, sin la necesidad de un conductor. Son sistemas concebidos para el transporte de materiales o supervisión de eventos o procesos, usualmente repetitivos, que pueden resultar peligrosos, tediosos o difíciles de realizar para un operador humano. Su popularidad dentro de la industria es creciente, dada su capacidad de realizar tareas de transporte de forma repetitiva y casi sin interrupciones. Es un sistema muy interesante de controlar y que se puede usar dentro de contextos científicos o industriales, de ahí su inclusión en el proyecto.

Para el análisis dinámico del sistema, se ha decidido dividirlo en dos parte, la dinámica longitudinal y la lateral. Se tratará cada sistema por aparte, para luego pasar a la obtención de los parámetros físicos de cada uno de los modelos y sus respectivos experimentos de control.

5.1 Modelo Construido

El modelo construido corresponde a una versión del vehículo creado por Laurens Valk [**?**]. Se trata de un carro construido con tres motores, dos para tracción y uno para dirección. La estructura sostiene el Brick, dando la forma característica de un vehículo de cuatro llantas. Los motores de tracción están sincronizados para garantizar una velocidad uniforme del móvil y hacer el movimiento semejante al de los carros comunes. La dirección cuenta con un sistema de cremallera para mover el eje de las llantas. El vehículo puede verse en la figura 5.1.

Figura 5.1: Sistema Construido: Vehículo Auto-Guiado

5.2 Dinámica Longitudinal

5.2.1 Modelo del sistema

La dinámica longitudinal del vehículo corresponde al comportamiento del sistema en un movimiento sobre un plano, visto desde un lado del vehículo, paralelo a la dirección de movimiento. Se asume que los ejes de la llantas son paralelos y están en un plano paralelo al piso. El eje x se encuentra en dicho plano y es perpendicular a los ejes. Si el plano de movimiento tiene pendiente, el eje z no se considera en la dirección de la gravedad, sino perpendicular al plano de movimiento. El esquema puede verse en la figura 5.2.

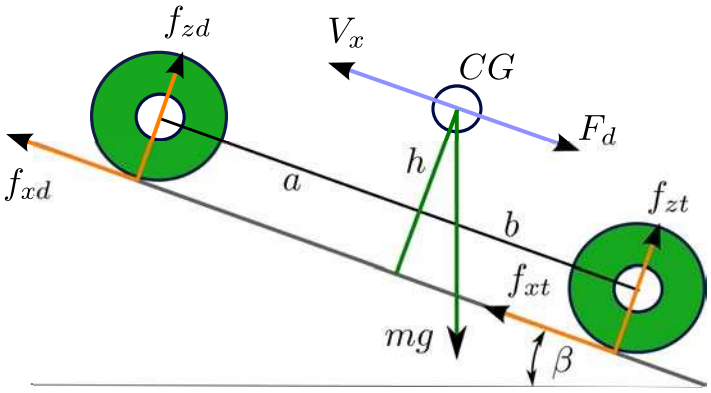


Figura 5.2: Esquema longitudinal del Vehículo[a]

[a] Adaptada de [39]

g	$[m/s^2]$	Aceleración de la gravedad
β	[rad]	Inclinación del plano de movimiento
m	[Kg]	Masa del vehículo
h	[m]	Altura del centro de gravedad del vehículo
a, b	[m]	Distancia del eje frontal y trasero al centro de gravedad
B	$[Ns/m]$	Coeficiente de rozamiento con el piso
V_x	$[m/s]$	Velocidad longitudinal del vehículo
f_{xd}, f_{xt}	[N]	Fuerzas longitudinales ejercidas sobre el vehículo
f_{zd}, f_{zt}	[N]	Fuerzas verticales ejercidas sobre el vehículo
C_d	$[Ns^2/Kgm]$	Coeficiente de arrastre aerodinámico
ρ	$[Kg/m^3]$	Densidad del aire
F_d	[N]	Fuerza de arrastre aerodinámico

Tabla 5.1: Parámetros físicos y variables de la dinámica longitudinal

En la tabla 5.1 se definen las variables y constantes del modelo.

5.2.2 Ecuaciones de movimiento

Para empezar, se puede tomar la sumatoria de fuerzas en la dirección x:

$$m\dot{V}_x = F_x + F_d - mg \cdot \sin\beta \tag{5.1}$$

Donde:

$$F_x = F_{xd} + F_{xt} \tag{5.2}$$

$$F_d = -\frac{1}{2} C_d \rho A {V_x}^2 \cdot \mathrm{sgn}(V_x) \tag{5.3}$$

Las fuerzas de contacto sobre el piso en la dirección z se definen como:

$$F_{zd} = \frac{+h(F_d - mg\sin\beta - m\dot{V}_x) + b \cdot mg\cos\beta}{a+b} \tag{5.4}$$

$$F_{zt} = \frac{-h(F_d - mg\sin\beta - m\dot{V}_x) + a \cdot mg\cos\beta}{a+b} \tag{5.5}$$

Nótese que por sumatoria de fuerzas en la dirección z se obtiene $F_{zd} + F_{zt} = mg \cdot \cos\beta$.

5.2.3 Simplificación del modelo

El modelo presentado tiene coeficientes que en la práctica son difíciles de medir, como los coeficientes de arrastre, razón por la cual se decidió simplificar el modelo. Si se asume que el ángulo de inclinación es cercano a cero, y que dada la baja velocidad que desarrolla el modelo construido, podemos considerar la fuerza de arrastre muy pequeña, la ecuación de movimiento en la dirección x es:

$$m\dot{V}_x = F_x \tag{5.6}$$

Donde F_x podemos definirla como $R\tau - BV(x)$, con τ como el torque producido por los motores y R, el radio de las llantas. Si asumimos además que dicho torque es proporcional al voltaje e del motor (se toma la dinámica del motor mucho más rápida que la del resto del sistema), podemos escribir $\tau = K_\tau V$. Aplicando transformada de Laplace a las anteriores relaciones obtenemos la función de transferencia:

$$G_l(s) = \frac{V_x(s)}{e(s)} = \frac{RK_\tau}{ms + B} \tag{5.7}$$

Con entrada el voltaje y salida la velocidad. Dados los resultados obtenidos en [32], se decidió identificar el sistema para obtener los parámetros del modelo, como se muestra en la siguiente sección.

5.2.4 Identificación del Sistema

El primer paso en la identificación del sistema es la generación de una señal binaria aleatoria[1] (RBS), que active al sistema y permita observar la mayor cantidad de características temporales (tiempo de establecimiento, subida, sobrepicos) y frecuenciales (oscilaciones, atenuación por frecuencia) de la dinámica del sistema. Dicha señal es aplicada al sistema y se miden los cambios en la variable de salida, velocidad para este caso. La señal seleccionada y su correspondiente respuesta del sistema son las mostrada en la figura 5.3.

Con los datos obtenidos, se procede a aplicar métodos de aproximación de mínimos cuadrados para la resolución de un sistema de ecuaciones sobredeterminado[40], que ajusten ciertas estructuras de modelo a la dinámica mostrada por las mediciones. Tales estructuras son escogidas de acuerdo a su similitud con el modelo teórico obtenido en la sección enterior. Las estructuras seleccionada y su nivel de ajuste a los datos reales se muestran en la figura 5.4.

Se puede ver que el nivel de ajuste a los datos es alto para todas las estructuras mostradas, por que se decide escoger la de más bajo orden y que mejor se acomode al modelo. Se trata de un modelo de proceso (P1 en la figura 5.4) de la forma:

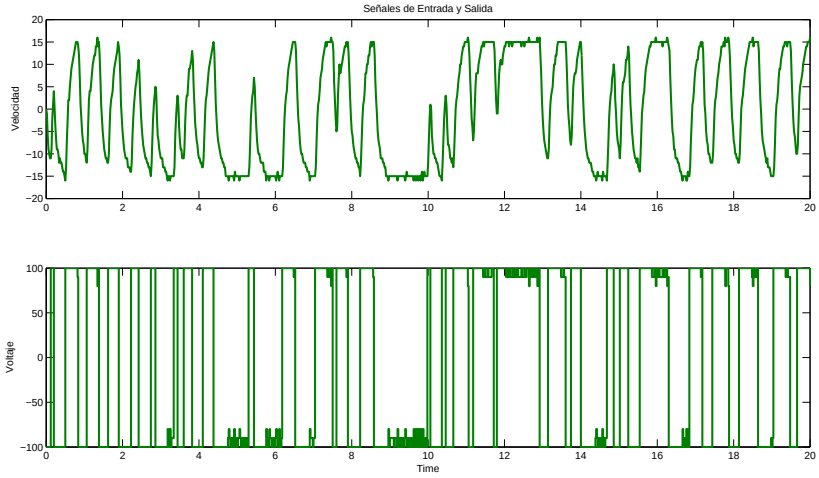


Figura 5.3: Identificación de la dinámica longitudinal

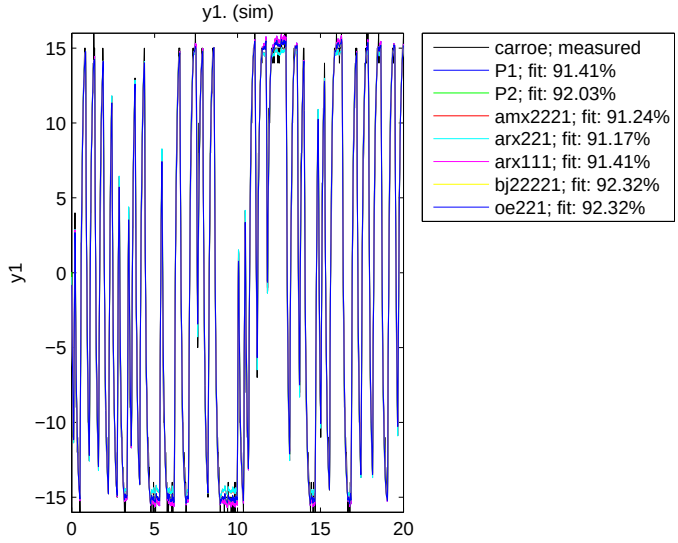


Figura 5.4: Ajuste a los datos de las diferentes estructuras

$$G(s) = \frac{K}{1 + T_p s} \tag{5.8}$$

5.2.5 Modelo numérico del Sistema

Al reemplazar los valores del sistema ajustado en la ecuación anterior, se llega a la siguiente función de transferencia:

$$G_{num}(s) = \frac{0{,}16641}{1 + 0{,}11411s} \tag{5.9}$$

Los parámetros numéricos del sistema se muestran en la tabla 5.2. Nótese que estos parámetros pueden no corresponder a variables físicas ya que son producto de un ajuste de regresión lineal.

K	RK_τ	0.16641
T_p	m	0.11411
	B	1

Tabla 5.2: Parámetros del sistema

5.2.6 Experimentos de Control

Con el modelo obtenido en la sección anterior, se procede a diseñar dos controladores, un controlador proporcional y un control PI (Proporcional Integral). Para el controlador proporcional se escogió una constante $K_p = 6{,}9$. La respuesta del sistema simulado y los datos reales obtenidos al probar el controlador en el sistema se muestran en la figura 5.5.

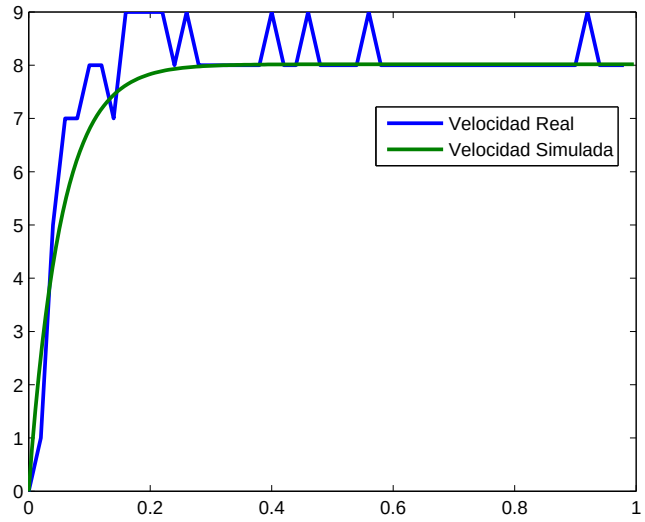


Figura 5.5: Respuesta del controlador Proporcional

La respuesta muestra en ambos casos un nivel de estado esacionario similar, con un error de posición cercano al 47 % (ref=15). Se presenta una discrepancia en el sobrepico presentado por los datos reales (12.4 %). No obstante, la similitud de las respuestas es notoria, lo que sirve como validación para el modelo obtenido. Para el controlador PI, se diseñó una función de transferencia dada por:

$$C_{PI} = \frac{5{,}9285(s + 10{,}77)}{s} \tag{5.10}$$

La respuesta del sistema con el mencionado controlador se muestra en el gráfico 5.6.

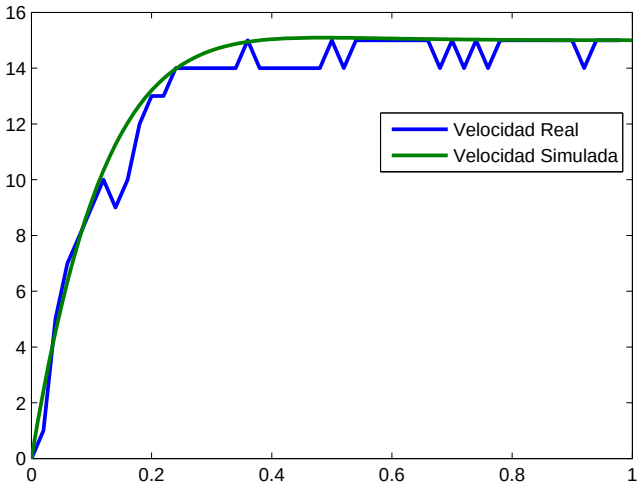


Figura 5.6: Respuesta del controlador PI

Se observa la dinámica presentada en ambos casos es prácticamente la misma (similar tiempo de establecmiento, cero error de estado estacionario). Este resultado confirma la validez del modelo obtenido, permitiendo obtener controladores que garanticen una velocidad lineal constante del móvil.

5.3 Dinámica Lateral

5.3.1 Modelo del Sistema

La dinámica lateral considera el movimiento del vehículo sobre el mismo plano del análisis mostrado en las secciones previas, pero visto de forma paralela a dicho plano. En este caso, se asume que el sistema se mueve con velocidad lineal constante U. En el estudio mostrado en la presente sección, se tiene en cuenta la dirección del carro sobre el plano (no se consideró anteriormente) y

su efecto sobre el movimiento del mismo. Se considera el sistema como de tres grados de libertad (DOF), y se usará la notación estándar de la SAE (Society of Automotive Engineers). Se trabajará con el sistema coordenado y las convenciones usadas en la figura 5.7 (Adaptada de [8]).

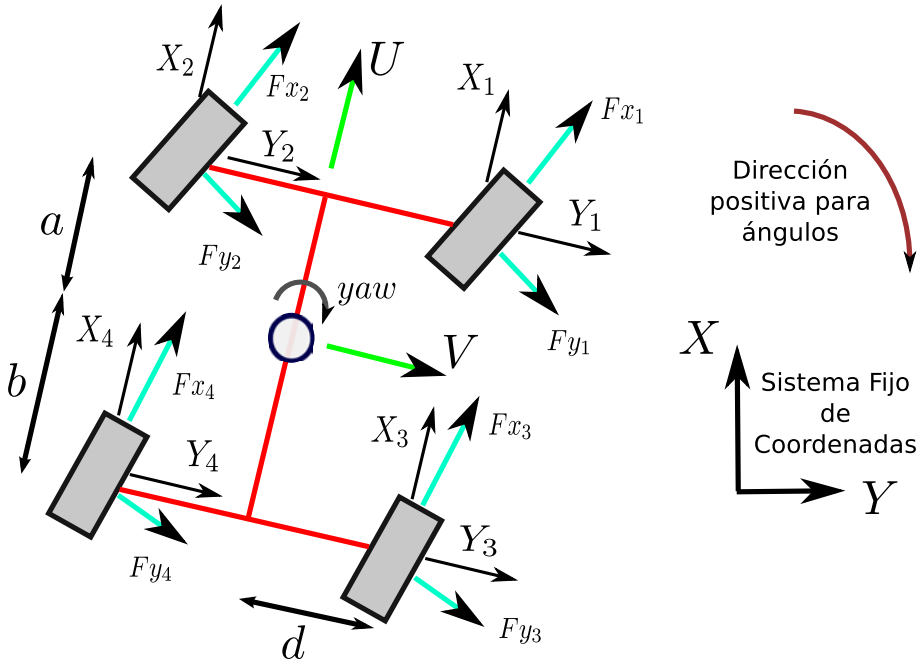


Figura 5.7: Esquema del Vehículo

Los parámetros y variables usados para deducir el modelo del sistema se pueden ver en la tabla 5.3.

a, b	[m]	Distancia del eje frontal y trasero al centro de gravedad
d	[m]	Distancia de la línea central del carro a cada llanta
X, Y		Coordenadas terrestres
U	$[m/s]$	Velocidad lineal del vehículo
V	$[m/s]$	Velocidad lateral del vehículo
m	[Kg]	Masa del Vehículo
δ_i	[rad]	Ángulo de entrada para la dirección (deseado)
ψ	[rad]	Ángulo de derrape (yaw) del carro (dirección real)
I_z	$[Kgm^2]$	Momento de inercia del carro con respecto a z
$C_{\alpha f}, C_{\alpha f}$		Coeficientes de rigidez de las llantas
$T_{nom,F,R}$	$[Nm]$	Torque nominal en las llantas
ΔT_i	$[Nm]$	Entrada de torque controlada

Tabla 5.3: Parámetros físicos y variables de la dinámica lateral

5.3.2 Dinámica en las Llantas

Una parte importante en el modelado de vehículos es el análisis dinámico de las llantas y su efecto sobre la totalidad del sistema. El número de modelos disponibles es considerable, en su mayoria de carácter no lineal. Sin embargo, para simplificar el modelo, se decide mostrar un análisis dinámico lineal, el modelo de Dugoff[8]. Hoy día, la identificación de la dinámica de las llantas recurre a técnica más elaboradas, como las redes neuronales, pero que escapan al alcance de este trabajo.

El modelo de Dugoff es una de las teoría más simples de modelado dinámico de llantas. Plantea que la fuerza ejercida por una llanta es proporcional al ángulo de deslizamiento de la misma, que es el ángulo que existe entre el plano de la rueda y su verdadera dirección de movimiento. El concepto de ve más claro en la figura 5.8.

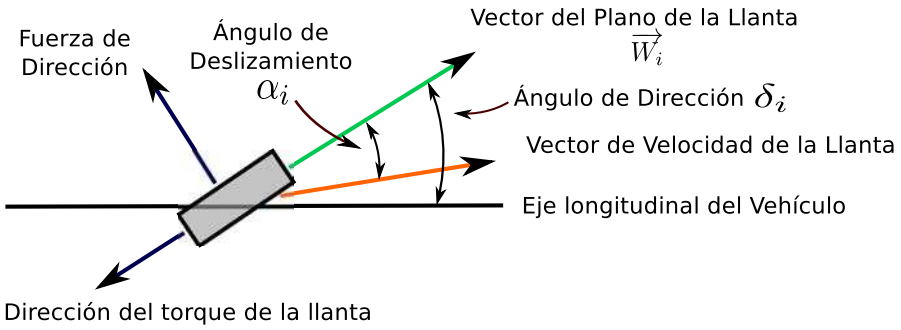


Figura 5.8: Modelo de la Llanta (Dugoff)[a]

[a] Adaptada de [8]

Por simplicidad, se asume que el torque entrante a cada llanta actua en el plano de la misma, y que las fuerzas de dirección son perpendiculares a dicho plano. Esta fuerzas se pueden escribir entonces como:

$$F_{xi} = \frac{T_i}{r_i} \tag{5.11}$$

$$F_{yi} = C_{\alpha i}\alpha_i \tag{5.12}$$

Donde α_i es el ángulo de deslizamiento de la i-ésima llanta[8]. Si se toman vectores unitarios $\hat{i}$ y $\hat{j}$ asociados a los ejes longitudinal y lateral del carro, la velocidad de cada llanta está dada por:

$$V_i = V_{CG} + \omega \times r_i \tag{5.13}$$

$$V_i = U\cdot \hat{i} + V\cdot \hat{j} + \dot{\psi}\cdot \hat{k} \times r_i \tag{5.14}$$

donde r_i es el vector radio de cada llanta, medido desde el centro de gravedad del carro. Los vectores radio para cada llanta son:

$$r_1 = a\cdot \hat{i} + d\cdot \hat{j} \qquad r_2 = a\cdot \hat{i} - d\cdot \hat{j} \tag{5.15}$$

$$r_3 = -b\cdot \hat{i} + d\cdot \hat{j} \qquad r_4 = -b\cdot \hat{i} - d\cdot \hat{j} \tag{5.16}$$

Conociendo la velocidad lateral de la i-ésima llanta V_i y en la dirección longitudinal U_i, el ángulo de deslizamiento es:

$$\alpha_i = \delta_i - \tan^{-1}\left(\frac{V_i}{U_i}\right) \tag{5.17}$$

donde V_i y U_i son los vectores de velocidad de cada llanta. Este ángulo está definido para cada llanta como [8]:

$$\alpha_1 = \delta_1 - \tan^{-1}\left(\frac{V + a\cdot \dot{\psi}}{U - d\cdot \dot{\psi}}\right) \qquad \alpha_2 = \delta_2 - \tan^{-1}\left(\frac{V + a\cdot \dot{\psi}}{U + d\cdot \dot{\psi}}\right) \tag{5.18}$$

$$\alpha_3 = \delta_3 - \tan^{-1}\left(\frac{V - b\cdot \dot{\psi}}{U - d\cdot \dot{\psi}}\right) \qquad \alpha_4 = \delta_4 - \tan^{-1}\left(\frac{V - b\cdot \dot{\psi}}{U + d\cdot \dot{\psi}}\right) \tag{5.19}$$

Para ángulos pequeños se puede realizar la siguiente simplificación:

$$\alpha_1 \approx \delta_1 - \frac{V + a\cdot \dot{\psi}}{U} \qquad \alpha_2 \approx \delta_2 - \frac{V + a\cdot \dot{\psi}}{U} \tag{5.20}$$

$$\alpha_3 \approx \delta_3 - \frac{V - b\cdot \dot{\psi}}{U} \qquad \alpha_4 \approx \delta_4 - \frac{V - b\cdot \dot{\psi}}{U} \tag{5.21}$$

5.3.3 Ecuaciones de Movimiento

Para comenzar, se usará la notación para el control del vehículo, como se define en la figura 5.9. Se puede obtener la dinámica del vehículo mediante las leyes de Newton (sumatoria de fuerzas). Dadas las fuerzas $F_x\hat{i} + F_y\hat{j}$, que pueden presentarse en la i-ésima llanta, se tiene que el momento para cada una de éstas es:

$$\vec{M_i} = \vec{r_i} \times \vec{F_i} \tag{5.22}$$

Sustituyendo:

$$\vec{M_i} = (x_i F_{yi} - y_i F_{xi}) \cdot \hat{k} \tag{5.23}$$

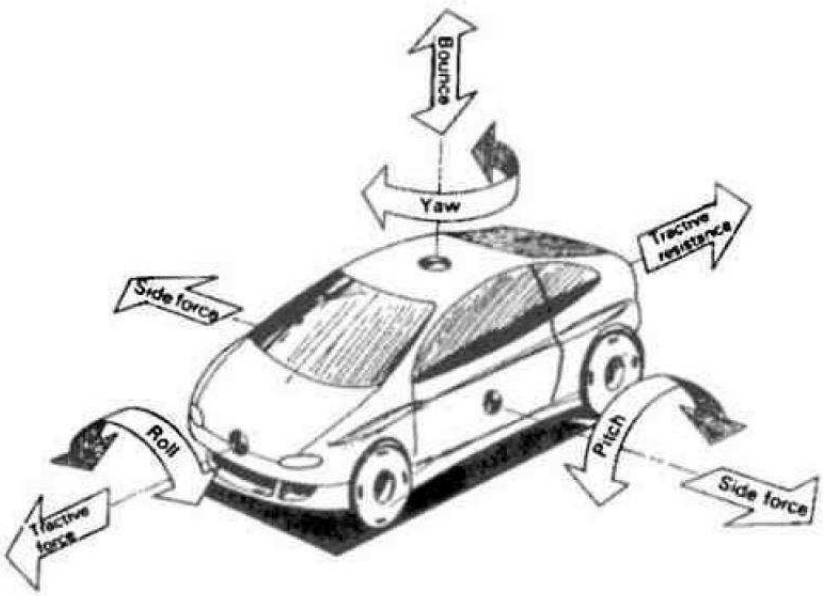


Figura 5.9: Grados de Libertad del Vehículo (Tomada de [8])

La suma de momentos para el vehículo debe ser entonces:

$$\sum \vec{M}_i = I_z \frac{d^2\psi}{dt^2} \tag{5.24}$$

Hay cierta dificultad en definir la aceleración del sistema en movimiento (indispensable para la definición de las fuerzas descritas anteriormente). Es necesario realizar una transformación de coordenadas entre un vector en un marco estacionario y uno en movimiento, de la siguente forma:

$$\vec{x}_{movimiento} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \vec{x} \qquad \vec{x} = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \vec{x}_{movimiento} \tag{5.25}$$

Donde ϕ es el ángulo entre el marco estacionario y el que se encuentra en movimiento. Para obtener los componentes de la aceleración en el sistema coordenado fijo se puede analizar como cambia la orientación del vehículo en un camino curvo (ver figura 5.10).

Para obtener la aceleración del sistema fijado al cuerpo se vale de un esquema basado en cambios diferenciales. Sean U y V los componentes de la velocidad en un tiempo t (longitudinal y lateral). Estos están referidos al sistema coordenado móvil del carro. Esto puede ser engañoso, ya que no hay movimiento del vehículo con respecto a sus propios ejes. El movimiento de los ejes mismos es relativo al sistema coordenado fijo. Para tener la aceleración, se observan los cambios diferenciales de U y V con respecto a los ejes x_1 y y_1. Siguendo tal razonamiento, el cambio en velocidad paralelo a x_1 es:

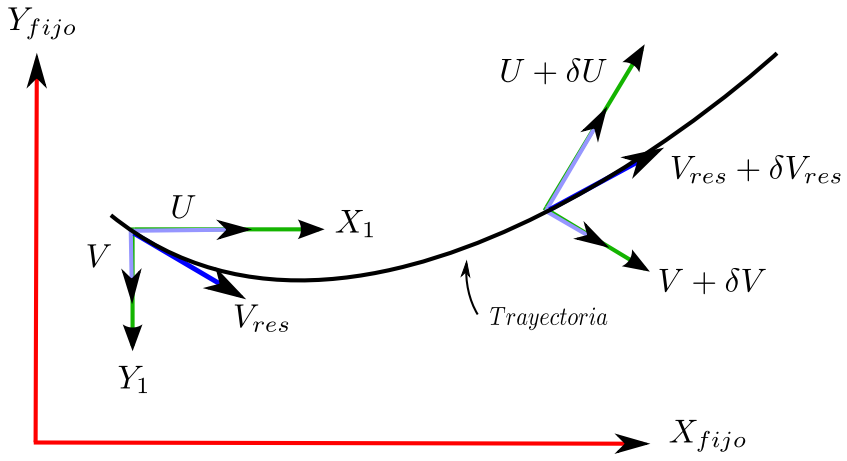


Figura 5.10: Coordenadas de Rotación y Traslación para encontrar la Aceleración[a]

[a] Adaptada de [8]

$$\delta V_{res,x_1} = (U + \delta U) \cdot \cos\delta\psi - U + (V + \delta V) \cdot \sin\delta\psi \tag{5.26}$$
$$\delta V_{res,x_1} = U\cos\delta\psi + \delta U\cos\delta\psi - U + V\sin\delta\psi + \delta V\sin\delta\psi \tag{5.27}$$

Dividiendo para δt y asumiendo que esta cantidad tiende a cero tenemos la aceleración para x:

$$a_x = \frac{dU}{dt} + V\frac{d\psi}{dt} \tag{5.28}$$
$$= \dot{U} + V\omega_z \tag{5.29}$$

La componente $(\dot{U})$ es debida al cambio de velocidad, mientras que el término restante se debe a la rotación los ejes [8]. De igual forma tenemos para la componente de aceleración en y_1:

$$a_y = \dot{V} - U\omega_z \tag{5.30}$$

Nótese que si las componentes de aceleración angular son mucho mayores que los de aceleración posicional, las ecuaciones de momento también requerirán de transformaciones. Además se asume que U es constante. Las ecuaciones de movimento se pueden escribir como:

$$\dot{\bar{x}} = A_X\bar{x} + B_X\bar{F} \tag{5.31}$$

Donde

$$\begin{aligned}
\bar{x} &= \begin{bmatrix} V & \dot{\psi} \end{bmatrix}^T \\
\bar{F} &= \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & y_1 & y_2 & y_3 & y_4 \end{bmatrix}^T \\
A_X &= \begin{bmatrix} 0 & -U \\ 0 & 0 \end{bmatrix} \\
B_X &= \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{1}{m} & \frac{1}{m} & \frac{1}{m} & \frac{1}{m} \\ -\frac{d}{I_z} & \frac{d}{I_z} & -\frac{d}{I_z} & \frac{d}{I_z} & \frac{a}{I_z} & \frac{a}{I_z} & -\frac{b}{I_z} & -\frac{b}{I_z} \end{bmatrix}
\end{aligned} \tag{5.32}$$

Es poco usual controlar las cuatro ruedas de manera independiente; a las ruedas delanteras se les aplica un ángulo δ_f y a las traseras δ_r. Esto provoca que las fuerzas laterales sean las mismas. Si asumimos las fuerzas delanteras como y_f y las traseras y_r, el modelo se reduce a:

$$\bar{F} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & y_r & y_f \end{bmatrix}^T \tag{5.33}$$

$$B_X = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{1}{m} & \frac{1}{m} \\ -\frac{d}{I_z} & \frac{d}{I_z} & -\frac{d}{I_z} & \frac{d}{I_z} & \frac{a}{I_z} & -\frac{b}{I_z} \end{bmatrix} \tag{5.34}$$

Se puede resolver las fuerzas x_i y y_i a partir de F_{xi} y F_{yi} así:

$$x_i = F_{xi} \cdot \cos\delta_i - F_{yi} \cdot \sin\delta_i \tag{5.35}$$

$$y_i = F_{xi} \cdot \sin\delta_i + F_{yi} \cdot \cos\delta_i \tag{5.36}$$

Las fuerzas en X se consideran como productoras de momentos que finalmente se cancelan. Entonces, para ángulos de dirección pequeños se aproximan a:

$$x_i \approx F_{xi} \tag{5.37}$$

$$y_i \approx F_{xi}\delta_i + F_{yi} \tag{5.38}$$

Sustituyendo por las fuerzas del modelo dinámico Dugoff:

$$x_i \approx \frac{T_i}{r_i} \tag{5.39}$$

$$y_i \approx \frac{T_i}{r_i}\delta_i + C_{\alpha i} \cdot \alpha_i \tag{5.40}$$

y aplicando la representación lineal de los ángulos de deslizamiento:

$$y_f \approx \left(\frac{T_1}{r_1} + \frac{T_2}{r_2}\right) \cdot \delta_f - \frac{2 \cdot C\alpha_f}{U} \cdot V - \frac{2 \cdot a \cdot C\alpha_f}{U} \cdot \dot{\psi} + 2 \cdot C\alpha_f \cdot \delta_f \tag{5.41}$$

$$y_r \approx \left(\frac{T_3}{r_3} + \frac{T_4}{r_4}\right) \cdot \delta_r - \frac{2 \cdot C\alpha_r}{U} \cdot V - \frac{2 \cdot b \cdot C\alpha_r}{U} \cdot \dot{\psi} + 2 \cdot C\alpha_r \cdot \delta_r \tag{5.42}$$

donde

$$C\alpha_f = \frac{C\alpha_1 + C\alpha_2}{2} \qquad C\alpha_r = \frac{C\alpha_4 + C\alpha_4}{2} \tag{5.43}$$

Es claro que las entradas de torque y fuerza delantera están acopladas. Para desacoplarla, se asume que el torque se compone de:

$$T_i = T_{nom,i} + \Delta T_i \tag{5.44}$$

El torque en cada eje queda afectado solo por el término transitorio, ya que los componentes de estado estable generan momentos iguales y opuestos:

$$x_i \approx \frac{\Delta T_i}{r_i} \tag{5.45}$$

Finalmente se asume que el torque diferencial multiplicado por la entrada de control es mucho menor que el torque de estado estable multiplicado por ésta misma. Fundamentalmente, se asume que la entrada de control de dirección no produce fuerzas laterales significativas[8]. También se asume que las llantas tienen radios aproximadamente iguales. Las fuerzas laterales son entonces:

$$y_f \approx \frac{2 \cdot T_{nom,f}}{r} \cdot \delta_f - \frac{2 \cdot C\alpha_f}{U} \cdot V - \frac{2 \cdot a \cdot C\alpha_f}{U} \cdot \dot{\psi} + 2 \cdot C\alpha_f \cdot \delta_f \tag{5.46}$$

$$y_r \approx \frac{2 \cdot T_{nom,r}}{r} \cdot \delta_r - \frac{2 \cdot C\alpha_r}{U} \cdot V - \frac{2 \cdot b \cdot C\alpha_r}{U} \cdot \dot{\psi} + 2 \cdot C\alpha_r \cdot \delta_r \tag{5.47}$$

De esta manera quedan completamente desacopladas las entradas de control y las fuerzas en las llantas son completamente lineales. El sistema es entonces:

$$F = A_F \bar{x} + B_F \bar{u} \tag{5.48}$$

Donde:

$$F = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & y_r & y_f \end{bmatrix}^T \tag{5.49}$$

$$\bar{x} = \begin{bmatrix} V & \dot{\psi} \end{bmatrix}^T \tag{5.50}$$

$$\bar{u} = \begin{bmatrix} \delta_f & \delta_r & \Delta T_1 & \Delta T_2 & \Delta T_3 & \Delta T_4 \end{bmatrix}^T \tag{5.51}$$

$$A_F = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -\frac{2 \cdot a \cdot C\alpha_f}{U} & -\frac{2 \cdot a \cdot C\alpha_f}{U} \\ -\frac{2 \cdot a \cdot C\alpha_r}{U} & \frac{2 \cdot b \cdot C\alpha_r}{U} \end{bmatrix} \tag{5.52}$$

$$B_F = \begin{bmatrix} 0 & 0 & \frac{1}{r} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{r} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{r} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{r} \\ \frac{2 \cdot T_{nom,f}}{r} + 2 \cdot C\alpha_f & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{2 \cdot T_{nom,r}}{r} + 2 \cdot C\alpha_r & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.53}$$

Se puede entonces escribir la dinámica lineal como:

$$\dot{\bar{x}} = A_X \bar{x} + B_X \bar{F} \tag{5.54}$$

$$= A_X \bar{x} + B_X (A_F \bar{x} + B_F \bar{u}) \tag{5.55}$$

$$= (A_X + B_X A_F) \cdot \bar{x} + B_X B_F \cdot \bar{u} \tag{5.56}$$

$$= A\bar{x} + B\bar{u} \tag{5.57}$$

Donde:

$$A = \begin{bmatrix} -2\frac{C_{\alpha f} + C_{\alpha r}}{m \cdot U} & -U - 2\frac{a \cdot C_{\alpha f} - b \cdot C_{\alpha r}}{m \cdot U} \\ -2\frac{a \cdot C_{\alpha f} - b \cdot C_{\alpha r}}{I_z \cdot U} & -2\frac{a^2 \cdot C_{\alpha f} + b^2 \cdot C_{\alpha r}}{I_z \cdot U} \end{bmatrix} \tag{5.58}$$

$$B = \begin{bmatrix} \frac{2}{m}\left(\frac{\cdot T_{nom,f}}{r} + C\alpha_f\right) & \frac{2}{m}\left(\frac{\cdot T_{nom,r}}{r} + C\alpha_r\right) & 0 & 0 & 0 & 0 \\ \frac{2 \cdot a}{I_z}\left(\frac{\cdot T_{nom,f}}{r} + C\alpha_f\right) & -\frac{2 \cdot b}{I_z}\left(\frac{\cdot T_{nom,r}}{r} + C\alpha_r\right) & -\frac{d}{I_z \cdot r} & \frac{d}{I_z \cdot r} & -\frac{d}{I_z \cdot r} & \frac{d}{I_z \cdot r} \end{bmatrix} \tag{5.59}$$

$$\tag{5.60}$$

Este modelo linealizado es lo que se conoce como el modelo de la bicicleta[1]. En algunos casos se requiere que el sistema coordenado quede fijo, lo que cambia un poco la descripción del sistema.

[1]Se le llama así porque es igual al modelo de una bicicleta con el movimiento restringido a un solo plano [8]

Con las relaciones anteriores y usando la expresión del control clásico $C(SI - A)^{-1}B$, se pueden obtener la funciones de transferencia del sistema. El polinomio resultante de calcular determinante de la matriz $(SI - A)$ corresponde al polinomio característico del sistema, o denominador de las funciones de transferencia. Para este caso es:

$$polos = s^2 + \left(\frac{2(C_{\alpha f} + C_{\alpha r})}{mU} + \frac{2(C_{\alpha f}a^2 + C_{\alpha r}b^2)}{I_z U}\right)s + \frac{4C_{\alpha f}C_{\alpha r}L^2}{mI_z U^2} - \frac{2}{I_z}(aC_{\alpha f} - bC_{\alpha r}) \tag{5.61}$$

Las funciones de transferencia son entonces:

$$\frac{V(s)}{\delta_f(s)} = \frac{\frac{2}{m}\left(\frac{T_{nom,f}}{r} + C_{\alpha f}\right)s + \frac{2}{mUI_z}\left(\frac{T_{nom,f}}{r} + C_{\alpha f}\right)\left[2bLC_{\alpha f} - amU^2\right]}{polos} \tag{5.62}$$

$$\frac{V(s)}{\delta_r(s)} = \frac{\frac{2}{m}\left(\frac{T_{nom,r}}{r} + C_{\alpha r}\right)s + \frac{2}{mUI_z}\left(\frac{T_{nom,r}}{r} + C_{\alpha r}\right)\left[2aLC_{\alpha r} + bmU^2\right]}{polos} \tag{5.63}$$

$$\frac{\dot{\psi}(s)}{\delta_f(s)} = \frac{1}{s}\frac{\frac{2a}{I_z}\left(\frac{T_{nom,f}}{r} + C_{\alpha f}\right)s + \frac{4C_{\alpha f}L}{mUI_z}\left(\frac{T_{nom,f}}{r} + C_{\alpha f}\right)}{polos} \tag{5.64}$$

$$\frac{\dot{\psi}(s)}{\delta_r(s)} = \frac{1}{s}\frac{-\frac{2b}{I_z}\left(\frac{T_{nom,r}}{r} + C_{\alpha r}\right)s - \frac{4C_{\alpha r}L}{mUI_z}\left(\frac{T_{nom,r}}{r} + C_{\alpha r}\right)}{polos} \tag{5.65}$$

En este punto, se asume que el ángulo de entrada para las llantas traseras es cero, por lo que las funciones con el término δ_r se desprecian. Igualmente, dadas las velocidades bajas que desarrolla el móvil, se asume que la velocidad lateral V es despreciable, por lo que finalmente, se trabajará con el modelo dado en la ecuación (5.64).

Finalmente, se asume que los cambios en la velocidad lineal no van a cambiar considerablemente los polos del sistema, ya que el rango de velocidades alcanzables por el vehículo no es amplio. Por otra parte, algunos de los parámetros del modelo seleccionado, son difíciles de medir u obtener (T_{nom}, $C_{\alpha r}$, $C_{\alpha f}$), por lo que se decidió identificar el sistema, seleccionando una estructura que se acomode al modelo escogido. Este procedimiento se discute en la siguiente sección.

5.3.4 Identificación del Sistema

Inicialmente, para realizar el procedimiento de identificación, se utiliza el controlador PI del sistema anterior para fijar la velocidad lineal constante. La onda escogida para realizar el procedimiento, es la señal binaria aleatoria (RBS) mostrada en la figura 5.11.

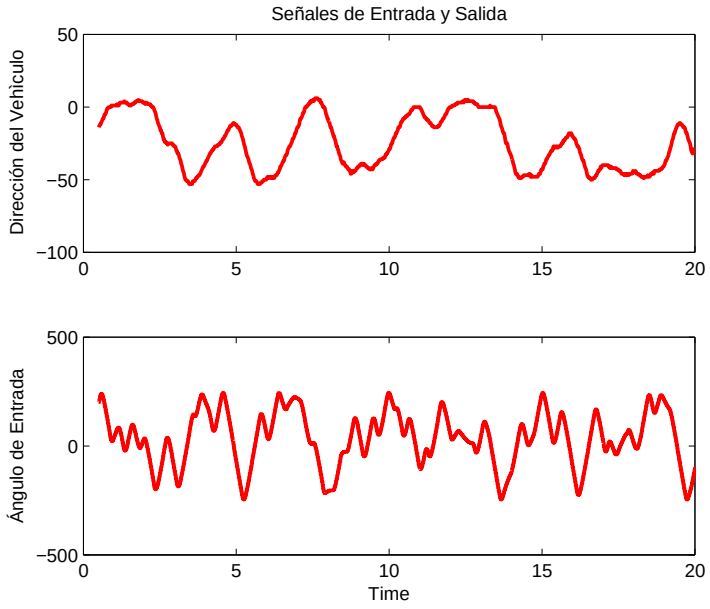


Figura 5.11: Identificación de la dinámica lateral

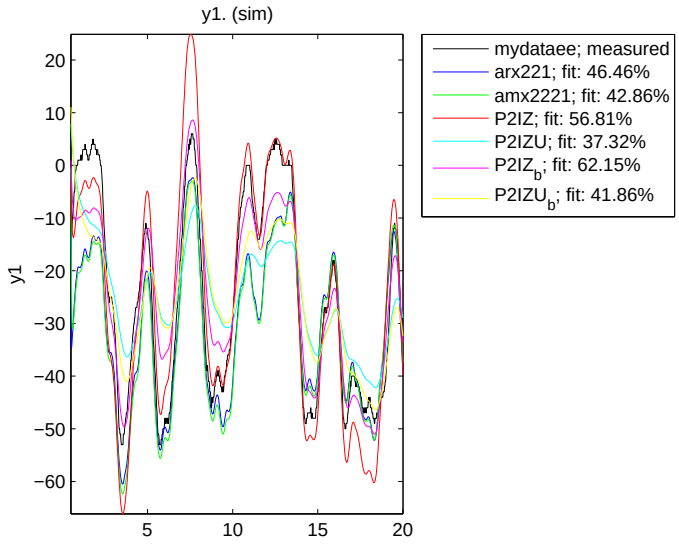


Figura 5.12: Ajuste a los datos de las diferentes estructuras

El nivel de ajuste alcanzado por los diferentes modelos escogidos (con estructura similar a la del modelo teórico de la sección anterior) se muestra en la figura 5.12.

Dados los niveles de ajuste obtenido, y su similitud en estructura con el modelo teórico, se escogió el modelo de proceso ($P2IZ_b$ en 5.12) de la forma:

$$\begin{aligned} G(s) &= K_p \frac{1 + T_z s}{s(1 + T_{p1} s)(1 + T_{p2} s)} \\ &= \frac{\frac{K_p}{T_{p1} T_{p2}} + \frac{K_p T_z}{T_{p1} T_{p2}} s}{s(\frac{1}{T_{p1} T_{p2}} + \frac{(T_{p1} + T_{p2})}{T_{p1} T_{p2}} s + s^2)} \end{aligned} \tag{5.66}$$

Con $K_p = -962{,}2257$, $T_{z1} = -111{,}61$, $T_{p1} = 40904$ y $T_{p2} = 0{,}0222$.

5.3.5 Modelo numérico del Sistema

Reemplazando los valores del sstema ajustado en la ecuación anterior, se tiene la diguiente función de transferencia:

$$G_{num}(s) = \frac{1{,}183s - 0{,}01059}{s(s^2 + 4{,}504s + 1{,}101 * 10^{-5})} \tag{5.67}$$

La tabla 5.4 muestra la equivalencia entre los parámetros del modelo numérico y del modelo teórico.

$\frac{K_p T_z}{T_{p1} T_{p2}}$	$\frac{2a}{I_z}\left(\frac{T_{nom,f}}{r} + C_{\alpha f}\right)$	1.183
$\frac{K_p}{T_{p1} T_{p2}}$	$\frac{4C_{\alpha f} L}{mUI_z}\left(\frac{T_{nom,f}}{r} + C_{\alpha f}\right)$	-0.01059
$\frac{(T_{p1}+T_{p2})}{T_{p1} T_{p2}}$	$\left(\frac{2(C_{\alpha f}+C_{\alpha r})}{mU} + \frac{2(C_{\alpha f} a^2 + C_{\alpha r} b^2)}{I_z U}\right)$	4.504
$\frac{1}{T_{p1} T_{p2}}$	$\frac{2}{I_z}(aC_{\alpha f} - bC_{\alpha r})$	$1{,}101 * 10^{-5}$

Tabla 5.4: Parámetros del sistema

5.3.6 Experimentos de Control

Con el modelo de la sección anterior, se procede a realizar los experimentos de control que confirmen la validez del mismo. Inicialmente, se prueba con un controlador proporcional $K_p = 10$. Se programa el vehículo para que siga en línea recta (ref=0°) y luego de un giro de 90°. Los resultados se observan en la figura 5.13.

Los resultados muestran cierta discrepancia entre la respuesta de los modelos en estado estacionario, pero el estado estable y el tiempo de establecimiento son similares, otorgando una idea de

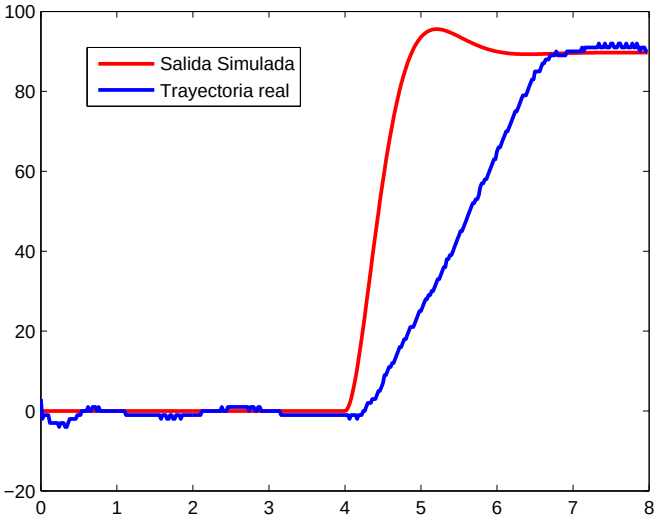


Figura 5.13: Prueba de dirección con Control P

dinámicas similares. La sdiferencias pueden deberse a saturación del sistema, debida a la imposición de mantener el ángulo de giro de las llantas en máximo 45°(Lo que evita problemas de desarmado o atasco en la estructura del vehículo). La trayectoria seguida se muestra en la figura 5.14.

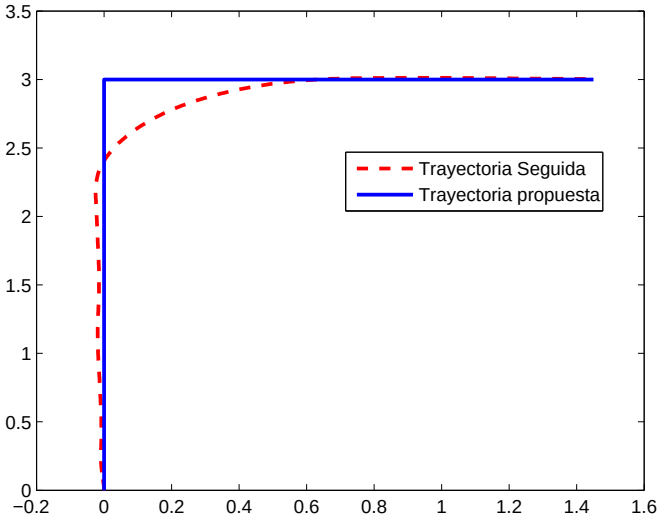


Figura 5.14: Trayectoria seguida con Control P

El móvil es capáz de realizar el giro, no en forma perfecta, ya que sus restricciones contructivas

se lo impiden, tal como sucede con los automóviles y otros vehículos de calle. La siguiente prueba fue con un controlador PI (Proporcional Integral), con una función de transferencia dada por:

$$C(s) = \frac{7s + 0{,}1}{s} \tag{5.68}$$

La figura 5.15 muestra el resultado de utilizar el controlador en el seguimiento de la trayectoria. El sistema tiene un sobrepico apreciable, y el seguimiento de ángulos si bien es bueno, no es apto para trayectorias cortas. La trayectoria mostrada en 5.16, confirma los resultados, el tiempo de establecimiento es algo lento pero el seguimiento en estado estable es bueno.

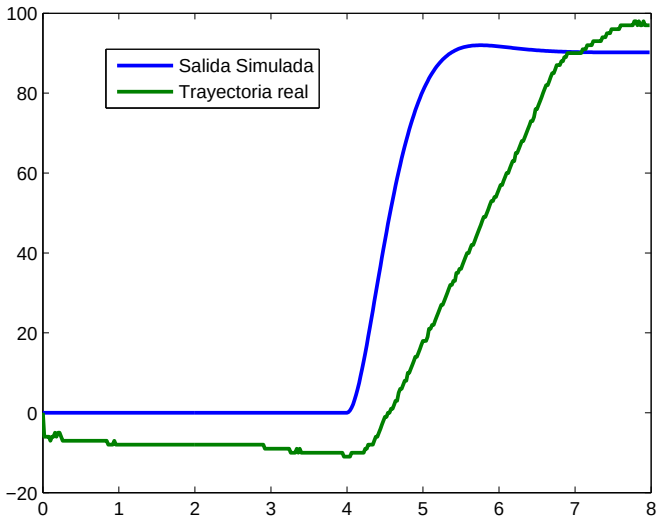


Figura 5.15: Prueba de dirección con Control PI

El expermimento final consistió en el seguimiento de una trayectoria un poco más compleja. Este debía realizar una figura similar a un cuadrado o rectángulo. Para tal fin, se decidió utlizar el controlador proporcional del primer experimento. El resultado de aplicar referencias escalonadas ascendentes, se ve en la figura 5.17.

La respuesta en términos generales es buena, logrando el estdo estable deseado, y sin sobrepico, que en el caso de seguimiento de caminos o trayectorias se considera ideal. La trayectoria seguida finalmente se muestra en la figura 5.18.

Se observa que el vehículo nuevamente es capaz de cumplir con el seguimiento de la trayectoria, usando únicamente un controlador proporcional. El sistema está en capacidad de seguir diversos

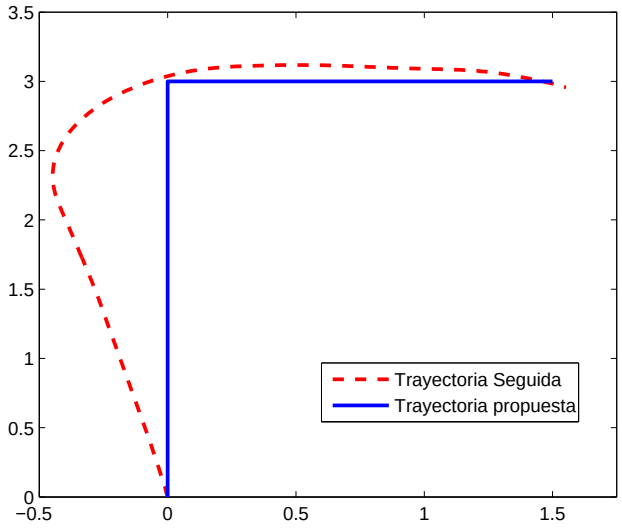


Figura 5.16: Trayectoria seguida con Control PI

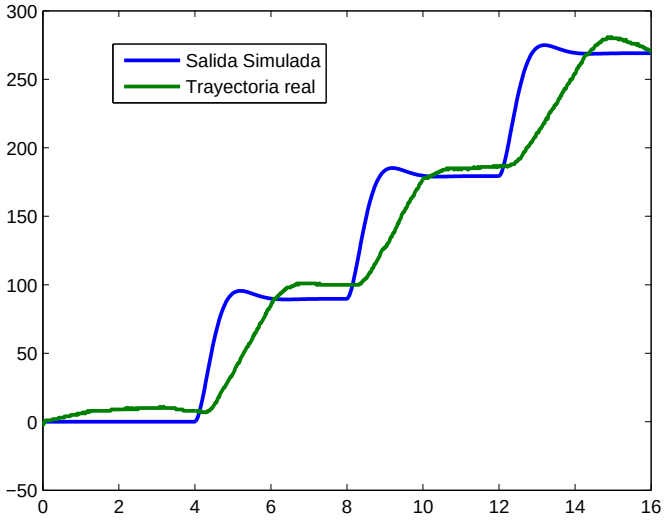


Figura 5.17: Prueba de dirección con referencia incremental

tipos de camino, siempre y cuando estos no vayan más allá de las limitaciones físicas de su contrucción.

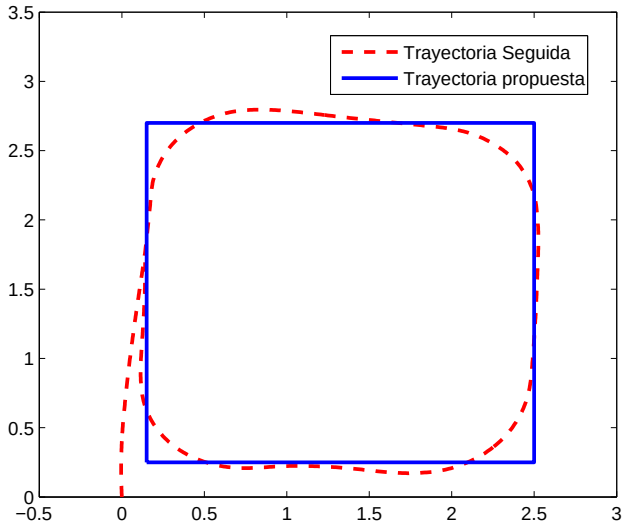


Figura 5.18: Trayectoria Cuadrada

5.4 Conclusiones

El modelado dinámico de vehículos es una tarea compleja que requiere de submodelos especializados, y su medición, obtención y/o identificación de parámetros no es una labor trivial. No obstante, permiten visualizar la dinámica de eventos comunes, como lo es el movimiento de vehículos en su espacio, y los diferentes elementos que intervienen en el mismo.

Las estrategias de control muestran un desempeño aceptable, limitado exclusivamente por las limitaciones de la cosntrucción mecánica, y que le son comunes y afines a los de cualquier vehículo con dirección delantera y ruedas traseras de tracción. El sistema muestra un seguimiento de trayectorias razonable, lo que permite que le sean programados diferentes tipos de camino, y éste sea capaz de sortearlos sin mayores problemas.

El modelo utilizado es suceptible del uso de estrategias de control sencillas, lo que convierte al sistema como ideal para la introducción e ilustración de conceptos básicos del control en lazo cerrado y prácticas introductorias a las estrategias de control más simples.

6
Sistema de Balance: Segway

Al hablar de sistemas de balance nos referimos a todo tipo de sistemas que oscilen o realizen un movimiento de balanceo alrededor de un punto de equilibrio o de rotación. Para completar la definición, se prefiere que dicho movimiento requiera de una señal externa (fuerza) para desarrollarse de forma estable, es decir, la dinámica natural del sistema sea inestable. Pueden ser llamados robots o sistemas equilibristas [28]. Dentro de esta definición podemos hallar al péndulo invertido, el segway, el pendubot, el nBot, y en general, a todo tipo de vehículos de dos ruedas (scooters) con autobalanceo.

Una analogía muy simple puede explicar la motivación y el origen para este tipo de sistemas. Muchos niños han intentado balancear una vara o bastón en la palma de su mano o en la punta de sus dedos y poder mantenerla en posición vertical; para lograrlo, deben ajustar constantemente la posición de la mano. Dispositivos como el péndulo invertido o el Segway realizan la misma acción básica para sostenerse, solo que limitada a un menor número de dimensiones (grados de libertad).

De igual manera que ocurre en el sistema del bastón sobre la mano, los sistemas de autobalanceo (Segway, péndulo invertido, etc.) cuentan con un comportamiento dinámico inestable. La señal de entrada, que en este caso es una fuerza, debe ser aplicada apropiadamente para mantener el sistema en posición vertical. Todo esto implica la utilización de una teoría de control apropiada y convierte a los sistemas de autobalance en el escenario ideal para aplicar y comparar diferentes técnicas de control.

Sistemas como el péndulo invertido o el Segway representan algunos de los retos más difíciles e interesantes que se pueden presentar dentro de la ingeniería de control, y de ahí su importancia en dicho campo. Es tal su relevancia que ha sido escogido en innumerables ocasiones para ser objeto de estudio tanto en tareas de modelado como de control, en el caso del péndulo invertido. Por tratarse de sistemas inestables, son ideales como tarea de control propuesta para estudiantes de pregrado y posgrado, ya que por su dinámica, se hace vital la implementación de estrategias de control de diferente complejidad.

Las razones principales para seleccionar este tipo de sistemas como retos de control son [26]:

- Se trata de sistemas de alta disponibilidad en los laboratorios académicos existentes alrededor del mundo.
- La teoría desarrollada para estos sistemas es amplia y muy variada, por lo que es relativamente fácil documentar proyectos relacionados con los mismos.
- Son sistemas no lineales, que pueden ser tratados como lineales sin variaciones o errores considerables para un amplio rango de operación.
- Constituyen una práctica de aplicación de muy buen nivel para los estudiantes control.

Los anteriores postulados hacen que los sistemas de balanceo sean una parte indispensable en un laboratorio de prácticas de control automático y la variedad de estrategias de control implementables en estos sistemas los convierte en una fuente inagotable de experimentación. A continuación se describirá el sistema de balanceo más común, el péndulo invertido, el pendubot, y el Segway, una aplicación interesante de este tipo de sistemas, y que es la escogida para formar parte de la plataorma descrita en el presente trabajo.

6.1 El Péndulo Invertido

El más importante y famoso de los sistemas de balance es el péndulo invertido. Aparece en una gran cantidad de libros de texto y publicaciones de investigación, y a pesar de estar presente en el ámbito científico durante muchos años, no pierde validez como base de investigación y experimentación de diferentes formas y estrategias de control, tanto a nivel universitario como de posgrado. El péndulo invertido pretende mantener una barra en posición horizontal, la cual se encuentra pivotada sobre una masa o carro. A éste último se le aplica la fuerza que ayuda a equilibrar la barra descrita anteriormente. El esquema del sistema descrito se puede ver en la figura 6.1.

El punto inicial para entender la dinámica del sistema es la obtención de su modelo matemático. Se empieza por las ecuaciones de movimiento de la masa o carro. Por sumatoria de fuerzas en la dirección horizontal del carro se obtiene:

$$M\ddot{x} + B\dot{x} + P_x = F \tag{6.1}$$

Donde x es la varaible de movimiento horizontal, M es la masa del carro, B el coeficiente de fricción con la superficie de movimiento, P_x la fuerza que ejerce la barra sobre el carro en la dirección de moviento y F la fuerza aplicada al carro. Ahora, se definen las coordenadas del centro de

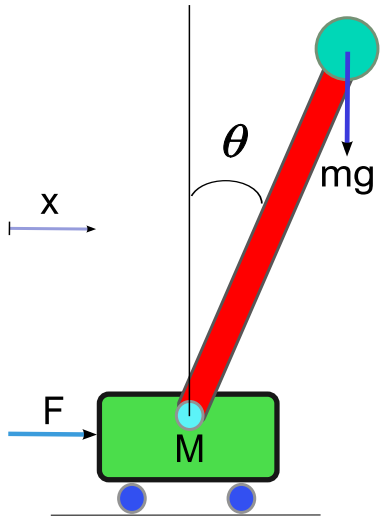


Figura 6.1: Péndulo Invertido

gravedad del péndulo:

$$x_c = x + l\sin\theta \tag{6.2}$$

$$y_c = \frac{l}{2}\cos\theta \tag{6.3}$$

El siguiente paso es obtener las ecuaciones de movimiento de la barra. Por sumatoria de torques alrededor del centro de masa de la barra se tiene:

$$I\ddot{\theta} = P_y\frac{l}{2}\sin\theta - P_x\frac{l}{2}\cos\theta \tag{6.4}$$

Donde I es el momento de inercia de la barra y P_y la fuerza ejercida por el carro sobre la barra en dirección vertical. Entonces, las ecuaciones de movimiento horizontal del péndulo son:

$$\begin{aligned} P_x &= m\ddot{x}_c \\ P_x &= m\frac{d^2}{dt^2}\left(x + \frac{1}{2}l\sin\theta\right) \\ P_x &= m\ddot{x} + \frac{1}{2}ml\ddot{\theta}\cos\theta - \frac{1}{2}ml\dot{\theta}^2\sin\theta \end{aligned} \tag{6.5}$$

Donde m es la masa de la barra. Las ecuaciones de movimiento vertical del péndulo son:

$$\begin{aligned} P_y - mg &= m\ddot{y}_c \\ P_y - mg &= m\frac{d^2}{dt^2}\left(\frac{1}{2}l\cos\theta\right) \\ P_y &= mg - \frac{1}{2}ml\ddot{\theta}\sin\theta - \frac{1}{2}ml\dot{\theta}^2\cos\theta \end{aligned} \tag{6.6}$$

A partir de las relaciones obtenidas anteriormente, se llega a las ecuaciones de movimiento del sistema completo:

$$F = (M+m)\ddot{x} + B\dot{x} + \frac{1}{2}\ddot{\theta}\sin\theta - \frac{1}{2}\dot{\theta}^2\cos\theta \tag{6.7}$$

$$\left(I + \frac{1}{4}ml^2\right)\ddot{\theta} = \frac{1}{2}mlg\sin\theta - \frac{1}{2}\ddot{x}\cos\theta \tag{6.8}$$

Este modelo es la base para entender el comportamiento dinámico del péndulo y de buena parte de los sistemas de balanceo. Para cualquier otro sistema de este tipo, se pueden seguir los pasos anteriores y llegar a modelos similares.

6.1.1 Aplicaciones de Péndulo Invertido

Simulación de Brazos Robóticos

El problema representado por el péndulo invertido es muy similar al de algunas extremidades o brazos robóticos. La dinámica mostrada por el péndulo simula la dinámica de un brazo robótico cuando el punto de presión se encuentra por debajo del centro de gravedad, lo que hace que el sistema se vuelva inestable. El brazo se comporta de forma muy parecida al péndulo invertido bajo estas condiciones [26].

Modelos de caminata Bípeda

El péndulo invertido es ampliamente aceptado como un modelo adecuado para la caminata bípeda (como modelo para la pierna y su movimiento). El péndulo invertido es inestable (sin interferencias externas como fuerzas o sistemas resorte/amortiguador sujetos al mismo), lo que hace indispensable implementar un sistema realimentado para estabilizar su dinámica [26].

6.2 Pendubot

El pendubot o Pendulum Robot es un dispositivo mecatrónico usado en la educación de ingeniería de control e investigación en robótica y control no lineal. Este aparato es un robot planar de dos eslabones, con actuador en el hombro, pero no en el codo [37], haciéndolo un sistema subactuado, con carácterísticas dinámicas interesantes. Su modelo se obtiene de forma similar a la del péndulo invertido. El esquema del pendubot corresponde a la figura 6.2.

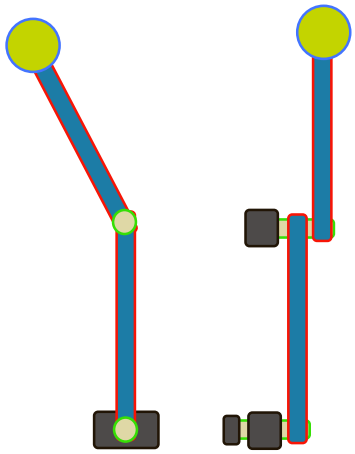


Figura 6.2: Esquema del Pendubot

6.3 Segway

El segway es una forma muy interesante de péndulo invertido. Inventado en 2001 por Dean Kamen y presentado por la compañía Segway Inc. [1], consiste en un sistema de transporte personal autobalanceado, con dos motores que mantienen la posición vertical del sistema de mando y al usuario, que actuan como péndulo. Para dirigir el movimiento, el usuario solo debe inclinarse en la dirección deseada. Una forma especial es el segway autónomo, el cual mantiene la posición del péndulo de forma estable, a manera de robot. En [25] se ve un ejemplo de control LQR muy interesante, con desarrollo en Lego. Es un reto muy interesante y original, en el cual se puede ofrecer un problema de control doble, la estabilización vertical y el control de movimiento. El esquema básico del Segway se ve en la figura 6.3:

[1] Fuente: http://es.wikipedia.org/wiki/Segway

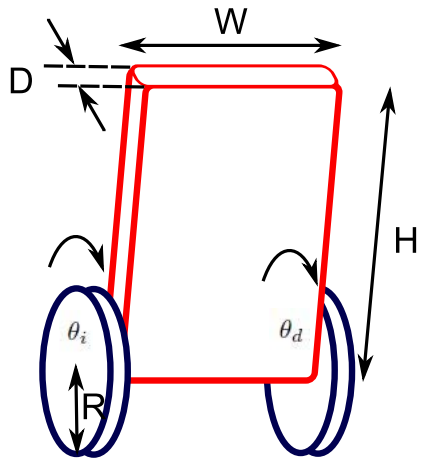


Figura 6.3: Esquema del Segway

Donde H, W, y D son la altura, el ancho y la profundidad del cuerpo del Segway respectivamente, R es el radio de las llantas y $\theta_{i,d}$ son los ángulos de giro de cada llanta.

6.3.1 Sistema Construido

Una vez se cuenta con el esquema básico del sistema, se procede a la construcción del mismo, usando las piezas y sensores del kit de Lego Mindstorms. Se escogió como base para el diseño la construcción del NXTWay-GS que aparece en [25]. Este sistema utiliza el Giroscópio de Hitechnic (descrito en el capítulo 2) para medir la velocidad angular del cuerpo y el sensor ultrasónico de Lego para detectar obstáculos y medir distancias de los mismos con respecto a la posición del Segway. Éste puede verse en la figura 6.4. La forma del modelo construido es asimilable a un paralelepípedo, como el mostrado en la figura 6.3, lo cual permite aproximar la geometría del sistema a la de dicho esquematico, haciendo posible su uso en la derivación de los diferentes parámetros geómetricos y ecuaciones de movimiento del sistema.

6.3.2 Modelo del Sistema

A continuación se describe el modelo del Segway y se obtienen las ecuaciones de movimiento del mismo. Este sistema puede considerarse como un péndulo invertido de dos llantas [25]. Para comenzar, se define el sistema coordenado para el cual se obtendrán las ecuaciones de movimiento, como se ve en la figura 6.5. Los parámetros mostrados en el mencionado esquema se listan en la tabla 6.1 (Tomados de [25]).

6. Sistema de Balance: Segway

Figura 6.4: Sistema Construido: Segway

g=9.81	$[m/seg^2]$	Aceleración de la gravedad
m=0.03	[Kg]	Peso de las llantas
R=0.04	[m]	Radio de las llantas
$J_w = mR^2/2$	$[Kgm^2]$	Momento de inercia de las llantas
M=0.6	[Kg]	Peso del cuerpo
H=0.144	[m]	Altura del cuerpo
W=0.14	[m]	Ancho del cuerpo
D=0.04	[m]	Profundidad del cuerpo
H=0.14	[m]	Altura del cuerpo
$L = H/2$	[m]	Distancia del centro de gravedad del cuerpo al centro de la llanta
$J_\psi = ML^2/3$	$[Kgm^2]$	Momento de inercia del cuerpo (inclinación)
$J_\phi = M(D^2 + W^2)/12$	$[Kgm^2]$	Momento de inercia del cuerpo (giro)
$J_m = 10^{-5}$	$[Kgm^2]$	Momento de inercia del motor
$R_m = 6{,}69$	$[\Omega]$	Resistencia eléctrica del motor
$K_b = 0{,}468$	$[V\ seg/rad]$	Constante electromotriz del motor
$K_t = 0{,}317$	$[Nm/A]$	Constante de torque
n		Relación de reducción del motor
$f_m = 0{,}0022$		Coeficiente de fricción entre el motor y el cuerpo
$f_w = 0$		Coeficiente de fricción entre las llantas y el piso

Tabla 6.1: Parámetros físicos del Segway

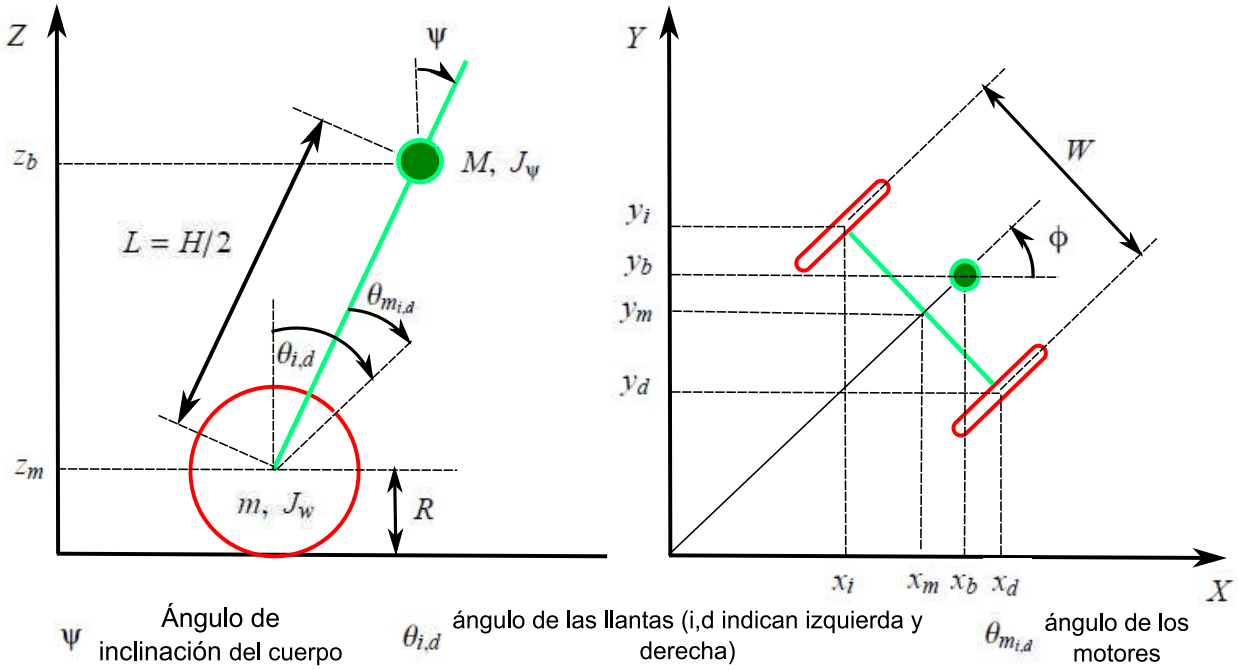


Figura 6.5: Sistema coordenado para el Segway[a]

[a] Adaptada de [25]

Los parámetros R_m, K_b y K_f fueron obtenidos por los experimentos realizados por Ryo Watanabe[2], lo que simplifica la búsqueda de parámetros para el sistema. Los parámetros J_ψ y J_ϕ son estimados con la aproximación al paralelepípedo que se hizo de la forma del Segway en la figura 6.3. Esto permite la estimación rápida de los mismos y evita la realización de experimentos adicionales para su medición. Finalmente, los parámetros J_m, n, f_m y f_w se toman los estimados en [25], ya que se ha visto que son útiles en la descripción de la dinámica del modelo, pero no fueron medidos, debido a que su medición implica un proceso engorroso y difícil de realizar.

6.3.3 Ecuaciones de Movimiento

Se pueden obtener las ecuaciones de movimiento del péndulo invertido de dos llantas mediante la formulación mecánica de Lagrange (energías), basándose en en el sistema coordenado de la figura 6.5. Si la dirección de movimiento del péndulo está en la dirección positiva del eje x en $t = 0$, las coordenadas del sistema son:

[2] Disponible en http://web.mac.com/ryo_watanabe/iWeb/Ryo%27s%20Holiday/LEGO%20Mindstorms%20NXT.html

$$(x_m, y_m, z_m) = (R\theta\cos\phi, R\theta\sin\phi, R) \tag{6.9}$$

$$(\theta, \phi) = \left(\frac{1}{2}(\theta_i + \theta_d), \frac{R}{W}(\theta_d - \theta_i)\right) \tag{6.10}$$

$$(x_i, y_i, z_i) = \left(x_m - \frac{W}{2}\sin\phi, y_m + \frac{W}{2}\cos\phi, z_m\right) \tag{6.11}$$

$$(x_d, y_d, z_d) = \left(x_m + \frac{W}{2}\sin\phi, y_m - \frac{W}{2}\cos\phi, z_m\right) \tag{6.12}$$

$$(x_b, y_b, z_b) = (x_m + L\sin\psi\cos\theta, y_m + L\sin\psi\sin\theta, z_m + L\cos\psi) \tag{6.13}$$

Podemos definir la energía cinética traslacional T_1, la energía cinética rotacional T_2 y la energía potencial U como:

$$T_1 = \frac{1}{2}m(\dot{x}_i^2 + \dot{y}_i^2 + \dot{z}_i^2) + \frac{1}{2}m(\dot{x}_d^2 + \dot{y}_d^2 + \dot{z}_d^2) + \frac{1}{2}M(\dot{x}_b^2 + \dot{y}_b^2 + \dot{z}_b^2) \tag{6.14}$$

$$T_2 = \frac{1}{2}J_w\dot{\theta}_i^2 + \frac{1}{2}J_w\dot{\theta}_d^2 + \frac{1}{2}J_\psi\dot{\psi}^2 + \frac{1}{2}J_\phi\dot{\phi}^2 + \frac{1}{2}n^2J_m(\dot{\theta}_i - \dot{\psi})^2 + \frac{1}{2}n^2J_m(\dot{\theta}_d - \dot{\psi})^2 \tag{6.15}$$

$$U = mgz_i + mgz_d + Mgz_b \tag{6.16}$$

El quinto y sexto término en la expresión de T_2 corresponden a la energía cinética rotacional del motor DC. La expresión del lagrangiano L es entonces:

$$L = T_1 + T_2 - U \tag{6.17}$$

Se usan como variables generalizadas θ, el ángulo promedio de las llantas derecha e izquierda, ψ, el ángulo de inclinación del cuerpo y ϕ el ángulo de giro del sistema. Las ecuaciones de Lagrange son las siguientes:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial\dot{\theta}}\right) - \frac{\partial L}{\partial\theta} = F_\theta \tag{6.18}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial\dot{\psi}}\right) - \frac{\partial L}{\partial\psi} = F_\psi \tag{6.19}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial\dot{\phi}}\right) - \frac{\partial L}{\partial\phi} = F_\phi \tag{6.20}$$

Al evaluar las expresiones anteriores, se obtiene:

$$[(2m+M)R^2 + 2J_w + 2n^2J_m]\ddot{\theta} + (MLR\cos\psi - 2n^2J_m)\ddot{\psi} - MLR\dot{\psi}^2\sin\psi - [(2m+M)R^2\theta + MLR\sin\psi] = F_\theta \tag{6.21}$$

$$(MLR\cos\psi - 2n^2J_m)\ddot{\theta} + (ML^2 + J_\psi + 2n^2J_m)\ddot{\psi} - MgL\sin\psi - (MLR\theta + ML^2\sin\psi)\dot{\phi}^2\cos\psi = F_\psi \tag{6.22}$$

$$\left[\frac{1}{2}mW^2 + J_\phi + \frac{W^2}{2R^2}(J_w + n^2J_m) + (2m+M)R^2\theta^2 + 2MLR\theta\sin\psi\right]\ddot{\phi} + 2[(2m+M)R^2\theta\dot{\theta} + ML^2\dot{\psi}\sin\phi\cos\phi + MLR(\dot{\theta}\sin\psi + \theta\dot{\psi}\cos\psi)]\dot{\phi} = F_\phi \tag{6.23}$$

Teniendo en cuenta la fricción viscosa y el torque del motor, las fuerzas generalizadas son:

$$(F_\theta, F_\psi, F_\theta) = \left(\frac{1}{2}(F_i + F_d), F_\psi, \frac{R}{W}(F_d - F_i)\right) \tag{6.24}$$

$$F_i = nK_t i_i + f_m(\dot{\psi} - \dot{\theta}_i) - f_w\dot{\theta}_i \tag{6.25}$$

$$F_r = nK_t i_r + f_m(\dot{\psi} - \dot{\theta}_r) - f_w\dot{\theta}_r \tag{6.26}$$

$$F_w = -nK_t i_l - nK_t i_r - f_m(\dot{\psi} - \dot{\theta}_l) - f_m(\dot{\psi} - \dot{\theta}_r) \tag{6.27}$$

donde $i_{i,d}$ es la corriente del motor. No es posible usar la corriente del motor para controlarlo, ya que estos usan PWM. Por lo tanto se evalúa la relación entre la corriente $i_{i,d}$ el voltaje $v_{i,d}$ usando la ecuación del motro eléctrico. Si se considera que la fricción dentro del motor es despreciable, se tiene que:

$$L_m\frac{di_{i,d}}{dt} = v_{i,d} + K_b(\dot{\psi} - \dot{\theta}_{i,d} - R_m i_{i,d}) \tag{6.28}$$

Si se asume que la inductacia del motor es aproximadamente cero, se tiene la siguiente expresión para la corriente:

$$i_{i,d} = \frac{v_{i,d} + K_b(\dot{\psi} - \dot{\theta}_{i,d})}{R_m} \tag{6.29}$$

Ahora las fuerzas generalizadas pueden ser expresadas como:

$$F_\theta = \frac{\alpha}{2}(v_i + v_d) - (\beta + f_w)\dot{\theta} + \beta\dot{\psi} \tag{6.30}$$

$$F_\psi = -\alpha(v_i + v_d) + 2\beta\dot{\theta} - 2\beta\psi \tag{6.31}$$

$$F_\phi = \frac{R}{W}\alpha(v_d - v_i) - \left(\beta + \frac{W}{R}f_w\right)\dot{\phi} \tag{6.32}$$

$$\tag{6.33}$$

Donde:

$$\alpha = \frac{nK_t}{R_m} \qquad \beta = \frac{nK_tK_b}{R_m} + f_m \tag{6.34}$$

Con las expresiones anteriores, es posible obtener las ecuaciones de estado del sistema. Se procede linealizando las ecuaciones alrededor del punto de equilibrio del Segway. Esto implica evaluar el límite cuando $\psi \to 0$ ($\sin\psi \to 0$, $\cos\psi \to 1$), y considerar nulos los términos diferenciales elevados al cuadrado. Entonces, las ecuaciones de movimiento son:

$$[(2m + M)R^2 + 2J_w + 2n^2J_m]\ddot{\theta} + (MLR - 2n^2J_m)\ddot{\psi} = F_\theta \tag{6.35}$$

$$(MLR - 2n^2J_m)\ddot{\theta} + (ML^2 + J_\psi + 2n^2J_m)\ddot{\psi} - MgL\psi = F_\psi \tag{6.36}$$

$$\left[\frac{1}{2}mW^2 + J_\phi + \frac{W^2}{2R^2}(J_w + n^2J_m)\right]\ddot{\phi} = F_\phi \tag{6.37}$$

Podemos re-escribir las ecuaciones así:

$$E\begin{bmatrix}\ddot{\theta}\\ \ddot{\psi}\end{bmatrix} + F\begin{bmatrix}\dot{\theta}\\ \dot{\psi}\end{bmatrix} + GE\begin{bmatrix}\theta\\ \psi\end{bmatrix} = H\begin{bmatrix}v_i\\ v_r\end{bmatrix} \tag{6.38}$$

$$E = \begin{bmatrix}(2m + M)R^2 + 2J_w + 2n^2J_m & MLR - 2n^2J_m\\ MLR - 2n^2J_m & ML^2 + J_\psi + 2n^2J_m\end{bmatrix}$$

$$F = \begin{bmatrix}\beta + f_w & -\beta\\ -2\beta & 2\beta\end{bmatrix}$$

$$G = \begin{bmatrix}0 & 0\\ 0 & -MgL\end{bmatrix}$$

$$H = \begin{bmatrix}\alpha/2 & \alpha/2\\ -\alpha & -\alpha\end{bmatrix}$$

Donde:

$$K\ddot{\phi} + I\dot{\phi} = J(v_d - v_i) \tag{6.39}$$

$$I = \beta + \frac{W}{R}f_w$$

$$J = \frac{R}{W}\alpha$$

$$K = \frac{1}{2}mW^2 + J_\phi + \frac{W^2}{2R^2}(J_w + n^2J_m)$$

Si asumimos como variables de estado x_1 y $x2$, y u la salida, donde:

$$x_1 = \begin{bmatrix} \theta, & \psi, & \dot{\theta}, & \dot{\psi} \end{bmatrix}^T \qquad x_2 = \begin{bmatrix} \phi, & \dot{\phi} \end{bmatrix}^T \qquad U = \begin{bmatrix} v_i, & v_d \end{bmatrix}^T \tag{6.40}$$

se puede llegar a la siguiente expresión en variables de estado:

$$\dot{x}_1 = A_1 x_1 + B_1 u \tag{6.41}$$
$$\dot{x}_2 = A_2 x_2 + B_2 u \tag{6.42}$$

Con las matrices A_1, A_2, B_1 y B_2:

$$A_1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & A_1(3,2) & A_1(3,3) & A_1(3,4) \\ 0 & A_1(4,2) & A_1(4,3) & A_1(4,4) \end{bmatrix} \qquad B_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ B_1(3) & B_1(3) \\ B_1(3) & B_1(3) \end{bmatrix} \tag{6.43}$$

$$A_2 = \begin{bmatrix} 0 & 1 \\ 0 & -I/K \end{bmatrix} \qquad B_2 = \begin{bmatrix} 0 & 0 \\ -J/K & J/K \end{bmatrix} \tag{6.44}$$

Donde:

$$A_1(3,2) = \frac{-MgLE(1,2)}{det(E)}$$
$$A_1(4,2) = \frac{MgLE(1,1)}{det(E)}$$
$$A_1(3,3) = \frac{-[(\beta + f_w)E(2,2) + 2\beta E(1,2)]}{det(E)}$$
$$A_1(4,3) = \frac{[(\beta + f_w)E(1,2) + 2\beta E(1,1)]}{det(E)}$$
$$A_1(3,4) = \frac{\beta(E(2,2) + 2E(1,2))}{det(E)}$$
$$A_1(4,4) = \frac{-\beta(E(1,2) + 2E(1,1))}{det(E)}$$
$$B_1(3) = \frac{\alpha(E(2,2)/2 + E(1,2))}{det(E)}$$
$$B_1(4) = \frac{-\alpha(E(1,2)/2 + E(1,1))}{det(E)}$$
$$det(E) = E(1,1)E(2,2) - E(1,2)^2$$

Este modelo coincide con el que aparece en [25] y representa la dinámica del sistema.

6.3.4 Modelo Numérico del Sistema

Cuando se introducen los datos de la tabla 6.1 en el modelo obtenido anteriormente, se llega a las siguientes expresiones:

$$\dot{x}_1 = \begin{bmatrix} 0 & 0 & 1{,}0000 & 0 \\ 0 & 0 & 0 & 1{,}0000 \\ 0 & -409{,}7184 & -104{,}6299 & 104{,}6299 \\ 0 & 269{,}6273 & 54{,}5833 & -54{,}5833 \end{bmatrix} x_1 + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 101{,}6952 & 101{,}6952 \\ -53{,}0523 & -53{,}0523 \end{bmatrix} u \tag{6.45}$$

$$\dot{x}_2 = \begin{bmatrix} 0 & 1{,}0000 \\ 0 & -15{,}6030 \end{bmatrix} x_2 + \begin{bmatrix} 0 & 0 \\ -8{,}6659 & \\ 8{,}6659 & \end{bmatrix} u \tag{6.46}$$

Este modelo, que coincide con el encontrado en [25], es el usado en la siguiente sección para el diseño de los controladores. De acuerdo a la teoría de control, todos los valores propios de las matrices A_1 y A_2 deben ser mayores a cero para que el sistema sea estable. Para la primer matriz se tienen como valores propios 0, -160.6649, 6.8021 y -5.3505, lo que implica la inestabilidad del sistema. Para la segunda matriz, se tienen 0 y -15.6030, lo que hace a dicho sistema marginalmente estable. Los controladores realizados en la siguiente sección deberán estabilizar la dinámica de la planta.

6.3.5 Experimentos de Control

En el primer experimento realizado, se diseñó un controlador en variables de estado, por colocación de polos. El vector de polos seleccionado y el vector de realimentación de polos obtenido se muestran a continuación:

$$p = \begin{bmatrix} -160 & -0{,}7 + 0{,}7 * j & -0{,}7 - 0{,}7 * j & -14 \end{bmatrix} \tag{6.47}$$

$$K = \begin{bmatrix} -0{,}1931 & -26{,}3277 & -0{,}8053 & -1{,}6963 \end{bmatrix} \tag{6.48}$$

Los polos con parte compleja sirven para suavizar la respuesta, los polos con parte real más grande, son útiles para dar velocidad a la dinámica del sistema. La respuesta del sistema con dicho controlador se observa en la figura 6.6.

La simulación fue hecha con una condición inicial $\psi = 0{,}125$ Los resultados muestran una oscilación alrededor de un punto de equilibrio $\psi = 0$. Dicho movimiento se presenta por la cuantización de los datos y el trabajo con punto flotante. No obstante la oscilación, el sistema tiene un comportamiento estable, consiguiendo uno de los propósitos del controlador. El ángulo no supera los 8 grados, manteniendo el Segway en equilibrio. El esquema de control cumple con estabilizar el ángulo de inclinación del aparato, pero no su posición lineal, permitiendo que se mueva de forma indefinida,

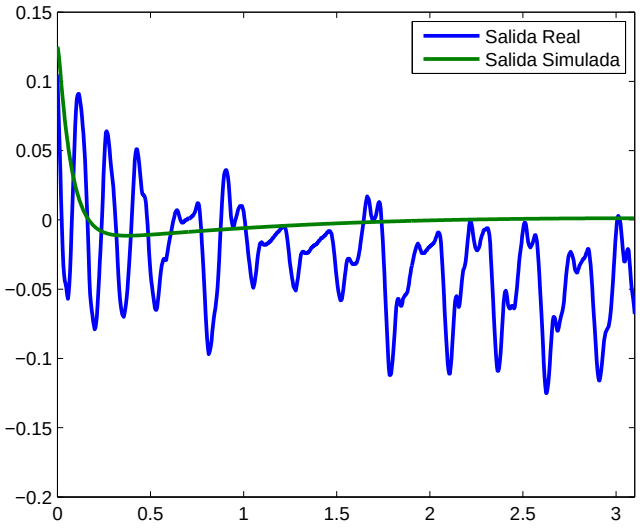


Figura 6.6: Controlador con colocación de polos

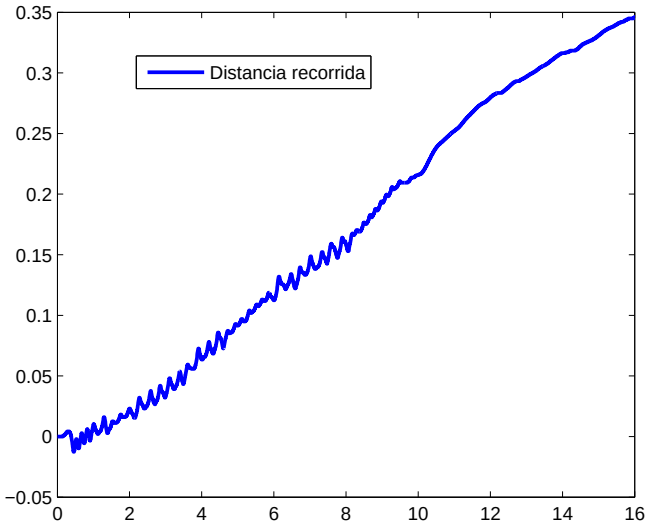


Figura 6.7: Distancia recorrida por el móvil

como se aprecia en la figura **??**.

En el segundo experimento de control, se decidió agregar efecto integral al sistema con el diseño de un PI vectorial. Éste fue sintetizado usando la misma filosofía de colocación de polos del

experimento anterior. Los polos escogidos y el vector de realimentación de estados se muestran a continuación:

$$p = \begin{bmatrix} -160 & -0{,}7 + 0{,}7 * j & -0{,}7 - 0{,}7 * j & -3 & -14 \end{bmatrix} \tag{6.49}$$

$$K = \begin{bmatrix} -1{,}0658 & -32{,}9598 & -1{,}4715 & -3{,}0014 \end{bmatrix} \quad K_i = -0{,}5794 \tag{6.50}$$

Donde K_i es la ganancia integral del controlador. Hay un polo adicional con respecto al caso anterior debido a la inclusión del integrador. Los resultados del trabajo con este controlador se ven en la figura 6.8.

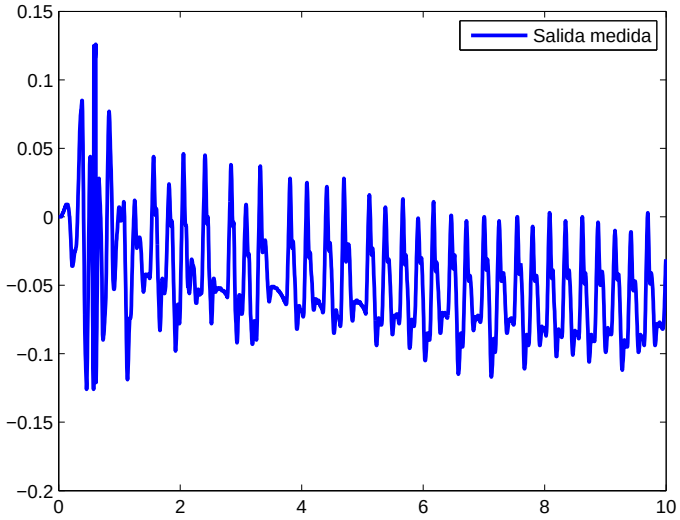


Figura 6.8: Respuesta con el PI vectorial

En las mediciones es notorio el comportamiento oscilatorio, pero es siempre cercano a cero, lo que implica la estabilización del sistema. Nuevamente la oscilación no supera los ocho grados manteniendo el equilibrio del Segway. Adicionalmente, la inclusión del efecto integral provoca la estabilización de la posición, disminuyendo su movimiento a apenas un par de centímetros, como se ve en la figura 6.9.

Para el último experimento de control, se decidió utilizar el sensor ultrasónico para medir la distancia del Segway a cualquier obstáculo (30 cms). La idea es controlar la posición lineal del sistema. Con este fin, se implementó la siguiente ley de control proporcional:

$$u_{dist} = k_p * (ref_{dist} - y_{dist}) \tag{6.51}$$

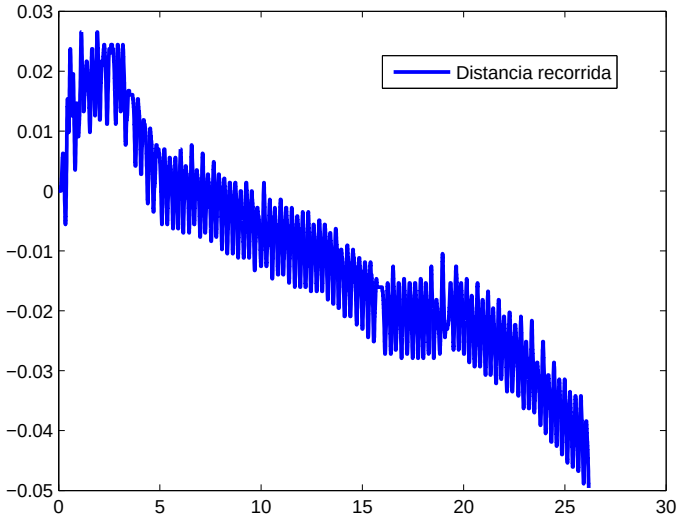


Figura 6.9: Distancia recorrida por el móvil

La señal de control generada es adicionada a la señal de control total del sistema, por lo que se le considera una perturbación a la entrada del sistema, permitindo probar la capacidad de rechazo a perturbaciones del PI vectorial. Se utilizó un $k_p = 5$. La respuesta del sistema en posición lineal se muestra en la figura 6.10.

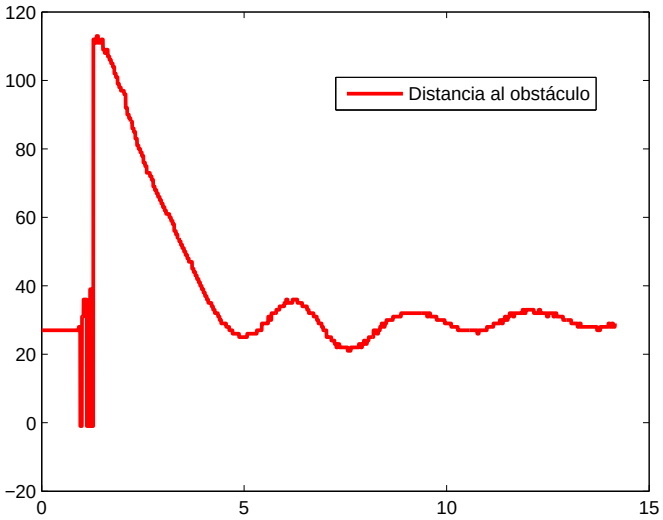


Figura 6.10: Distancia a un obstáculo

Los resultados muestran que el Segway effectivamente persigue al ostáculo hasta situarse a una distancia fija del mismo, con cierta oscilación. En la figura 6.11 se observa que el controlador PI vectorial mantiene en equilibrio al sistema, comprobando su capacidad de rechazar perturbaciones.

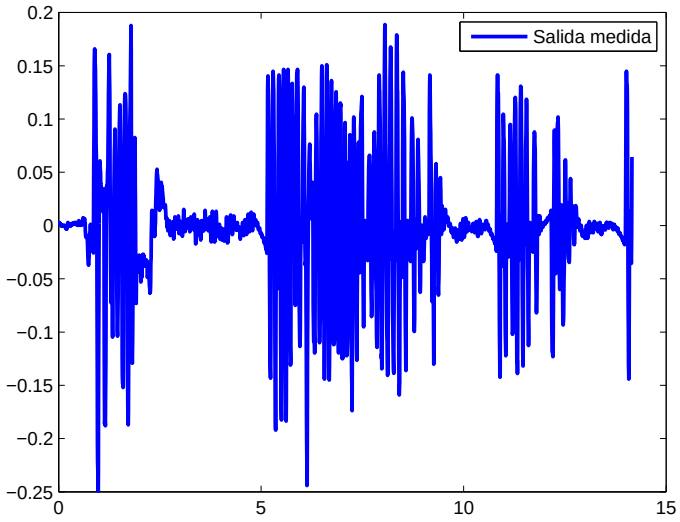


Figura 6.11: Ángulo de Inclinación

6.4 Conclusiones

El segway constituye un ejemplo muy interesante se sistema inestable, que requiere de estrategias de control que garanticen su funcionamiento alrededor de un punto de equilibrio. La tarea de modelado del mismo es una labor que necesita de la aplicación de un buen número conocimientos propios de la física, haciéndola muy cautivadora.

Los controladores desarrollados lograron el objetivo de estabilizar la planta, e incluso resolver tareas adicionales como mantener el sistema a una distancia fija de un obstáculo, con un desempeño muy aceptable. El Segway representa la oportunidad propicia para aplicar técnicas de control en variables de estado, algo muy interesante y de suma importancia, ya que no siempre es posible mostrar estos conceptos mediante explicaciones teóricas o simulaciones.

El sistema es ideal para realizar prácticas y proyectos avanzados de control, dado que no se trata de un sistema trivial y cuyo control requiere de especial atención.

6. Sistema de Balance: Segway

7
Bola y Viga

El sistema de la Bola y la Viga (Ball & Beam) es uno de los modelos de laboratorio más populares e importantes en la enseñanza del control automático en ingeniería. Es ampliamente usado por tratarse de un sistema muy fácil de entender, y sin embargo, la gran cantidad de técnicas de control que permite estudiar cubren una porción valiosa de los métodos de diseño clásicos y modernos existentes. La Bola y la Viga tiene una propiedad particular, es inestable en lazo abierto.

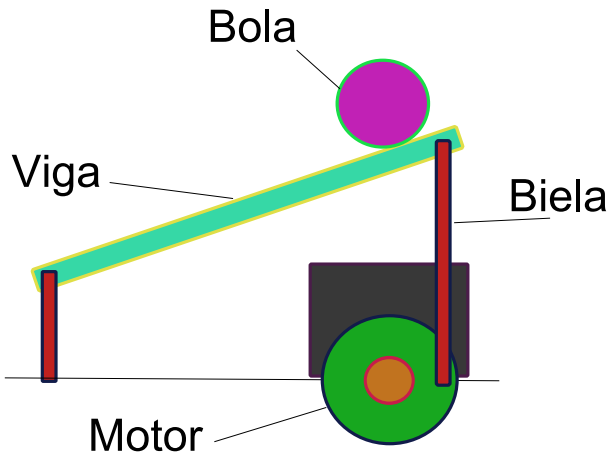


Figura 7.1: Esquema del Ball & Beam

El sistema es muy simple (fig. 7.1); se trata de una esfera rodando sobre una viga o riel. La viga se encuentra conectada mediante una biela a un motor, el cual permite cambiar la inclinación de la misma con respecto a la horizontal. Ésto provoca que la esfera se mueva, cambiando su posición, lo que puede ser detectado o medido con sensores especiales.

La tarea de control se reduce a regular la posición de la bola sobre la viga modificando el ángulo de inclinación de esta última. Se trata de una labor complicada debido a que la bola no se mantiene

en un lugar fijo, sino que se mueve con una aceleración proporcional a la inclinación de la viga. De acuerdo a la teoría de sistemas dinámicos, la posición de la esfera es inestable, dado que se incrementa sin límite alguno para una entrada fija, representada por el ángulo de la viga. Es necesario realimentar el sistema para mantener la bola en un aposición fija.

7.1 Importancia de la Bola y la Viga

Una cantidad considerable de los problemas de control a los que se pueda enfrentar un ingeniero son relativamente sencillos de resolver. Para entradas fijas, la salida se mantiene relativamente constante. Sin embargo, hay sistemas que por su naturaleza o diseño son inestables, y es necesario que el control de lazo cerrado garantice su operación de forma segura. Muchos sistemas tecnológicos o procesos industriales son inestables y que no podrían ser usados sin la estabilización del control realimentado.

En la vida práctica se pueden encontrar ejemplos de sistemas inestables, entre los que podemos nombrar[29]:

- **Procesos químicos industriales:** cuando se quiere controlar reacciones químicas exotérmicas, la generación de calor provoca que la reacción se acelere a medida que la temperatra aumenta. Este hecho hace necesaria la implementación de una estrategia de control que estabilice la temperatura y evite una reacción de escape rápido (que pueden resultar en explosiones). Las reaciones exotérmicas están presentes en la produción de una gran cantidad de productos usados de forma cotidiana, que sin la existencia del control realimentado no estarían disponibles para nuestro uso.

- **Generación de energía:** algunos de los sistemas experimentales de generación energética, requieren de la manipulación de plasma. La posición de esta sustancia debe ser controlada mediante la manipulación de campos magneticos a su alrededor con gran precisión, ya que debe ser impactada con sistemas de laser o similares para provocar la producción de energía. El problema radica en el movimiento inestable del plasma dentro de los campos magnéticos. Sin el control adecuado, este tipo de proyectos podrían no ser útiles.

- **Sistemas aeroespaciales:** muchos sistemas aeroespaciales autónomos, como cohetes o misiles, necesitan de la corrección constante de su trayectoria mediante la modificación del ángulo de sus aletas o alerones, para evitar que caigan, se desvien o vuelquen. Sin la realimentación para estabilizar sus movimientos, tal vez no existiría la exploración espacial o los aviones modernos.

El control de sistemas inestables es una tarea fundamental dentro de las tareas de control más difíciles y debe ser estudiado en el laboratorio [29]. Por obvias razones, no es posible traer los sistemas

inestables al laboratorio, por lo que sistemas como la Bola y la Viga o el péndulo invertido han sido desarrollados; permiten resolver la paradoja de tener sistemas inestables en los laboratorios, pero sin los riesgos o complicaciones de los verdaderos sistemas. Se trata de un dispositivo simple y seguro que tiene la dinámica de un sistema inestable.

7.2 Sistema Construido

El sistema construido trata de reproducir lo más fielmente posible el diagrama mostrado en la figura 7.1. El sistema motriz tiene acoplado una biela que le permite cambiar la inclinación de la viga. Para medir la posición de la bola, se usan los sensores infrarrojos EOPD (ver capítulo 2). Estos sensores cuentan con la capacidad de realizar detección de obstáculos y medición de distancias, sin tener mayor influencia de las fuentes externas de luz. La construcción final del dispositivo, se muestra en la imagen 7.2.

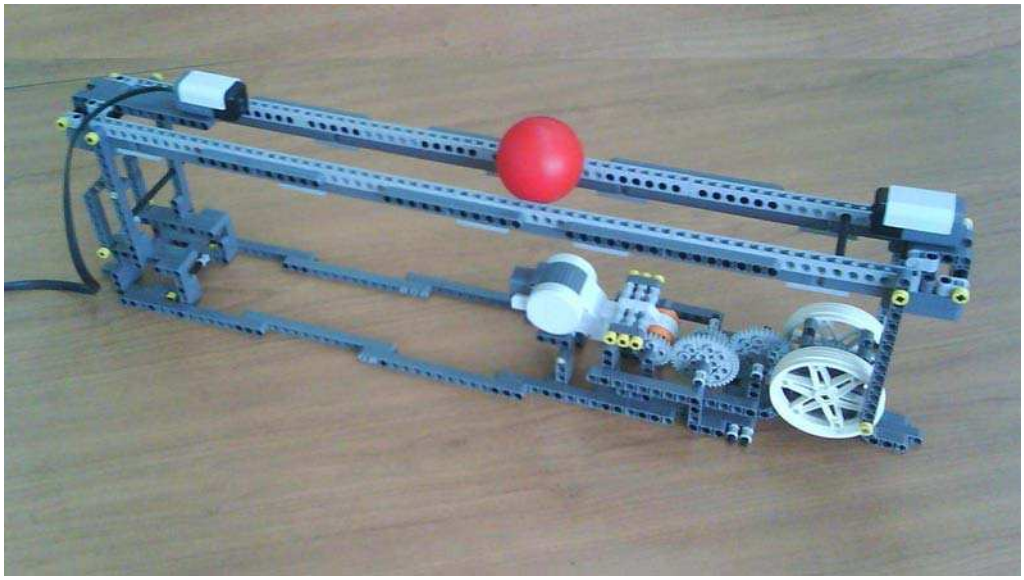


Figura 7.2: Modelo del Ball & Beam

La ecuación que relaciona el giro de la biela con el giro del motor está dada por la relación de reducción en los engranes que acoplan estas partes del sistema.

$$\frac{\theta_{motor}}{\theta_{biela}} = -\frac{12}{40}\frac{12}{40} = -0{,}36$$
$$\theta_{motor} = -0{,}36\theta_{biela} \tag{7.1}$$

Esta relación es útil para calcular el ángulo que debe moverse el actuador (motor) y de esta manera generar la señal de control adecuada.

7.3 Caracterización de Sensores

En general, los sensores que vienen incluidos en los paquetes de Lego Mindstorms y los ofrecidos por la compañía HiTechnic, no requieren de caracterización, calibración o linealización alguna, ya que no es necesaria por su contrucción física o porque las funciones de lectura que están incorporadas en los lenguajes de programación para Lego ya realizan esas tareas. Sin embargo, el NXT OSEK no cuenta con una función propia para la lectura del EOPD, por lo que fue inevitable realizar las correpondientes pruebas para con los sensores y desarrollar la función de lectura apropiada.

En primer lugar, se tomaron varias medidas de cada sensor con la bola roja del kit de Lego Mindstorms. Las lecturas se realizaron cada 5 milímetros y fueron registradas tal como se observa el las imágenes 7.3 y 7.4.

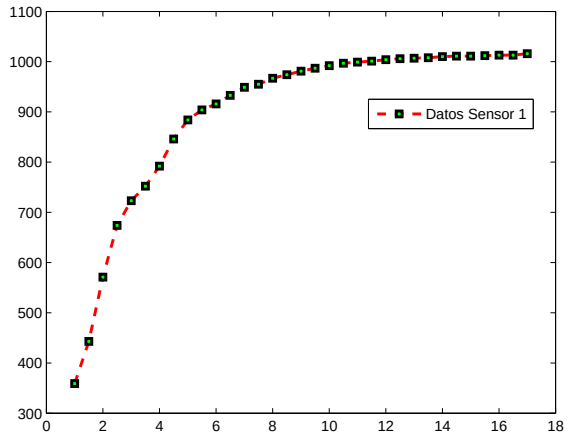


Figura 7.3: Datos sensor 1

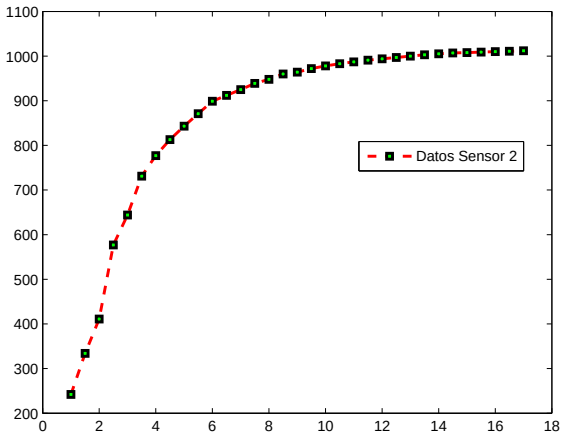


Figura 7.4: Datos sensor 2

Posteriormente, se buscó ajustar cada una de las curvas mostradas a un conjunto de funciones lineales que permitieran su lectura en una medida apta para el trabajo de los controladores. Cada curva se dividió en 4 partes, que fuesen cada una fácilmente ajustables a una recta. La forma de cada ecuación es:

$$y = ax + b \tag{7.2}$$

Los valores a y b para cada recta fueron encontrados usando las diferencias entre valores de medidos (y) y las diferencias entre las correspondientes posiciones (x). Los coeficientes para cada recta se muestran en la tabla 7.1.

	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8
a	214.6	79.0	18.3	2.6549	202.3429	65.8286	19.8333	4.7143
b	136.2	480.4286	815.1556	971.2088	34.5619	509.6476	785.4444	936.0238

Tabla 7.1: Constantes para cada función lineal

Con estas ecuaciones, es posible implementar las funciones en el Brick para la lectura de los sensores. Una vez que la bola entre en el rango de lectura de cualquiera de los sensores, el programa verifica en que rango de medidas se encuentra el valor obtenido, y devuelve el valor de la posición, mediante la ecuación:

$$x = \frac{y - b}{a} \tag{7.3}$$

Para el primer sensor, el valor de salida es directo; una vez la bola salga de su rango de trabajo y entre a la del sensor dos, la salida es igual a la longitud de la viga menos el valor medido. El ajuste de cada polinomio a las curvas se ve en las figuras 7.5 y 7.6.

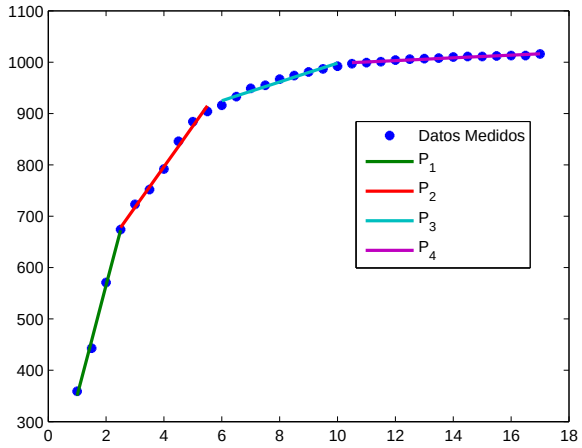


Figura 7.5: Polinomios para el sensor 1

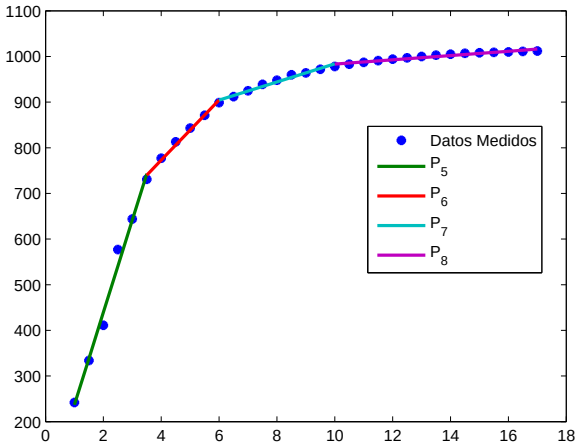


Figura 7.6: Polinomios para el sensor 2

7.4 Modelo del Sistema

Para obtener el modelo del sistema, se recurre a la mecánica de Lagrange, entre diversos métodos disponibles. Las coordenadas generalizadas de sistema son d, el desplazamiento de la esfera y α el ángulo de inclinación de la viga con respecto a la horizontal. Éstas se pueden observar en la figura 7.7. En la tabla 7.2 se listan los parámetros medidos y obtenidos de la literatura del sistema de bola

g=9.81	$[m/seg^2]$	Aceleración de la gravedad
m=0.013	[Kg]	Masa de bola
R=0.0255	[m]	Radio de la bola
R_i=0.0215	[m]	Radio interno de la pelota
$I_b = \frac{2(R^5-R_i^5)}{5(R^3-R_i^3)}m$	$[Kgm^2]$	Momento de inercia de la bola
r=0.025	[m]	Radio de la biela
r_e=0.013	[m]	Radio efectivo de giro de la bola
L=0.385	[m]	Longitud de la viga

Tabla 7.2: Parámetros físicos del Ball & Beam

y viga, y que son necesarios para el modelo.

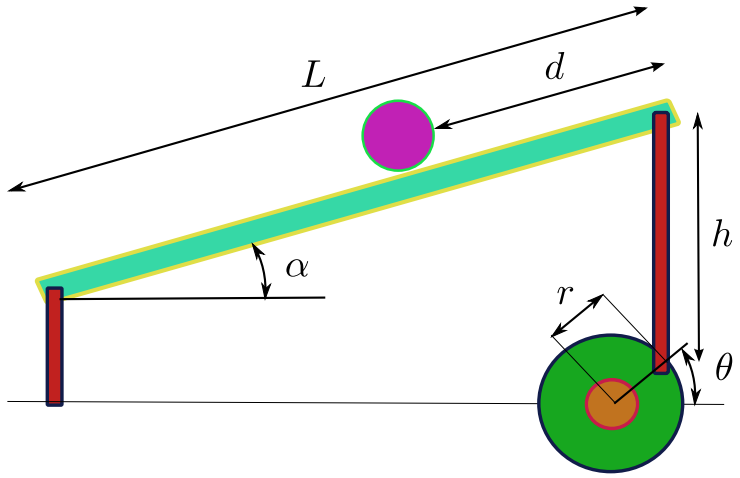


Figura 7.7: Modelo del Ball & Beam

7.5 Ecuaciones de Movimiento

Inicialmente, se calcula la energía cinética del sistema:

$$T = \frac{1}{2}mv^2 + \frac{1}{2}I_b\omega^2 + \frac{1}{2}I_v\dot{\alpha}^2 \tag{7.4}$$

donde I_b es el momento de inercia de la bola, m la masa de la misma, I_v es el momento de inercia de la viga, ω es la velocidad angular de la pelota y v su velocidad lineal.

Dado que los sensores son capaces de medir la posición traslacional de la pelota d y el ángulo del actuador θ, es aconsejable dejar el modelo en términos de dichas variables. Con la mencionada finalidad, se puede hallar la relación entre d, ω y la distancia entre el punto de contacto de la esfera con la viga hasta su centro de gravedad, a la que denominaremos r. Se puede notar entonces que la distancia recorrida por la pelota estará dada por:

$$d = r_e \phi \tag{7.5}$$

donde ϕ es el ángulo de rotación de la esfera con respecto al punto de contacto con la viga y r_e es la distancia del centro de gravedad de la bola al punto de contancto con la viga (radio efectivo de giro). El ángulo de giro total de la bola es la suma de ϕ y α. La velocidad rotacional está dada entonces por:

$$\omega = \dot{\phi} + \dot{\alpha} = \frac{\dot{d}}{r_e} + \dot{\alpha} \tag{7.6}$$

La velocidad lineal de la pelota se puede tomar como:

$$v = \sqrt{\dot{d}^2 + (d\dot{\alpha}^2)} \tag{7.7}$$

La expresión para el Lagrangiano se escribe entonces:

$$L = \frac{1}{2}\left\{ m\left[\dot{d}^2 + (d\dot{\alpha}^2)\right] + I_b\left[\frac{\dot{d}}{r_e} + \dot{\alpha}\right] + I_v\dot{\alpha}^2 \right\} \tag{7.8}$$

Entonces, se define la ecuación de Lagrange para el sistema:

$$\frac{d}{dt}\left(\frac{\partial}{\partial d}L\right) - \frac{\partial}{\partial d}L = mg\sin(\alpha) \tag{7.9}$$

Al evaluar la expresión del Lagrangiano, se tiene la ecuación de movimiento del sistema

$$\left(m + \frac{I_b}{r_e^2}\right)\ddot{d} + \left(\frac{I_b}{r_e^2}\right)\ddot{\alpha} - md\dot{\theta}^2 = mg\sin(\alpha) \tag{7.10}$$

Para obtener la función de transferencia del sistema, es necesario linealizar la ecuación anterior alrededor de un punto de equilibrio. Si se asume que los cambios en el ángulo de inclinación de la viga son pequeños ($\ddot{\alpha} \to 0$) y que los términos de cuadráticos son despreciables ($\dot{\alpha}^2 \to 0$), se llega a la siguiente expresión:

$$\left(m + \frac{I_b}{r_e^2}\right)\ddot{d} = mg\alpha \tag{7.11}$$

Teniendo en cuenta la relación entre el ángulo de inlinación de la viga y el ángulo de la biela:

$$\alpha = \frac{r}{L}\theta_{biela} \tag{7.12}$$

y reemplazando, se llega a la siguiente función:

$$\left(m + \frac{I_b}{r_e^2}\right)\ddot{d} = mg\frac{r}{L}\theta_{biela} \tag{7.13}$$

Aplicando transformada de Laplace, se tiene la función de transferencia del sistema:

$$G(s) = \frac{D(s)}{\Theta_{biela}} = \frac{mgr}{\left(m + \frac{I_b}{r_e^2}\right)Ls^2} \tag{7.14}$$

Sin embargo, se debe tener en cuenta que el sistema finalmente es accionado por el motor, por lo que reemplazando por la expresión que relaciona el ángulo del motor con el devla biela, llegamos al modelo final del sistema:

$$G(s) = \frac{D(s)}{\Theta_{motor}} = \frac{-mgr}{0{,}36\left(m + \frac{I_b}{r_e^2}\right)Ls^2} \tag{7.15}$$

7.6 Modelo numérico del Sistema

Reemplazando los valores de parámetros de la tabla 7.2, se encuentra el modelo numérico del sistema:

$$G_{num}(s) = \frac{-0{,}5521}{s^2} \tag{7.16}$$

Este modelo es el utilizado en la siguiente sección para la realización de los experimentos de control. Cabe anotar que se trata de un modelo marginalmente estable (doble integrador), que requiere de estrategias de control acertadas para garantizar un comportamiento estable y un desempeño aceptable, haciéndolo muy apropiado como reto de control.

7.7 Experimentos de Control

Para el primer experimento, se diseñó un controlador de adelanto, con función de transferencia dada por:

$$C(s) = \frac{-1{,}0865(s + 0{,}2017)}{(s + 0{,}7773)} \tag{7.17}$$

Los resultados de utilizar dicho controlador se muestran en la figura 7.8 (ref=200mm). La salida muestra un comportamiento oscilatorio para ambos sistemas, que es más notorio en el sistema real. No obstante, los tiempos de establecimiento y el estado estable alcanzado son similares, mostrando que la dinámica del modelo obtenido es cercana a la del sistema real. Además, se logra tener un buen desempeño del sistema y lo más importante, consigue estabilizar la planta de acuerdo a los resultados simulados.

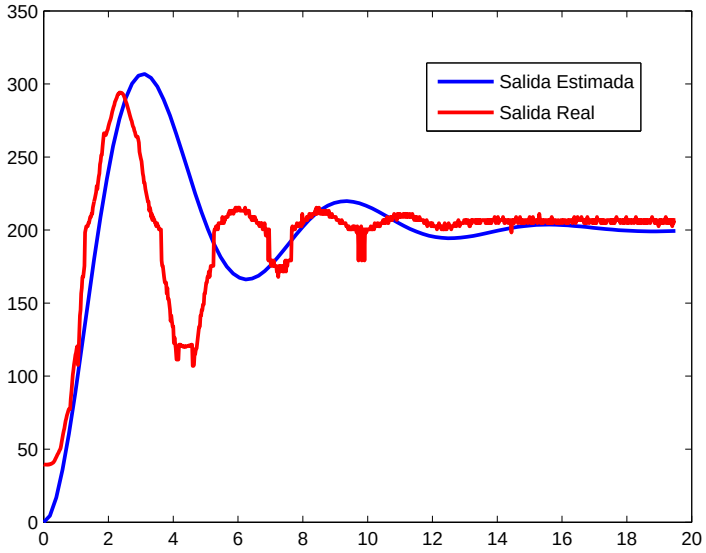


Figura 7.8: Salida con controlador de adelanto

Más adelante se sintetizó un controlador PID (Proporcional Integral Derivativo) dado por la expresión:

$$C(s) = \frac{-20{,}5391(s^2 + 0{,}8067s + 0{,}1631)}{s(s + 5{,}637)} \tag{7.18}$$

En la imágen 7.9 se observa la prueba realizada con el controlador PID. Tal como en el caso anterior, el sistema tiene un comportamiento oscilatorio con frecuancia más alta de la esperada, pero

con un comportamiento en estado estable aceptable, y con un sobrepico menor al simulado, lo cual es un indicio de un desempeño aceptable.

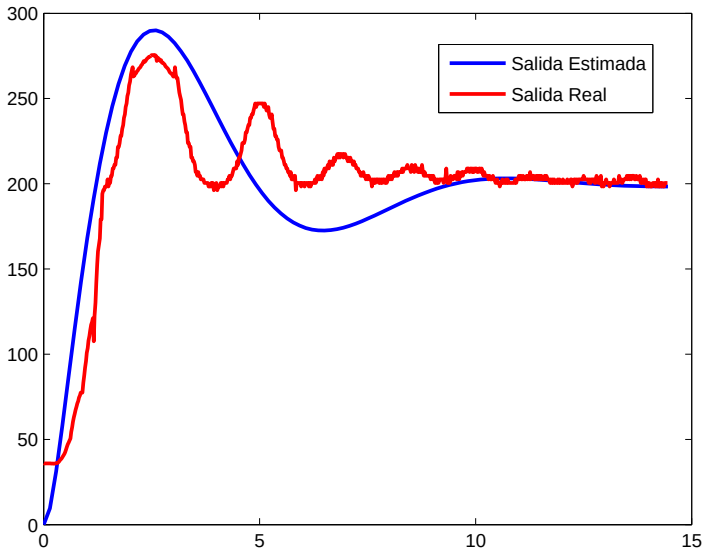


Figura 7.9: Salida con controlador PID

Luego, fue diseñado un controlador algebraico por realimentación unitaria que colocara los polos del sistema en lazo cerrado en $s = -1$, -2 y -3. La función de transferencia para el controlador está dada por:

$$C(s) = \frac{-19{,}79s - 10{,}79}{s + 6} \tag{7.19}$$

Los resultados mostrados el la figura 7.10, indican nuevamente un sobrepico inferior y un tiempo de establecimiento pequeño. El desempeño del sistema con esta estrategia de control es aceptable, y aunque se observa cierta oscilación en el estado estable, su amplitud es apenas perceptible y no modifica en mayor medida el comportamiento del sistema.

En el siguiente experimento, fue probado un controlador algebraico de dos parámetros, que implementara la función de lazo cerrado:

$$T(s) = \frac{1}{s^2 + 1{,}505s + 1} \tag{7.20}$$

con el polinomio observador $p_o(s) = s + 1$. La expresión para el controlador resultante es:

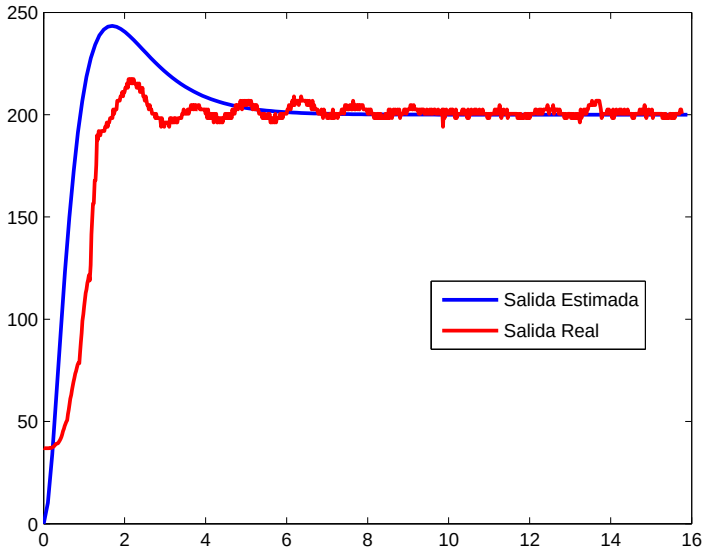


Figura 7.10: Salida con controlador algebarico

$$C(s) = \left[\frac{-1{,}799s-1{,}799}{s+2{,}505} \quad \frac{4{,}506s+1{,}799}{s+2{,}505}\right] \tag{7.21}$$

La imagen 7.11 muestra la respuesta del sistema con el controlador diseñado. Es notable la velocidad de respuesta del sistema, con un tiempo de subida y establecimiento inferior al simulado. El controlador permite a la planta alcanzar un estado estable similar al del sistema simulado, con poca oscilación. La oscilación es muy baja y el desempeño en términos de velocidad y seguimiento de la referencia es bueno.

Para la última prueba de control, se utilizó el mismo controlador del experimento anterior, ahora con una referencia cambiante. Consitió en conectar un motor adicional al sistema, cuyo encoder actuaría como referencia para el sistema, los grados leidos, serían ahora los milímetros que debería moverse la bola. Los resultados se observan en la imagen 7.12.

Es claro que el cambio continuo de referencias provoca oscilaciones en la respuesta del sistema. También se nota que pese a las vibraciones, el sistema logra seguir las referencias, otorgando al sistema la capacidad de actualizar dinámicamente el valor que debe alcanzar.

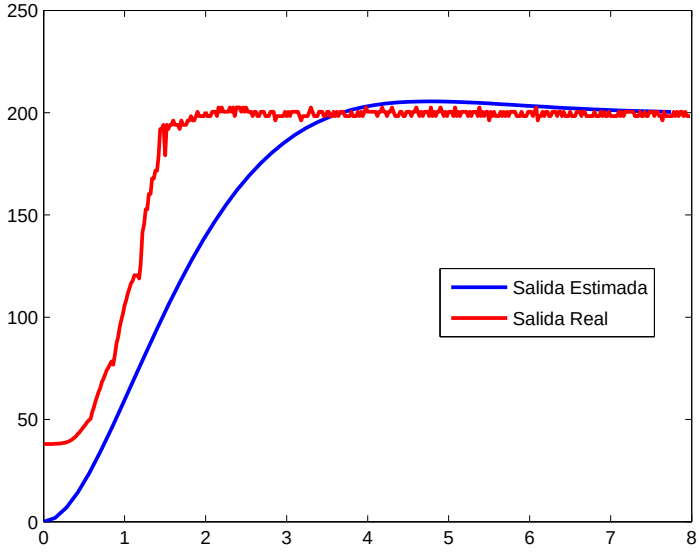


Figura 7.11: Salida con controlador de dos parámetros

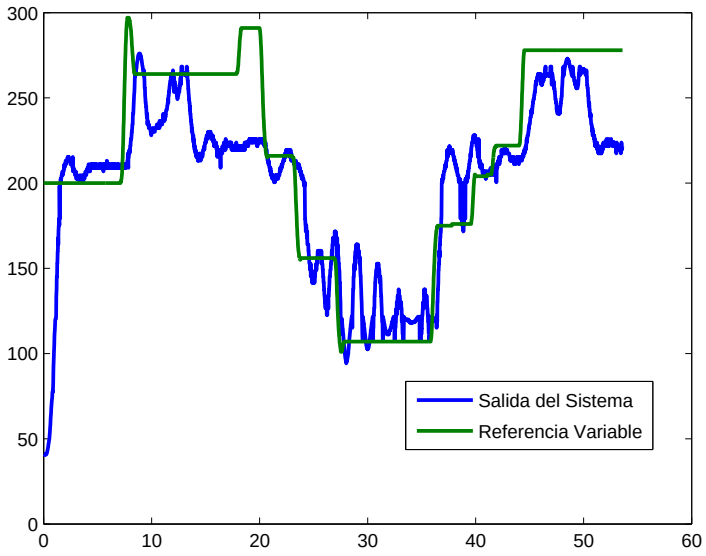


Figura 7.12: Salida con referencia variable

7.8 Conclusiones

Los sistemas inestables o marginalmente estables como el bola y viga mostrado, constituyen un reto interesante de control, dado que las estrategias de control diseñadas e implementadas, tienen que

garantizar antes que desempeño, la estabilización del sistema, por lo que su inclusión en los cursos de control es muy importante, interesante, y tal vez, indispensable.

La obtención de modelos para este sistema requiere de la medición y observación de parámetros físicos, cuya consecución es relativamente sencilla, permitiendo realizar de forma rápida pruebas y modificaciones en los parámetros de la planta.

Los controladores obtenidos permiten estabilizar el ball & beam y adicionalmente, conseguir un desempeño aceptable en cuanto a seguimiento de referencias, tiempos de subida y de establecimiento, por lo que las pruebas de control que se pueden realizar son diversas e interesantes. La variedad de controladores obtenidos, garantiza que el sistema pueda ser incluido en un curso de control como una práctica útil y muy enriquecedora, que puede ser usada sin problemas a lo largo de un plan curricular dedicado a la enseñanza de control automático.

7. Bola y Viga

8
Sistema de Acoples Flexibles o Elásticos

Los sistemas de transmisión mecánica son una de las aplicaciones más comunes dentro de los mecanismos. Permiten transformar y transmitir movimiento (energía mecánica) desde un punto motriz hasta un elemento impulsado, formando lo que se conoce como una cadena cinemática. Es común encontrarlos en prácticamente todos los mecanismos industriales existentes, y casi siempre requieren de un control preciso para regular las variables que implican su movimiento.

Los sistemas de transmisión o acople pueden ser fijos, separados por accionamiento (embragues) o auto separables (de seguridad) [38]. Los fijos requieren el desmontaje total o parcial del acople para su conexión o desconexión y pueden ser Rígidos (espárragos, chavetas, conos, cilíndros), que requieren alineación estricta entre las parte móviles e impulsada, Flexibles (cadenas, cárdanes), que permiten transmitir potencia, tolerando desalineaciones, y Elásticos (bandas) que absorben desalineaciones, variaciones torsionales y vibraciones.

Este capítulo se centra en el control de posición de un sistema de transmisión por acoples flexibles o elásticos. Los sistemas elásticos tienen la habilidad de transmitir potencia mecánica aún en presencia de desalineaciones angulares y axiales, y vibraciones, por lo que es muy común hallarlos en numerosos dispositivos mecánicos. Sin embargo, estos sistemas tienen modos vibracionales propios por efecto de su naturaleza elástica, lo que puede dificultar su control preciso. Además, en presencia de variaciones de la carga impulsada, el modelo puede tener cambios significativos en su dinámica y comportamiento. Esta razón convierte a los sistemas flexibles en un interesante reto de modelado y control.

Las mencionadas variaciones dinámicas han hecho que sea muy común encontrarlo en diversos libros de texto [3] (ver figura 8.1) y publicaciones científicas donde se mustran diversas técnicas de control avanzado [33].

No obstante su modelo matemático es relativamente fácil de obtener, los parámetros no son necesariamente triviales de medir, por lo que es usual que sea usado en experimentos de identificación de sistemas. En este capítulo se describirá un procedimiento de identificación paramétrica para conocer

Figura 8.1: Sistema de acoples Flexibles[a]

[a] Tomada de [3]

el modelo del dicho sistema.

8.1 Sistema construido

El aparato que aparece en la figura 8.1 es la base para la construcción del sistema aquí presentado. En ese sistema un motor DC servocontrolado está conectado a una polea, la cual está sujeta por una banda a una segunda polea. Ésta a su vez se conecta mediante una segunda banda con una tercer polea, y el objetivo es controlar la posción angular de esta última. Para su armado se utilizaron motores y sensores de la versión RCX del Lego Mindstorms, debido a su disponibilidad en el material presente en el Laboratorio de Control de la Universidad Nacional de Colombia. Adicionalmente, los motores de la mencionada versión, manejan un menor torque, por lo que es más fácil transmitir movimiento mediante las bandas elásticas del kit de Lego. El dispositivo se ve en la figura 8.2.

En el sistema original[3], el motor cuenta con un lazo cerrado para controlar su posición angular, y se considera el ángulo de la primer polea θ_1 como a entrada al sistema. Aquí se considerará el sistema como un todo, y la señal de entrada será el voltaje en el motor. La salida seguirá siendo θ_3 (ver siguiente sección).

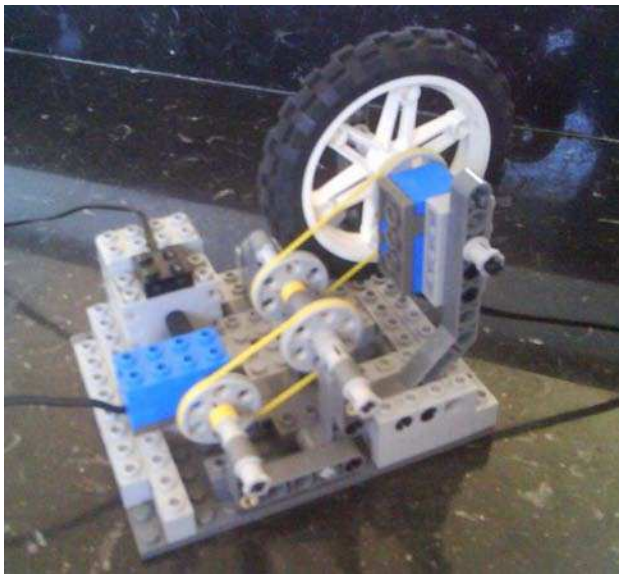


Figura 8.2: Sistema de acoples Flexibles contruido

8.2 Modelo del Sistema

Para modelar el sistema se tiene en cuenta el esquema que aparece en la figura 8.3. Se tienen en cuenta las inercias de las 3 poleas o ejes y sus respectivos rozamientos. Para la definición de las ecuaciones de movimiento, además se considerarán los radios de las poleas (son iguales) y las constantes elásticas de las bandas. Los parámetros y variables del modelos se listan en la tabla 8.1.

$\theta_1,\ \theta_2,\ \theta_3$	[rad]	Ángulo de giro de cada una de las poleas
$J_1,\ J_2,\ J_3$	$[Kgm^2]$	Inercia de cada una de las poleas
r	[m]	Radio de las poleas
$k_1,\ k_2$	$[N/m]$	Constante elástica de las bandas

Tabla 8.1: Parámetros físicos y variables del sistema de acoples flexibles

8.3 Ecuaciones de Movimiento

En el modelo original, se toma θ_1 como entrada al sistema. En el sistema aquí mostrado, la entrada es el voltaje de alimentación del motor. Por lo tanto, la función de transferencia que relaciona el voltaje con la dinámica de θ_1 es:

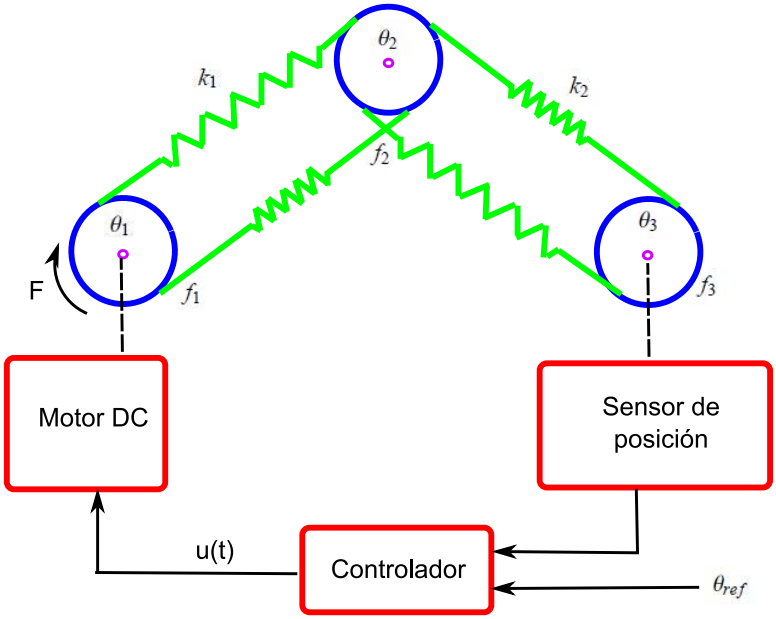


Figura 8.3: Esquema: Sistema de acoples Flexibles

$$G_m(s) = \frac{\theta_1(s)}{V(s)} = \frac{K_m}{s(s + p_e)(s + p_m)} \tag{8.1}$$

Donde p_e y p_m son los polos debidos a la parte mecánica y eléctrica del motor respectivamente. Si se asumen los polos como suficientemente rápidos con respecto a la dinámica del sistema total, se puede reescribir la función de transferencia como:

$$G_m(s) = \frac{K_{mr}}{s} \quad \rightarrow \quad \dot{\theta}_1 = K_{mr}V(t) \quad \rightarrow \quad \theta_1 = \int K_{mr}V(t)dt \tag{8.2}$$

De las ecuaciones anteriores se puede ver que en estado estable ($\dot{\theta}_1 = 0$), el voltaje de entrada al motor es cero, por lo que el área bajo la curva de la función $V(t)$ será una constante K_f. Entonces, para un sistema estable y con una dinámica lo suficientemente rápida (motor), podemos escribir:

$$\theta_{1f} = K_f K_{mr} = K \quad \rightarrow \quad \theta_1 = KV(t) \tag{8.3}$$

Para la obtención de las ecuaciones de movimiento, se suman los torques alrededor de cada eje, asumiendo que la dinámica de $theta_1$ es más rápida que la del resto del sistema, así:

$$2k_1r^2(\theta_1 - \theta_2) = F \tag{8.4}$$

$$J_2\ddot{\theta_2} + f_2\dot{\theta_2} + 2k_1r^2(\theta_2 - \theta_1) + 2k_2r^2(\theta_3 - \theta_2) = 0 \tag{8.5}$$

$$J_3\ddot{\theta_3} + f_3\dot{\theta_3} + 2k_2r^2(\theta_3 - \theta_2) = 0 \tag{8.6}$$

Donde F es el torque de entrada al sistema. Con las relaciones obtenidas, y tomando como variables de estado $x = \begin{bmatrix}\theta_2 & \dot{\theta_2} & \theta_3 & \dot{\theta_3}\end{bmatrix}^T$ se llega a la representación en variables de estado:

$$\begin{aligned}\dot{x} &= Ax + Bu \\ y &= Cx\end{aligned} \tag{8.7}$$

Donde:

$$\begin{aligned}A &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ -2r^2\frac{k_1+k_2}{J_2} & -\frac{f_2}{J_2} & \frac{2k_2r^2}{J_2} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{2k_2r^2}{J_3} & -\frac{2k_2r^2}{J_3} & 0 & -\frac{f_3}{J_3} \end{bmatrix} \\ B &= \begin{bmatrix} 0 \\ \frac{2k_1r^2}{J_2} \\ 0 \\ 0 \end{bmatrix}, \qquad C = \begin{bmatrix}0 & 0 & 1 & 0\end{bmatrix}\end{aligned} \tag{8.8}$$

Con $u = \theta_1$. Este modelo coincide con el que aparece en [33]. Usando la relación entre el voltaje del motor (asumiendo que el sistema será estabilizable), podemos escribir la entrada en términos del voltaje como:

$$u = K * V(t) \tag{8.9}$$

Por lo que la matriz B es finalmente:

$$B = \begin{bmatrix} 0 \\ \frac{2k_1r^2}{J_2}K \\ 0 \\ 0 \end{bmatrix} \tag{8.10}$$

Dada la popularidad del sitema como ejemplo en experimentos de identificación y a la dificultad de medir parámetros como las fricciones en las poleas, se presenta en la siguiente un experimento de identificación para obtener el modelo numérico del sistema.

8.4 Identificación del sistema

Para identificar este sistema, se aplicó la señal binaria aleatoria (RBS) que se muestra en la figura 8.4. La señal de entrada es el nivel de PWM (Voltaje) y la salida corresponde al ángulo del eje de la tercer polea (grados).

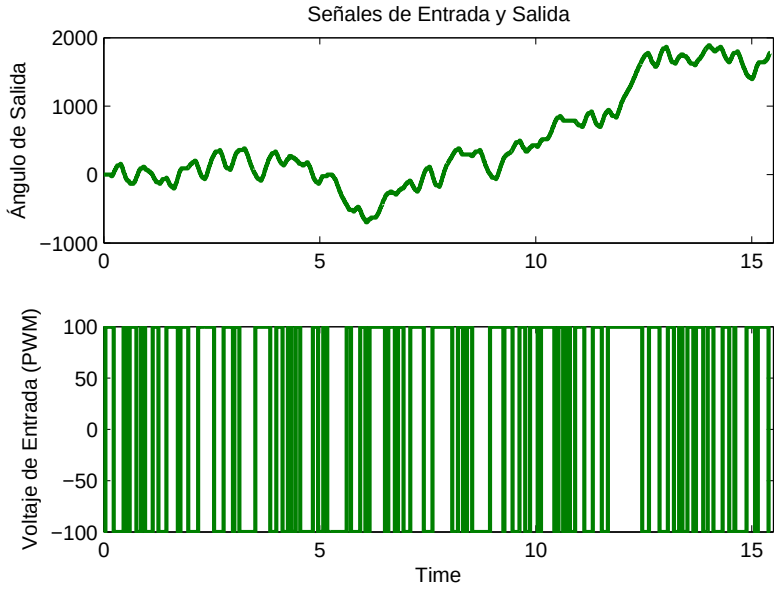


Figura 8.4: Identificación del Sistema

Para aplicar los algoritmos de ajuste por mínimos cuadrados, fueron seleccionadas varias estructuras de tercer y cuarto orden, conocidas como modelos de caja negra[1], y que ya fueron usados en capítulos anteriores para identificar la dinámica del vehículo, mostrando los resultados de la figura 8.5.

Pese al alto grado de ajuste, se decidió probar con una estructura de caja blanca[1], la cual toma el modelo teórico (en variables de estado) y trata de ajustar los datos a una estructura igual a la del mencionado modelo. Los resultados de ajuste se muestran en la figura 8.6.

El sistema que acompaña a la estructura de caja blanca (reducido en 8.6), es una versión reducida del mismo[41], que permite conservar parte de las características dinámicas del sistema, pero con un orden menor haciendo más fácil trabajar con el modelo. Los datos numéricos de ambos sistemas se muestran en la siguiente sección. El nivel de ajuste logrado con ambos sistemas hace que sean los seleccionados para representar al modelo teórico, ya que represntan mejor su dinámica. La forma básica del modelo de caja blanca está dada por la siguiente ecuación:

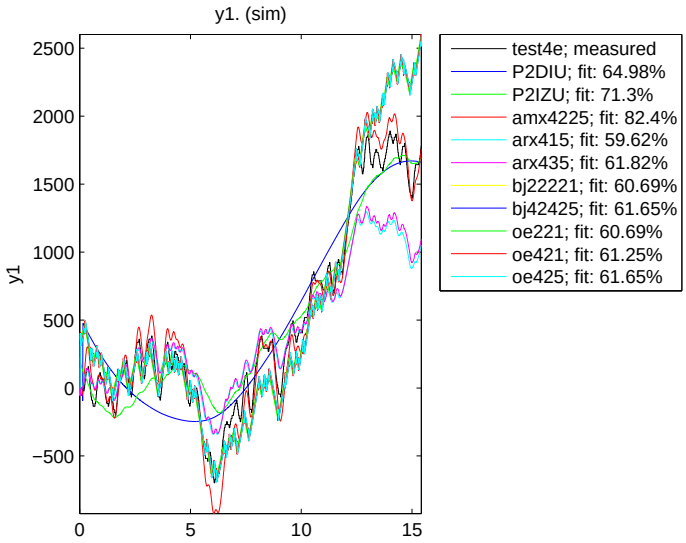


Figura 8.5: Nivel de ajuste con modelos de Caja negra

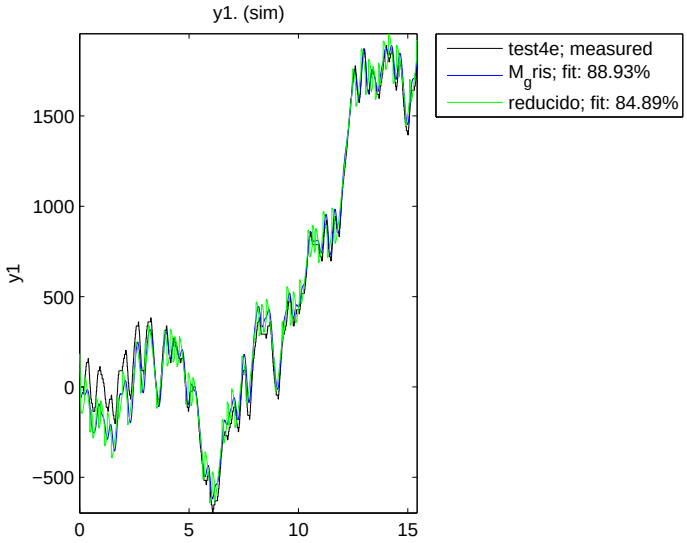


Figura 8.6: Nivel de ajuste con el modelo de Caja Blanca

$$\dot{x} = Ax(t) + Bu(t) + Ke(t) \tag{8.11}$$
$$y(t) = Cx(t) + Du(t) + e(t)$$

En la estructura mostrada se tienen en cuenta los términos K y $e(t)$, debidos a la parte de la

salida que el modelo no puede explicar (error de estimación). En la siguiente sección se muestran sus expresiones numéricas, más no se consideran necesarias para la realización de los controladores, ya que para tal fin solo se requiere de las matrices A y B[5].

8.4.1 Modelo Numérico del Sistema

El modelo numérico del sistema se obtiene al reemplazar los valores numéricos obtenidos durante la identificación en la estructura seleccionada:

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -165{,}05 & 0{,}49613 & 311{,}61 & 0 \\ 0 & 0 & 0 & 1 \\ 163{,}57 & 0 & -308{,}68 & -33{,}946 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 390{,}88 \\ 0 \\ 0 \end{bmatrix} u(t) + \begin{bmatrix} 371{,}81 \\ 54{,}575 \\ 150{,}85 \\ -12392 \end{bmatrix} e(t) \tag{8.12}$$

$$y(t) = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} x(t) + 0u(t) + e(t)$$

Nótese que la estructura original no contempla las matriz K. Ésta hace parte del modelo desviado o porción de la salida que no puede ser explicada por el modelo. En general, hace parte de un sistema paralelo que seencarga de tomar los errores en la salida del sistema, pero que en la práctica, no es usada ya que no hace parte del modelo real, es solo una explicación numérica del error entre la salida real y la simulada. Por tal motivo, no se usará en los experimentos de control. En la tabla 8.2 se muestran los valores obtenidos durante el ajuste de mínimos cuadrados y su equivalencia con el modelo teórico.

$2r^2\frac{k_1+k_2}{J_2}$	165.05	$\frac{2k_2r^2}{J_3}$	163.57
$\frac{f_2}{J_2}$	-0.49613	$\frac{2k_2r^2}{J_3}$	308.68
$\frac{2k_2r^2}{J_2}$	311.61	$\frac{f_3}{J_3}$	33.946
$\frac{2k_1r^2}{J_2}K$	390.88		

Tabla 8.2: Parámetros del sistema

Para faciltar el uso del modelo en las tareas de control, es recomendable utilizar la versión reducida del sistema, cuyas expresión viene dada por:

$$\dot{x}(t) = \begin{bmatrix} 1{,}9457 & 6{,}4917 \\ -5{,}8595 & -19{,}588 \end{bmatrix} x(t) + \begin{bmatrix} -0{,}31016 \\ -0{,}064854 \end{bmatrix} u(t) + \begin{bmatrix} -0{,}37351 \\ -0{,}012714 \end{bmatrix} e(t) \tag{8.13}$$

$$y(t) = \begin{bmatrix} -20{,}171 & 39{,}04 \end{bmatrix} x(t) - 0{,}52984u(t) + e(t) \tag{8.14}$$

Con las mismas consideraciones que se hicieron sobre el modelo completo sobre el vector K. La versión en función de transferencia de la expersión anterior es:

$$G_{num}(s) = \frac{0{,}8362s^2 - 43{,}17s + 654{,}8}{s^2 + 55{,}83s - 0{,}233} \tag{8.15}$$

8.5 Experimentos de Control

El primer experimento de control para este sistema, consistió en el diseño de un controlador PI, con una función de transferencia dada por:

$$C(s) = \frac{0{,}75114(s + 0{,}1461)}{s} \tag{8.16}$$

Los resultados de la prueba realizadas con el contolador PI muestran en la figura 8.7.

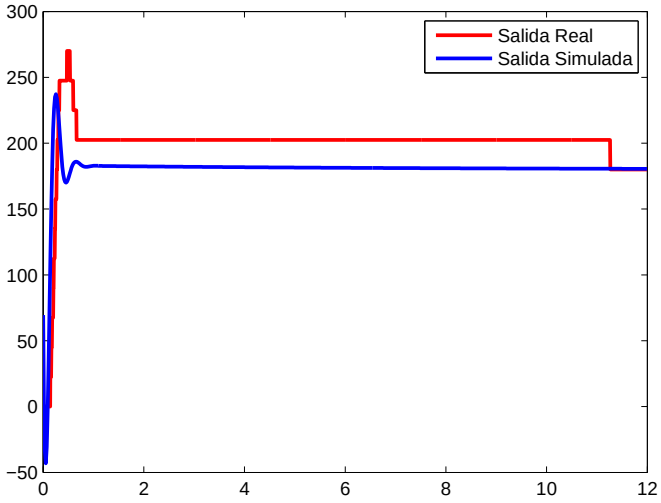


Figura 8.7: Prueba con el controlador PI

Se observa una diferencia entre los valores de sobrepico y tiempo de subida y establecimiento. Esto puede deberse a la poca resolución del sensor de posición utilizado (22.5°) y a la dificultad que existe en controlar los motores del RCX con el Brick NXT. No obstante, el controlador logra llegar a cero error de estado estacionario, la dinámica en general no difiere en forma a la del sistema simulado. La siguiente prueba corresponde a un controlador de atraso dado por la función de transferencia:

$$C(s) = \frac{1{,}386s + 1}{1{,}873s + 1} \tag{8.17}$$

Los resultados pueden verse en la figura 8.8.

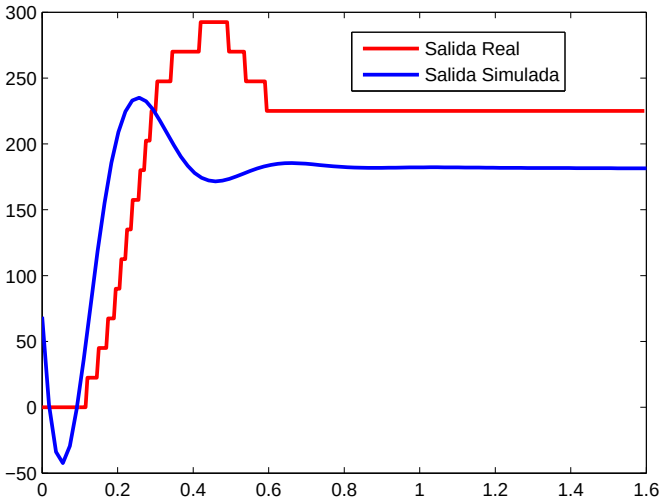


Figura 8.8: Prueba con el controlador de atraso

Es evidente la diferencia en estado estacionario de las dos respuestas. Las razones son las mismas del caso anterior. Adicionalmente, se puede considerar el efecto de la zona muerta del motor, una región en la que el nivel de voltaje no es suficiente para mover el mismo. Finalmente se prueba con un controlador algerbraico por realimentación unitaria, con polos en -1, -20, -30, y -40. La función de transferencia del controlador es:

$$C(s) = \frac{0{,}805s^2 + 43{,}07s + 36{,}65}{0{,}3268s^2 + 71{,}5s} \tag{8.18}$$

En la figura 8.9 se visualiza la prueba realizada al sistema con el mencionado regulador. Los resultados muestran la diferencia en estado transitorio de las dos respuestas. Sin embargo, el estado estacionario muestra resultados similares, lo que valida la eficiencia del controlador diseñado.

8.6 Conclusiones

La planta presentada en este capítulo representa un interesante reto de modelado e identificación de sistemas. Es propicia para la implementación de diversas estrategias de control, con un montaje muy sencillo. Permite utilizar los elementos de versiones anteriores del Lego Mindstorms e incorporarlos a las tareas de control. Un hecho interesante en la utilización de dichos elementos, es la

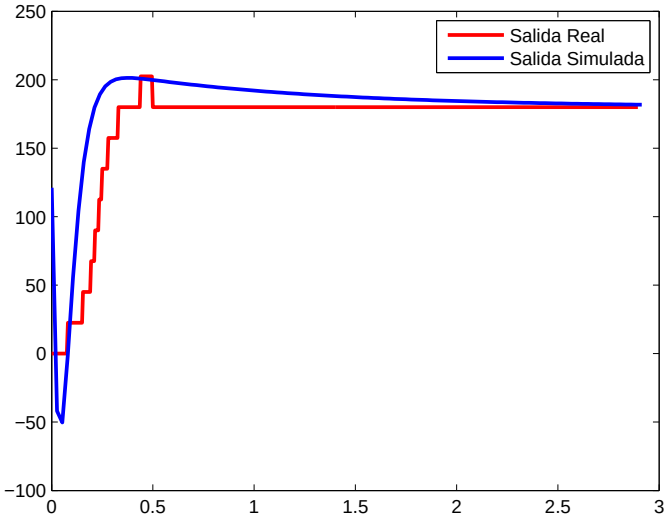


Figura 8.9: Prueba con el controlador algebraico

pérdida de precisión en la labor de control debida a la baja resolución de los sensores utilizados y a la calidad menor del motor empleado como actuador.

Pese a los problemas evidenciados en el desempeño de los sistemas de control, las diferencias no llevan a pensar en resultados poco satisfactorios; por el contrario, las pruebas realizadas reflejan dinámicas similares a la de los modelos teóricos y/o simulados, validando tanto el trabajo de identificación como de simulación.

En el futuro, podrían ser empleados sensores y actuadores de mejor calidad que mejoren aún más el desempeño observado. El sistema permite la implementación y aplicción de diferentes estrategias de control, tanto básicas como avanzadas, facilitando su uso en un laboratorio dedicado a la enseñanza del control automático.

8. Sistema de Acoples Flexibles o Elásticos

9
Software Desarrollado

El software desarrollado en el proyecto contempla dos partes principales, las funciones y librerias realizadas para las diferentes tareas llevadas a cabo sobre el Brick, y la interfaz de usuario, donde se reciben los parámetros de los controladores y se descargan los archivos al procesador. Esta configuración evita que el usuario tenga que involucrarse con detalles complejos de programación y hace más fácil la utilización de la plataforma.

En este capítulo se describirán los modulos realizados y su funcionalidad. También se comentará sobre la metodología utilizada en su proceso de discretización e implementación de los controladores analógicos en el Brick (en lenguaje de alto nivel).

9.1 Implementación de controladores

Una vez el usuario ha diseñado su controlador en el dominio analógico, se requiere de un procedimiento apropiado que le permita introducirlo como una expresión matemática en un lenguaje de alto nivel. El primer paso es la selección del tiempo de muestreo y el método de conversión del modelo continuo a discreto. Debido a que el procesador del Brick puede ofrecer un frecuencia de muestreo para rutinas de usuario de hasta un kilohertz, el tiempo de muestreo para los sistemas diseñados va desde los 4 segundos, como en el caso del segway, hasta los 100 milisegundos en algunas tareas de lectura de sensores. En general, se trabajó con 20 ms para la mayor parte de los experimentos mostrados en este documento.

La selección del tiempo de muestreo también implica conocimientos sobre el ancho de banda del sistema. No puede ser muy bajo, para evitar problemas de aliasing, pero uno muy alto puede degradar el desempeño del sistema [4], por lo que la selección de los 20 milisegundos parece ser apropiada.

Dentro de los métodos de conversión podemos encontrar la invaranza al paso (ZOH), la invarianza a la rampa (FOH) o la aproximación bilineal o de Tustin entre otras[3]. Para el proyecto se usa ésta última, ya que ofrece un nivel de aproximación aceptable, sin incrementar el orden del sistema

y sin permitir la pérdida de las principales características dinámicas del sistema (niveles de ganancia estática, frecuencia de trabajo, etc.). La expresión de la aproximación de Tustin es:

$$s = \frac{2}{T_s}\frac{z-1}{z+1} \tag{9.1}$$

con T_s como tiempo de muestreo. Al reemplazar en la ecuación del controlador, se tiene una expresión discreta del mismo, pero que aún no puede ser implementada directamente como ecuación. Se cuenta con varios procedimientos para llevar a cabo dicha tarea, los cuales conllevan a escribir el controlador como un conjunto de ecuaciones en diferencias. Aquí se utiliza el paso a variables de estado del controlador para tal fin, buscando las matrices A, B, C, y D del mismo. Estas ecuaciones de diferencia en versión de variables de estado, son las utilizada en el proyecto para implementar en lenguaje de alto nivel. Para evitar errores de cuantificación de coeficientes, lo más recomendables es que la realización digital del controlador sea hecha en punto flotante[40].

Uan práctica recomendable en la implementación de controladores en plataformas digitales, es el balanceo de la realización. Esta consiste en llevar la representación del sistema a una forma que garantice que sus coeficientes tienen órdenes de magnitud similares, y de esta forma evitar el desbordamiento de variables o el truncamiento de datos. Es un paso opcional, pero muy deseable. El procedimiento de balanceo utilizado se describe en [42].

9.1.1 Ejemplo de Discretización e implementación

Para el controlador diseñado algebraico por realimentación unitaria que aparece en el capítulo 7, se obtuvo la expresión:

$$C(s) = \frac{-19{,}79s - 10{,}79}{s+6} \tag{9.2}$$

Con un tiempo de muestreo de 20 ms y utilizando la expresión para la transformación bilineal se obtiene:

$$C_d(z) = \frac{-18{,}77z + 18{,}57}{z - 0{,}8868} \tag{9.3}$$

Ahora, se procede al paso a la realización balanceada en variables de estado discretas:

$$\begin{aligned} x_1[k] &= 0{,}8868x_1[k-1] + 1{,}386e[k-1] \\ u[k] &= 1{,}386 * x_1[k-1] - 18{,}77e[k-1] \end{aligned} \tag{9.4}$$

con $e[k]$ la señal de error, $u[k]$, la señal de control y $x_1[k]$, la variable de estado del controlador. Esta expresión, ya se puede escribir en forma de lenguaje de alto nivel. Todas las funciones descritas, son implementadas en GCC para NXT, y basados en el lenguaje NXT-Osek. No obstante, su escritura corresponde a un proceso automático en el cual el usuario no interviene, ya que es levada a cabo por la interfaz de usuario, descrita en la siguiente sección.

9.2 Interfaz de Usuario

El propósito de la interfaz de usuario, es el de recibir los parámetros del controlador, como coeficientes de numeradores y denominadores, tiempo de muestreo y coeficientes de vectores de realimentación de estados, entre otros. Está compuesta por pestañas, cada una permitiendo introducir los controladores a cada uno de los cuatro sistemas, además de dos pestañas adicionales que permiten, la primera, realizar controladores con forma de función de transferencia para el motor NXT, y una al final para cargar los arhivos de texto para realizar la identificación de sistemas.

La primera de las pestañas, permite implementar controladores en forma de función de transferencia, recibiendo por parámetros los coeficientes del numerados y denominador en cada unas de las entradas de la parte superior. Más abajo aparecen los botones de creación de archivos, con lo cual se generan automáticamente los archivos para NXTOsek con los controladores implementados de acuerdo al procedimiento mostrado en la seccción anterior y las funciones de lectura de sensores, de ser necesarias. Las pestañas del sistema de acoples flexibles y el bola y viga tienen aspecto similar. Esta parte de la interfaz muestra en la imagen 9.1.

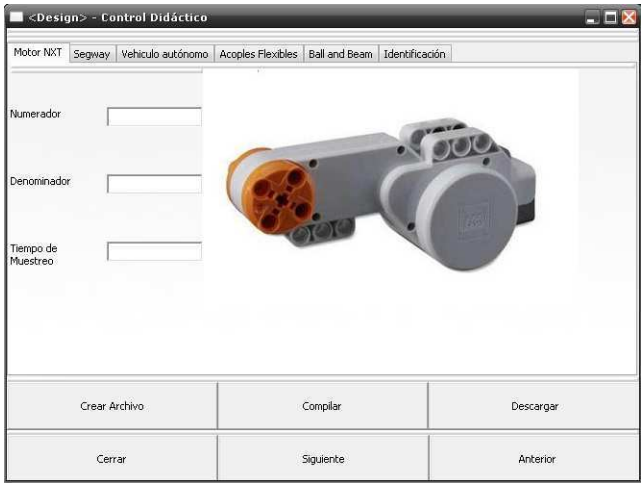


Figura 9.1: Esquema básico de las pestañas

La pestaña del vehículo autónomo solo difiere de la anterior en la cantidad de entradas, ya que

esta requiere de dos sistemas de control, tal como se observa en la figura 9.2.

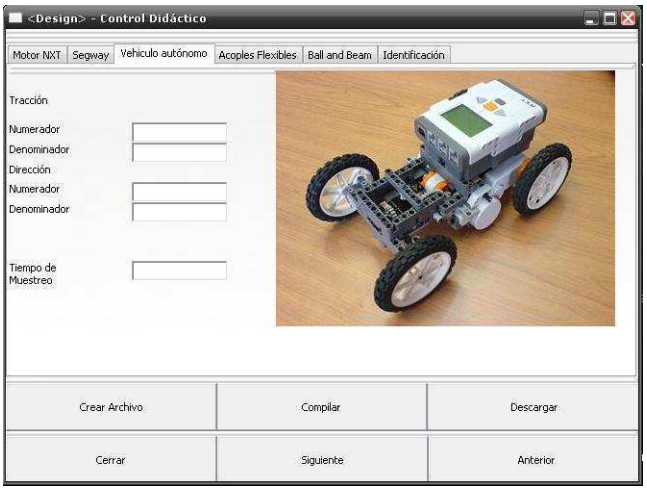


Figura 9.2: Pestaña del Vehículo Autónomo

La parte de la interfaz dedicada al segway, tiene tres entradas, una para el vector de realimentación de estados, la constante para el controlador PI con efecto integral y la ganancia para la parte del control de posición, tal como se observa en la figura 9.3.

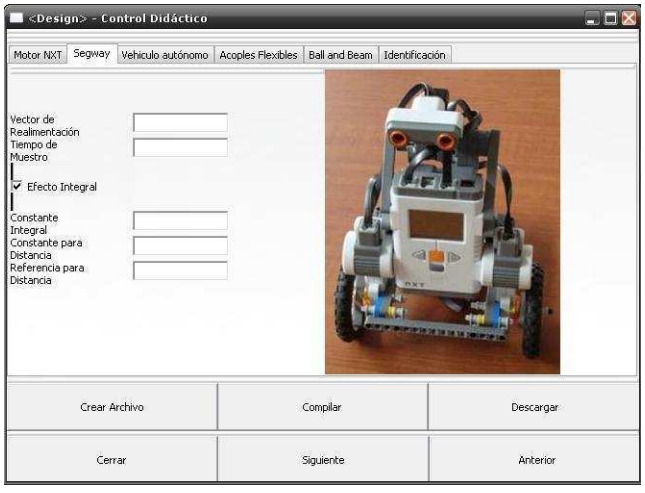


Figura 9.3: Pestaña del Segway

La pestaña de identificación es muy simple, contiene un selector para escoger el archivo de texto que será usado para la identificación, y un boton que invoca la función de descarga del archivo, ba-

sada en el programa NextTool. Esta se ve en la imagen 9.4.

Figura 9.4: Pestaña de Identificación

Las pestañas del vehículo, el bola y viga, y el sistema de acoples flexibles tienen pestañas para definir las referencias, que en el caso del Segway, es la distancia al obstáculo que se quiere mantener.

9.3 Funciones de NXTOSEK

Los procedimientos para NXTOsek desarrollados comprenden funciones de control, lectura de referencias desde archivos de texto, visualización de variables, lectura de sensores y definición de archivos cabecera para alguans variables y constantes usadas en algunos programas. Las funciones de visualización manipulan el display del Brick y permiten ver en pantalla valores como la señal de control o la salida del sistema. Están basada en las funciones de visualización de NXTOsek, y permite una visión personalizada del display, en lugar de la básica, que muestra puertos y otra información quizá no tan importante para nuestro proyecto.

Las funciones de control son el corazón de los programas. Son el lugar en el que son implementadas las estrategias de control. Allí se escriben las ecuaciones en diferencias para el caso de los controladores en representación de función de transferencia, o se implementa el vector de realimentación de estado para el caso del los controladores para el segway. Cuentan además con el acceso a los puertos de motores (con funciones nativas del NXTOsek).

Las funciones de lectura (o calibración) de sensores son básicamente tres, una para calcular el offset del giroscopio usado en el segway, otra para leer el sensor de rotación RCX usado en el sistema de acoples flexibles y la última para linealizar la lectura del sensor EOPD utilizado en el bola y

viga. La primera simplemente consiste en la toma de muestras del giroscopio, cada 4 milisegundos durante un segundo. Una vez terminada esta función, se da inicio a la tarea de control principal; todo esto para evitar la entrada de datos erróneos al controlador.

La siguiente función permite obtener la lectura aproximada en grados del sensor de rotación RCX, el cual no tiene una función definida en NXTOsek para su lectura, por lo que fue inevitable su implementación. Primero se identificaron los 4 valores crudos (raw) que emite el sensor, en orden 356, 515, 1023, y 772, aproximadamente. Estos valores fueron guardados en un vector. Las variaciones entre dichos números indicaban un aumento o una disminución de 22.5 grados en el giro del eje de salida (se generan 16 pulsos por vuelta). Si el cambio se producía en dirección ascendente (el vector está indexado) en los valores del vector de secuencias, la salida aumentaba 22.5 grados. En caso contrario, la salida decrementa su valor en una cantidad igual de grados. En ausencia de cambios, la salida permanece con valor constante igual al producido por el último cambio registrado.

La función de lectura para el EOPD consiste en la implementación de los polinomios mostrados en el capítulo 7. Consiste en realizar una comparación entre el valor crudo de la lectura del sensor ubicado en el extremo pivotado del bola y viga, y los diferentes rangos de trabajo definidos para cada uno de los polinomios calculados. Así se devuelve la lectura directa en milímetros. Sino se encuentra en el rango del mencionado sensor, compara con los rangos del segundo sensor, y devuelve la diferencia entre la longitud de la viga y la lectura directa en milímetros del sensor.

En cuanto a la lectura de los archivos de identificación, se requiere que dichos archivos sean escritos en forma de texto plano, con cada dato de entrada escrito en formato entero, y separados uno del otro por un salto de linea.

10
Conclusiones y Trabajo Futuro

Los kits de Lego Mindstorms se presentan como una alternativa para implementar laboratorios y esquemas prácticos de enseñanza a bajo costo. Ofrecen la posibilidad de construir, modelar, programar y probar diferentes tipos de sistemas y situaciones, permitiendo de una forma rápida y segura, modificar la dinámica y el comportamiento de los mismos. Esta capacidad es fundamental en un entorno educativo, dado que se facilita la exploración, ilustración y comprobación de diversos conceptos.

El Lego Mindstorms contiene los elementos necesarios para la conformación de lazos de control realimentado, como sensores, actuadores, procesadores y piezas constructivas, todo en un mismo paquete, con la capacidad de ampliación, consecución e incluso fabricación de partes adicionales. Esto le brinda total libertad al usuario para copiar diseños, adaptarlos a su acomodo y hacer creaciones propias que le permitan no solo probar conceptos nuevos y afianzar los actuales, sino también proponerlos. Los sistemas mostrados pueden ser modificados físicamente de una manera muy rápida, comparada con las soluciones disponibles para prácticas de control automático, que son de estructura cerrada, o su ampliación o cambio requiere la consecución de costosos equipos.

Una ventaja fundamental del Lego Mindstorms, es que no requiere de la consecución de complejas herramientas para su programación, y las existentes, cuentan con curvas de aprendizaje muy veloces. Cada una de estas tiene sus capacidades y desventajas propias, en rango tan amplio que se acomoda casi a cualquier tipo de usuario, desde principiantes hasta expertos. La profundidad de la programación puede ser la que requiera el consumidor, debido a que en casi todas las herramientas existentes se permite realizar desde rutinas muy sencillas, hasta programación de tareas multiples y desarrollo de códigos complejos. Tal habilidad, fue la que permitió la implementación de las ecuaciones propias de cada uno de los experimentos de control mostrados.

El equipo de Lego, brinda la oportunidad al usuario de sentirse en un proceso de aprendizaje mediante el juego, librándolo de muchas las presiones de la enseñanza tradicional, y le da confianza para realizar cambios y pruebas, algo que no ocurre con muchos sistemas didácticos para control automático.

10. Conclusiones y Trabajo Futuro

La estrategia de diseño puesta en práctica durante el proyecto es aplicable a casi cualquier producto o maquinaria que se pueda desarrollar. Parte desde unas necesidades básicas, y pasa por todo un proceso metodológico, que permite aplicar conocimientos y conceptos en un esquema muy práctico, realizar pruebas de funcionamiento y desempeño, e incluso iteraciones muy rápidas sobre los esquemas diseñados, hasta obtener el correcto cumplimiento de los objetivos propuestos. La ventaja de realizar las actividades mencionadas con el Lego radica en el hecho de contar con un conjunto de piezas y partes ya construidas, y generalmente probadas, sin tener que ocuparse por tareas dispendiosas, e incluso costosas, como la fabricación, caracterización y ensayo de dispositivos adicionales, que sin duda entorpecerían el propósito educativo del proyecto. La filosofía del proyecto consiste en la habilidad de construir las plantas, conectar los diferentes dispositivos y comenzar a probar y comparar diferentes estrategias de control, sin preocuparse por la instrumentación o la resolución de detalles de complejos de construcción mecánica. Esto sin lugar a dudas permite al estudiante centrarse en el problema de control, incrementando las probabilidades de éxito en la apropiación de los conocimientos de dicha temática.

La convergencia de diferentes disciplinas de la ingeniería en un mismo proyecto (programación, diseño mecánico, utilización de sensórica e instrumentación, síntesis de controladores) como el mostrado en este documento, permite un acercamiento a nuevas y modernas metodologías de desarrollo como la mecatrónica. Faculta al estudiante de control, el poder conjugar todas esas ramas de la ingeniería, sin necesidad de tener un conocimiento basto en todas ellas, proporcinándole la capacidad de interactuar con conceptos que van más allá de su propia base de conocimientos, sin implicar un esfuerzo adicional considerable. Se puede considerar incluso, que enriquece el bagaje intelectual del estudiante y complementa su formación.

La plataforma implementada permite la realización de pruebas que abarcan un amplio espectro de conceptos propios del control clásico y moderno, pasando desde estrategias de control simples como los controladores proporcionales y PID, hasta esquemas de realimentación por variables de estado. La habilidad de representar y probar tal caudal de conceptos, permite al sistema hacer parte activa de un plan curricular dedicado a la enseñanza de control automático. Cada una de la plantas o retos de control mostrados en este proyecto tiene su dinámica y necesidades de control propias, que se ajustan fácilmente a la temática ofrecida en los cursos impartidos en el campo de dichas líneas de conocimiento.

Las diferentes plantas expuestas cuentan con diferentes cualidades y características dinámicas, tal como se ha mencionado. Estas características incluyen con polos estables, resonantes, en el origen (integradores), en el semiplano derecho, retardos y/o ceros de fase no mínima. Todo esto implica la síntesis y experimentación con diversas estrategias de control, y la posibilidad de realizar comparaciones sobre los resultados y desempeño de las mismas sobre un mismo sistema. Todas las plantas permiten aplicar diferentes conceptos de diseño de controladores, sin restringir las posibilidades de experimentación sobre su comprtamiento y dinámica.

10. Conclusiones y Trabajo Futuro

El vehículo autónomo permite la implementación de estrategias de control básicas, hacéndolo ideal para práctica introductorias al control automático. El sistema es familiar para casi cualquier usuario e ilustrar de maneras muy rápida el control de variables presentes en muchos problemas de ingeniería, como el control de posición y velocidad de mecanismos. Requiere de la implementación de dos controladores simultáneos, lo que le otorga la posibilidad de mostrar el efecto de un sistema sobre el comportamiento del otro (dirección y tracción), y viceversa. Además, pueden realizarse sobre él pruebas de seguimiento de trayectorias, que pueden ser útiles para validar el funcionamiento de los controladorees diseñados, y la realización de pruebas de navegación más complejas.

Los sistemas como el bola y viga y el sistema de acoples flexibles permiten implementar muchos tipos o formas de control, como controladores PID, controladores diseñados por métodos frecuenciales y controladores obtenidos por métodos algebraicos. Esto cubre casi la totalidad de la temática contenida en un curso de control automático, por lo que pueden estar involucrados en prácticas durante la duración del curso. Dichos sistemas tienen la particularidad de ser inestables o marginalmente estables, por lo que la labor de control sobre ellos es perticularmente interesante, y muestra conceptos de control más complejos. Estos sistemas dejan abierta la posibilidad de probar técnicas de control robusto y moderno, como aparece en infinidad de referencias bibliográficas, lo que permitiría su uso en cursos más avanzados.

La utilización de sensores y actuadores de calidad más baja en el caso del sistema de acoples flexibles hace que el desempeño del sistema no sea al más adecuado, pero antes de hacerlo inútil, hacen más interesante la tarea de control, requiriendo de un mayor esfuerzo por parte del diseñador del sistema de control, quien se vería forzado a depurar y mejorar las estrategias que llegue a implementar.

El segway es una de las plantas más interesantes mostradas en el proyecto. Se trata de un sistema con dinámica inestable, que requiere de estrategias de control más avanzadas para lograr su correcto funcionamiento. Es muy útil para mostrar los conceptos del control en variables de estado, una técnica de control moderno. Puede ser usado como una práctica para el final del curso o como proyecto. Permite además, realizar pruebas de seguimiento de trayectorias y de rechazo a perturbaciones, una cualidad muy deseable en cualquier sistema bajo los efectos del control realimentado.

Una parte importante del proceso, fueron los diferentes modelos matemáticos obtenidos para cada uno de los sistemas implementados. Los modelos permiten una descripción cualitativa de la dinámica de los procesos y facilitan la realización de conjeturas y predicciones acerca del comportamiento de los mismos. Son el primer paso y la base para vislumbrar las técnicas de control necesarias para el funcionamiento de los sistemas. La posibilidad de hacer experimentos de identificación de modelos, ayudan a precisar los diferentes parámetros de las plantas, y podrían ser considerados como una práctica o componente adicional dentro de los cursos de control automático.

10. Conclusiones y Trabajo Futuro

Los experimentos de control realizados tienen resultados que sirven para validar los modelos obtenidos y son una clara ilustración de las diferentes técnicas de control implementables con cada uno de los sistemas. Todos los sistemas presentados son suceptibles de ser controlados mediante diversas formas de control, e incluso permiten la realización de experimentos adicionales a la simple prueba de cada controlador.

La interfaz de software relizada permite la rápida introducción de los diferentes parámetros de cada controlador, sin necesidad que los usuarios tengan que preocuarse por aspectos complejos de programación. La rápida inserción de parámetros, brinda la capacidad al usuario de implementar diferentes estrategias de control en un mismo sistema y hacer comparaciones en un periodo de tiempo muy corto, ayudadando a ilustrar y probar muchos conceptos, en sesiones de poca extensión.

Como trabajo futuro, se pueden realizar experimentos de control más complejos e interesantes. Es posible llevar a cabo pruebas de tele control en sistemas como el segway o el vehículo autónomo, el desarrollo de controladores robustos o inteligentes, para ser probados sobre el bola y viga o el sistema de acoples flexibles, implementación de estrategias de navegación complejas sobre el carro, o la introducción de los sistemas en cursos complejos o proyectos para un estudio más profundo que lleve a la comprobación y utilización de estrategias avanzadas de control.

También, se pueden desarrollar mejoras sobre el sistema de registro de datos, y sobre la interfaz con el usuario en general, dado que la parte gráfica se realizó pensando en facilitar el ingreso por parte de datos del usuario, pero no era una parte primordial del proyecto.

Por último, se pueden considerar reformas a las plantas mostradas, que implique un cambio en su dinámica, o un mejoramiento de su desempeño, como en el caso del sistema de acoples flexibles, mediante la utilización de mejores sensores y actuadores. Cualquier mejora al sistema será bien recibida.

La plataforma muestra ser un elemento ideal para la realización de pruebas y prácticas de control, sobre sistemas de comportamiento dinámico diverso, uno de los pilares fundamentales de este proyecto. Es a bajo costo y de arquitectura reconfigurable, lo que permite realizar variantes sobre modelos y estrategias de control implementables sobre la misma. La capacidad realizar experimentos con diferentes diferentes estrategias de control muestra y afianza los conceptos, lo que constituye la base para la inclusión de las plantas y la plataforma desarollada en un esquema de prácticas para control automático.

11 Anexos

El primer anexo corresponde a la guía de instalación del software Cygwin y NXTOsek, necesarios para la ejecución de los programas desarrollados para cada experimento de control mostrado en este documento.

En esta sección pueden encontrarse también, los artículos correspondientes a las presentaciones del trabajo en diferentes eventos. El primero corresponde al artículo **Utilización e Implementación de Herramientas Didácticas para Prácticas de Control Automático**, presentado en el 3^{rd} Colombian Workshop on Circuits and Systems, en Bogotá, el 5 de octubre de 2009. En segundo lugar, aparece el artículo **Aplicaciones Didácticas en Control Automático con Lego Mindstorms NXT**, presentado en el II South American Congress on Computational Mechanical, en Buenos Aires, Argentina, el 16 de Noviembre de 2010.

Finalmente, se muestran las guías de armado de las plantas Vehículo autónomo y Bola y Viga.

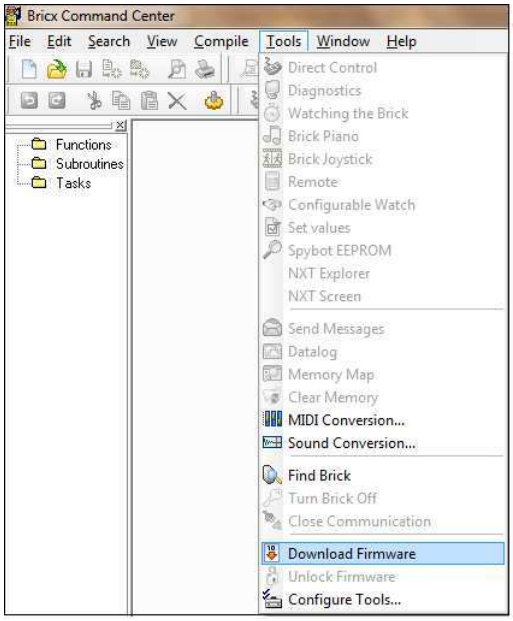
Con el uso del NXTOsek el objetivo es implementar sistemas más precisos mediante cálculos en punto flotante.

11.1 Firmware

Para poder implementar cálculos de punto flotante en el procesador del NXT, es necesario descargar un nuevo bloque de instrucciones que use los recursos más eficientemente. El firmware que se usa para poder hacer y descargar programas en NXTOSEK se llama enhanced NXT firmware y puede ser descargado de internet.

El firmware deberá ser descargado al brick con la ayuda de dos herramientas: NeXTTool y BricXCC. NeXTTool Es un programa de comandos en Windows que permite la comunicación con el brick mediante la utilización del driver de LEGO Mindstorms. NeXTTool puede ser descargado de la página de BricX Command Center y deberá ser guardado en un carpeta cuyo directorio no contenga espacio o caracteres especiales. Esta carpeta debe coincidir con la carpeta en la que se ha guardado previamente el archivo de firmware. Mediante BricX Command Center se descargará el

firmware usando el comando Download Firmware en el menú Tools como se muestra en la figura a continuación

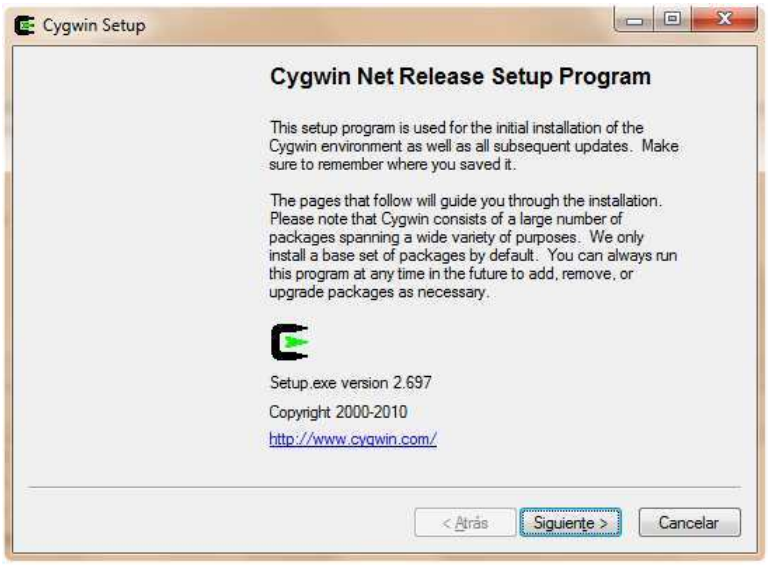


11.2 Instalación

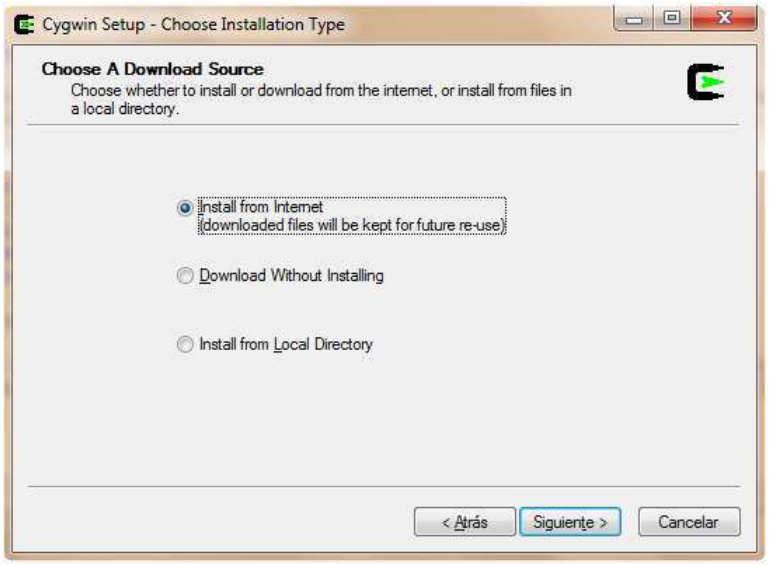
NXTOSEK necesita software adicional para poder ser usado. Estos programas deberán ser instalados como se muestra a continuación.

Cygwin Cygwin es un programa que emula un ambiente Linux para Windows. Es necesaria su instalación pues el compilador de código para procesadores ARM está basado en Linux. Cygwin puede ser desacargado de internet. Para su instalación, se deben seguir las siguientes instrucciones:

1. Ejecutar el archivo descargado y hacer click en siguiente

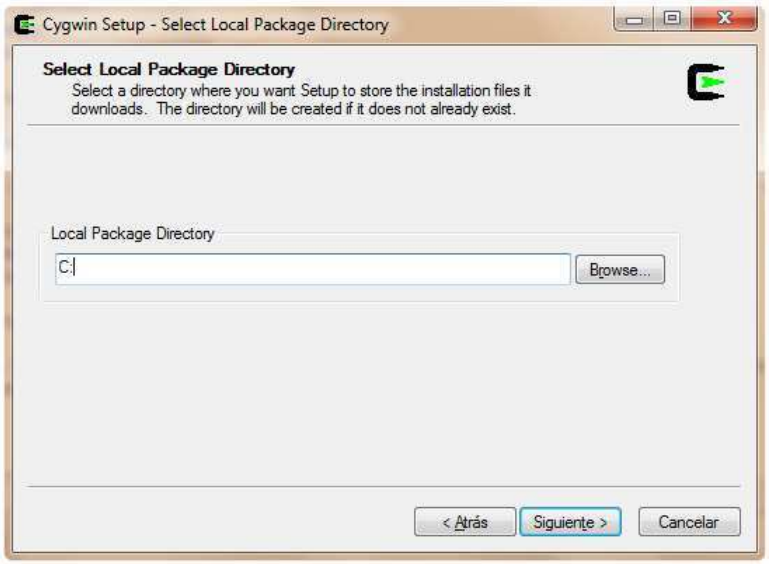


2. En el paso de tipo de instalación escoger instalar desde internet (*Install from Internet*) si el programa ha sido descargado o instalar de un directorio local (*Install from Local Direectory*) si el programa se encuentra en el disco duro. Todos los instaladores se encuentran en la carpeta instaladores.

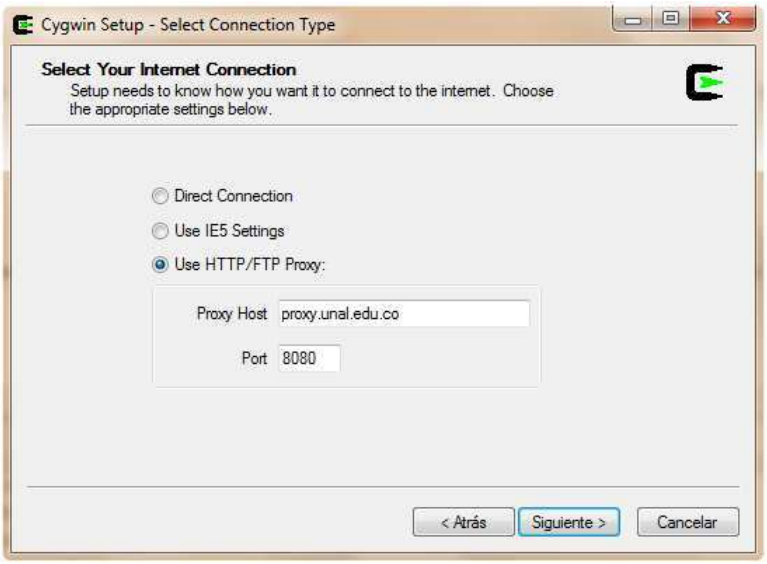


3. El programa será instalado en C:\cygwin como se muestra en la siguiente pantalla. Hacer click en siguiente

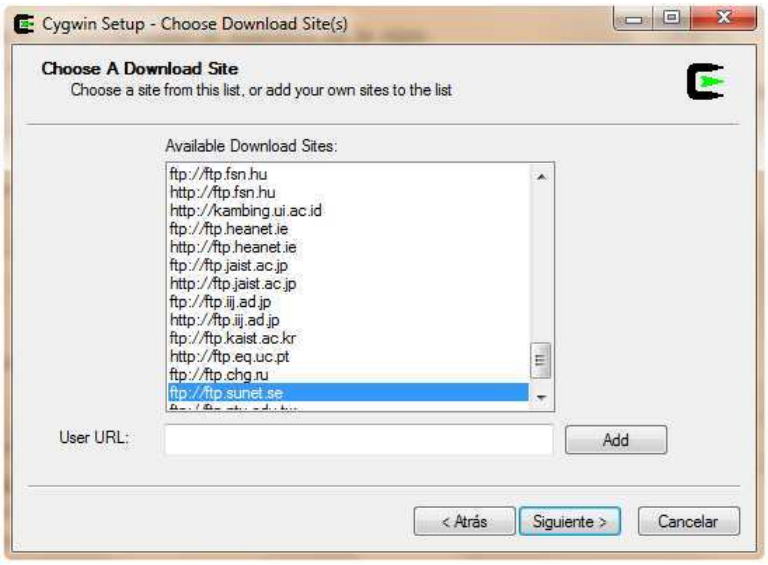
4. En la siguiente pantalla se debe seleccionar una ruta en la cual se guardarán los archivos de instalación. Hacer click en siguiente



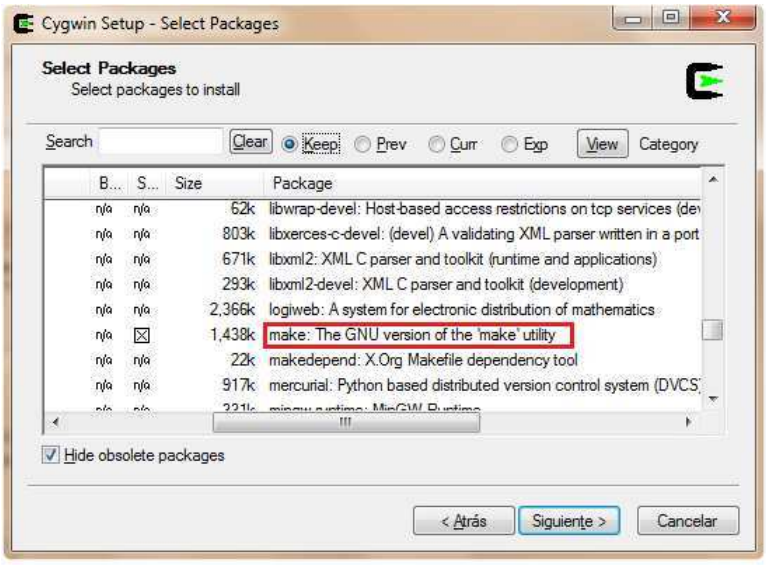
5. Elegir las opciones de conexión a internet para descargar el programa. hacer click en siguiente



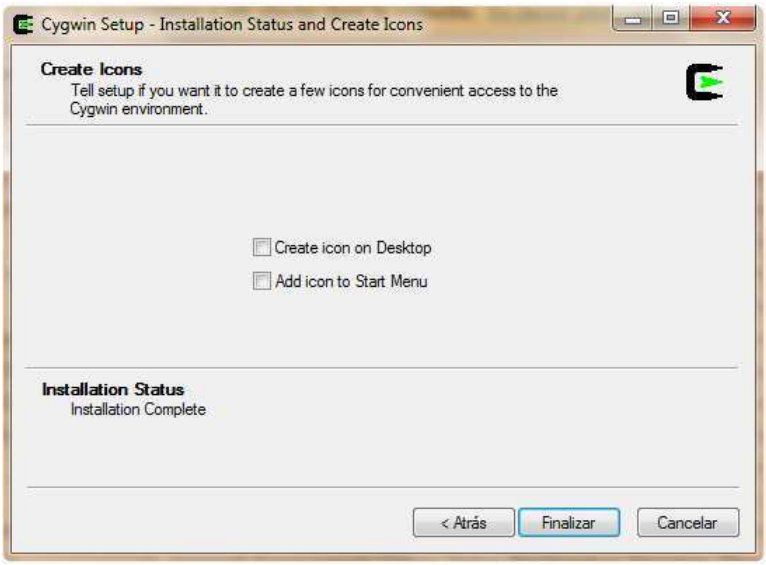
6. Escoger un sitio web para la descarga de los archivo. Hacer click en siguiente.



7. Después de un tiempo aparece la siguiente pantalla de selección de paquetes en donde se debe asegurar que el paquete make sea instalado. Este paquete se encuentra en el menú Devel



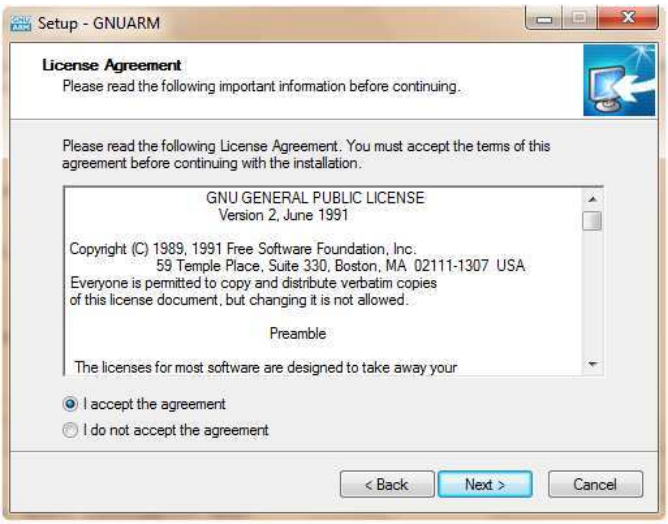
8. Finalizar la instalación



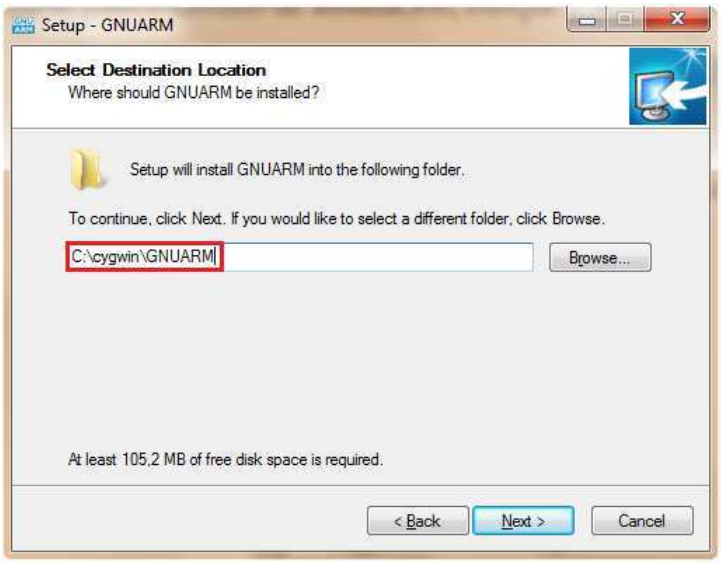
GNUARM GNUARM es un compilador de C que permite traducir los programas escritos al idioma del procesador ARM. La versión actualizada puede ser descargada de internet. A continuación se muestran las instrucciones para su instalación.

1. Ejecutar el archivo descargado y hacer click en siguiente

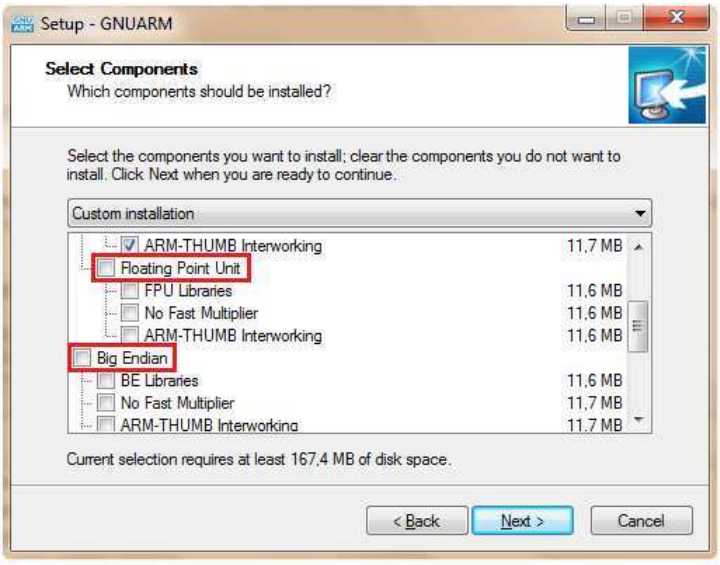
2. Aceptar el acuerdo de licencia y hacer click en siguiente



3. Asegurarse de guardar los archivos de instalación en la carpeta donde cygwin ha sido instalado. C:\cygwin\GNUARM

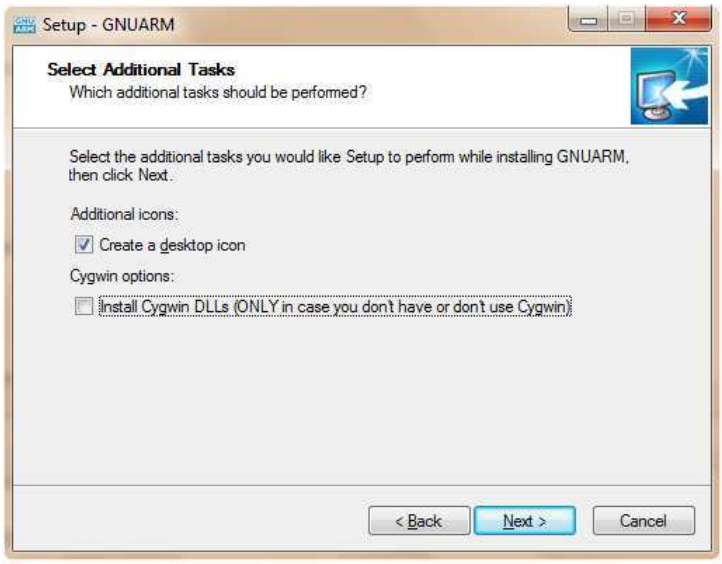


4. En el paso de selección de componentes se debe seleccionar instalación personalizada (*Custom installation*) y desmarcar las casillas *big endian* y *floating point*

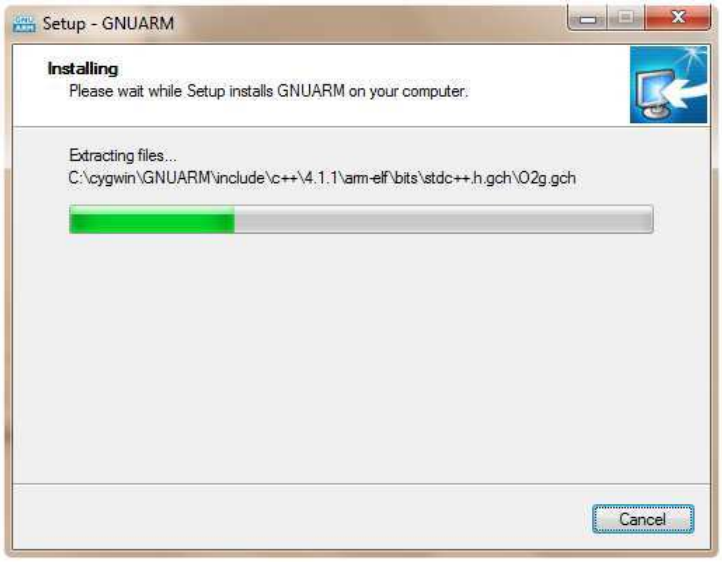


5. Continuar la instalación

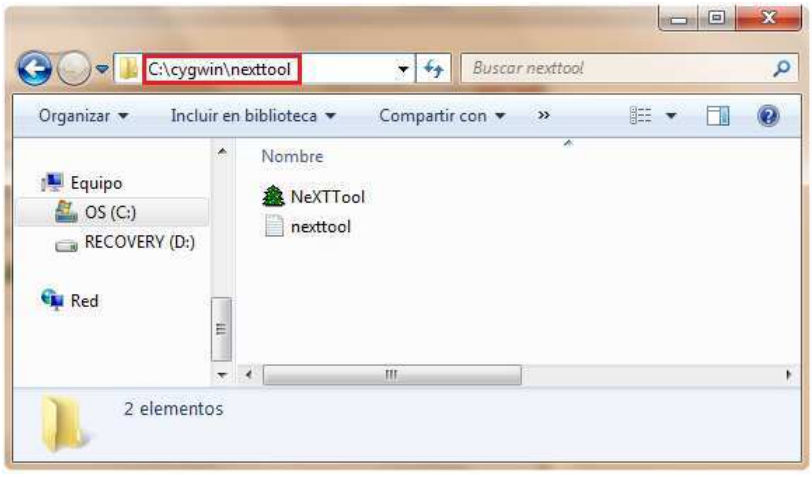
6. Desactivar la casilla de instalación de DLLs de cygwin



7. Finalizar la instalación

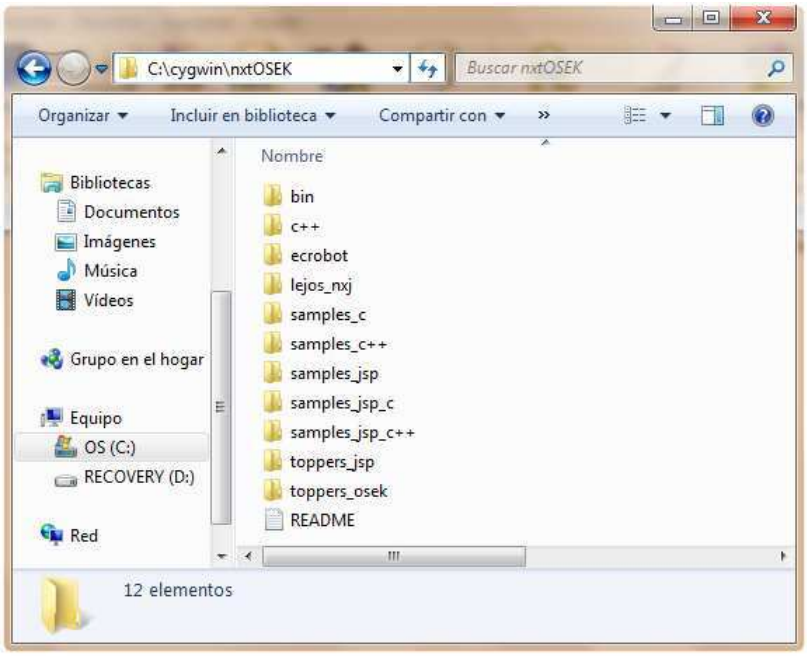


NeXTTool Los archivos de NeXTTool descargados anteriormente deberán ser guardados dentro de una carpeta llamada nexttool dentro de la carpeta de cygwin para poder ser reconocidos por el emulador.



NXTOSEK Finalmente, los archivos de NXTOSEK, descargados de internet deben ser copiados en la misma ruta. NXTOSEK permite la utilización del estándar OSEK al compilar los programas.

Se debe tener en cuenta que si BricXCC no ha sido instalado anteriormente, es necesaria la instalación del driver de LEGO MINDSTROMS disponible en la página oficial.



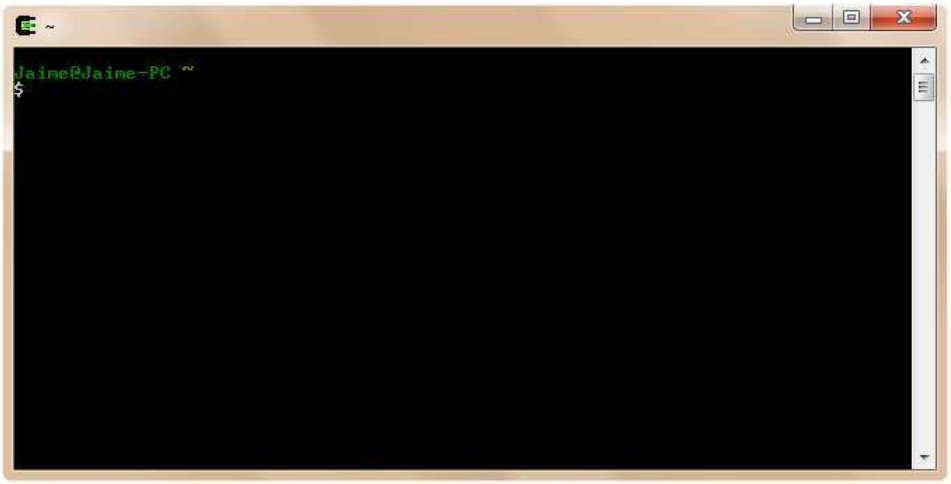
11.3 Programación

Para poder compilar y descargar programas al brick, se deben crear tres archivos con extensiones diferentes: El archivo con extensión .c permite describir el funcionamiento del robot en lenguaje C para ser compilado por GNUARM; El archivo con extensión .oil permite especificar algunas características de las funciones y objetos contenidas en .c; y finalmente el archivo Makefile permite a cygwin tomar todos estos archivos y compilarlos de forma que la información sea descargable al brick. La estructura general de estos archivos se explicará más adelante con la ayuda de los ejemplos.

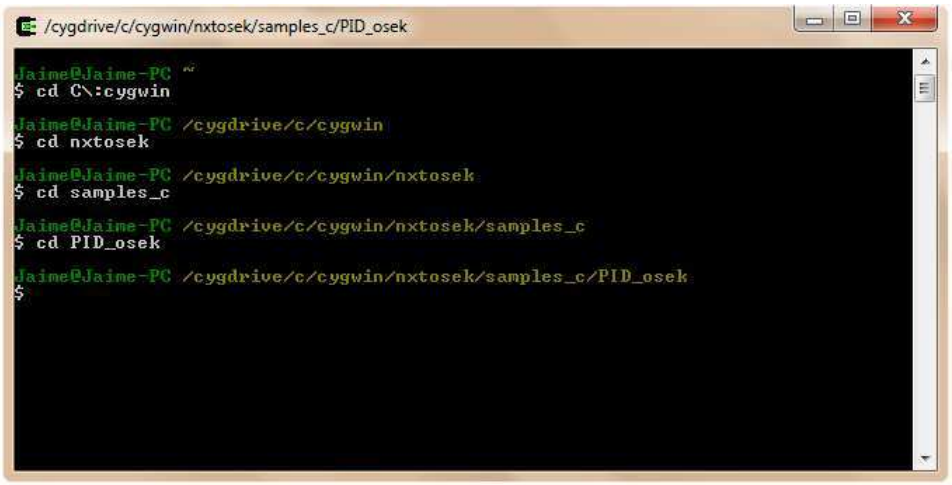
Una vez creados los archivos, será necesario usar cygwin para poder hacer la compilación con la ayuda de GNUARM. Los pasos son los siguientes:

1. Guardar los tres archivos dentro de una carpeta contenida en la ruta C:/cygwin/nxtOSEK/samples_c

2. Abrir el bash de cygwin. El ejecutable se encuentra en Inicio, Todos los programas, Cygwin, Cygwin Bash Shell



3. Desplazarse hasta la carpeta donde se encuentran los srchivos con la ayuda del comando cd.



4. Usar el comando *make all* para compilar los archivos. La generación del ejecutable (*.rxe) indica que no hay errores en el programa.

5. Finalmente se usa el comando *./rxeflash.sh* para descargar los archivos en el brick.

IX ... [illegible] ... I ... [illegible]

Aspectos Prácticos en la utilización e implementación de Herramientas Didácticas para Control Automático

David Herrero Alfonso, Leonardo Herrero Clavijo

Resumen— La ingeniería de control es una disciplina que implica grandes conocimientos tanto prácticos como teóricos, siendo la enseñanza de estos últimos, un camino largo y en ocasiones complejo, debido a la falta de una conexión conceptual y de aplicación con su contraparte práctica. El propósito del presente escrito es el de ilustrar la primera fase de un proyecto de implementación de esa parte práctica a bajo costo. Este documento describe algunas herramientas de software de carácter libre y hardware económico, que pueden ser utilizadas en el desarrollo e implementación de prácticas de control automático, basadas en los retos clásicos existentes en esta materia, y que permitan además la síntesis de los controladores obtenidos por el usuario en una plataforma digital, mediante interfaces de uso muy sencillas, incluyendo que el éste no se involucre en problemas más allá de una simple prueba de compensadores.

Index Terms—AGV, control automático, identificación de sistemas, mecatrónica, retos de control, síntesis de controladores digitales.

I. INTRODUCCIÓN

En la industria se presentan de manera recurrente todo tipo de procesos que implican el control de un conjunto de variables en dominios físicos múltiples, como flujos de caudal, temperaturas, presiones, niveles de líquido y voltajes, entre muchos otros, lo cual exige entonces un profundo tratamiento de los temas relacionados con la teoría de control y sus posibles aplicaciones prácticas a seno de los cursos que las universidades y otros centros tengan destinados para la enseñanza de estos temas.

Durante mucho tiempo, la educación en control se ha centrado en el aleccionamiento de los conceptos teóricos, acompañado en épocas recientes de simulaciones con software especializado. Sin embargo la teoría básica requiere un esfuerzo intelectual considerable, el cual en caso de no verse reflejado en ejercicios prácticos y/o de simulación, puede mostrar problemas de apropiación del conocimiento por parte de los estudiantes. De acuerdo a [1], se nota que la mayoría de los aprendices (de control) encuentran la teoría de control conceptualmente difícil, la perciben como periférica y tienen problemas integrándola con otro material, agregando además que en la instrucción de control se está muy preocupado con la "parte analítica" del proceso de control porque no se sabe como enseñar otros problemas más complejos en la síntesis del mismo, es decir, existe un problema de contextualización y de integración de conocimientos.

Tal dificultad, se ha venido supliendo mediante la implementación de las plantas clásicas en laboratorios dedicados, dado que la enseñanza de la ingeniería de control se beneficia enormemente mediante el uso de laboratorios de control, con experimentos en los que se utiliza hardware real, como parte integral de los cursos [4], [6]. Además, algunas empresas han creado diversos módulos que recrean las mismas, productos de muy buena calidad y fiabilidad. Sin embargo, no son muchas las compañías que trabajan en este campo y sus soluciones aunque de gran nivel, son costosas y limitar la creatividad del estudiante a la hora de intentar nuevos retos, dado que se trata en su mayoría, de plantas de arquitectura cerrada o con pocas posibilidades expansión sobre piezas prefabricadas.

Una posible solución se evidencia en las construcciones de bajo precio y buenos resultados (algunos ejemplos propuestos se pueden ver en [3], [4] y [7], entre otros), pero que caen de nuevo en la carencia de posibilidades de ampliación o mejora. Por lo tanto el problema se concentra en la necesidad de un conjunto centralizado, unificado y coherente de prácticas de control (que aunque existen, se restringen a centros especializados que implican grandes inversiones [9]), portátil, con una flexibilidad constructiva que brinde posibilidades de ampliación, modificación y nuevos estudios, todo a un costo razonable y al alcance de quienes ni siquiera tiene la posibilidad de contar con esquemas de práctica de control.

Aquí se propone la construcción de diferentes plantas de control con Lego^{TM}, para la enseñanza y práctica del Control, comenzando por un ejercicio sencillo, un vehículo autónomo construido con Lego, sobre el cual se lleva a cabo un proceso de identificación, seguido de la implementación digital de diferentes controladores de la posición del móvil, y por último una prueba de auto navegación que incluye la utilización de sensores ultrasónicos. Las posibilidad de realizar practicas durante los periodos de entrenamiento, pone al aspirante a ingeniero de control en el contexto de los problemas que se deben enfrentar a la hora de implementar un controlador, además que le brinda las herramientas teóricas suficientes para afrontarlo y poner en práctica los conocimientos adquiridos en un caso que lo lleva desde el modelado hasta la entrega de un control funcional. Problemas como la identificación de sistemas y la implementación final son algunos de los casos que puede preparar al ingeniero para un futuro desafío en el mundo laboral. En este punto es importante que el desarrollo propuesto va más allá de un sistema de práctica de

Ingeniero Mecatrónico, Estudiante Maestría en Ingeniería-Automatización Industrial, Unidades Tecnológicas de Santander, e-mail: dherrero@correo.udes.edu.co.
Ingeniero Electrónico, M.Sc., Profesor Asociado, Universidad Nacional de Colombia, e-mail: lherrero@unal.edu.co.

control, sino que adicionalmente implica el enfrentarse con un problema de integración de disciplinas, desde la programación hasta el diseño mecánico, pasando por asuntos relacionados con la identificación de sistemas y la implementación de los controladores, colocando a esta propuesta en un marco mucho más amplio, en un contexto de diseño mecatrónico.

II. Estrategia Propuesta

La realización de cualquier esquema de prácticas de control implica el desarrollo de una metodología completa que permita de una forma concreta realizar el diseño inicial del reto de control, conocer y modelar adecuadamente la planta construida, calcular e implementar las estrategias de control apropiadas, y finalmente, probar y validar el trabajo realizado con ensayos apropiados. En el marco del presente trabajo, se propone la puesta en marcha de un conjunto de plantas basadas en los retos de control, lo que hace necesario poner los anteriores postulados dentro de una metodología completa de desarrollo para cada una de las plantas que se proponga construir.

El proceso comienza con la selección del reto de control y su respectivo modelo matemático genérico. Esto puede corresponder a la revisión de la literatura existente o a la obtención de un modelo específico de acuerdo a las necesidades o modificaciones que impliquen las características de la plataforma de desarrollo. La siguiente fase corresponde al diseño mecánico de la planta o sistema a controlar. En esta parte, se plantea el reto de control, la disposición física de los componentes y se eligen los actuadores y sensores más apropiados para la tarea propuesta. Dada la flexibilidad de la plataforma Lego, esta parte se puede llevar a cabo de forma recursiva y a la par de otras partes del proceso, a medida que este avance, es decir, se desarrolla una estrategia de diseño iterativo, lo que va de la mano con un modelo matemático de los sistemas construidos.

El siguiente paso, es la identificación del sistema. El modelo analítico previo, contiene parámetros que por lo general son desconocidos, y que pueden ser estimados mediante este tipo de procesos, o simplemente se desea contar con modelos precisos con los que sea posible diseñar estrategias de control más exactas. El modelado e identificación son tareas que deben ir de la mano, dado que la primera indica el orden del sistema y los parámetros que deben ser tenidos en cuenta, permitiendo la obtención de modelos adecuados. Un buen modelo garantiza que el siguiente paso, el diseño e implementación del controlador, se lleve a cabo de forma más confiable.

Por último, se validan los resultados mediante un conjunto de pruebas que dependen del tipo de planta y los requerimientos de diseño iniciales. No obstante, estas pruebas sirven para evaluar y determinar si los modelos y controladores obtenidos se ajustan al desempeño deseado. Es fácil notar que esta estrategia se puede aplicar al diseño de cualquier máquina, prototipo o proceso, y poder llevarlas a cabo durante el periodo de entrenamiento del ingeniero, de manera sencilla y económica, le otorga a este tipo de iniciativas un gran valor agregado. La figura 1 resume el proceso descrito.

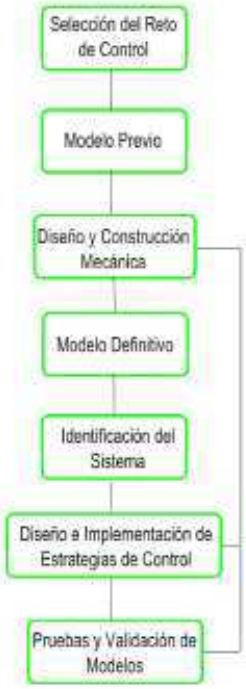


Figura 1. Esquema Propuesto

III. Hardware Propuesto

La plataforma escogida para construir las plantas corresponde al producto Lego Mindstorms. Se trata de un conjunto completo desarrollado por la compañía Lego para la creación e implementación de robots por parte de niños a partir de los doce años. Sin embargo, la herramienta ofrece tal flexibilidad que su uso se ha extendido en el campo de la educación desde nivel de secundaria hasta cursos de posgrado. En la Universidad Nacional de Colombia ya se viene trabajando en el área de control con sistemas más sencillos que los aquí planteados, con buenos resultados tanto en pregrado como posgrado. Las principales ventajas que presenta frente a otros productos, son su bajo costo y enorme flexibilidad.

IV. Herramientas de Software Propuestas

IV-A. NXT-OSEK

Se trata de una herramienta que concentra la versatilidad del conocido lenguaje de desarrollo C, uno de los más conocidos por los desarrolladores de software, el uso de compiladores de caracter libre como gcc. Aquí se escriben y compilan las rutinas que permiten controlar las plantas construidas, para lo cual el sistema NXT-OSEK ya cuenta con un buen número de ejemplos y el desarrollo tras esta herramienta es considerable. Junto al Lejos (Java) son quizá la plataformas de desarrollo para Lego con mayor trabaja y detalle, además que cuentan con la ventaja de ser escritos en lenguajes de programación muy comunes para cualquier ingeniero.

IV-B. *Python*

Se trata de un lenguaje de programación orientado a objetos, que puede ser utilizado con multiples propósitos de desarrollo de software. Esta versatilidad y la capacidad de integrarse con otros lenguajes y herramientas, lo hace ideal para la propuesta aquí mostrada. Se pretende que sea esta la interfaz entre el usuario y el sistema construido, realizado por funciones la recepción del controlador diseñado (función de transferencia, por ejemplo), la implementación del mismo en forma de algoritmo en C++ para NXT-OSEK, su compilación y descarga al Brick, todo en un mismo programa, cosa que no permite actualmente el Matlab por sí mismo.

IV-C. *Matlab*

Es una de las herramientas más comunes dentro de la ingeniería de control, por su gran poder de cómputo y los incontables paquetes (toolbox) con los que cuenta, no solo para el área de control, sino también en muchas otras áreas más allá de la ingeniería. Aquí se desarrolla, en la actualidad, el proceso de identificación de sistemas y síntesis de controladores. Las interfaces de desarrollo e interacción con las plantas se desarrollan en este software, dada la gran cantidad de información y herramientas disponibles en éste para los fines de la propuesta, pero tiene la desventaja de ser software propietario y de un costo relativamente elevado.

IV-D. *Brics*

Este sistema es empleado por los desarrolladores Lego por tratarse de una herramienta libre. Se basa en la codificación del Not exactly C (NXC), contiene un conjunto completo de funciones que permite acceder a las principales funciones y características del Brick de Lego, y tiene una curva de aprendizaje muy rápida. Sin embargo, se ve limitado por la incapacidad de trabajar con variables en punto flotante (aunque esto no le resta demasiada funcionalidad). En este momento, aquí se escriben las rutinas de control e identificación.

V. Caso de estudio: Vehículo Tipo Controlado

La planta propuesta corresponde a un vehículo robótico tele-controlado o auto-guiado de cuatro llantas en dos dimensiones. Este consiste en un robot en forma de carro, con dos motores, uno de dirección y otro de tracción. La idea es conseguir que el móvil sea capaz de moverse a través de un espacio dado usando una serie de sensores, y donde el usuario deberá encontrar los controladores de posición y/o velocidad necesarios para cumplir con el objetivo mencionado. Sobre este sistema es posible implementar controladores proporcionales, PID, con dos parámetros en lugar de tres. La motivación de este reto es la posibilidad de enfrentar al usuario a un problema muy usual en robótica móvil, que es el posicionamiento en el espacio para tareas específicas, como los vehículos auto guiados (AGV) existentes en la industria.

Inicialmente, se construye el modelo mostrado en la figura 2, el cual pasa por un proceso de obtención de un modelo aproximado, aplicando las técnicas de identificación contenidas en [2]. La parte de tele-control será hecha utilizando las estrategias de control implementadas en lenguaje Brics (una implementación de C para el Brick de Lego) y activadas desde un PC usando Matlab. Una a parte de esta guía, se implementará una estrategia en C que deberá mantenerse a una distancia de 30 centímetros de un blanco y ser capaz de dirigir o cambiar su dirección de movimiento a voluntad del blanco. El sistema Brics permite trabajar con datos enteros con signo de 32 bits.

Figura 2. Vehículo Tipo Controlado

V-A. *Identificación del Sistema*

El sistema fue dividido en dos partes principales: una dedicada a la tracción del vehículo y otra a la dirección, suponiendo que el sistema es no acoplado, para facilidad en la implementación de los controladores. Esto reduce el problema de identificación y control a dos experimentos separados, en los cuales se busca hallar el modelo aproximado de dos motores eléctricos con carga mecánica. En general, se aplicarán algunas señales de prueba al sistema de cada motor en lazo abierto y se aplicarán los métodos descritos en [2] para obtener los modelos deseados.

Se crearon programas en Brics para aplicar las señales de prueba al sistema. Estos consistían en rutinas de integración de Matlab con el Brick (NXC) para identificación del sistema y que permitieron ubicar las señales de prueba dentro del Brick (descarga) para luego ser recuperadas e ingresadas en Matlab para aplicar los respectivos algoritmos de identificación.

Para el sistema de tracción se aplican ondas de identificación Chirp y seudo-aleatorias (PRBS), con cuyos datos se obtuvieron una serie de modelos lineales mostrados en las figuras 3 y 4, en la que se muestran también el porcentaje de validación de cada modelo con respecto a un conjunto de datos de prueba obtenidos. Estos modelos fueron escogidos entre ARX, ARMAX, OE y BJ, de acuerdo al orden, prefiriendo de orden 2 y 3, ya que es el orden del motor eléctrico, suponiendo que la caja reductora a lo mucho un grado a sistema, para garantizar la sencillez en los controladores obtenidos, y por tanto facilitar su implementación digital en el Brick, además del mencionado porcentaje de validación.

Al final se escogió un modelo obtenido con la onda Chirp del tipo Output Error de la forma:

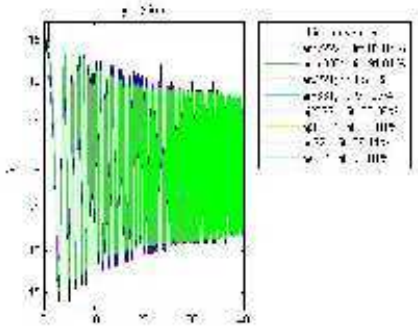


Figura 3. Identificación con Onda Chirp

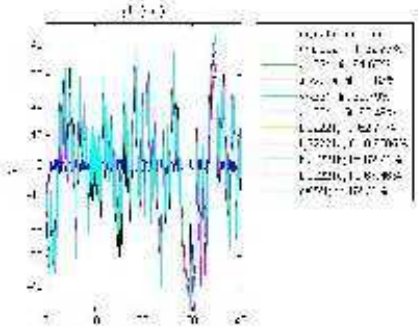


Figura 5. Identificación con Onda RBS

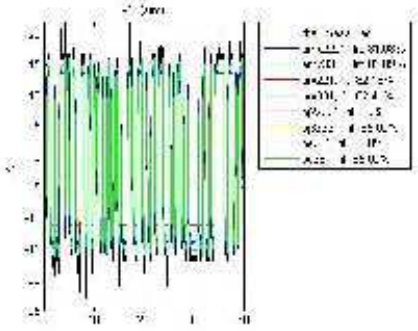


Figura 4. Identificación con Onda PRBS

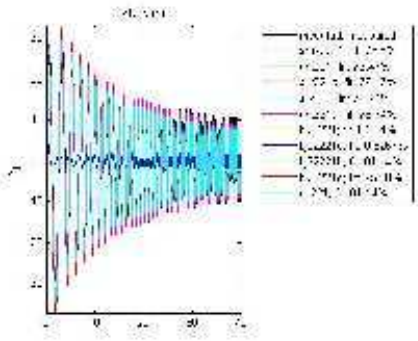


Figura 6. Identificación con Onda Chirp

$$y(t) = \frac{B(q)}{F(q)}u(t) + e(t)$$

$$B(q) = 0.02581q^{-1} + 0.02163q^{-2} + 0.0057\_5q^{-3}$$

$$F(q) = 1 - 2.115q^{-1} + 1.647q^{-2} - 0.4951q^{-3}$$

Para la parte de la dirección del vehículo se siguió el mismo procedimiento mostrado con anterioridad. Primero se aplicó una onda RBS y luego una Chirp, con la que se consiguieron los resultados mostrados en las figuras 5 y 6.

Finalmente se escogió el modelo Box-Jenkins estimado con la onda chirp que presentaba los siguientes parámetros.

$$y(t) = \frac{B(q)}{F(q)}u(t) + \frac{C(q)}{D(q)}e(t)$$

$$B(q) = 0.0337q^{-1} - 0.0326q^{-2}$$

$$C(q) = 1 - 0.5936q^{-1} + 0.2842q^{-2}$$

$$D(q) = 1 - 0.897q^{-1} + 0.4527q^{-2}$$

$$F(q) = 1 - 1.555q^{-1} + 0.586q^{-2}$$

Con los modelos descritos anteriormente, se procedió a buscar las estrategias de control que garantizaran el poder gobernar sobre la posición del robot en dos dimensiones, es decir, del motor de tracción y del de dirección.

VII. Estrategias de Control Implementadas

Primero se utilizó la técnica del lugar de las raíces para encontrar un controlador proporcional que presentara un buen tiempo de respuesta y cero sobrepico. La ganancia sintonizada e implementada es la siguiente:

$$C_p = 2.0584$$

Los resultados mostrados en la figura 7, en la que se compara la respuesta del controlador real con el modelo simulado, indican que el controlador implementado es muy similar al obtenido en las simulaciones, pero que no refleja completamente el contenido frecuencial del sistema. Sin embargo, el comportamiento es bueno y muy aceptable.

Para la implementación digital, no fue necesaria la escalización de las variables. La siguiente estrategia de control fue sintonizada de la misma forma que la anterior, pero se logró un controlador PID descrito por la función de transferencia:

$$C_{PID} = \frac{1.0775z^2 - 1.8722z + 0.8251}{z^2 - z + 0.5}$$

El resultado de la utilización de este controlador (figura 8), indica que el sistema tiene una zona muerta, para señales

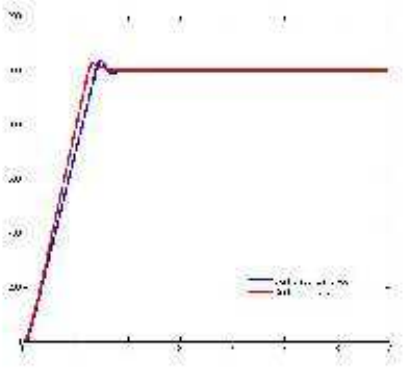


Figura 7. Prueba del Controlador Proporcional

de control baja, es decir, una región en la que no alcanza a realizar movimientos debido a un control escaso. No obstante, se lográn los objetivos de estabilización y error de posición cero.

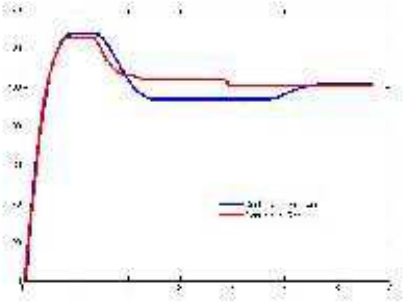


Figura 8. Prueba del Controlador PID

La implementación digital del controlador requiere de un paso a variables de estado del mismo y la escalización de las mismas, ya que los componentes eran números reales menores a cero. Si bien el controlador reacciona muy bien, no se recomienda ser usado para referencias grandes, ya que presenta problemas de windup, lo que se puede resolver con una estrategia de control de dos parámetros o de dos grados de libertad. Esto se basa en el principio de colocación de polos. El controlador obtenido es el siguiente:

$$C_{PD} = \left| \frac{\quad}{\quad} \right|$$

Este nuevo controlador no presenta windup y muestra un comportamiento muy bueno para seguimiento de referencias de posición (figura 9). Para su implementación digital fueron necesarios los mismos pasos del controlador PID.

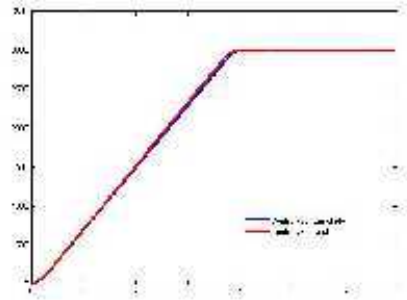


Figura 9. Prueba del Controlador de dos Parámetros

En el caso de la dirección, el sistema se encuentra restringido mecánicamente a moverse dentro del rango de 30 grados tanto a derecha como izquierda. Se decidió sintonizar e implementar una estrategia de control proporcional, mediante la aplicación del lugar de las raíces. El controlador obtenido es el siguiente:

$$C_p = 4$$

Los anexos (figura 10) muestran que el controlador implementado responde algo más rápido y con mayor oscilación que el controlador simulado. Si bien el resultado es muy aceptable, indica que el contenido frecuencial del modelo no fue capturado por completo durante la identificación, es decir, el modelo puede ser mejorado.

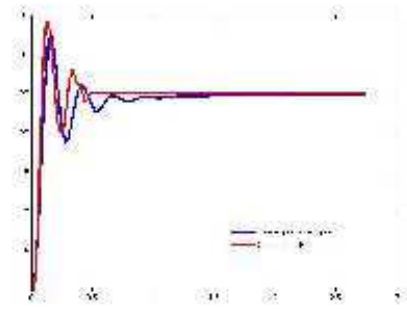


Figura 10. Prueba del Controlador Proporcional

Para lograr el sistema completo de tele control, se desarrolló un programa en Matlab® capaz de enviar vía Bluetooth® la interfaz de usuario mostrada en la imagen 11, escribir un código que es capaz de mandar una trama que contiene las referencias (en grados) introducidas por el usuario, y las envía al controlador montado en el Brick; es un ejemplo muy sencillo de control embebido. En este, se dirigen simplemente valores de setup para los sistemas y el control abordo se encarga de gestionar y cumplir los requerimientos del usuario.

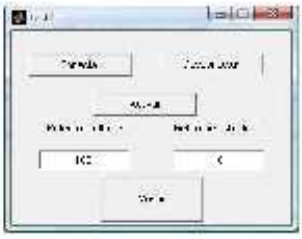


Figura ... Interfaz de Usuario en Matlab

Las pruebas realizadas muestran la efectividad de los controladores y que si es posible tele comandar el robot desde Matlab.

V-C. *Sistema Simple de Auto-Mantenimiento*

Luego de lograr el tele-control, se implementó una estrategia que permitiera al robot seguir un obstáculo (mueble o pared) móvil sin importar la inclinación del mismo y mantenerse a una distancia fija de 30 centímetros de éste. Para este objetivo se colocaron dos sensores ultrasónicos en la parte delantera del robot, tal como se muestra en la figura 12.

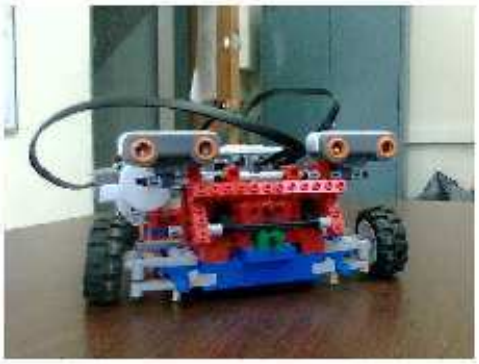


Figura 12. Ubicación de los Sensores.

La estrategia de seguimiento fue implementada de acuerdo a la figura 13.

Donde D es la distancia a la que los sensores, d_1 es la lectura de un sensor y d_2 la de su compañero. La manera de calcular la referencia (error en este caso) es simple, se toma el promedio de la lectura de los sensores, se compara con la referencia y el error resultante se transforma en grados. La forma de calcular el ángulo de giro se aplica la siguiente ecuación:

$$\theta = \sin\left(100\frac{d_1 - d_2}{D^2 + (d_1 - d_2)^2}\right)$$

Las pruebas realizadas con controladores proporcionales se muestran en la figura 14. Esta evidencia algunos problemas de ciclo límite, que se deben a que los sensores incrementan el tiempo de muestreo del controlador, ya el sistema se encuentra a una tasa de muestreo de 20 milisegundos, y cada sensor se

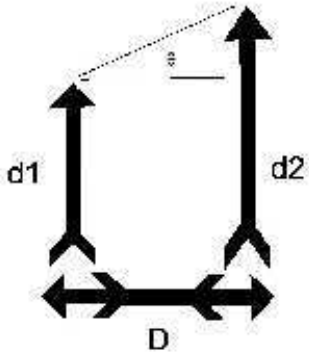


Figura ... Implementación Estrategia de Auto-Seguimiento

tarda 20 ms aproximadamente en tomar su respectiva lectura, incluyendo un retardo equivalente a una vez y media la tasa de muestreo.

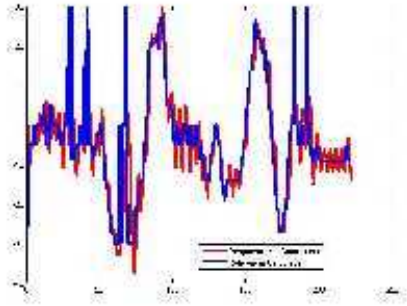


Figura 14. Prueba de Ejecución del robot

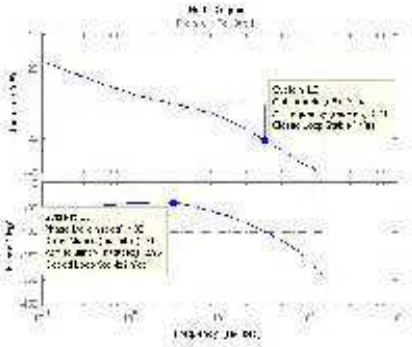


Figura 15. Diagrama de Bode Sistema de Referencia

En este caso, la asíntota vertical del ciclo límite queda totalmente excluida y razonable, ya que el margen de retardo para este sistema es de apenas 2.95 muestras, peligrosamente

cerca a las 1.5 muestras de retardo. Aquí es imperativo calcular un nuevo controlador, pese a que el actual es funcional, con el objetivo de obtener un mejor desempeño.

VI. Conclusiones y Trabajo Futuro

La identificación realizada se puede considerar como muy aceptable de acuerdo a los resultados mostrados por las estrategias de control implementadas. La posibilidad de proponer ejercios similares a estudiantes de control (por lo menos en cuanto a la implementación de los reguladores), puede ser un ejercicio muy interesante que puede incrementar las habilidades que estos puedan adquirir. Es una prueba relativamente sencilla y puede cumplir apropiadamente con los objetivos educativos y didácticos originales.

Los controladores obtenidos son aceptables en general. Sin embargo, para mejorar la calidad de los controladores es imperativo mejorar el proceso de identificación, acompañándolo de un modelado exhaustivo del móvil, lo que daría no solo mejores muestras del orden que debe tener cada sistema, sino un significado físico más intuitivo a los modelos mostrados.

La implementación de los controladores digitales depende del nivel de escalamiento y la plataforma de desarrollo, lo que si bien plantea una limitante, representa un reto muy enriquecedor desde el punto de vista de aprendizaje, y podría incluirse dentro del ciclo de enseñanza de control.

Para mejorar la calidad de la comunicación vía Bluetooth, se deberá buscar un reemplazo al sistema montado en Matlab. Para esto se ha venido trabajando en plataformas más robustas como Python, con resultados prometedores.

Todos los datos y resultados obtenidos hasta aquí serán de fundamental ayuda para el montaje de las plantas restantes de la plataforma de enseñanza, y obviamente, en el resultado final de la aquí mostrada.

Referencias

[1] G. Franklin, J. Powell, M. Workman. *Digital Control of Dynamic Systems*. Addison Wesley Longman Inc., Third Edition, 1998.

[2] L. Ljung. *System Identification: Theory for the User*. Prentice Hall, 1987.

[3] S. Moon, D. R. [illegible], M. [illegible]. "*Process Control Kits: A Hardware and Software Resource*". 35th ASEE/IEEE Frontiers in Education Conference, October 19-22, 2005, Indianapolis, IN.

[4] D. [illegible], M. Arrington, L. Ledgewood. *Rotational Inverted Pendulum: A New Control Experiment*, Proceedings of the American Control Conference Seattle, Washington, June 1995.

[5] A. Lee. "*[illegible]*", Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005 Seville, Spain, December 12-15, 2005.

[6] O. Miglino, H. Lund, M. Cardaci. *La robótica como herramienta para la educación*, 2000.

[7] T. [illegible], J. Kujala, A. [illegible], M. [illegible]. *[illegible]*. [illegible] Technology Centre 2002-2003.

[8] B. [illegible]. "*Design for Control Systems*". Disponible en [illegible]

[9] Centro de Investigación y de Estudios Avanzados del IPN. "*[illegible]*", Departamento de Control Automático. Disponible en [illegible]

Asociación Argentina
de Mecánica Computacional

Mecánica Computacional Vol XXIX, págs. 2039-2060 (artículo completo)
Eduardo Dvorkin, Marcela Goldschmit, Mario Storti (Eds.)
Buenos Aires, Argentina, 15-18 Noviembre 2010

APLICACIONES DIDÁCTICAS EN CONTROL AUTOMÁTICO CON LEGO® MINDSTORMS NXT

Leonardo Bermeo[a], David Herrera[b] y Ricardo Ramirez[c,d]

[a]*LCA - Laboratorio de Controle e Automação, Programa de Engenharia Elétrica, COPPE, Universidad Federal de Rio de Janeiro, Sala I-148, Cidade Universitaria, Ilha do Fundão, CEP 21945-970, Rio de Janeiro, RJ, Brasil, lbermeo@poli.br, www.gmar.unal.edu.co*

Laboratorio de Control, Universidad Nacional de Colombia, Departamento de Ingeniería Eléctrica y Electrónica, Edificio 453, Sala 202, Bogotá, Colombia, lbermeo@unal.edu.co, dherrera@unal.edu.co, http://www.gmar.unal.edu.co

[b]*Grupo de Plataformas Robóticas UnRobot , Universidad Nacional de Colombia,Departamento de Ingeniería Mecánica y Mecatrónica, Of. 453-201, Ciudad Universitaria, Bogotá,Colombia, rramirezp@unal.edu.co, http://www.unrobot.unal.edu.co/*

[c]*Laboratorio de Robótica y Diseño de Maquinas Labloh, Programa de Ingeniería Mecánica, COPPE, Universidade Federal do Rio de Janeiro, Sala I-104, Cidade Universitaria, Ilha do Fundão, CEP 21945-970, Rio de Janeiro, RJ, Brasil, ramirez@poli.br, http://www.labrob.coppe.ufrj.br*

Palabras Clave: Síntesis de controladores, Control PID, Identificación de Sistemas, Educación.

Resumen.

En este trabajo se desarrolla una propuesta didáctica completa para un curso experimental en el tema de control automático, usando los módulos didácticos LEGO NXT. La propuesta incluye software, un conjunto de plantas experimentales y una metodología de trabajo. El software presentado está desarrollado en herramientas GNU. Las plantas experimentales están basadas en LEGO y tienen ventajas importantes en costo, flexibilidad y reciclabilidad. Algunas de las plantas presentadas son servomecanismo, péndulo invertido, robot móvil y sistema de viga y bola.

Complementariamente, se propone una metodología con la cual es posible alcanzar en un mismo curso práctico la integración de conceptos importantes como simulación, dinámica e identificación de sistemas, diseño mecánico, diseño de algoritmos de control y sistemas operativos de Computadores.

1. INTRODUCCIÓN

Existe un declarado interés mundial en presentar propuestas y proyectos para realizar en laboratorios de control virtual de acceso remoto, teniendo en cuenta las limitaciones en tiempo y espacio (Gawthrop y E.McGookin (2004); Benetrin (2005b,a); K.J. Åström (2004)). La conclusión de quienes trabajan en enseñanza de control es unánime: el área del control automático es ideal para la unificación praxis-teoría. El notable profesor Dennis Bernstein de la Universidad de Michigan ha puntualizado esta conclusión de manera contundente en la introducción de una edición especial del IEEE Control Magazine dedicada a la educación en control (Bernstein (2005a)).

> *"El control es un tema que tradicionalmente se enseña de una manera excepcional general. Paradójicamente, es un área que prospera solamente cuando las estudiantes pueden ver aplicaciones a través casos de estudio prácticos".*

Uno de los problemas relacionados con la enseñanza experimental de control es su costo. Los costos de los sistemas construidos por las empresas dedicadas a desarrollar distintos hacen difícil su acceso masivo en las universidades.

El propósito fundamental de la enseñanza de control experimental es que un estudiante recorra los pasos metodológicos de síntesis de un controlador (Bermeo y Díaz (2007)), como son:

Paso 1 *Implementación de la planta*. Se realiza la integración de los componentes constitutivos (mecánicos, mecánicos, electrónicos, etc) del sistema a ser controlado, sensores y actuadores.

Paso 2 *Modelamiento y/o Identificación*. En este paso se obtiene un modelo lineal continuo o discreto del proceso a controlar.

Paso 3 *Diseño preliminar*. Con el modelo obtenido y especificaciones dadas (en el tiempo ó en la frecuencia) para el sistema de lazo cerrado, se hace un diseño preliminar del controlador en el dominio continuo o discreto (Franklin et. al. (1997)).

Paso 4 *Discretización*. Si el controlador se diseñó en el dominio análogo se usa genéricamente la transformación bilineal. El objetivo principal de este paso es seleccionar cuidadosamente el tiempo de muestreo T, ponderando el tiempo de cálculo del procesador con la pérdida de margen de fase debida al retardo introducido por el controlador digital.

Paso 5 *Implementación digital en un lenguaje de alto nivel*. Normalmente se usa C y un sistema operativo de tiempo real (RTOS) (Lurie y Enright (2000)).

Paso 6 *Pruebas de Hardware*. En esta parte se realizan directamente las pruebas del controlador diseñado para todas las condiciones de operación previsibles del sistema. Las pruebas realizadas deben mostrar un cierto grado de similaridad entre las predicciones del paso 2. Lamentablemente, si existe mucha divergencia entre las predicciones del

Mecánica Computacional Vol XXXIX, págs. 2035-2052 (2021)

paso 3 y los resultados experimentales será necesario refinar el modelo del sistema desde el paso 1.

El propósito fundamental de la enseñanza de control experimental es que un estudiante recorra los pasos metodológicos de síntesis de un controlador de manera ágil y sin tener que abordar problemas secundarios (calibración de sensores, diseño de acoples eléctricos o mecánicos, etc) que desvíen la atención del problema central. La componente experimental de un curso de control debe perseguir dos objetivos básicos:

1. Aplicar sobre un mismo sistema varias estrategias de control para establecer comparativamente límites y ventajas de cada una.
2. Realizar montajes para sistemas de diferente dinámica y dominio físico. Además se deben incluir experimentos sobre sistemas con polos estables, resonantes, en el origen, con semiplano derecho, retardo y/o ceros no mínima fase.

Este artículo propone un curso experimental, que complementa la componente teórica de los cursos de control. La propuesta incluye software, material didáctico, modelos matemáticos y guías de construcción de los experimentos. Estas prácticas han sido probadas por dos años en los cursos de laboratorio de control de Ingeniería Eléctrica de la Universidad Nacional de Colombia.

Este artículo se estructura de la manera siguiente:

En la sección 2 se presenta brevemente las herramientas de software utilizadas y la plataforma Mindstorms. En la sección 3 se presenta una práctica introductoria para demostrar el diseño de los sistemas ON-OFF y PID en el control de un servomecanismo. En la sección 4 se describe una práctica más avanzada para el control de un robot móvil que contiene todos los pasos de la metodología de síntesis de un controlador expuesta anteriormente. Estos pasos van de la identificación paramétrica de sistemas a la implementación digital en C y el RTOS NXT-OSEK. En la sección 5 se describen todos los montajes instalados con los cuales se experimenta al final del curso: una versión del péndulo invertido (SEGWAY) y el clásico problema de viga y bola. Finalmente, en la sección 6 se hacen consideraciones acerca de los resultados obtenidos en los cursos impartidos.

2. HERRAMIENTAS USADAS

En esta sección se describen brevemente las herramientas utilizadas en el desarrollo de las prácticas propuestas. A nivel de software, se puede utilizar Matlab, ó, si se prefiere FOSS, Python y Octave para el diseño de controladores y de difusión de sistemas. La programación de del sistema embebido de LEGO® se obtenía con las herramientas de programación NXC y NXT-OSEK. A nivel de hardware, los experimentos son montados únicamente con componentes de LEGO® Mindstorms.

2.1. Matlab® y GNU Octave

Se usan para el diseño de sistemas y controladores, simulación y discretización de controladores.

2.2. Python

Tiene el mismo uso que MATLAB. Parte de este trabajo es el desarrollo de una librería en Pyt on enlazada con el software GUTAV para la identificación de sistemas, diseño, simulación y discretización de controladores. Esto para el caso de usar solamente software GNU, que los autores consideran muy importante en el campo educativo.

2.3. NXC

NXC (Not eXactly C) es un lenguaje de programación estándar para los Bricks NXT. Es un conjunto completo de funciones que permiten acceder al hardware del Brick, y que además ofrece una curva de aprendizaje muy rápida. Esto limitado a variables en punto fijo.

Con NXC se pueden escribir rutinas de control básico. La diferencia fundamental con el código C estándar, radica en las funciones creadas para el manejo de los Bricks y el tipo de datos que soporta (solo hasta un long).

2.4. NXT-OSEK

Es un sistema operativo de tiempo real (RTOS) para el Brick NXT basado en el estándar OSEK de la industria automotriz. Este provee un ambiente de programación en C, una API para los sensores y motores NXT así como para otros dispositivos, soporte para operaciones en punto flotante, y capacidad de ejecutar tareas múltiples en tiempo real.

Con nxtOSEK se pueden desarrollar filtros, controladores avanzados, y rutinas de identificación.

2.5. LEGO® Mindstorms

Existe un conjunto de piezas, motores, sensores y actuadores, desarrollado por LEGO, para el diseño e implementación de robots; inicialmente pensado para niños mayores de diez años. Sin embargo, la herramienta ofrece tal flexibilidad en la educación en tecnología, que su uso se ha extendido desde el nivel de secundaria hasta cursos de postgrado en ingeniería.

Las principales ventajas que presenta son su bajo costo y enorme flexibilidad teniendo en cuenta el número de experimentos que pueden obtenerse a partir de un mismo kit. En nuestro caso ha sido usado el kit LEGO MINDSTORMS® para educación. El kit incluye varios sensores, 3 servomotores y más de 700 piezas. El bloque embebido programable, que en el resto del artículo se denomina Brick, es basado en un procesador ARM de 32-bits con una capacidad de cómputo notable. Una descripción completa del kit diferente se encuentra en la página oficial (LEGO (2010)).

Revista Iberoamericana de Automática e Informática Industrial (2010) 1–8

Figura 1: Kit de extensión 9797 de LEGO®. Cortesía de S. Briceño.

3. ACCIONES FUNDAMENTALES APLICADAS EN EL CONTROL DE UN SERVOMECANISMO

Las prácticas propuestas son una introducción a las acciones básicas de control, y un ejercicio de profundización que incluye la identificación de sistemas y la aplicación de técnicas de cálculo y ajuste analítico de controladores digitales.

3.1. Servomotor: Acciones básicas de control

En esta práctica se busca comprender, en una primera aproximación intuitiva, las acciones básicas de control. Es una convención que la literatura denomina *acciones básicas de control a la acción ON-OFF* y *las acciones proporcional (P), integral (I) y derivativa (D)* (Åström y Wittenmark (1997)).

El experimento inicial propone el control de posición y velocidad angular de un servomotor de LEGO®. Por ser el ejercicio para empezar, no se introduce un modelo matemático de la planta (el servomotor) porque en una primera aproximación al problema del control realimentado es de carácter cualitativo. Su propósito es mostrar las acciones básicas de control, el efecto de la realimentación, el error estacionario, los ciclos límite, las perturbaciones, entre muchas otras nociones fundamentales.

Como parte del trabajo en laboratorio, los estudiantes construyen la planta (su construcción toma unos 20 minutos aproximadamente) utilizando un computador hídico a la sesión práctica. Una secuencia detallada de pasos de construcción puede ser generada mediante el software CAD LEGO® DIGITAL DESIGNER, provisto por el fabricante. Mediante el mismo software se produce la figura 2 que muestra el ensamble mínimo necesario para el primer experimento. El servomecanismo es estático, robusto y satisface los requerimientos para ser la planta de este experimento.

Figura 2: Servomecanismo para el primer experimento

3.1.1. Control ON-OFF

Se trata de la estrategia más simple de control en lazo cerrado (conocida desde los griegos), en la que se usa toda la energía correctiva disponible para llevar la salida del sistema al punto deseado.

$$u(t) = \begin{cases} u_{max} & \text{si } e(t) > 0 \\ u_{min} & \text{si } e(t) < 0 \end{cases} \quad (1)$$

Esta estrategia se programa fácilmente en un lenguaje de alto nivel; requiriendo un conocimiento elemental de programación. El código del algoritmo de control ON-OFF en NXC se muestra en el algoritmo 1, y en NXT-OSEK en el algoritmo 2. Tales códigos hacen parte del material didáctico entregado a los estudiantes para que, durante la sesión, ellos solo tengan que hacer algunas modificaciones de los parámetros; por ejemplo, el porcentaje de energía entregada al motor. Los estudiantes hacen las modificaciones, compilan y programan el BRICK NXT en un proceso que tarda un par de minutos, casi como si el ejercicio fuera hecho solo a nivel de simulación.

Un factor que merece consideración es que los estudiantes, desde esta fase inicial, están conscientes de que la realización de controladores no está un mundo pitagórico intangible de transformadas z o de Laplace, sino que es un problema práctico de programación que requiere de compiladores de alto nivel e incluso sistemas operativos en tiempo real como el NXT-OSEK.

Las implementaciones mostradas en el algoritmo 1 y en el algoritmo 2 son programadas en C. Solo que en el segundo caso se un sistema operativo de tiempo real que involucra definiciones adicionales como la duración y ejecución de las tareas. Desde un punto de

Mecánica Computacional Vol XXIX, págs. 2935-2953 (2010) 3045

Algorithm 1: Controlador ON OFF en NXC.

```
long r, e, umax, umin;

task main()
{
 umax=100;
 umin=  0;
 r=150;
 while(true)
        {
        y = MotorRotationCount(0);
        e = r - y;
        if(e>0){u=umax;}
        if(e<0){u=umin;}
        OnFwd(OUT_A,u);
        }
}
```

vista pedagógico es mejor iniciar con la implementación en NXC (algoritmo 1), pero introducir el uso de una herramienta avanzada como el NXT-OSEK muestra las ventajas de tener un RTOS para hacer control. Así mismo, los estudiantes van siendo preparados en el manejo de una herramienta que necesitarán para los proyectos más avanzados en el final del curso.

Como un ejemplo de los comportamientos dinámicos obtenidos en esta práctica, en la Figura 3 se muestran respuestas para diferentes valores máximo y mínimo en la energía de la señal de control (variando u_{max} y u_{min}), y sus efectos sobre el ciclo límite y el tiempo de respuesta del sistema. Aquí es importante resaltar que por medio del software desarrollado, el alumno puede obtener directamente una gráfica como la mostrada; sin embargo, lo más importante en esta etapa es la percepción directa de los fenómenos involucrados.

3.1.2. Control PID

La estructura de Control PID es predominante en las aplicaciones industriales. Más del 90% de las aplicaciones de control de procesos en la industria están basadas en el PID (Åström y Hagglund (2006)). Esta realidad implica que los futuros ingenieros que tengan que tratar con procesos controlados tengan conceptos muy elaborados de los efectos producidos por cada una de las acciones de control en un PID. El hecho de poder percibir y prever a un nivel intuitivo los efectos de cada acción, permite que se inicie adecuadamente el desarrollo de estos importantes conceptos.

El control PID implementado en el laboratorio es de la forma no interactuante (llamada a veces de textbook (Åström y Hagglund (2006)):

Algorithm 2: Controlador ON OFF en NXC.

```
#include "kernel.h"
#include "kernel_id.h"
#include "ecrobot_interface.h"
#include "math.h"

DeclareCounter(SysTimerCnt);
DeclareTask(Task1);

void user_1ms_isr_type2(void)
{
  StatusType ercd;
  ercd = SignalCounter(SysTimerCnt);
  if (ercd != E_OK)
  {
    ShutdownOS(ercd);
  }
}

int v=0;
int ref= 80;
double e, u, umin, umax;
umin = -100;
umax=100;

TASK(Task1)
{
  v = nxt_motor_get_count(NXT_PORT_A);
  e = (double)ref - (double)v;
  if(e>0){u=umax;}
  if(e<0){u=umin;}
  nxt_motor_set_speed(NXT_PORT_A,(int)u,0);
  TerminateTask();
}
```

$$u = k_p e + k_i \int_0^t e\,dt - k_d \frac{d}{dt} e \tag{2}$$

Esta no es la implementación usual en un controlador industrial ya que debe incluir muchos otros detalles, pero es la más adecuada para entender los efectos individuales de cada acción.

Las acciones integral y derivativa son aproximadas usando diferencias finitas:

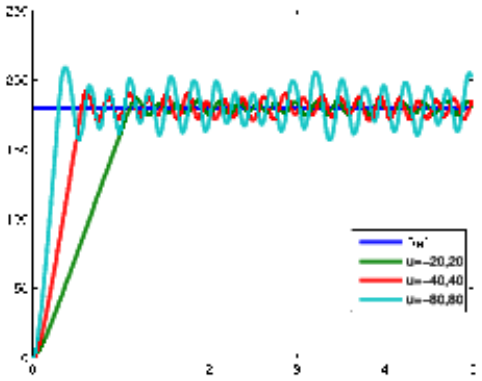


Figura 3: Respuesta del servomecanismo con control ON-OFF

$$u_d^k = k_d \Delta t \cdot e^k - e^{k-1} \tag{3}$$

$$u_i^k = k_i \Delta t \cdot e + u_i^k \tag{4}$$

Al momento de realizarlas a prácticas, no es necesario que los estudiantes tengan conocimientos en control digital. Los conocimientos necesarios para entender estas implementaciones ya han sido desarrollados previamente en los cursos de cálculo diferencial e integral de cualquier programa de ingeniería. En el primer caso (ecuación 3), por la aproximación de la derivada por la recta secante y en el segundo (ecuación 4), mediante el cálculo de integrales por métodos numéricos, usando la suma de Riemann. La acción integral está representada por la acumulación en la variable u_i que representa la señal de control. La acción derivativa u_d por la diferencia del error entre dos instantes de tiempo que pueden considerarse "cercanos" en la aproximación de la derivada. *Este es un buen momento y lugar para ver la acción, importancia y potencia de una herramienta vital del cálculo diferencial, en el mundo real.*

La implementación del control PID en NXC se muestra en el algoritmo 3, y en NXT OSEK en el algoritmo 4. Estos programas son entregados a los estudiantes como parte del material de la guía. Nótese que sintonizar unas nuevas constantes del PID es solo cambiar los números en las primeras líneas de código.

En la experimentación del servomecanismo controlado por un PID se explora el efecto conjunto y separado de cada una de las acciones de control. En la metodología propuesta, los estudiantes modifican cada una de las constantes, verifican la dinámica y en la misma aula de laboratorio sintonizan el controlador con buenos resultados dinámicos.

Debe notarse que reprogramar el Brick NXT para lograr un controlador PID con nuevas constantes tarda menos de 30 segundos. Esto permite que en la misma sesión se proponga sintonizar el PID hasta obtener diferentes comportamientos dinámicos. Dada la

Algorithm 3: Controlador PID en NXC

```
long kp, ki, kd, up, ui, Ts;
long e, r, y, u, ud, uiant, eant;

sub inicializa()
{
 SetSensor(S1,SENSOR_TYPE_ROTATION);
 ClearSensor(S1);
 r   180;
 Ts = 20;
 kp = 2;
 ki   1;
 kd = 1;
}

sub control()
{
 y   MotorRotationCount(OUT_A);
 e = r-y;
 up = kp*e;
 ui = uiant + ki*Ts*e;
 ud = kd*(e-eant)/Ts;
 u   up + ui + ud;
 if (u > 100) {u = 100;}
 if (u <   100) {u      100;}
 OnFwd(OUT_A,u);
 eant   e;
 uiant = ui;
}

task main()
{
 inicializa();
 while(true)
        {
        control();
        Wait(Ts);
        }
}
```

facilidad de programación y reprogramación del Brick, esto se hace con el mismo nivel de dificultad de un simulador, pero con la diferencia cognitiva profunda de percibir el funcionamiento real.

Las figuras 4, 5, y 6 muestran las respuestas del motor usando controladores P, PI, y

Mecánica Computacional Vol XXXIX, págs. 2035-2052 (2021) 2045

Algorithm 4: Controlador PID en nxtOSEK.

```
#include "kernel.h"
#include "kerne_id.h"
#include "ecrobot_interface.h"
#include "math.h"

DeclareCounter(SysTimerCnt);
DeclareTask(Task1);

void user_1ms_isr_type2(void)
{
  StatusType ercd;
  ercd = SignalCounter(SysTimerCnt);
  if (ercd != E_OK)
  {
    ShutdownOS(ercd);
  }
}

int v=0;
int ref= 80;
double e,up,ui,ud,u,uiant=0;
kp = 2; ki = 0.5; kd = 0.1;

TASK(Task1)
{
  y = nxt_motor_get_count(NXT_PORT_A);
  e = (float)ref - (double)y;
  up = kp*e;
  ui = uiant + ki*Ts*e;
  ud = kd*(e - eant)/Ts;
  u = up + ui + ud;
  if(u>100){u=100;}
  if(u<-100){u = -100;}
  eant = e;
  uiant = ui;
  nxt_motor_set_speed(NXT_PORT_A,(int)u,0);
  TerminateTask();
}
```

PID, respectivamente. En la misma sesión de sesión de laboratorio puede verse el efecto sobre el servomecanismo de aplicar una acción de control integral (error de estado estacionario cero, incremento de sobrepico), y una acción de control derivativa (reducir las oscilaciones, mejora el tiempo de respuesta). También se pide a los estudiantes reproducir comportamientos no deseados; por ejemplo, sin incluir la ganancia integral y una-

por ciclo) hasta llevar el sistema a la estabilidad marginal o la inestabilidad. El objetivo es caracterizar experimentalmente tanto los comportamientos deseados, como aquellos que deben evitarse.

Algunos resultados relevantes de la guía de experimentación sobre este experimento son:

- "Ajustar un control proporcional para una referencia de 180°. Aumente K_p hasta llevar el error de posición al 5%."
- "Ajustar un controlador PI para una referencia de 720°, de manera que la respuesta en lazo cerrado sea suave (sin oscilaciones)."
- "Ajustar un controlador PID para una referencia de $-$ 80, de manera que la respuesta en lazo cerrado sea rápido y con muy poca oscilación."
- "Verificar y explicar el comportamiento si la señal medición fuera de signo contrario, es decir cuando $e = r + y$"

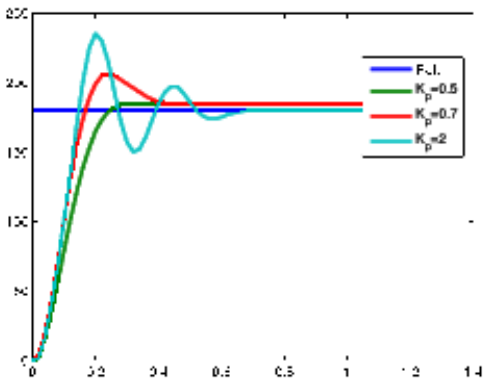


Figura 7: Respuesta del servomecanismo para diferentes valores de K_p

3.2. Control de Velocidad

Ahora se busca controlar la velocidad angular del servomotor. Esto podría hacerse con el mismo ensamble mecánico anterior; sin embargo, para estudiar los efectos de los controladores en una aplicación algo más interesante, se construyó el pequeño robot mostrado en la Figura 7. Nuevamente la estructura del robot agrega una componente difícil a la realización del experimento.

El robot incluye un sensor de ultrasonido para detectar obstáculos. Las variables de salida son la velocidad lineal estimada del robot, y la posición medida con respecto a una pared fija que se detecta con el sensor de ultrasonido.

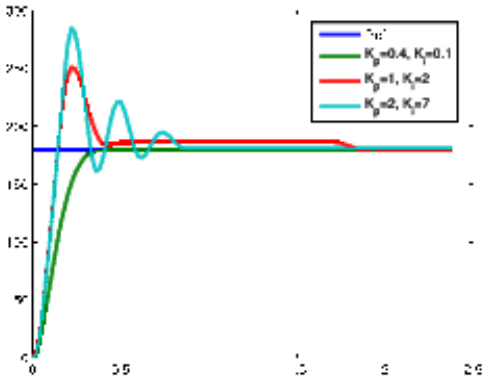


Figura 5: Respuesta del sistema con control PI

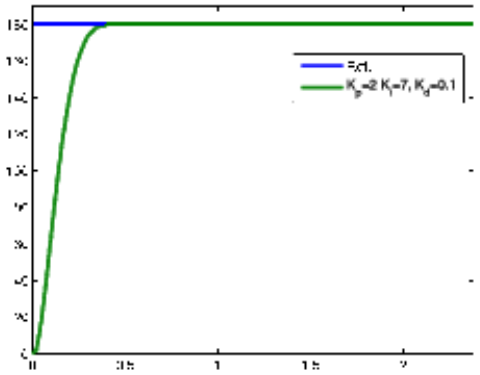


Figura 6: Respuesta del sistema con control PID

La velocidad se estima a partir de las medidas de posición angular:

$$\omega^k = \frac{\theta^k - \theta^{k'}}{\Delta t} \tag{5}$$

Al hacer la aproximación de la ecuación 5, se producen errores numéricos típicos en la estimación numérica de derivadas. Por esta razón se observa una ligera oscilación en la respuesta de estado estable. Los autores consideran de gran utilidad la baja calidad de la medición de la velocidad por aproximación de la derivada para ilustrar uno de los principios fundamentales de la teoría de la realimentación, conocido desde Black:

> *"El desempeño de un sistema de lazo cerrado está cuantificando y limitado por la calidad del sensor".*

Figura 7: Pequeño robot para control de Velocidad

La Figura 8 muestra la respuesta del sistema con un controlador PI para velocidad. La oscilación sobre el estado estable es una consecuencia de la calidad de la estimación de la velocidad a partir de la derivada de la ecuación 5.

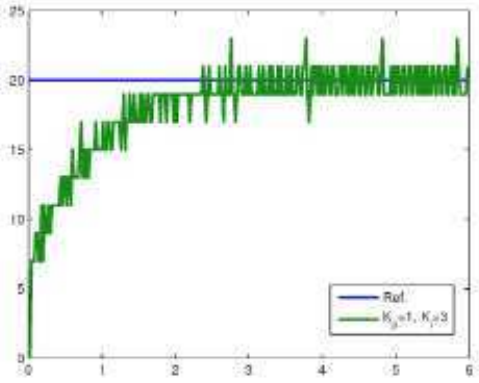


Figura 8: Vehículo con control de Velocidad (PI)

Mecánica Computacional Vol XXIX, pág. 2999-3009, 2010 2051

4. EXPERIMENTOS MÁS AVANZADOS

Para desarrollar controladores más avanzados que un PID, es necesario obtener un buen modelo de la planta. El método ARX de identificación de sistemas puede ser usado. Los fundamentos para usar este herramienta son los conceptos de control digital y la aproximación de mínimos cuadrados de un sistema de ecuaciones lineales sobredeterminado. Con estos conceptos se puede abordar la identificación de sistemas para obtener un modelo discreto de buena calidad $G(z)$.

La entrada $u(t)$ de la planta es el porcentaje de voltaje aplicado ($u(t) \in [0, 100]$) y la salida $y(t)$ es la velocidad del robot, estimada a través de la posición detectada por el sensor de ultrasonido. Es parte de este trabajo la programación de tareas que entregan una señal binaria pseudo-aleatoria o una onda "chirp" al robot, detectar la salida y enviar los datos a un computador vía USB o bluetooth, para su posterior procesamiento.

4.1. Identificación de Sistemas

En este experimento se usa una señal binaria pseudo aleatoria RBS (Ljung (1987)), que varía entre los niveles máximo y mínimo de PWM del motor NXT y con una frecuencia inferior a unos 5 Hz. La señal de identificación se muestra en la figura 9. El tiempo para ejecutar este experimento es cerca de un minuto. La señal se almacena en un archivo de texto en el Brick para ser enviada al servomotor, y al mismo tiempo el programa descargado crea otro archivo de texto con las muestras de posición del motor. En este caso, el uso del Brick para obtener los datos de entrada-salida del servomotor, elimina la necesidad de usar tarjetas de adquisición de datos dedicada. Los archivos de texto con los datos de entrada-salida de la planta se transfieren a un computador vía USB o BLUETOOTH.

Los datos se capturan y envían al PC en un archivo de texto CSV, que puede ser fácilmente leído en MATLAB u OCTAVE para usar las funciones especializadas de identificación de sistemas.

En el laboratorio se hace uso del toolbox de identificación de sistemas de Matlab (ident) o del Sys con Identification Toolbox de Octave. Este software permite obtener diferentes tipos de modelos a partir de los datos de entrada y salida del sistema.

Elegimos un modelo sencillo y que se ajuste a la forma de la función de transferencia típica de un sistema mecánico sin componente estática. La función de transferencia en posición es

$$G(s) = \frac{0.18536}{s(1 \quad 0.002695s)} \tag{3}$$

4.2. Implementación del Controlador

Se diseña un controlador basado en el modelo de la ecuación 3 para cumplir características de sobrepico y tiempo de establecimiento. Las especificaciones de diseño fueron un sobrepico y tiempo de establecimientos grandes (para poder observarlos en el vehículo). El controlador obtenido es

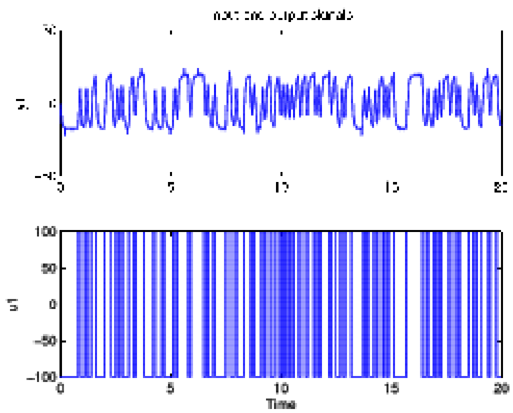


Figura 9: Parte inferior: onda PRBS usada para identificación. Parte superior: salida $y(t)$ del sistema

$$C(s) = \frac{1,1224(s-0,4295)(s+7,2171)}{s(s+2,5091)}$$

Seguidamente se discretiza el controlador y se lleva a su representación en variables de estado para convertirlo en un algoritmo en C. El sistema discreto obtenido es

$$C(z) = \frac{1,1224(z-0,9914)(z-0,8609)}{(z-1)(z-0,9195)}$$

La representación en variables de estado discreta para el controlador $C(z)$ está dado por

$$\begin{aligned} x_1[k+1] &= x_1[k] + 1,4\,e[k] \\ x_2[k+1] &= 0,9495\,x_2[k] - 0,2874\,e[k] \\ u[k+1] &= 0,001891\,x_1[k] + 0,2874\,x_2[k] + 1,122\,e[k] \end{aligned}$$

Para evitar errores de cuantificación de coeficientes es mejor que la realización digital del controlador sea hecha en punto flotante. Para ello se usa el sistema operativo de tiempo real NXT-OSEK que incluye tipos de datos float de 32 bits. El código es muy sencillo y es equivalente para cualquier controlador implementado con esta herramienta (Algoritmo 5). El estudiante puede modificar fácilmente las líneas que definen el controlador y el tiempo de muestreo para probar muchos otros controladores. Esto toma apenas unos minutos, no implica un conocimiento profundo de programación y deja claras las problemas y las herramientas necesarias que se deben tener en cuenta al enfrentar un problema de control.

En la figura 10 se aprecia la comparación entre la respuesta del sistema, comparada con la respuesta simulada. El sistema responde razonablemente de acuerdo con lo proyectado. Es de resaltar la rapidez y facilidad de implementación digital. Por otra parte se comprueba experimentalmente la validez del modelo de $G(s)$ en la expresión 6, obtenido en

Mecánica Computacional Vol XXIX, pags. 3039-3060 (2010) 2055

Algorithm 5: Control PID en NXT-OSEK.

```
#include "kernel.h"
#include "kernel_id.h"
#include "ecrobot_interface.h"
#include "math.h"


DeclareCounter(SysTimerCnt);
DeclareTask(Task1);

void user_1ms_isr_type2(void)
{
  StatusType ercd;
  ercd = SignalCounter(SysTimerCnt);
  if(ercd != E_OK)
  {
    ShutdownOS(ercd);
  }
}

int y = 0;
int ref=180;
double e,x1,x2,xlast,x2last,u;

TASK(Task1)
{
// Controller code
  y = nxt_motor_get_count(NXT_PORT_A);
  e=(double)ref-(double)y;
  x1 = xlast + 1.4*e;
  x2=0.9*x2last - 0.2871*e;
  u=0.0029*xlast + 0.287*x2last + 1.122*e;
  nxt_motor_set_speed(NXT_PORT_A,(int)u,0);
  xlast = x1;
  x2last = x2;
  TerminateTask();
}
```

el procedimiento de identificación. Según el punto de vista de los autores de este artículo, se vitan para los estudiantes que inician su camino en la teoría de control, el hecho de obtener concordancias entre teoría-práctica y simulación-realidad. Esto crea confianza en los métodos de la teoría de control automático.

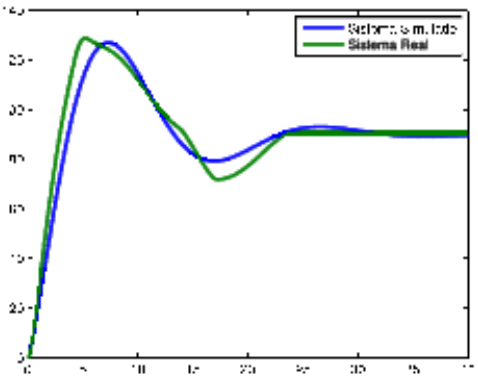


Figura 10: Respuesta con Controlador PID

En el material didáctico que acompaña estos experimentos se pide diseñar zonas de estabilidad, error en régimen, controladores algebraicos y de realimentación de estados. El ciclo completo de identificación, diseño y realización de controladores se puede cumplir eficientemente usando el Brick de LEGO® con con las herramientas NXC y NXT-OSEK y los programas para control e identificación desarrollados en esta propuesta.

Finalmente, se muestran 2 experimentos que pueden ser usados para cursos de control avanzados y cursos de posgrado. Estos son implementados en NXT-OSEK debido a su capacidad de trabajar en punto flotante. Estos ejercicios incluyen el desarrollo de habilidades en el análisis y modelado de sistemas, estabilización de sistemas con polos en el semi plano derecho, diseño e implementación de controladores digitales y filtrado digital.

4.3. Ball & Beam

La figura 11 muestra una planta avanzada: la bola y la viga, donde se busca controlar la posición de la bola a lo largo de la viga. En este ejercicio se desarrolla una rutina de identificación de la planta y diferentes tipos de controladores. La implementación del controlador en NXT-OSEK se muestra en el algoritmo 6.

4.4. Segway

Otra planta interesante es el segway mostrado en la figura 12. La implementación se muestra en el algoritmo 7. El diseño mecánico de esta planta y software mucho más avanzado fue desarrollado en este artículo, el cual fue hecho por Takeshi Chikamasa [1].

[1] http://lejos-osek.sourceforge.net/

Mecánica Computacional Vol XXIX, pags. 3039-3060 (2010) 2057

Algorithm 6: Control del Ball & Beam en NXT-OSEK.

```
#include "kernel.h"
#include "kernel_id.h"
#include "ecrobot_interface.h"
#include "math.h"

DeclareCounter(SysTimerCnt);
DeclareTask(Task1);

void user_1ms_isr_type2(void)
{
  StatusType ercd;
  ercd = SignalCounter(SysTimerCnt);
  if(ercd != E_OK)
  {
    ShutdownOS(ercd);
  }
}

int y=0;
int ref= 25;
double e,x1,u;

TASK(Task1)
{
  y = ecrobot_get_sonar_sensor(NXT_PORT_S1);
  e=r-y;
  x1= 0.9607*x1 + 1.561*e
  u=-1.561*x1+1.52*e
  nxt_motor_set_speed(NXT_PORT_A,(int)u,0);
  TerminateTask();
}
```

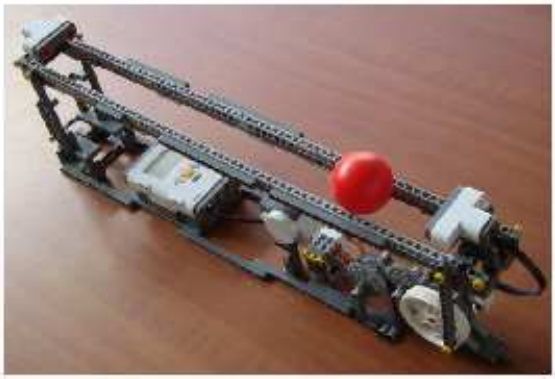


Figura 11. Bal & Beam con LEGO ©

En la página del autor se incluyen diseños realmente impresionantes como una bicicleta autoestable.

Figura 12. Segway con LEGO ©

Mecánica Computacional Vol XXIX, pags. 2039-2062 (2010) 2059

Algorithm 7: Control del Ball & Beam en NXT OSEK.

```
#include "kernel.h"
#include "kernel_id.h"
#include "ecrobot_interface.h"
#include "math.h"

DeclareCounter(SysTimerCnt);
DeclareTask(Task1);

void user_1ms_isr_type2(void)
{
  StatusType ercd;
  ercd = SignalCounter(SysTimerCnt);
  if (ercd != E_OK)
  {
    ShutdownOS(ercd);
  }
}

int y=0;
int ref=95;
double e,xl,u,linear_pos,ang_velocity,angle,scale;

TASK(Task1)
{
  scale=0.85
  y = ecrobot_get_sonar_sensor(NXT_PORT_S1);
  y2= ecrobot_get_gyro_sensor(U8 port_id);
  linear_pos = nxt_motor_get_count(NXT_PORT_A);
  linear_vel = estimate_vel(linear_pos);
  ang_velocity = passbandfilter(y2);
  angle=estimate_angle(ang_velocity);
  u=(0.8844*linear_pos + 38.1749*linear_vel + 8.55*angle + 1.09*ang_velocity)*s
  if(u>100){u=100;}
  if(u<-100){u=-100;}
  nxt_motor_set_speed(NXT_PORT_A,(int)u,0);
  nxt_motor_set_speed(NXT_PORT_B,(int)u,0);
  TerminateTask();
}
```

5. CONCLUSIONES Y TRABAJO FUTURO

Se reporta satisfactoriamente una alternativa de bajo costo para la experimentación en prácticas de control automático. Mediante el uso de esta herramienta se logra reducir el tiempo de un curso de experimental de control, pues hacer todos los pasos de un ciclo de diseño: *construcción de la planta, modelamiento e identificación, diseño e implementación de controladores y pruebas de hardware.*

Se ha diseñado una serie de planes para trabajar en un curso experimental de control, cuyas instrucciones y guías de construcción están disponibles (consultar a los autores). Las plantas diseñadas pueden ser realizadas con una caja estándar del set educativo de LEGO ®Mindstorms. El uso de LEGO ®Mindstorms ha mostrado ser una alternativa viable por costo y calidad para hacer cursos de control experimental.

Las prácticas mencionadas son realizadas actualmente en los cursos de control para pregrado de la Universidad Nacional de Colombia. La constante realimentación de información y la continua esfuerzo en el mejoramiento de estas prácticas ha generado un incremento significativo en el interés por parte de los estudiantes hacia la teoría de control y una mejor comprensión de los conceptos claves en el estudio de sistemas dinámicos.

La propuesta aquí presentada es un proceso permanentemente activo cuyos objetivos son: aumentar su cobertura, calidad y número de experimentos por semestre realizados en el curso de laboratorio de control.

REFERENCIAS

Bermúdez A. y Díaz H. Sistema para prácticas de control experimental. *Memorias del VII congreso de la Asociación Colombiana de Automática*, 2007.

Bernstein D.S. Innovations in undergraduate control education. *Control Magazine*, 25/1, 2006a.

Bernstein D.S. The quanser dc motor control trainer. *Control Magazine*, 25/1:90–98, 2006b.

Franklin G., Powell J., y Workman M. *Digital Control of Dynamic Systems*. Addison Wesley, 3a ed. Hill Road, U.S., 3rd ed'ición, 1997.

Gawthrop P. y E.McGookin. A lego based control experiment. *Control Magazine*, 24/3:43–56, 2004.

K.J. Astrõm J.A. A laptop servo for control education. *Control Magazine*, 24/3:70–73, 2004.

LEGO. Lego® mind-storms® education base set. Website, 2010. http://www1.lego.com/education/search/default.asp?l2id=0_1&page=7_1&productid=8797.

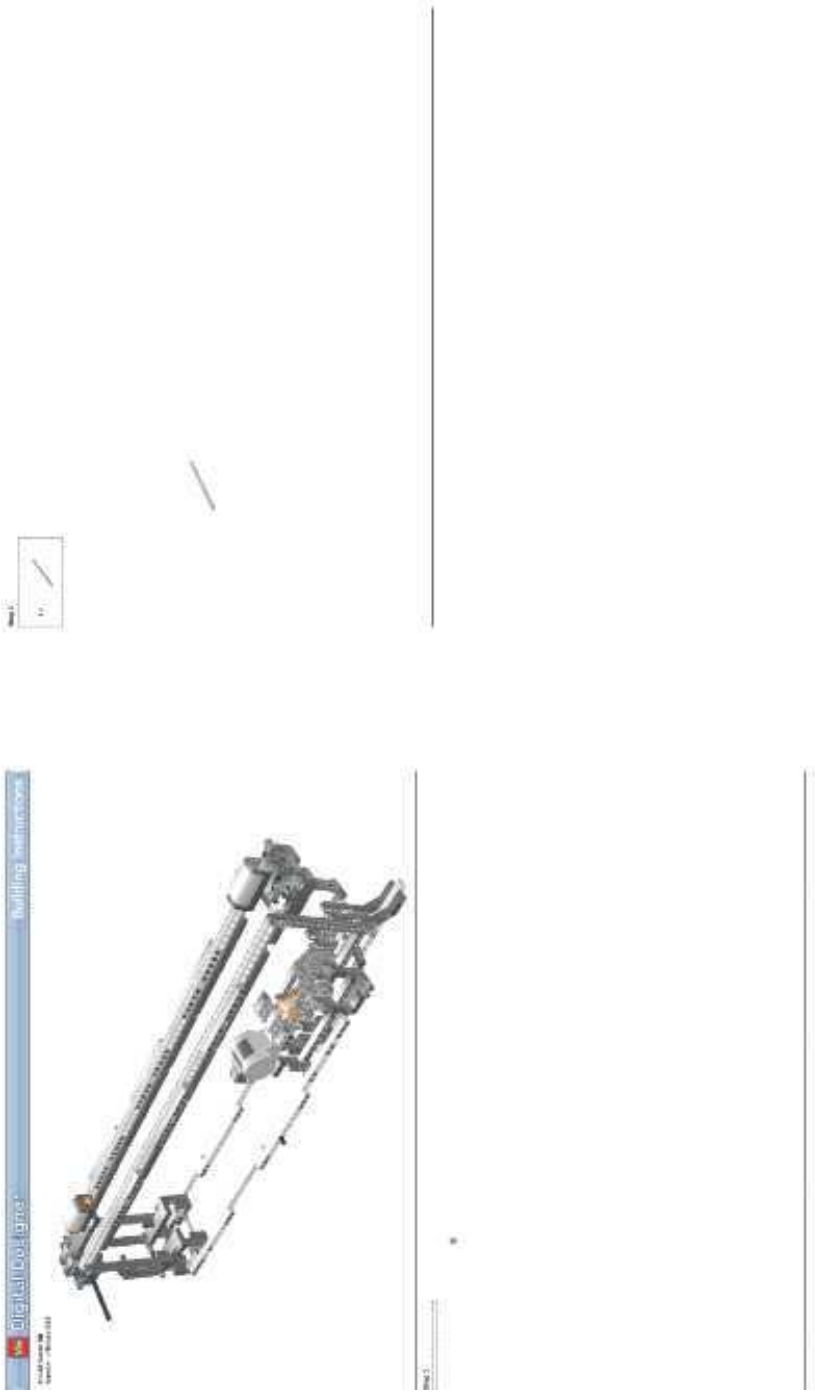
Ljung L. *System Identification Theory For User*. Prentice Hall, 1987.

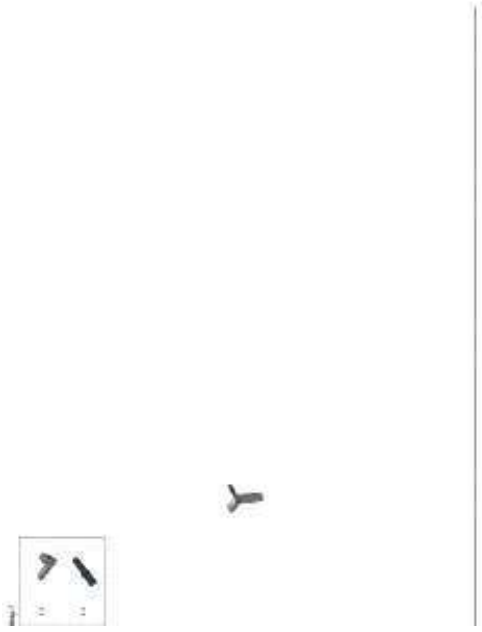
Lurie B. y Enright P. *Classical feedback Control with Matlab*. Marcel Dekker, New York, 2001.

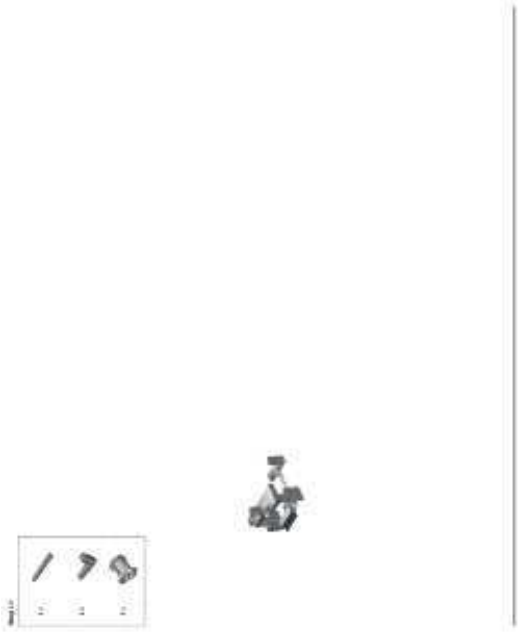
Astrõm K. y Hagglund T. *Advanced Pid Control*. ISA - Instrumentation, Systems and Automation Society, Research Triangle Park,NC 27709, U.S., 1rst edición, 2006.

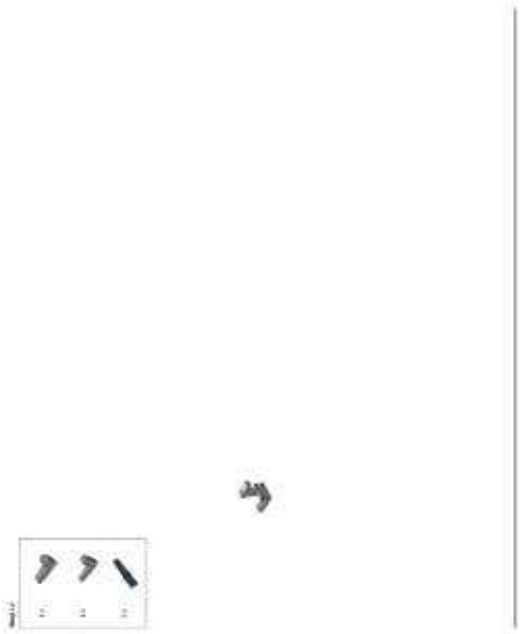
Astrõm K. y Wittenmark B. *Computer Controlled Systems*. Prentice Hall, Upper Saddle River, N.J., U.S., third edición, 1997.

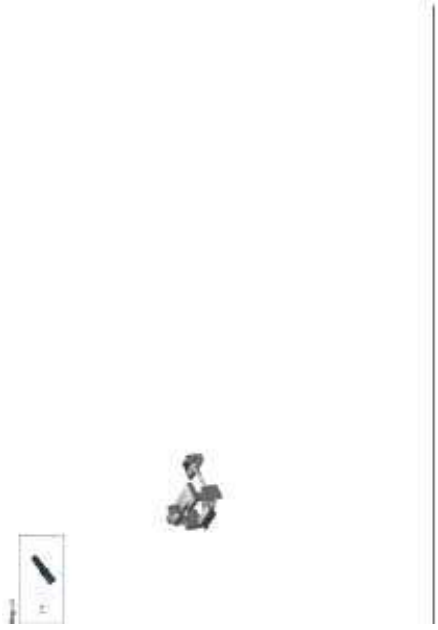
11. Anexos

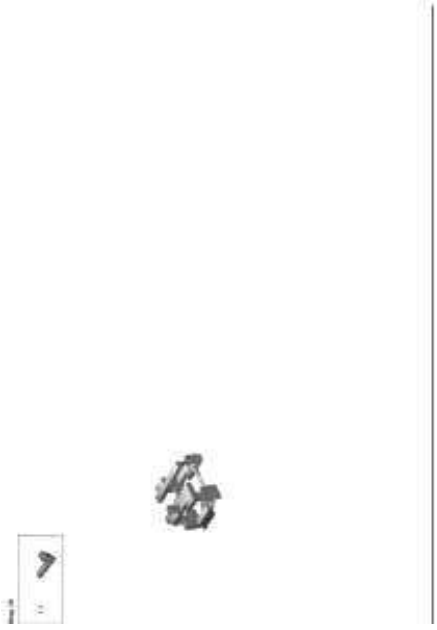


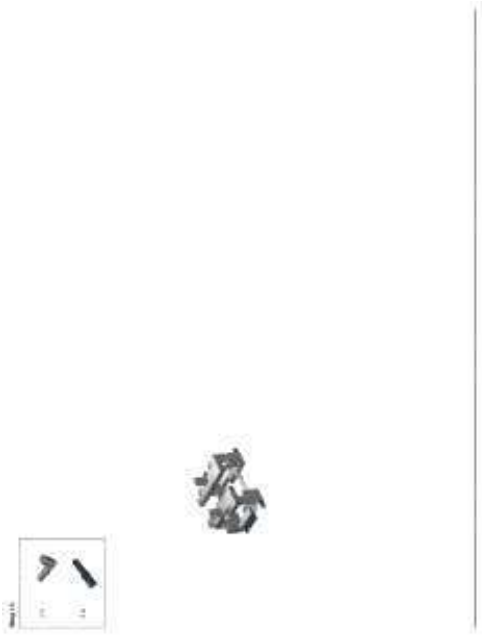


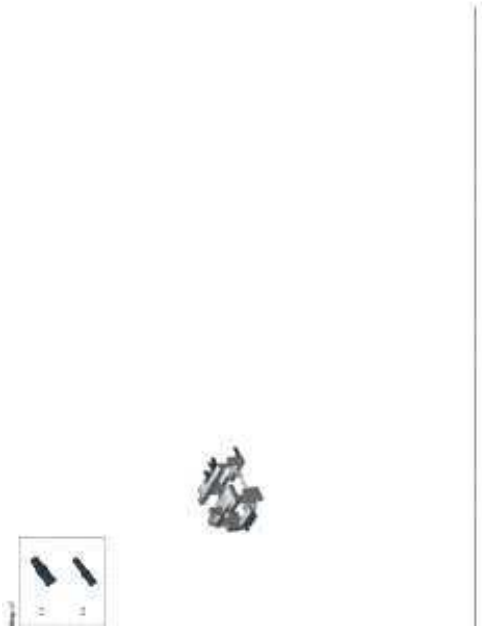


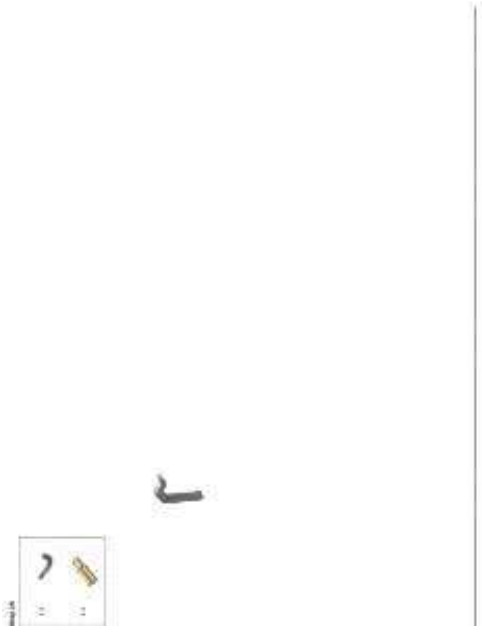


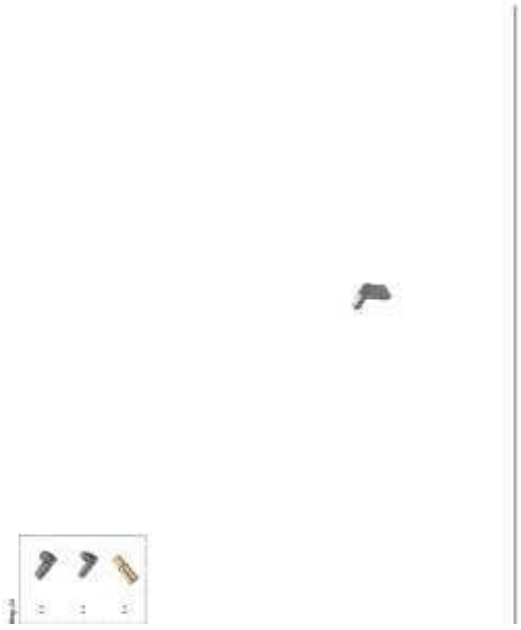


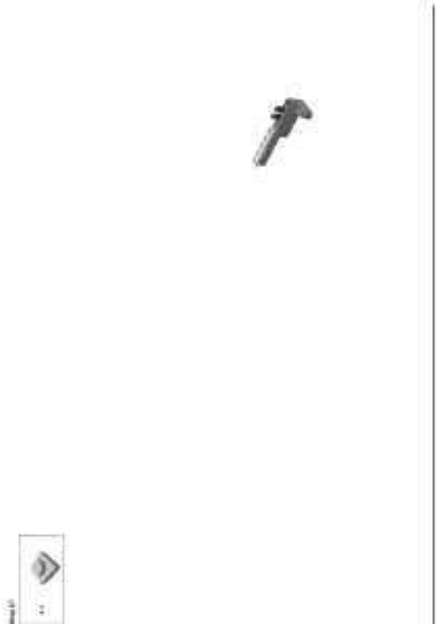


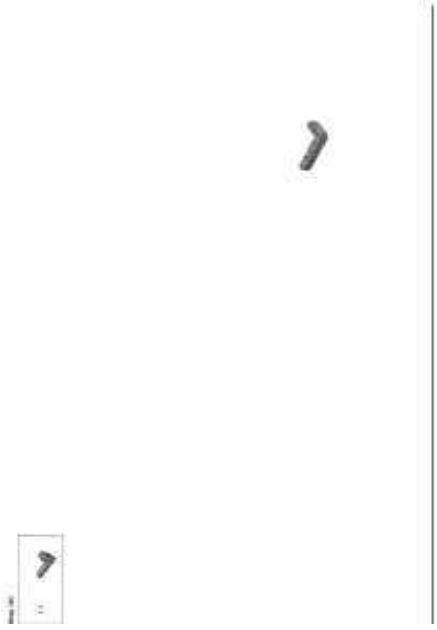


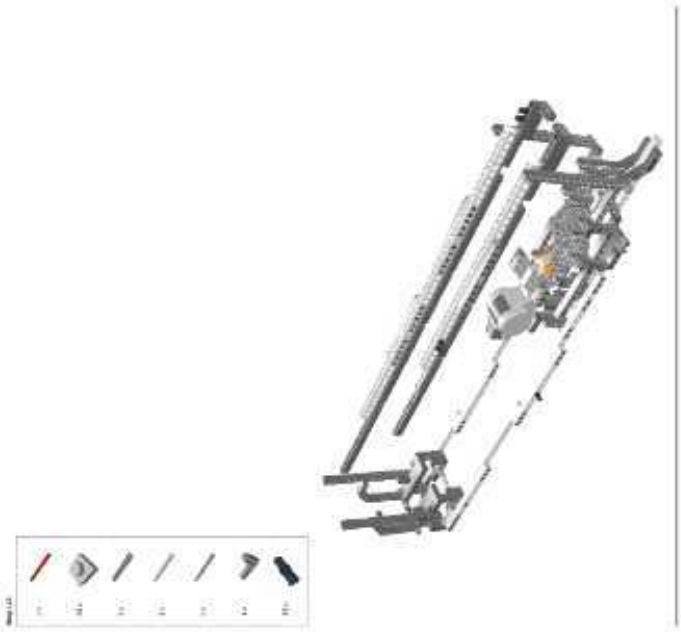


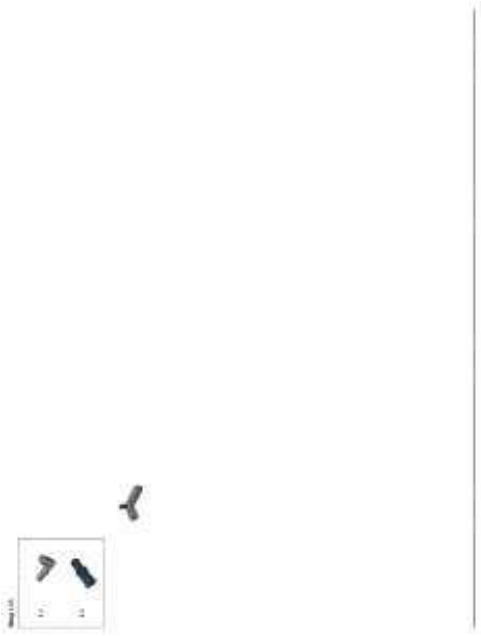


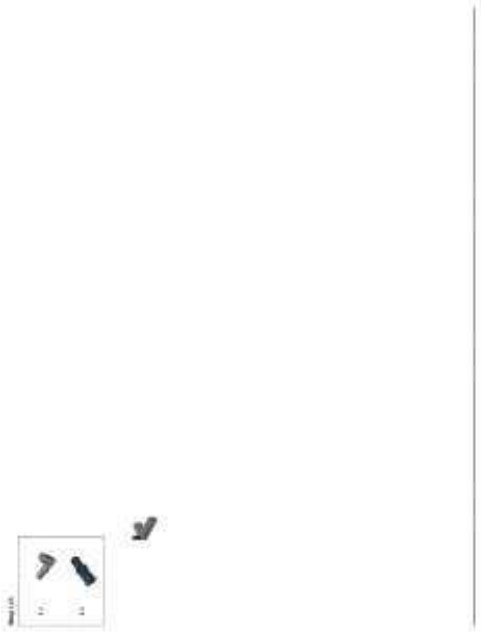


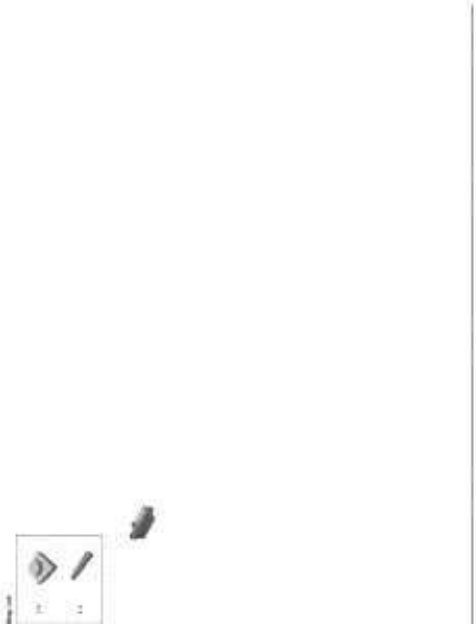


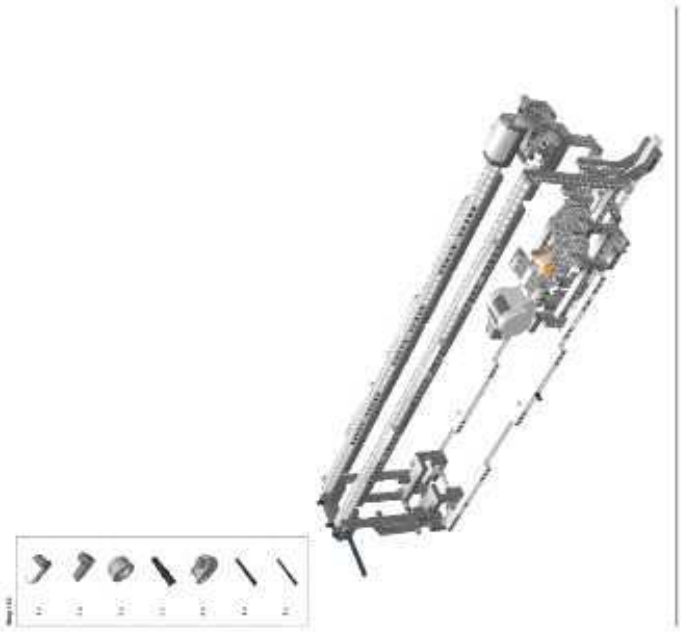




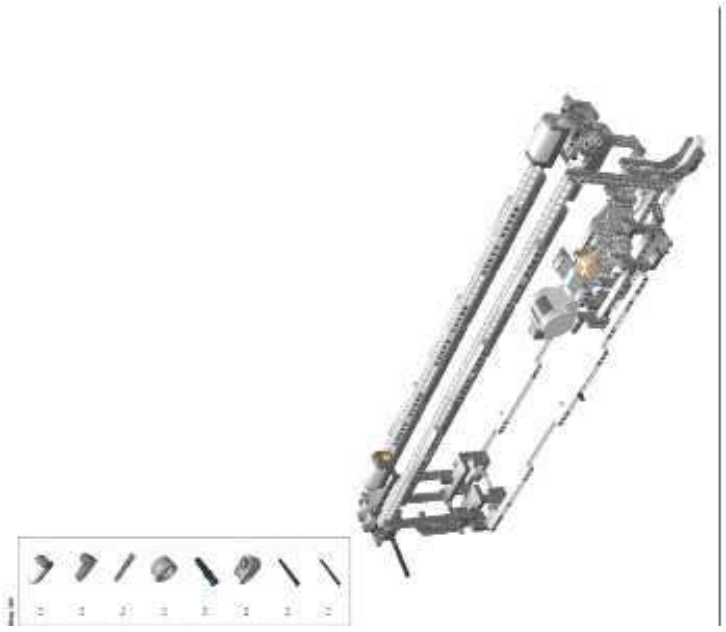


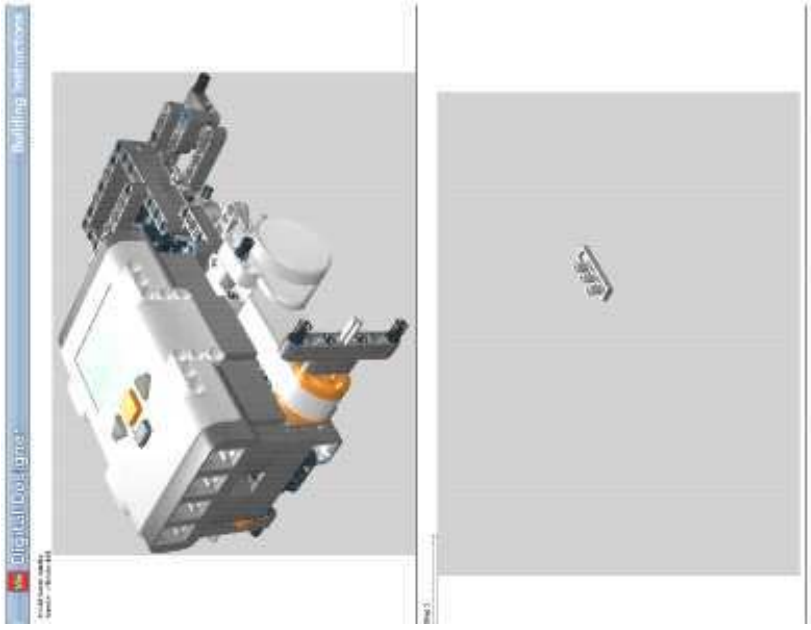




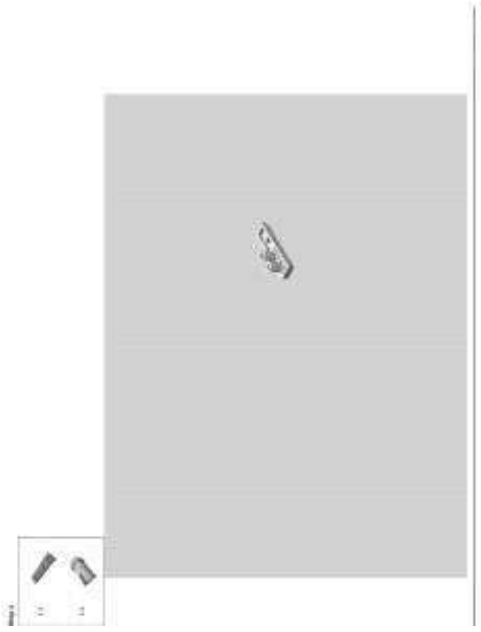


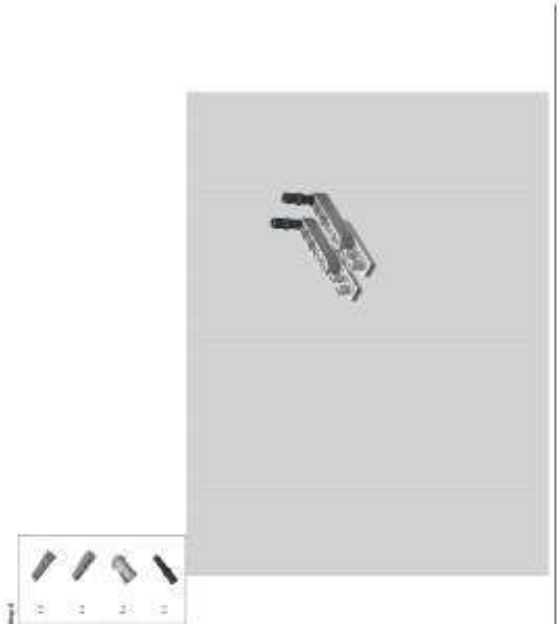
11. Anexos

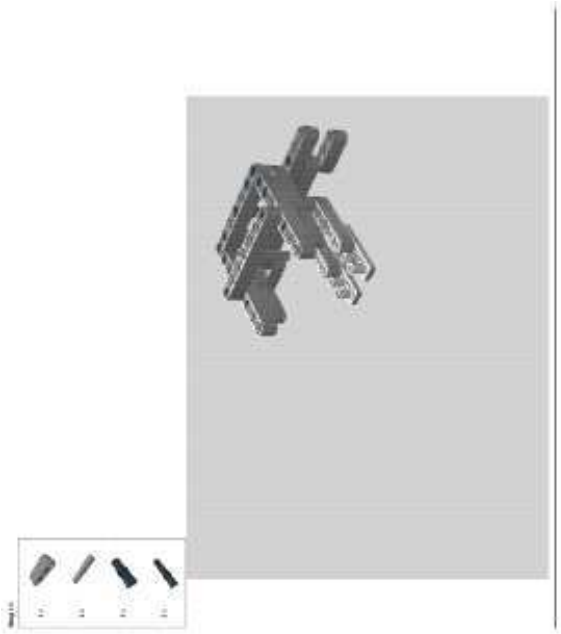


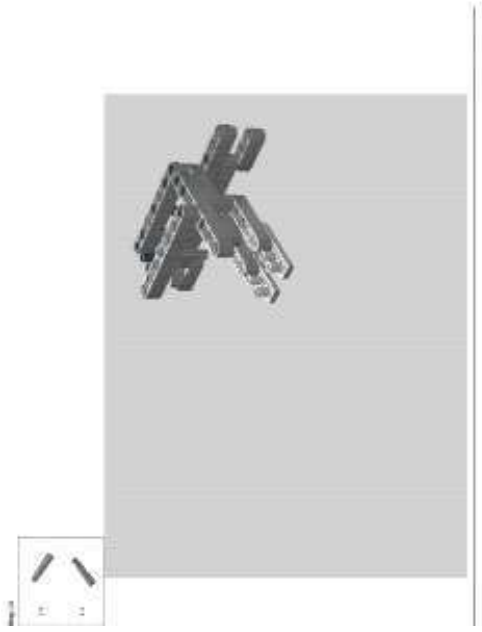


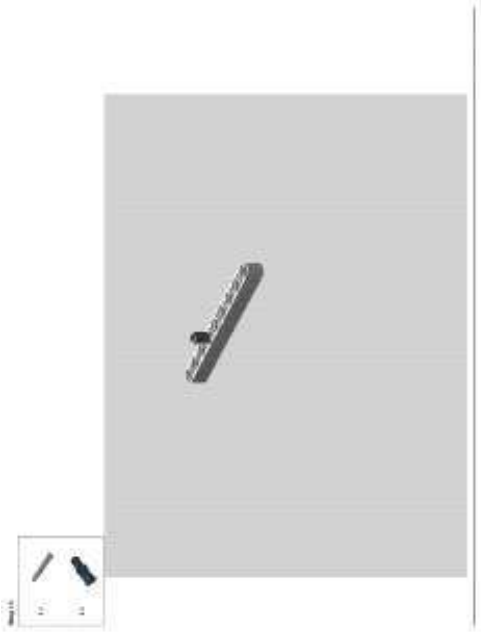
!Ht

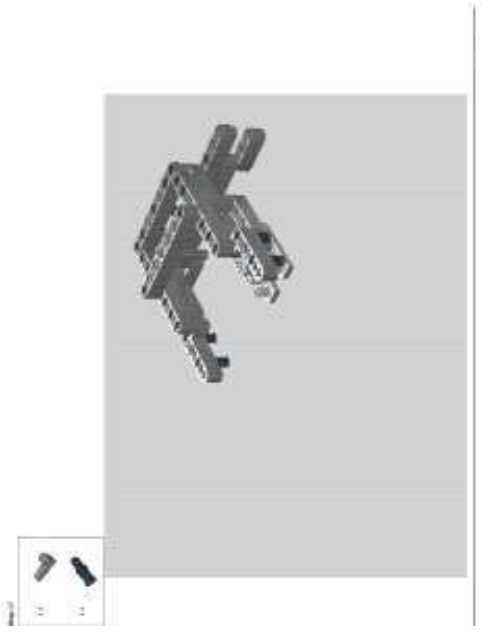


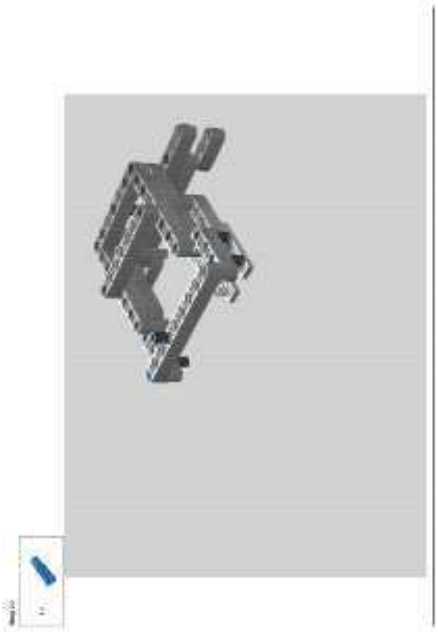


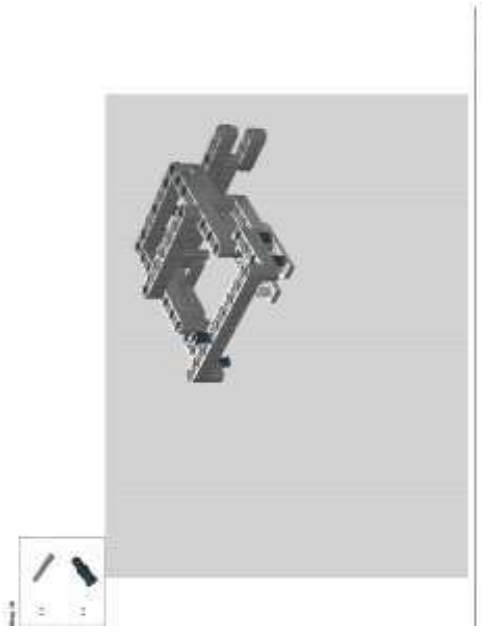


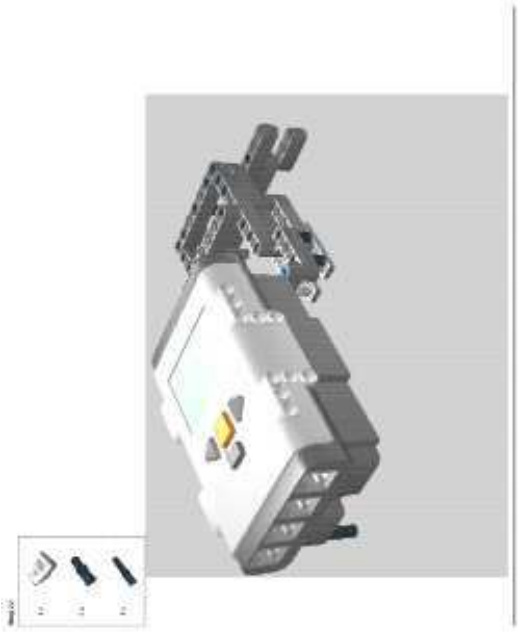


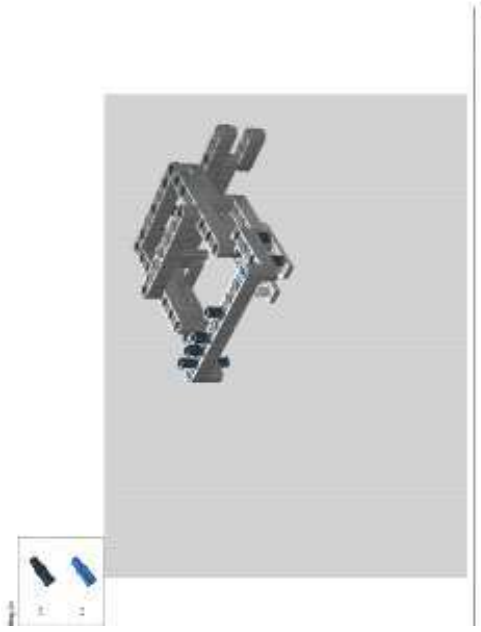


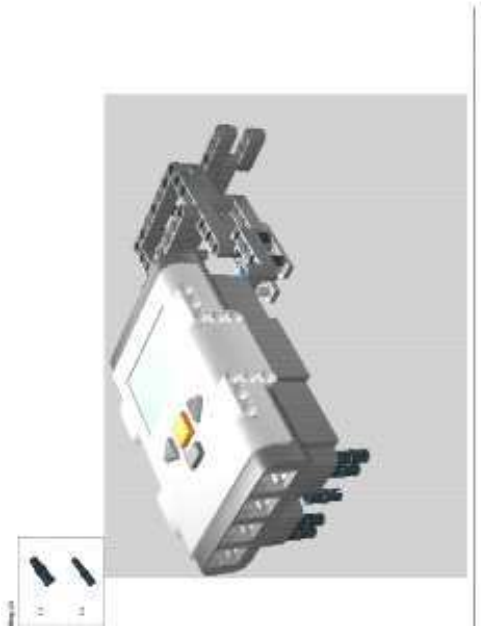


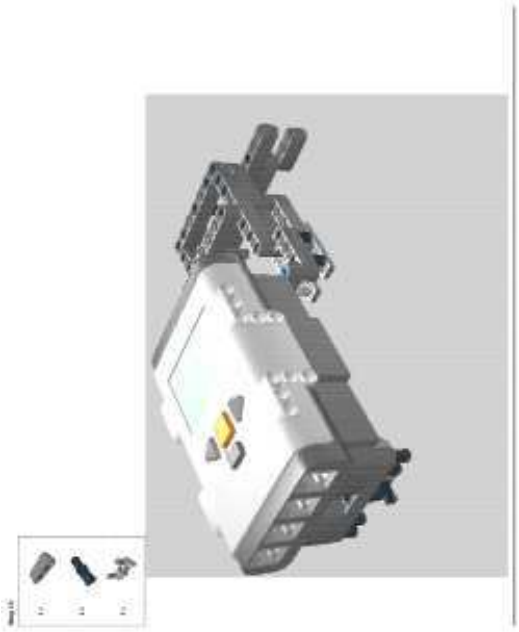


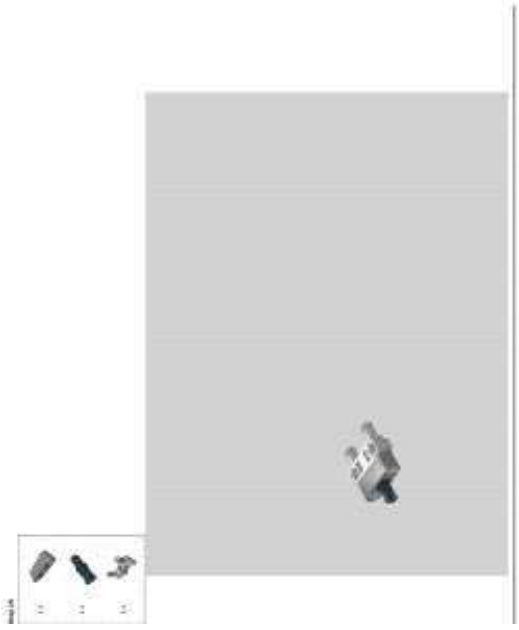


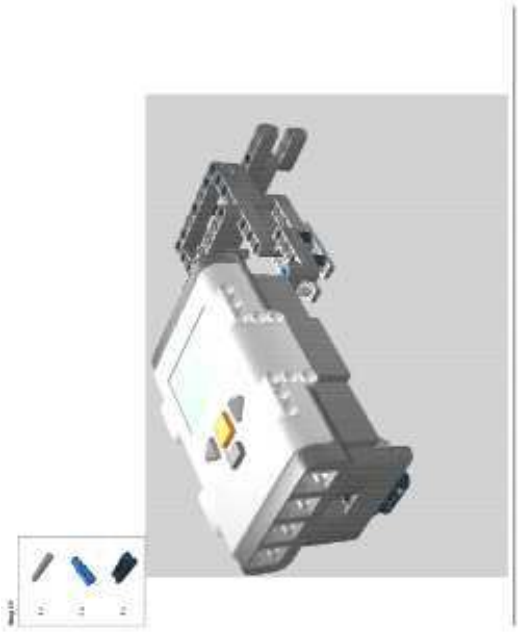


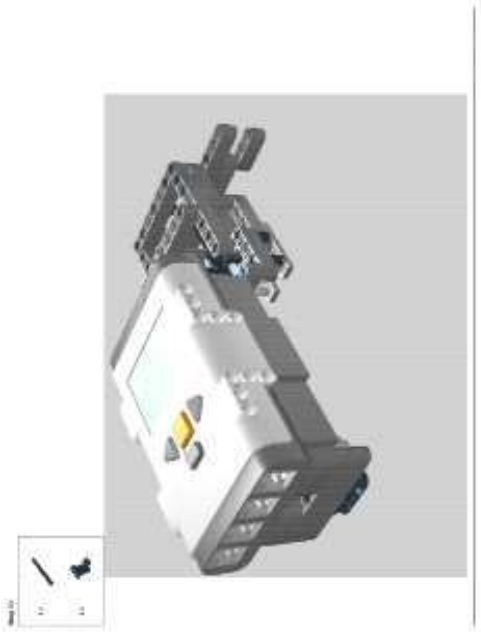


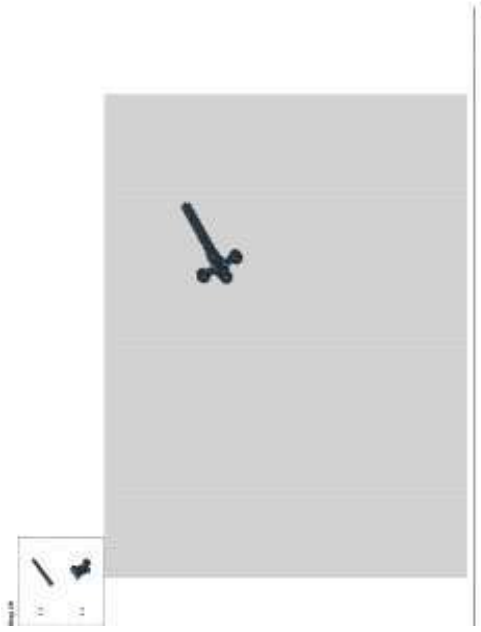


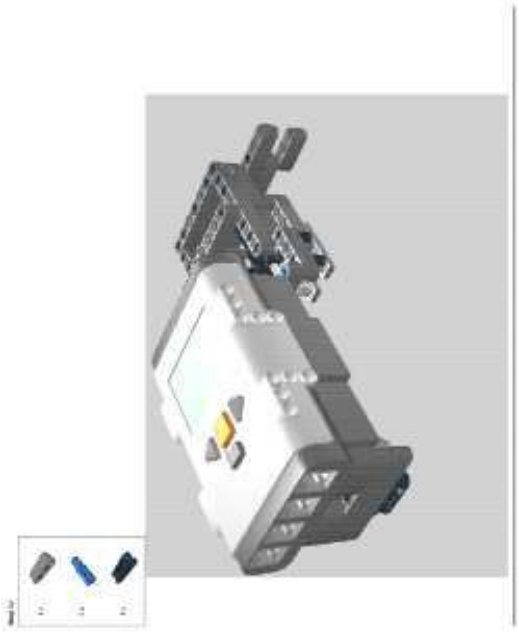


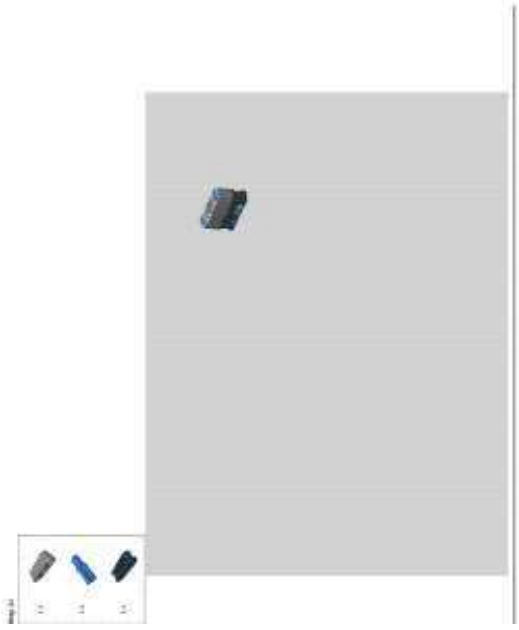


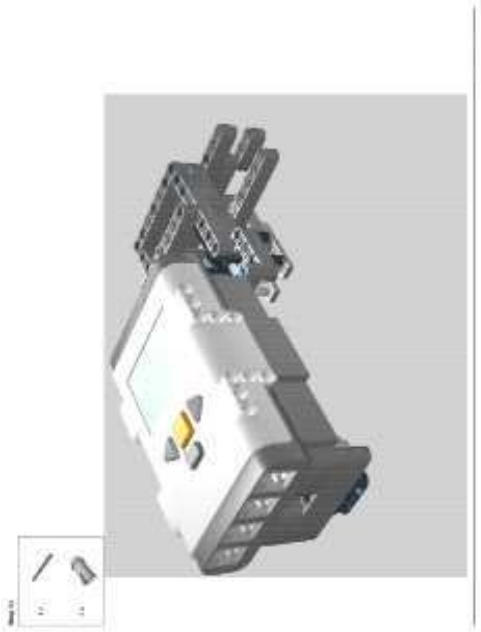


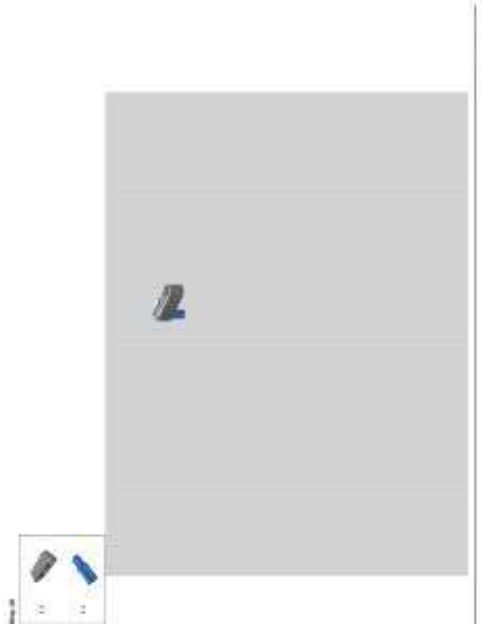


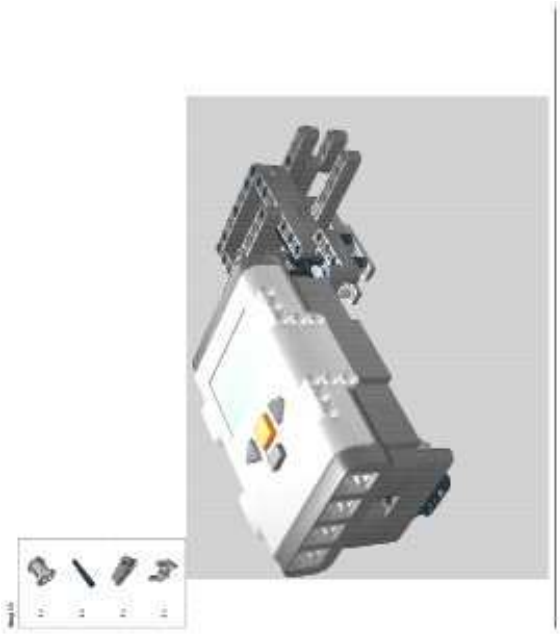


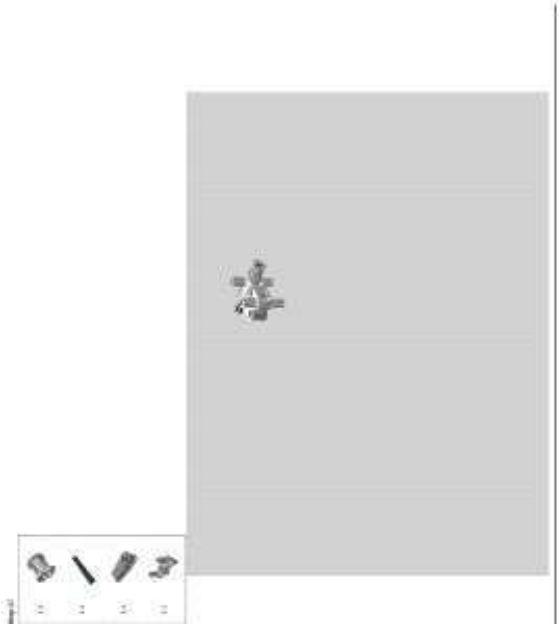


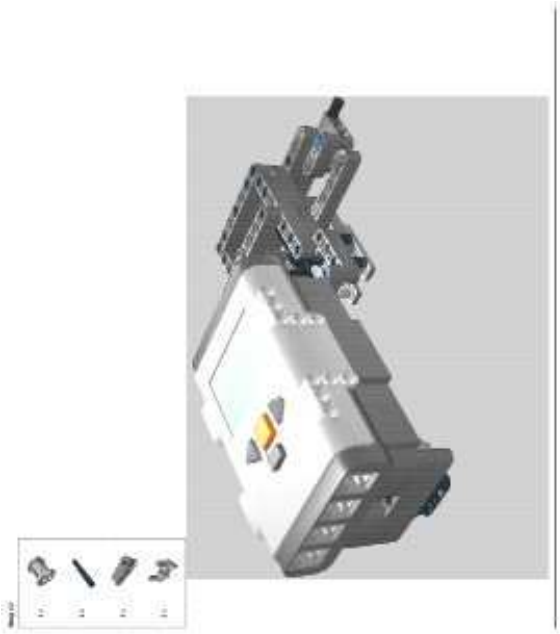


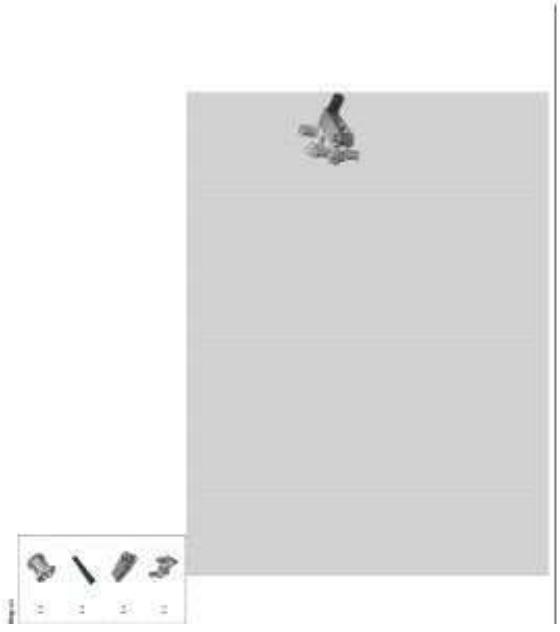


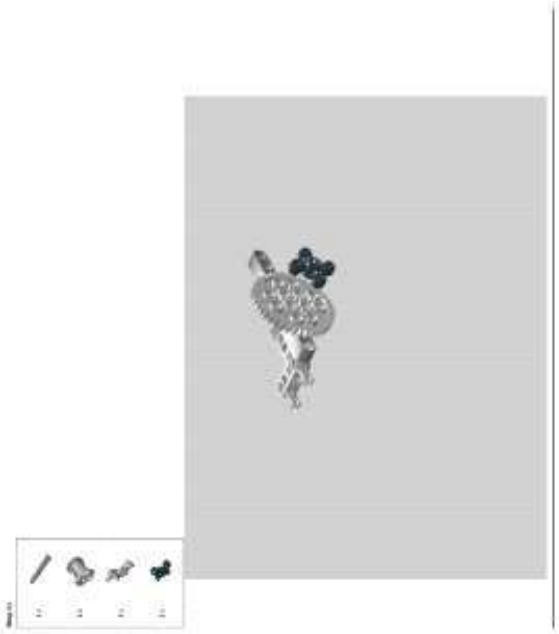


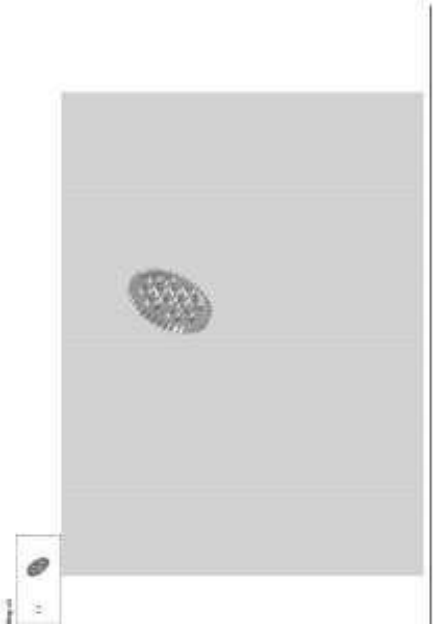


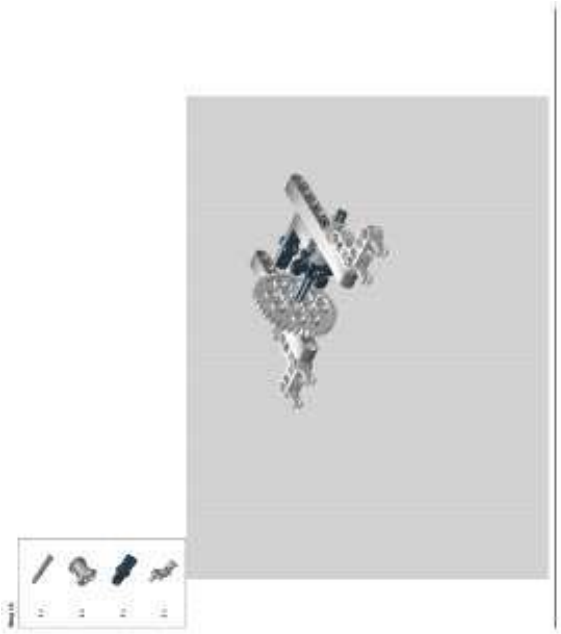


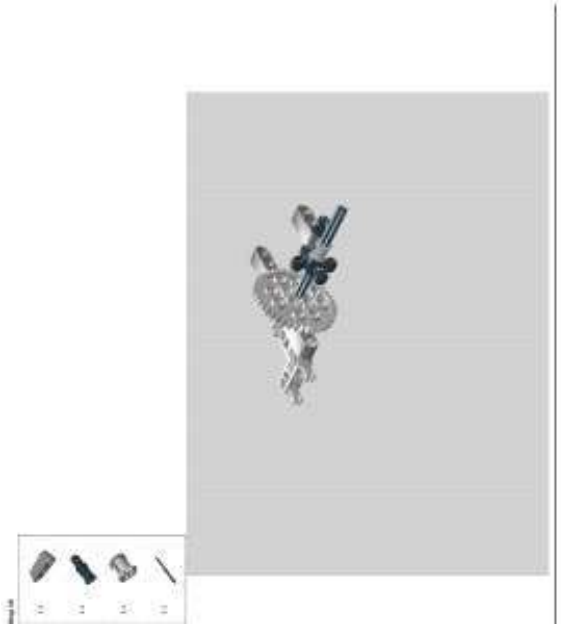


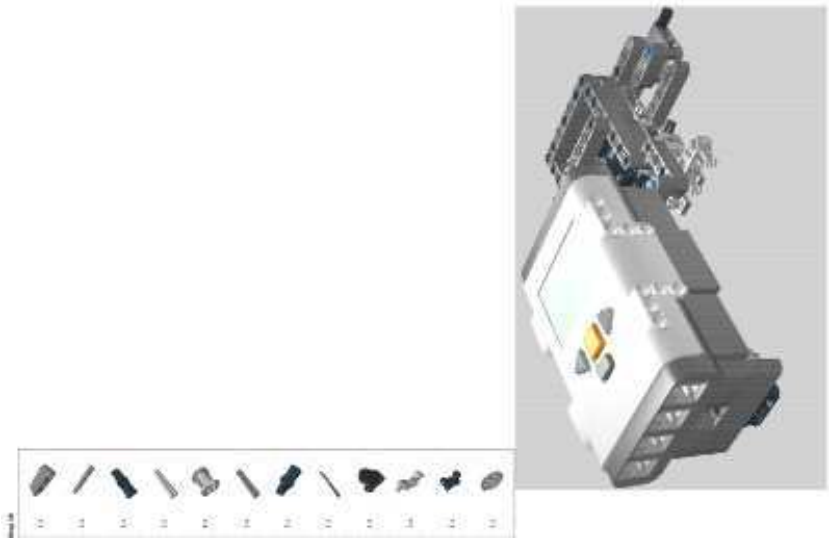


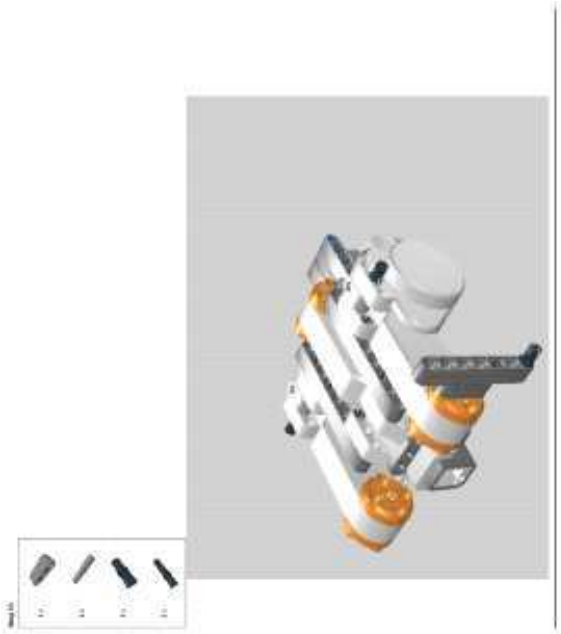


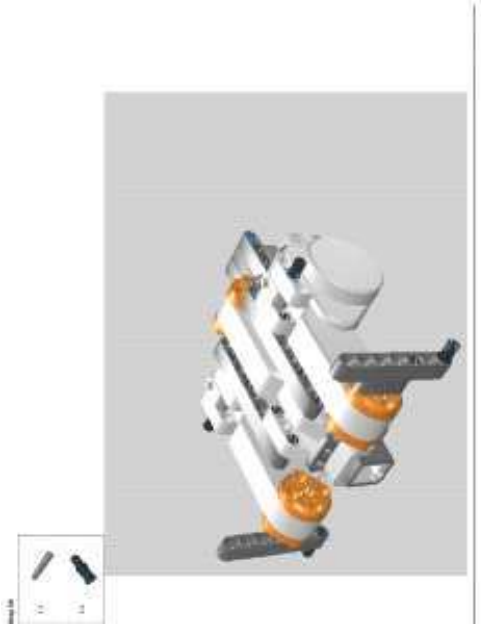


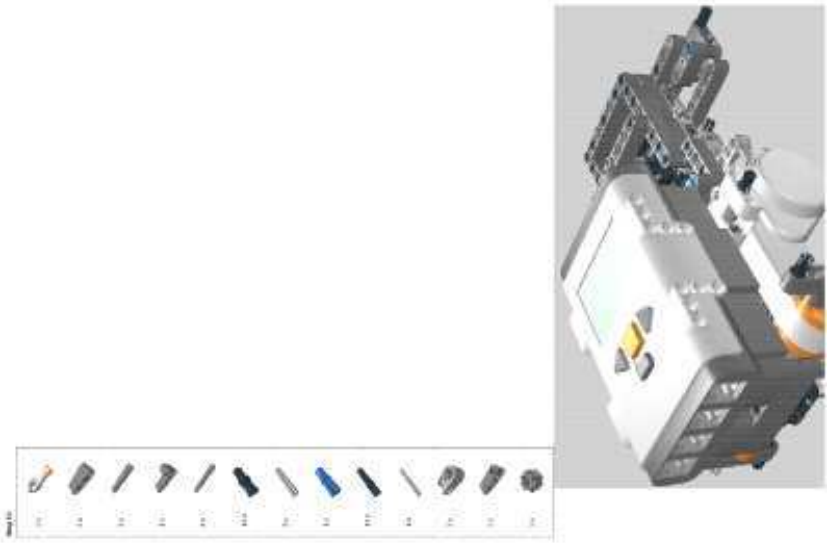


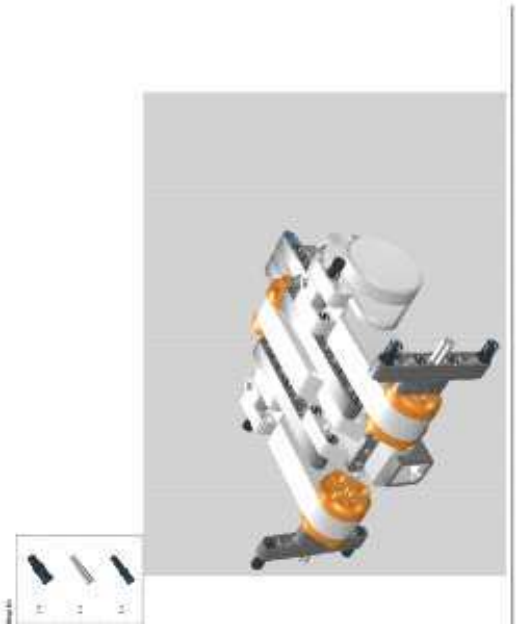












11. Anexos

Bibliografía

[1] L. Ljung, *System Identification Theory For User*., Prentice Hall, 1987.

[2] L. Ljung, T. Glad, *Modeling of Dynamical Systems*., Prentice Hall, 1994.

[3] I. Landau, G. Zito, *Digital Control Systems: Design, Identification and Implementation*., Springer-Verlag London Limited, 2006.

[4] K. Åström, B. Wittenmark, *Computer Controlled Systems.* Prentice Hall, Upper Saddle River, U.S., third edition, 1997

[5] B. Lurie, P. Enright, *Classical Feedback Control with Matlab*., Marcel Dekker, New York, 2000.

[6] G. Franklin, J. Powell, M. Workman, *Digital Control of Dynamic Systems*., Adisson Wesley, Sand Hill Road, US. Third Edition, 1997.

[7] K. Lilienkamp, *Lab Experiences for Teaching Undergraduate Dynamics*., Massachusetts Institute of Technology, Master Thesis, February 2003.

[8] S. Brennan, *Modeling and Control Issues associated with Scaled Vehicles*., University of Illinois at Urbana-Champaign, Master Thesis, 1999.

[9] S. Moor, P. R. Piergiovanni, M. Metzger *Process Control Kits: A Hardware and Software Resource*., 35^{th} ASEE/IEEE Frontiers in Education Conference, October 19-22, 2005, Indianapolis, IN .

[10] K. McLeod, *APRIL (A Pid Robot Implemented with Lego)*., University of British Columbia, December, 2007.

[11] A. Leva *Turning a toy into a didactic industrial regulator*., Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005 Seville, Spain, December 12-15, 2005.

[12] E. Misawa, M. Arrington, T. Ledgerwood *Rotational Inverted Pendulum: A New Control Experiment* ., Proceedings of the American Control Conference Seattle, Washington, June 1995 .

[13] B. Heck, N. Scott, A. Ferri. *A LEGO Experiment for Embedded Control System Design.,* IEEE Control Systems Magazine, October 2004.

[14] T. Lukka, J. Kujala, A. Soukka, M. Katila *Fast, Cheap and In Control.,* Agora Human Technologies Center 2002-2003.

[15] C. Vibet *Control Teaching via Low Cost Setups.,* IEEE TRANSACTIONS ON EDUCATION, VOL. 31, NO. 3, August 1994.

[16] C. Deltheil, D. Leandri, E. Moreau, I. Yousef *Mechatronics Training on Source Separation Analysis Using a Gyroscopic Motion of a LEGO,s Robot.,* IEEWASME International Conference on Advanced Intelligent Mechatronics Proceedings 8-12 July 2001 Como, Italy.

[17] D. Benedettelli, N. Ceccarelli, A. Garulli, A. Giannitrapani *Experimental validation of a decentralized control law for multi-vehicle collective motion.,* Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, USA, Oct 29 - Nov 2, 2007.

[18] W. White, M. Foss, X. Guo *Educational Robotics in a Systems Design Masters Program.,* Proceedings of the 2007 American Control Conference, New York City, USA, July 11-13, 2007.

[19] P. Fiorini *LEGO Kits in the Lab.,* IEEE Robotics & Automation Magazine, December 2005.

[20] O. Miglino, H. Lund, M. Cardaci. *La robótica como herramienta para la educación.,* 2000.

[21] P. Gawthrop, E. McGookin. *A LEGO-Based Control Experiment.,* IEEE Control Systems Magazine, October 2004.

[22] M. Resnick, F. Martin, R. Sargent, B. Silverman. *Programmable Bricks: Toys to think with.,* IBM SYSTEMS JOURNAL, VOL 35, NOS 3&4", 1996.

[23] M. Gasperi, P. Hurbain. *Extreme NXT.,* Apress, 2007.

[24] E. Papadoupoulos, V. Papadimitriou *Putting Low Cost Commercial Robotics Components to the Test.,* IEEE Robotics and Automation Magazine, September 2007.

[25] Y. Yamamoto *NXTway-GS Model-Based Design-Control of self-balancing two-wheeled robot built with LEGO Mindstorms NXT-.,* CYBERNET SYSTEMS CO., LTD., 2008.

[26] K. Sultan, A. Mirza, *Inverted Pendulum: Analysis, Design and Implementation.,* IIEE Visionaries, Document Version 1.0, 2003.

[27] National Centre for Technology in Education. *NCTE Advice Sheet - Programmable Bricks Advice Sheet 33.,* June 2007.

[28] A. Cardi, M. Wagner, *Control of an RC Segway*., 2005.

[29] P. Wellstead, *Ball and Beam 1: Basics*., Control Systems Principles.co.uk.

[30] L. Bermeo, H. Díaz, *Sideco: una propuesta para la enseñanza de control experimental*., Memorias del VII congreso de la Asociación Colombiana de Automática, 2007.

[31] G. Nieves Molina, *Estudio de las posibilidades didácticas en ingeniería de control del LEGO Mindstorms NXT*., Universidad Politécnica de Cartagena, Escuela Técnica Superior de Ingeniería Industrial, Tesis de Grado, 2008.

[32] D. Herrera, L. Bermeo, *Utilización e Implementación de Herramientas Didácticas para Prácticas de Control Automático*, 3^{rd} Colombian Workshop on Circuits and Systems, October 5-6, 2009.

[33] N. Sadati, A. Talasaz *Sliding Mode Control for a Flexible Transmission System*, IEEE International Conference on Systems, Man and Cybernetics, 2004

[34] E. Díaz, I. Esperabe, R. Fernández, D. Gualda, J. Manzano, J. Martín, J. Mateos, L. de Santiago *Introducción al Diseño de Microrobots Móviles*, Trabajo realizado para la asignatura de Diseño de Microrobots Móviles de la Universidad de Alcalá, Noviembre, 2006

[35] N. Cambon *Investigación Lego: Sensores y Actuadores*, disponible en http://www.saladeteletipos.com/twiki/bin/viewfile/ProcesadorLego/InvestigacionLego?rev=1;filename=sensores.pdf Marzo, 2010

[36] J. Hansen *NXC Reference Guide*, Version 1.2.1 r4, 2010

[37] M. Spong, D. Block *The Pendubot: A Mechatronic System for Control Research and Education*, Coordinated Science Laboratory, University of Illinois at Urbana, Champaign

[38] M. Estévez *Sistemas de Transmisión Mecánica*, Tecno Metal.

[39] Matlab Documentation *Longitudinal Vehicle Dynamics block*, 1984-2008 The MathWorks, Inc.

[40] L. Bermeo, D. Herrera, R. Ramirez, *Aplicaciones Didácticas en Control Automático con Lego Mindstorms NXT*, II South American Congress on Computational Mechanical, Mecánica Computacional Vol XXIX, págs. 2039-2060, Buenos Aires, Argentina, 15-18 Noviembre 2010

[41] A. Varga, *Balancing-Free Square-Root Algorithm for Computing Singular Perturbation Approximations* Proc. of 30th IEEE CDC, Brighton, UK (1991), pp. 1062-1065.

[42] A. Laub, *Computation of Balancing Transformations* Proc. ACC, San Francisco, Vol.1, paper FA8-E, 1980.