UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO

INSTITUTO DE CIENCIAS BÁSICAS E INGENIERÍA

"Diseño e integración de una plataforma experimental para evaluación de algoritmos de control de robots"

Tesis para obtener el título de:

Ing. en Electrónica y Telecomunicaciones

Presenta: *Vite*

Medécigo Silvionel

Director de tesis:

Dr. Ornar A. Domínguez Ramírez

Director externo:

Dr. Rene V. Mayorga U.
Regina, Saskatchewan, Canadá

Pachuca, Hgo., México, Enero del 2006

A mis

PADRES Y HERMANOS

Agradecimientos

Gracias a mis Padres Silvionel Vite E. y Herminia Medécigo L. quienes me han brindado un sinfín de oportunidades para emplear mis energías, actividades, mi voluntad y haberlo hecho con placer, por que hasta ahora se han preocupado por educarme como un hijo pobre para enriquecerme, por su infinito amor y comprensión en todo momento.

A mis hermanos Alex y Karys por su inmenso cariño e invaluable apoyo, por servirme de ejemplo y hacerme sentir orgulloso de ustedes.

Al Dr. Omar Arturo Domínguez Ramírez, mi director de tesis, con abundante respeto y profunda admiración, por que antes que nada me brindó una amistad sincera. Por todos los conocimientos aportados y formar parte de la idea esencial de este proyecto de tesis, por su inigualable constancia y extremada paciencia.

Agradezco las finas atenciones del Dr. Rene V. Mayorga, director externo de tesis, por sus conocimientos e ideas aportadas, mi sincero respeto y admiración.

A Lorena con mucho cariño, por ser parte de mi inspiración, a mis eternos amigos Axel, Juan Carlos, Rafael y Mirsha con aprecio, quienes siempre responden cuando es necesario, con un acto de afecto, comprensión y aun de sacrificio.

A mis compañeros y amigos de licenciatura Alfio, Iván, Beto, Cesar, David y Jorge por los imperecederos recuerdos. A mis compañeros tesistas Liz, Areli, Jair, José Guadalupe, Herbert y Víctor por su apoyo y confianza.

Resumen

Se realizó en Mechanical Desktop 6 el diseño en tres dimensiones de cada una de las piezas utilizadas para la construcción de una plataforma experimental robótica que ayudó a la evaluación de algoritmos de control de robots a partir de un diseño fundamentado en el modelo matemático del robot (modelo cinemático y dinámico); su diseño mecánico está basado en un robot planar de dos grados de libertad, sin efecto gravitatorio (R2SG). Para el modelo matemático del robot se realizaron estudios de simulación digital y se corroboraron los resultados de manera experimental, asimismo se comprueban la manipulabilidad cinemática y las propiedades dinámicas del robot R2SG. Se evaluaron estrategias de control clásicas (control PD y PID) para que posteriormente se realizara un estudio comparativo con un control moderno PID no lineal o control por modos deslizantes de segundo orden para regulación a una coordenada operacional, regulación basada en seguimiento a una coordenada operacional y seguimiento de una trayectoria cerrada (circunferencia). Cabe mencionar que la parte de integración numérica de las estrategias de control se obtuvo mediante el método de Runge-Kutta de 4° orden. Se desarrolló la interfaz electrónica de potencia y el acondicionamiento de señales para lograr la correspondencia tecnológica entre la tarjeta de adquisición de datos y control con los servomotores empleados. Finalmente todos los experimentos fueron desarrollados con el auxilio de una interfaz de usuario diseñada para este propósito y que es parte de la integración de esta plataforma experimental, esta interfaz permite la visualización virtual del robot en línea y sintonizar las ganancias del controlador, así como el tiempo de convergencia antes de cada experimento.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

1.1. Antecedentes

La necesidad cada vez con mayor peso de aumentar la productividad y conseguir productos acabados de una calidad uniforme, hacen que la industria gire hacia una automatización basada en sistemas digitales como una computadora. En la actualidad, la mayoría de las tareas de fabricación automatizadas se realizan mediante máquinas de uso especial diseñadas para realizar funciones predeterminadas en un proceso de manufactura. La flexibilidad y generalmente el alto costo de estas máquinas, a menudo llamados sistemas de automatización duros, han llevado a un interés creciente en el uso de robots capaces de efectuar una variedad de funciones de fabricación en un entorno de trabajo más flexible y a menor costo de producción.

La palabra robot proviene de la palabra checa robota, que significa siervo o esclavo. Un robot es un manipulador reprogramable de uso general con sensores externos que pueden efectuar diferentes tareas de manejo. Con esta definición un robot debe poseer inteligencia que se debe normalmente a los algoritmos de la computadora asociados con un sistema de control y sensorial.

Los primeros trabajos que condujeron a los robots industriales de hoy en día se remonta al periodo que siguió inmediatamente a la segunda guerra mundial. Durante los años finales de la década de los cuarenta, comenzaron programas de investigación en Oak Ridge y Argone National Laboratories para desarrollar manipuladores mecánicos controlados de forma remota para manejar materiales radioactivos. Estos sistemas eran de tipo maestro/esclavo, diseñados para reproducir fielmente los movimientos de mano y brazo realizados por un operario humano. El manipulador maestro era guiado por el usuario a través de una secuencia de movimientos, mientras que el manipulador esclavo duplicaba la unidad maestra tan fidedignamente tal como lo era posible.

El trabajo sobre manipuladores maestro/esclavo fue seguida por sistemas más sofisticados capaces de operaciones repetitivas autómatas. A mediados de los años cincuenta, George C. Devol. desarrollo un dispositivo que llamo “dispositivo de transferencia programada articulada”, un manipulador cuya operación podía ser programada y que podía seguir una secuencia de pasos de movimientos determinados por las instrucciones en el programa. Posteriores desarrollos, Devol

y Joseph F. Engelberger condujo al primer robot industrial, introducido por Unimation Inc. en 1959. La clave de este dispositivo era el uso de una computadora en conjunción con un manipulador para producir una máquina que podía ser "enseñada"para realizar una variedad de tareas de forma automática. Estos robots se podían reprogramar y cambiar de herramienta a un costo relativamente bajo para efectuar otros trabajos cuando cambiaban los requisitos de fabricación.

Aunque los robots programados ofrecían una herramienta de fabricación nueva y potente, se hizo patente en los años sesenta que la flexibilidad de estas máquinas se podían mejorar significativamente mediante el uso de una retroalimentación sensorial. Al comienzo de esta década, H. A. Ernst [1962] publicó el desarrollo de una mano mecánica controlada por una computadora con sensores táctiles. Este dispositivo llamado el MH-1, podía sentir bloques y usar esta información para controlar la mano de manera que apilaba los bloques sin ayuda de un operario. Este trabajo es uno de los primeros ejemplos de un robot capaz de conducta adaptativa en un entorno razonable no estructurado. Este programa de investigación posteriormente evolucionó como parte del proyecto MAC, y se le añadió una cámara de televisión para comenzar la investigación sobre la percepción en la máquina. Durante el mismo periodo, Tomovic y Boni [1962] desarrollaron una mano prototipo provista con un sensor de presión que detectaba el objeto y proporcionaba una señal de alimentación de entrada a un motor para iniciar uno dos modelos de aprehensión. En 1963, la American Machine y Foudry Company (AMF) introdujo el robot comercial Versatran. Comenzando en este mismo año se desarrollaron diversos diseños de brazos para manipuladores, tales como el brazo Roehampton y el de Edinburgh. En [1968] McCarthy así como en el Stanford Artificial Intelligence Laboratory publicaron el desarrollo de una computadora con manos, ojos y oídos (manipuladores, cámaras de T.V., micrófonos); demostrando un sistema que reconocía mensajes hablados, veía bloques distribuidos sobre una mesa, y los manipulaba de acuerdo con instrucciones. En [1969] se desarrollo el brazo Boston y el año siguiente el brazo de Stanford, que estaba diseñado con una cámara y controlado por computadora. Algunos de los trabajos más serios en robótica comenzaron cuando estos brazos se utilizaron como robots manipuladores; fue así que en 1974 Cincinnati Milacron introdujo su primer robot industrial controlado por computadora el cual podía levantar más de 100 libras. En [1978] se introdujo el robot Puma para tareas de montaje, basándose en diseños obtenidos en el estudio de General Motors.

1.2. Planteamiento del problema

En la actualidad, las estrategias de control de movimiento en robots manipuladores han tenido una importante evolución, muchos de los resultados reportados en la literatura son totalmente teóricos, presentando las pruebas de estabilidad en lazo cerrado con la dinámica de movimiento para cada caso de estudio. Sin embargo los algoritmos de control son diseñados con base a modelos matemáticos de robots en condiciones ideales y sin consideración de dinámicas no modeladas y desconocidas al día de hoy, razón por la que de lo teórico a lo práctico, es decir, de la prueba matemática y simulación digital a los resultados experimentales existe un llano amplio de incertidumbre que en muchos de los casos se intenta justificar con las leyes físicas del movimiento hasta ahora reportadas en la literatura. Es por ello que se diseñó e integró una plataforma

experimental, con la cual se pretende reducir en base a distintas pruebas experimentales esta incertidumbre ya antes mencionada. Otra de las cosas importantes es que la adquisición de un robot comercial de cualquier tipo es muy costosa comparada con esta plataforma además que suelen tener restricciones mecánicas no favorables para la evaluación de distintas pruebas experimentales.

1.3. Solución propuesta

Con el afán de establecer un método experimental para la comprobación de los modelos cinemáticos y dinámicos a partir de sus propiedades para un robot manipulador, así como la evaluación de estudios comparativos de estrategias de control de movimiento se propone un diseño y la correspondiente integración de los subsistemas de un robot experimental, basado en los subsistemas electromecánico (robot planar de dos grados de libertad sin efecto gravitatorio), comunicación (visualización virtual y consola de entrada y salida), control (estructuras clásicas y modernas), percepción (sensor de posición y velocidad articular), así como la interfaz electrónica de potencia y el acondicionamiento de señales. Cabe mencionar que los métodos descritos para la obtención de los modelos matemáticos de este robot y de las distintas estrategias de control son totalmente utilizados para cualquier robot.

1.4. Estado del arte

El concepto de máquinas automatizadas se remonta a la antigüedad, con mitos de seres mecánicos vivientes. Los autómatas, o máquinas semejantes a personas, ya aparecían en los relojes de las iglesias medievales, y los relojeros del siglo XVIII eran famosos por sus ingeniosas criaturas mecánicas.

Algunos de los primeros robots empleaban mecanismos de realimentación para corregir errores, mecanismos que siguen empleándose actualmente. Un ejemplo de control por realimentación es un bebedero que emplea un flotador para determinar el nivel del agua. Cuando el agua cae por debajo de un nivel determinado, el flotador baja, abre una válvula y deja entrar más agua en el bebedero. Al subir el agua, el flotador también sube, y al llegar a cierta altura se cierra la válvula y se corta el paso del agua.

El primer controlador realimentado auténtico fue el regulador de Watt, inventado en 1788 por el ingeniero británico James Watt. Este dispositivo constaba de dos bolas metálicas unidas al eje motor de una máquina de vapor y conectadas con una válvula que regulaba el flujo de vapor. A medida que aumentaba la velocidad de la máquina de vapor, las bolas se alejaban del eje debido a la fuerza centrífuga, con lo que cerraban la válvula. Esto hacía que disminuyera el flujo de vapor a la máquina y por tanto la velocidad.

El control por realimentación, el desarrollo de herramientas especializadas y la división del

trabajo en tareas más pequeñas que pudieran realizar obreros o máquinas fueron ingredientes esenciales en la automatización de las fábricas en el siglo XVIII. A medida que mejoraba la tecnología se desarrollaron máquinas especializadas para tareas como poner tapones a las botellas o verter caucho líquido en moldes para neumáticos. Sin embargo, ninguna de estas máquinas tenía la versatilidad del brazo humano, y no podían alcanzar objetos alejados y colocarlos en la posición deseada.

El desarrollo del brazo artificial multiarticulado, o manipulador, llevó al robot moderno. El inventor estadounidense George Devol desarrolló en 1954 un brazo primitivo que se podía programar para realizar tareas específicas. En 1975, el ingeniero mecánico estadounidense Victor Scheinman, cuando estudiaba la carrera en la Universidad de Stanford, en California, desarrolló un manipulador polivalente realmente flexible conocido como Brazo Manipulador Universal Programable (PUMA, siglas en inglés). El PUMA era capaz de mover un objeto y colocarlo en cualquier orientación en un lugar deseado que estuviera a su alcance. El concepto básico multiarticulado del PUMA es la base de la mayoría de los robots actuales.

El diseño de un manipulador robótico se inspira en el brazo humano, aunque con algunas diferencias. Por ejemplo, un brazo robótico puede extenderse telescópicamente, es decir, deslizando unas secciones cilíndricas dentro de otras para alargar el brazo. También pueden construirse brazos robóticos de forma que puedan doblarse como la trompa de un elefante. Las pinzas están diseñadas para imitar la función y estructura de la mano humana. Muchos robots están equipados con pinzas especializadas para agarrar dispositivos concretos, como una gradilla de tubos de ensayo o un soldador de arco.

Las articulaciones de un brazo robótico suelen moverse mediante motores eléctricos. En la mayoría de los robots, la pinza se mueve de una posición a otra cambiando su orientación. Una computadora calcula los ángulos de articulación necesarios para llevar la pinza a la posición deseada, un proceso conocido como cinemática inversa.

Algunos brazos multiarticulados están equipados con servocontroladores, o controladores por retroalimentación, que reciben datos de un ordenador. Cada articulación del brazo tiene un dispositivo que mide su ángulo y envía ese dato al controlador. Si el ángulo real del brazo no es igual al ángulo calculado para la posición deseada, el servocontrolador mueve la articulación hasta que el ángulo del brazo coincida con el ángulo calculado. Los controladores y los ordenadores asociados también deben procesar los datos recogidos por cámaras que localizan los objetos que se van a agarrar o las informaciones de sensores situados en las pinzas que regulan la fuerza de agarre.

Cualquier robot diseñado para moverse en un entorno no estructurado o desconocido necesita múltiples sensores y controles (por ejemplo, sensores ultrasónicos o infrarrojos) para evitar los obstáculos. Los robots como los vehículos planetarios de la NASA necesitan una gran cantidad de sensores y unas computadoras de a bordo muy potentes para procesar la compleja información que les permite moverse. Eso es particularmente cierto para robots diseñados para trabajar en estrecha proximidad de seres humanos, como robots que ayuden a personas discapacitadas

o sirvan comidas en un hospital. La seguridad debe ser esencial en el diseño de robots para el servicio humano.

En 1995 funcionaban unos 700,000 robots en el mundo industrializado. Más de 500,000 se empleaban en Japón, unos 120,000 en Europa Occidental y unos 60,000 en Estados Unidos. Muchas aplicaciones de los robots corresponden a tareas peligrosas o desagradables para los humanos. En los laboratorios médicos, los robots manejan materiales que conlleven posibles riesgos, como muestras de sangre u orina. En otros casos, los robots se emplean en tareas repetitivas y monótonas en las que el rendimiento de una persona podría disminuir con el tiempo. Los robots pueden realizar estas operaciones repetitivas de alta precisión durante 24 horas al día sin cansarse. Uno de los principales usuarios de robots es la industria del automóvil. La empresa General Motors utiliza aproximadamente 16,000 robots para trabajos como soldadura por puntos, pintura, carga de máquinas, transferencia de piezas y montaje. El montaje es una de las aplicaciones industriales de la robótica que más está creciendo. Exige una mayor precisión que la soldadura o la pintura y emplea sistemas de sensores de bajo coste y computadoras potentes y baratas. Los robots se usan por ejemplo en el montaje de aparatos electrónicos, para montar microchips en placas de circuito.

Las actividades que entrañan gran peligro para las personas, como la localización de barcos hundidos, la búsqueda de depósitos minerales submarinos o la exploración de volcanes activos, son especialmente apropiadas para emplear robots. Los robots también pueden explorar planetas distantes. La sonda espacial no tripulada Galileo, de la NASA, viajó a Júpiter en 1996 y realizó tareas como la detección del contenido químico de la atmósfera joviana.

Ya se emplean robots para ayudar a los cirujanos a instalar caderas artificiales, y ciertos robots especializados de altísima precisión pueden ayudar en operaciones quirúrgicas delicadas en los ojos. La investigación en telecirugía emplea robots controlados de forma remota por cirujanos expertos; estos robots podrían algún día efectuar operaciones en campos de batalla distantes.

Los manipuladores robóticos crean productos manufacturados de mayor calidad y menor coste. Sin embargo, también pueden provocar la pérdida de empleos no cualificados, especialmente en cadenas de montaje industriales. Aunque crean trabajos en los sectores de soporte lógico y desarrollo de sensores, en la instalación y mantenimiento de robots y en la conversión de fábricas antiguas y el diseño de fábricas nuevas, estos nuevos empleos exigen mayores niveles de capacidad y formación. Las sociedades orientadas hacia la tecnología deben enfrentarse a la tarea de volver a formar a los trabajadores que pierden su empleo debido a la automatización y enseñarles nuevas capacidades para que puedan tener un puesto de trabajo en las industrias del siglo XXI.

Las máquinas automatizadas ayudarán cada vez más a los humanos en la fabricación de nuevos productos, el mantenimiento de las infraestructuras y el cuidado de hogares y empresas. Los robots podrán fabricar nuevas autopistas, construir estructuras de acero para edificios, limpiar conducciones subterráneas o cortar el césped. Ya existen prototipos que realizan todas esas tareas.

Una tendencia importante es el desarrollo de sistemas microelectromecánicos, cuyo tamaño va

desde centímetros hasta milímetros. Estos robots minúsculos podrían emplearse para avanzar por vasos sanguíneos con el fin de suministrar medicamentos o eliminar bloqueos arteriales. También podrían trabajar en el interior de grandes máquinas para diagnosticar con antelación posibles problemas mecánicos.

Puede que los cambios más espectaculares en los robots del futuro provengan de su capacidad de razonamiento cada vez mayor. El campo de la inteligencia artificial está pasando rápidamente de los laboratorios universitarios a la aplicación práctica en la industria, y se están desarrollando máquinas capaces de realizar tareas cognitivas como la planificación estratégica o el aprendizaje por experiencia. El diagnóstico de fallos en aviones o satélites, el mando en un campo de batalla o el control de grandes fábricas correrán cada vez más a cargo de ordenadores inteligentes.

1.4.1. Estado del arte de los robots manipuladores

Un manipulador industrial es de uso general controlado por una computadora que consiste en algunos elementos rígidos conectados en serie mediante articulaciones prismáticas o de revolución. El final de la cadena esta fijo a una base soporte, mientras el otro extremo esta libre y equipado con una herramienta para manipular objetos o realizar tareas de montaje.

El movimiento de las articulaciones produce, un movimiento relativo de los distintos elementos. Mecánicamente, un robot se compone de un brazo y una muñeca más una herramienta. Se diseña para alcanzar una pieza de trabajo localizada dentro de su volumen de trabajo. El volumen de trabajo es la esfera de influencia de un robot cuyo brazo puede colocar el submontaje de la muñeca en cualquier punto dentro de la esfera. El brazo generalmente se puede mover con tres grados de libertad. La muñeca consta normalmente de tres movimientos giratorios. La combinación de estos movimientos orienta a la pieza de acuerdo a la configuración del objeto para facilitar su recogida. Estos últimos tres movimientos se denominan a menudo elevación(pitch), desviación(yaw) y giro(roll), por tanto, para un robot de seis articulaciones, el brazo es el mecanismo de posicionamiento, mientras que la muñeca es el mecanismo de orientación.

1.4.2. Estado del arte del control de robots manipuladores

El control automático ha desempeñado una función vital en el avance de la ingeniería y la ciencia. Además de su extrema importancia en los sistemas de vehículos espaciales, de guiado de misiles, robóticos y similares, el control automático se ha vuelto una parte importante e integral de los procesos modernos industriales y de manufactura. Por ejemplo, el control automático es esencial en el control numérico de las máquinas-herramienta de las industrias de manufactura, en el diseño de sistemas de pilotos automáticos en la industria aeroespacial, y en el diseño de automóviles y camiones en la industria automotriz. También es esencial en las operaciones industriales como el control de presión, temperatura, humedad, viscosidad y flujo en las industrias de proceso.

Debido a que los avances en la teoría y la práctica del control automático aportan los medios para obtener un desempeño óptimo de los sistemas dinámicos, mejorar la productividad, aligerar la carga de muchas operaciones manuales repetitivas y rutinarias, así como de otras actividades, casi todos los ingenieros y científicos deben tener un buen conocimiento de este campo.

El primer trabajo significativo en control automático fue el regulador de velocidad centrífugo de James Watt para el control de la velocidad de una máquina de vapor, en el siglo XVIII. Minorsky, Hazen y Nyquist, entre muchos otros, aportaron trabajos importantes en las etapas iniciales del desarrollo de la teoría de control. En 1992, Minorsky trabajó en los controladores automáticos para dirigir embarcaciones, y mostró que la estabilidad puede determinarse a partir de las ecuaciones diferenciales que describen el sistema. En 1932, Nyquist diseñó un procedimiento relativamente simple para determinar la estabilidad de sistems de lazo cerrado, con base en la respuesta en lazo abierto en estado estable cuando la entrada aplicada es una senoidal. En 1934, Hazen, quien introdujo el témino servomecanismos para los sistemas de control de posición, analizó el diseño de los servomecanismos con relevadores, capaces de seguir con precisión una entrada cambiante.

Durante la década de los cuarenta, los métodos de la respuesta en frecuencia hicieron posible que los ingenieros diseñaran sistemas de control lineales en lazo cerrado que cumplieran con los requerimientos de desempeño. A finales de los años cuarenta y principios de los cincuenta, se desarrollo por completo el método del lugar geométrico de las raíces propuesto por Evans.

Los métodos de repuesta en frecuencia y del lugar geométrico de las raíces, que forman el núcleo de la teoría de control clásica, conducen a sistemas estables que satisfacen un conjunto más o menos arbitrario de requerimientos de desempeño. En general estos sistemas son aceptables pero no óptimos en forma significativa. Desde el final de la década de los cincuenta, el énfasis en los problemas de diseño de control se ha movido del diseño de uno de muchos sistemas que trabajen apropiadamente al diseño de un sistema óptimo de algún modo significativo.

Conforme las plantas modernas con muchas entradas y salidas se vuelven más y más complejas, la descripción de un sistema de control moderno requiere de una gran cantidad de ecuaciones. La teoría del control clásica, que trata de los sistemas con una entrada y una salida, pierde su solidez ante sistemas con entradas y salidas múltiples. Desde alrededor de 1960, debido a que la disponibilidad de las computadoras digitales hizo posible el análisis en el dominio del tiempo de sistemas complejos, la teoría de control moderna, basada en el análisis en el dominio del tiempo y la síntesis a partir de variables de estados, se ha desarrollado para enfrentar la creciente complejidad de las plantas modernas y los requerimientos limitativos respecto a la precisión, peso y costo en aplicaciones militares, espaciales e industriles.

Durante los años comprendidos entre 1960 y 1980, se investigaron a fondo el control óptimo tanto de sistemas determinísticos como estocásticos, y el control adaptable, mediante el aprendizaje de sistemas complejos. De 1980 a la fecha, los descubrimientos en la teoría de control moderna se centraron en el control robusto, y temas asociados.

Ahora que las computadoras digitales se han vuelto más baratas y más compactas, se usan como parte integral de los sistemas de control. Las aplicaciones recientes de la teoría de control moderna incluyen sistemas ajenos a la ingeniería, como los biológicos, biomédicos, económicos y socioeconómicos.

Los robots industriales se usan con frecuencia en la industria para mejorar la productividad. Un robot puede realizar tareas monótonas y complejas sin errores en la operación. Asimismo, puede trabajar en un ambiente intolerable para operadores humanos. Por ejemplo, puede funcionar en temperaturas extremas (tanto altas como bajas), en un ambiente de presión alta o baja, bajo el agua o en el espacio. Hay robots especiales, entre muchos otros.

El robot industrial, debe tener algunos dispositivos sensores. A los robots de nivel bajo, se les instalan microinterruptores en los brazos como dispositivos sensores. El robot toca primero un objeto y después, mediante los microinterruptores, confirma la existencia del objeto en el espacio y avanza al paso siguiente para asirlo.

En un robot de nivel alto se usa un medio óptico (como un sistema de televisión) para rastrear el fondo del objeto. El robot reconoce el patrón y determina la presencia y orientación del objeto. Se requiere de una computadora para procesar las señales del proceso de reconocimiento de patrones. En algunas aplicaciones, el robot computarizado, reconoce la presencia y orientación de cada parte mecánica mediante un proceso de reconocimiento de patrones que conciste en la lectura de los números de código que se fijan a cada parte. A continuación, el robot levanta la parte y la mueve a un lugar conveniente para su ensamble, después ensambla varias partes para formar un componente. Una computadora bien programada funciona como controlador.

Se han realizado diversos estudios acerca del control de robots manipuladores planares, este es el caso de un robot planar de dos grados de libertad sin término gravitatorio construido por el Centro de Investigaciones Avanzadas (CINVESTAV), el cual ha sido utilizado para múltiples trabajos de investigación, una de las más relevantes fue realizada en junio de 2003 en donde se implemento un controlador por modos deslizantes para seguimiento en tiempo finito. Este proyecto tuvo como propósito diseñar un controlador continuo para una dinámica no lineal del robot tal que garantice el error cero en el seguimiento de trayectorias en un tiempo finito arbitrario, obteniendo trayectorias deseadas uniformes asumiendo el no conocimiento del sistema dinámico del robot. Las contribuciones obtenidas en este trabajo fueron el seguimiento continuo de trayectorias libres de chattering utilizando como ley de control un PID no lineal para un sistema dinámico obtenido mediante la formulación de Euler-Lagrange. La diferencia respecto a este trabajo de investigación, es que no se elaboró un software en donde además de que se controlara al robot se visualizara en forma virtual de tal forma que pudiera verse en línea.[17]

Otros resultados experimentales utilizando un robot manipulador planar de dos grados de libertad fueron realizados por la División de Física Aplicada, CISECE y por la Universidad de Illinois, USA. En este proyecto de investigación presentan un nuevo controlador que resuelve el

problema de un servo control visual para robots planares que contienen una configuración de cámara fija. Para este nuevo controlador, no se utiliza ni la cinemática inversa ni el Jacobiano inverso. Para la posición deseada se caracterizó el atractor global en lazo cerrado utilizando la dinámica total del robot, y realizando la prueba de estabilidad asintótica local del error en el efector final. Se asume que el sistema de visión equipado con una cámara fija cubre el espacio de trabajo total del robot disponible para localizar al efector final del robot revelando la posición cartesiana de este y así llevarlo a una posición deseada utilizando un simple control diseñado y sustentado por rigurosos análisis de estabilidad tomando en cuenta la dinámica no lineal del robot.[13][14]

El Instituto Tecnológico de Puebla dio a conocer un avance del estudio relacionado con el análisis cinemático para el diseño de un robot paralelo planar de tres grados de libertad con el fin de ejecutar tareas de ensamble por colocación en dos posiciones angulares entre dos piezas en forma automatizada y precisa. Presentaron solo consideraciones básicas y generales, ya que la parte de análisis cinemático, está en proceso de realización en este momento. Dicho análisis, junto con la secuencia de los movimientos de ensamble denotada por los movimientos angulares y lineales de las articulaciones proporcionará las posiciones adecuadas de los eslabones para obtener ensambles en diferentes posiciones.[22]

1.5. Contribución de la tesis

Este trabajo de investigación y desarrollo tuvo como resultados las siguientes contribuciones:

1. Construcción del mecanismo de eslabones articulados de dos grados de libertad sin gravedad.
2. Desarrollo de una interfaz virtual para la visualización de movimientos en linea.
3. Construcción de una interfaz electrónica de potencia y acondicionamiento de señales para comunicación de entrada y salida con la tarjeta de adquisición de datos y control.
4. Obtención del modelo cinemático (posición y diferencial) y dinámico del robot, y su comprobación teórica y experimental.
5. Evaluación experimental de estrategias de control clásica y moderna para regulación a una coordenada operacional, regulación basada en seguimiento a una coordenada operacional y seguimiento de una trayectoria cerrada.
6. Se construyeron importantes bases experimentales para interacción hombre-máquina: guiado háptico local con propósitos de diagnóstico y rehabilitación neuropsicológica.

1.6. Organización de la tesis

La tesis está organizada en 7 capítulos los cuales se describen a continuación:

Capítulo 1 : Se presentan breves antecedentes, el planteamiento del problema, una propuesta para solucionar el problema, la descripción del estado del arte de los robots manipuladores así como las estrategias de control utilizadas, de igual forma, se dan a conocer las contribuciones relevantes de la tesis y finalmente la descripción del contenido y su organización.

Capítulo 2 : Se presenta una descripcíon de los conceptos importantes del modelo matemático de un robot que se basa en la cinemática y dinámica, haciendo una revisión de los modelos cinemáticos directos e inversos de posición, velocidad y aceleración definidos a través de la cadena cinemática del robot, mientras que el segundo modelo describe el movimiento del robot manipulador debido a las fuerzas que originan dicho movimiento recurriendo al método de Euler-Lagrange, tomando en cuenta las propiedades dinámicas del robot.

Capítulo 3 : Se hace un análisis general acerca de los métodos de control, revisando los conceptos básicos como los controles clásicos tales como el control proporcional y derivativo (PD), control proporcional, integral, derivativo (PID) y el control moderno por modos deslizantes de segundo orden (PID no lineal). Los cuales brindan a los robots presición y estabilidad en sus movimientos.

Capítulo 4 : Aquí se describe de forma detallada el diseño y construcción mecánica del robot R2SG, se da a conocer el tipo de servomotor utilizado en cada articulación para continuar describiendo detalladamente la interfaz electrónica de potencia y el acondicionamiento de señales y así lograr la comunicación con la computadora con la ayuda de una TAD. De igual forma se detalla el modelo cinemático y dinámico del robot R2SG y por último se describen los algoritmos de control experimentados con el robot R2SG.

Capítulo 5 : Este capítulo describe brevemente la realidad virtual en robots manipuladores y se detalla la realización mediante una computadora, de un mundo virtual creado con un lenguaje de programación específico y librerias para el diseño en 3D a fin de lograr la integración del mundo virtual activo en tiempo real suave del robot R2SG.

Capítulo 6 : Se analizan todos los experimentos realizados con el robot R2SG evaluando las distintas estrategias de control clásica y moderna para regulación a una coordenada operacional, regulación basada en seguimiento a una coordenada operacional y seguimiento de una trayectoria cerrada.

Capítulo 7 : Se describen las conclusiones de este trabajo de tesis, las perspectivas y proyección de los algoritmos propuestos.

Capítulo 2

Modelo matemático de robots manipuladores

2.1. Introducción

Para el modelado matemático del robot es necesario dividirlo en *Modelo Cinemático* y *Dinámico*. Para la cinemática se hace el estudio analítico de la geometría del movimiento con respecto a un sistema de coordenadas fijo como una función del tiempo sin considerar las fuerzas que originan dicho movimiento. Existen métodos clásicos para obtener el modelo dinámico de un dispositivo, como el método de Euler-Lagrange que es simple y sistemático. El modelo dinámico es empleado para el diseño de la ley de control dependiente o no del modelo dinámico. En esta sección se presenta de manera general y sintética el procedimiento para obtener el modelo cinemático y dinámico de un robot.

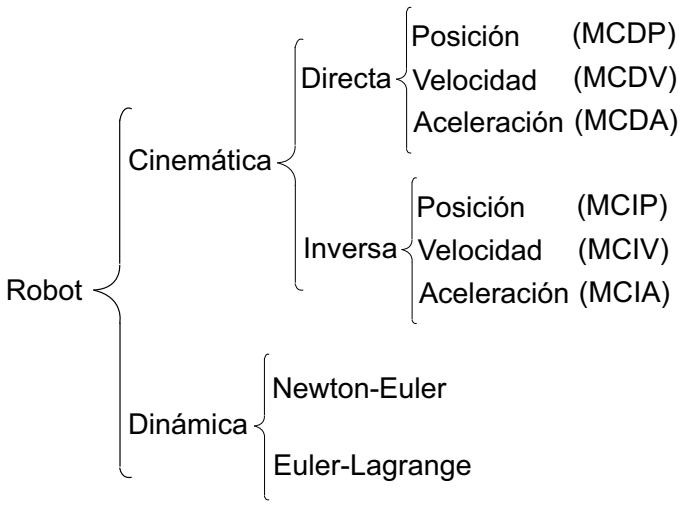

Figura 2.1: Modelado de un robot.

2.2. Modelo cinemático

La cinemática se ocupa de la descripción del movimiento sin tener en cuenta sus causas. La velocidad (la tasa de variación de la posición) se define como la distancia recorrida dividida entre el intervalo de tiempo. La magnitud de la velocidad se denomina celeridad, y puede medirse en unidades como kilómetros por hora, metros por segundo, etc. La aceleración se define como la tasa de variación de la velocidad, es decir, el cambio de la velocidad en relación al tiempo en que se produce. Por lo tanto, la aceleración tiene magnitud, dirección y sentido, y se mide en unidades del tipo metros entre segundo cuadrado.

En cuanto al tamaño o peso del objeto en movimiento, no se presentan problemas matemáticos si el objeto es muy pequeño en relación con las distancias consideradas. Si el objeto es grande, se emplea un punto llamado centro de masas, cuyo movimiento puede considerarse característico de todo el objeto. Si el objeto gira, muchas veces conviene describir su rotación en torno a un eje que pasa por el centro de masas.

Un mecanismo de eslabones articulados puede ser modelado como una cadena articulada (cadena cinemática) en lazo abierto con algunos cuerpos rígidos (eslabones) conectados en serie por una articulación. Trata con la descripción analítica del desplazamiento espacial del mecanismo como una función del tiempo, en particular las relaciones entre las variables espaciales de tipo articulación (variables articulares), la posición y orientación del efector final.

Los elementos de un mecanismo pueden girar y/o trasladarse con respecto a un sistema de coordenadas de referencia, el desplazamiento espacial total del efector final se debe a las rotaciones y traslaciones angulares de los elementos. Denavit y Hartenberg [1955] propusieron un método sistemático y generalizado de utilizar álgebra matricial para describir y representar la geometría espacial de los elementos del mecanismo con respecto a un sistema de referencia fijo. Este método utiliza una matriz de transformación homogénea de tamaño 4×4 para describir la relación espacial entre dos elementos mecánicos rígidos adyacentes y reduce el problema cinemático directo a encontrar una matriz de transformación homogénea equivalente que relaciona el desplazamiento espacial del sistema de coordenadas del efector final al sistema de coordenadas de referencia. Estas matrices de transformación homogénea también son útiles para derivar las ecuaciones de movimiento dinámico de un mecanismo de eslabones articulados.

Un procedimiento clásico para definir el modelo cinemático de un mecanismo de eslabones articulados, es la metodología de Denavit-Hartemberg. La cadena cinemática definida a partir del tipo de articulaciones y eslabones, permite identificar los marcos ortonormales por articulación y de ahí los parámetros Denavit-Hartemberg que se refiere a características físicas, su substitución en la matriz de transformación homogénea generalizada permite definir tantas matrices elementales como grados de libertad, el producto de las matrices elementales define la matriz de transformación homogénea del dispositivo de donde es posible determinar el modelo cinemático directo de posición y, mediante artificios algebráicos y trigonométricos es posible determinar el modelo cinemático inverso de posición. Las derivadas respecto del tiempo de estas ecuaciones definen al modelo cinemático directo e inverso de velocidad y aceleración.

2.2.1. Modelo cinemático de posición

El *Modelo Cinemático Directo de Posición* (MCDP) de un robot manipulador es la relación que permite determinar las coordenadas operacionales del robot en función de la configuración q del mismo, en donde q es la variable articular.

El MCDP es representado por la expresión sintética (2.1).

$$x = f(q) \tag{2.1}$$

Para describir la situación de un sólido en el espacio cartesiano se requieren m parámetros (coordenadas operacionales), distribuidos en un vector columna,

$$\vec{x} = [x_1, x_2, \ldots, x_m]^T \tag{2.2}$$

donde $m \geq 6$. En general, x_1, x_2 y x_3 definen la posición del sólido, mientras que x_4, x_5,...,x_m definen la orientación. Para el caso de un robot o mecanismo de eslabones articulados, el sólido se refiere al extremo superior del último eslabón libre. Las coordenadas de posición se pueden definir en coordenadas cartesianas, cilíndricas ó esféricas, y las coordenadas de orientación se pueden definir en términos de los ángulos de Euler o Bryant. Dada la matriz de transformación homogénea generalizada (2.3) del mecanismo, es posible determinar la ecuación (2.2),

$${}^0_nT = {}^0_1T\,{}^1_2T \cdots {}^{n-1}_nT = \begin{bmatrix} e_{11} & e_{12} & e_{13} & e_{14} \\ e_{21} & e_{22} & e_{23} & e_{24} \\ e_{31} & e_{32} & e_{33} & e_{34} \\ e_{41} & e_{42} & e_{43} & e_{44} \end{bmatrix} \tag{2.3}$$

donde ${}^0_1T\,{}^1_2T \cdots {}^{n-1}_nT$, corresponde al producto de las n matrices elementales del mecanismo de eslabones articulados, y n es el número de grados de libertad. Los elementos de la matriz ${}^0_nT$ definen el modelo cinemático directo de posición,

$$x_1 = e_{14} \tag{2.4}$$
$$x_2 = e_{24} \tag{2.5}$$
$$x_3 = e_{34} \tag{2.6}$$

$$x_4 = atan2\left(-e_{23}, e_{33}\right) \tag{2.7}$$
$$x_5 = atan2\left(e_{13}, e_{33}/\cos\left(x_4\right)\right) \tag{2.8}$$
$$x_6 = atan2\left(-e_{12}, e_{11}\right) \tag{2.9}$$

El *Modelo Cinemático Inverso de Posición* (MCIP) de un robot le permite obtener la configuración deseada, es decir las coordenadas generalizadas están en función de las coordenadas operacionales, donde las primeras son utilizadas como consignas de los actuadores para llevar al robot a la posición deseada.

El MCIP es representada por la expresión sintética (2.10), permite determinar las variables articulares ó ángulos en las articulaciones en función de las coordenadas operacionales.

$$q = f^{-1}\left(x\right) \tag{2.10}$$

Por definición, un robot ó mecanismo de eslabones articulados es resoluble si es posible determinar todas las configuraciones q correspondientes a una situación dada x (es decir, todas las q tales que $f\left(q\right) = x$). Los métodos de solución del modelo inverso de posición se clasifican en numéricos y analíticos. Los métodos numéricos tienen la desventaja de que, en caso de existencia de más de una solución, el usuario no tiene la posibilidad de controlar la solución a la cual converge el método. Sin embargo, los métodos analíticos permiten resolver la mayoría de las arquitecturas de los robots utilizados en la práctica.

2.2.2. Modelo cinemático de velocidad

El *Modelo Cinemático Directo de Velocidad* (MCDV) resulta de la primera derivada con respecto al tiempo de las ecuaciones que representan al MCDP, es decir

$$\frac{d}{dt}\left(x\right) = \frac{d}{dt}\left\{f\left(q\right)\right\} \tag{2.11}$$

La ecuación representativa clásica es

$$\dot{x} = J\dot{q} \tag{2.12}$$

donde $\dot{x} \in \Re^{(n\times 1)}$ representa la velocidad operacional en el extremo libre o efector final, $\dot{q} \in \Re^{(n\times 1)}$ es el vector de velocidades articulares y $J \in \Re^{(n\times n)}$ corresponde a la matriz jacobiana del robot.

El *Modelo Cinemático Inverso de Velocidad* (MCIV) es definido a partir de la ecuación (2.12), la expresión que representa a este modelo es definida en la ecuación (2.13)

$$\dot{q} = J^{-1}\dot{x} \tag{2.13}$$

donde $J^{-1} = adj\,(J)/\det(J)$ con $\det(J) \neq 0$. Este conjunto de ecuaciones permite determinar la velocidad articular en función de la velocidad operacional y a su vez definir los puntos singulares o inadmisibles.

2.2.3. Modelo cinemático de aceleración

El *Modelo Cinemático Directo de Aceleración* (MCDA) resulta de la segunda derivada de las ecuaciones que representan al modelo cinemático directo de posición, es decir

$$\frac{d^2}{dt^2}(x) = \frac{d^2}{dt^2}\{f(q)\} \tag{2.14}$$

a partir de la ecuación (2.12) se define la representación en términos de la matriz jacobiana y su derivada,

$$\ddot{x} = J\ddot{q} + \dot{J}\dot{q} \tag{2.15}$$

donde $\ddot{x} \in \Re^{(n\times 1)}$ representa la aceleración operacional y $\dot{J} \in \Re^{(n\times n)}$ corresponde a la derivada de la matriz jacobiana del robot. Este conjunto de ecuaciones permite determinar la aceleración operacional $\ddot{x}$ en función de la velocidad y aceleración articular $\dot{q}$ y $\ddot{q}$ respectivamente.

El *Modelo Cinemático Inverso de Aceleración* (MCIA) es definido a partir de la ecuación (2.15), la expresión que representa a este modelo es definida en la ecuación (2.16)

$$\ddot{q} = J^{-1}\left[\ddot{x} - \dot{J}\dot{q}\right] \tag{2.16}$$

Esta ecuación define la aceleración articular en términos de la aceleración operacional y de la velocidad articular.

2.3. Modelo dinámico

La dinámica es una rama de la mecánica que estudia el movimiento de los cuerpos mediante una acción de fuerza. El comportamiento dinámico de un mecanismo de eslabones articulados es descrito en términos de la razón de cambio de la configuración del dispositivo en relación a los pares ejercidos por los eslabones.

La dinámica de un robot es el estudio de la relación matemática entre el movimiento del robot y las fuerzas implicadas en él mismo:

- Localización del robot definida por sus variables articulares o por las coordenadas de localización de su extremo, y sus derivadas: velocidad y aceleración.
- Fuerzas y pares aplicados en las articulaciones o en l extremo del robot.
- Parámetros dimensionales del robot, como las longitudes, masas e inercias de sus elementos.

Para la obtención del modelo dinámico del robot se deben tener en cuenta los siguientes puntos:

- La complejidad aumenta con el número de grados de libertad.
- Interacción entre movimientos.
- No siempre es posible su obtención en forma cerrada.
- Procedimientos numéricos iteractivos.
- Necesidad de incluir los actuadores y su dinámica.
- Se tienen que realizar simplificaciones.
- Caso especial: Robots flexibles.

En cuanto a su utilidad:

- Simulación del movimieno del robot.
- Diseño y evaluación de la estructura mecánica del robot.
- dimencionamiento de los actuadores.

- Diseño y evaluación del control dinámico el robot.

Un actuador tiene que equilibrar los pares de cuatro fuentes: los pares dinámicos que surgen del movimiento, los pares estáticos que surgen de la fricción en los mecanismos, los pares gravitatorios que surgen por la acción de la gravedad sobre los eslabones y finalmente, los pares y fuerzas externas que actúan en el efector final y que dependen de la tarea. Existen tres tipos de pares dinámicos que surgen del movimiento del manipulador: par inercial, par centripeto y par de Coriolis. Los pares inerciales son proporcionales a la aceleración articular de acuerdo a la segunda ley de Newton. La inercia es la propiedad de los materiales a oponerse al cambio en el movimiento, y está cuantificada como la masa del material. Los pares centripetos surgen de las fuerzas centripetas las cuales restringen a un cuerpo a rotar alrededor de un punto, son dirigidas hacia el centro del movimiento circular uniforme y son proporcionales al cuadrado de la velocidad articular. Los pares de Coriolis surgen a partir de las fuerzas vortiginosas derivadas de dos eslabones en rotación , estas fuerzas son similares a las fuerzas causadas en un vórtice, y son proporcionales al producto de las velocidades articulares de esos eslabones.

Para sistemas dinámicos, las ecuaciones o el modelo dinámico puede ser formulado por medio de cantidades de energía. El modelo de un sistema mecanico rígido, con n grados de libertad, puede ser obtenido usando el método de Euler-Lagrange. El uso del método de Euler-Lagrange, para la obtención del modelo dinámico es simple y sistemático, proporciona las ecuaciones de estado en forma explícita y estas pueden ser utilizadas para analizar y diseñar estrategias de control.

$$\tau_i = \frac{d}{dt}\frac{\partial}{\partial \dot{q}_i}L - \frac{\partial}{\partial q_i}L \tag{2.17}$$

$i = 1, 2, ..., n$ donde n es el número de grados de libertad de nuestro robot, q es la coordenada generalizada o variable articular, $\dot{q}$ es la primera derivada respecto del tiempo de la coordenada generalizada q, τ corresponde a la fuerza generalizada aplicada a la articulación i para mover el eslabón i y L es el *Lagrangiano*, que se define como el balance de energias y que esta representado de la siguiente forma:

$$L = \sum_{i=1}^{n} k_i - P_i \tag{2.18}$$

Las energías son cinética (k) y potencial (P) que estan dadas por

$$k_i = \frac{1}{2} m_i V_i^2 \tag{2.19}$$

m es la masa y V la velocidad,

$$P_i = m_i g h_i \tag{2.20}$$

g es la constante de gravedad y h es la altura al centro de gravedad de los eslabones.Utilizando (2.17) se obtiene la representación del modelo dinámico

$$\tau : M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) \tag{2.21}$$

en donde $M(q) \in \Re^{nxn}$ es la matriz de inercia, $C(q,\dot{q}) \in \Re^{nxn}$ es la matriz de Coriolis y fuerzas centrípetas, $G(q) \in \Re^{nx1}$ es el vector de gravedad y $F(\dot{q}) \in \Re^{nx1}$ es el vector de fricción.

Para verificar la validez del modelo dinámico deben cumplirse sus propiedades. La matriz de inercia debe ser simétrica $M(q) = M(q)^T$ y definida positiva $|M(q)| > 0$, ademas debe cumplirse la propiedad de la matriz antisimétrica $x^T(\dot{M}(q) - 2C(q,\dot{q}))x \equiv 0 \quad \forall x \in \Re^n$.

2.4. Conclusiones

En este capítulo se dieron a conocer de manera genérica los conceptos básicos de cinemática y dinámica de un robot manipulador. La teoría que describe el comportamiento espacial de un cuerpo rígido con movimientos definidos mediante una rotación y translación asociado a robots manipuladores y el movimiento de los cuerpos mediante las fuerzas que lo originan. El uso del modelo de Euler-Lagrange para la obtención del modelo dinámico.

Las propiedades cinemáticas como dinámicas expuestas en este capítulo fueron descritas de manera general, esto quiere decir que pueden aplicarse para cualquier tipo de robot.

Para la obtención del modelo cinemático fue empleada la formulación de Denavit-Hartemberg, este método permitió de manera sistemática determinar la cinemática directa e inversa de posición, velocidad y aceleración del robot R2SG, cuyas ecuaciones son empleadas para planificar trayectorias en el espacio operacional, dado que los controles empleados en esta tesis son articulares, estos experimentos son reportados en el capítulo 6.

El modelo dinámico fue determinado con la formulación de Euler-Lagrage, cuyas ecuaciones describen el movimiento del robot, considerando los fenómenos inherentes a todo sistema electromecánico como los debidos a las fuerzas inerciales, de Coriolis, centrípetas y fricción. Los parámetros calculados exerimentalmente con mediciones directas y el modelo dinámico obtenido fueron corroborados experimentalmente con las propiedades dinámicas como la matriz de inercia definida positiva y la matriz antisimétrica cuyos resultados son presentados en el capítulo 4.

Para ambos modelos se evaluó, en el espacio de trabajo, a la manipulabilidad cinemática y dinámica, indicadores que permitieron conocer la operación del robot en el espacio de trabajo y definen su aproximación a regiones singulares, los correspondientes experimentos pueden apreciarse en el capítulo 4.

Capítulo 3

Control de robots manipuladores

3.1. Introducción

A pesar de la existencia de robots comerciales, el diseño de controladores para robots sigue siendo un área de intensos estudios por parte de los constructores de robots así como de los centros de investigación. Podría argumentarse que los robots industriales actuales son capaces de realizar correctamente una gran variedad de actividades, por lo que parecería innecesario, a primera vista, el desarrollo de investigaciones sobre el tema de control de robots. Sin embargo, este último tema no solo es interesante en sí mismo, sino que también ofrece grandes retos teóricos, y más importante aún, su estudio es indispensable en aplicaciones específicas que no pueden ser llevadas a cabo mediante los robots comerciales actuales.

Se deben de determinar las variables físicas del sistema cuyo comportamiento se desea gobernar tales como temperatura, presión, desplazamiento, velocidad, etc. Estas variables reciben el nombre de *salidas del sistema*. Además, también deben identificarse claramente aquellas variables físicas del sistema que se encuentran disponibles y que influyen en su evolución, y en particular de las salidas del sistema. Estas variables llamadas *entradas del sistema* pueden ser, por ejemplo, la apertura de la válvula, tensión, par o fuerza, etc.[12]

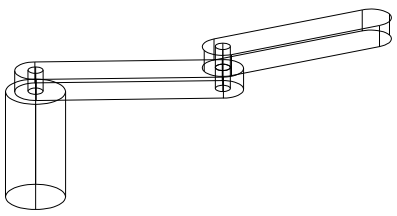

Figura 3.1: Robot R2SG.

En el caso particular de robots manipuladores, la variable de salida cuya conducta se desea modificar, ofrece un amplio espectro de elecciones tal y como se esquematiza en la Figura 3.1, por ejemplo, los destinados a tareas de pintado, traslado de objetos de un punto a otro, corte por rayo láser, etc., la salida puede corresponder simplemente a las posiciones y velocidades articulares, o también a la posición y orientación del órgano terminal o herramienta.

Para robots manipuladores como el mostrado esquemáticamente en la Figura 3.1 que involucran su interacción con el medio ambiente por contacto físico para realizar tareas como pulido de superficies, desbastado de materiales, ensamblaje de alta precisión, etc., la salida puede incluir los pares y fuerzas ejercidos por el extremo del último eslabón del robot sobre su medio ambiente.

Los robots manipuladores industriales pueden clasificarse según su aplicación en dos clases. La primera es aquella en la cual el robot se desplaza libremente en su espacio de trabajo realizando movimientos sin interaccionar con su medio ambiente, por ejemplo, aplicando una fuerza sobre éste. Esta clase de manipuladores realiza tareas como el pulido y el ensamblado de precisión.

En este capítulo se abordará exclusivamente el estudio de controladores de movimiento para robots manipuladores que se desplazan libremente en su espacio de trabajo, sin interaccionar con el medio ambiente.

Se considerarán robots manipuladores provistos de accionadores ideales sin dinámica, es decir, los accionadores proporcionan pares y fuerzas proporcionales a sus entradas. La idealización anterior es común en muchos de los trabajos teóricos sobre control de robots. Por otro lado, recientes desarrollos tecnológicos en la construcción de accionadores eletromecánicos, permiten contar hoy en día con motores de Corriente Continua (CC) de alto par, que pueden configurarse como fuentes ideales de pares en un amplio intervalo de operación. Por último, es importante mencionar que a pesar de tratarse en este capítulo el escenario de accionadores ideales, la mayor parte de los estudios sobre controldores que serán presentados, pueden extenderse con ligeras modificaciones, al de accionadores lineales de segundo orden, como los motores de CC.

3.2. Control clásico

3.2.1. Control PD

El control Proporcional (P) con retroalimentación de velocidad es el controlador de malla cerrada más sencillo que pueda emplearse en el control de robots manipuladores. La aplicación conceptual de esta estrategia de control es común en el control de posición angular de motores de corriente continua. En dicha aplicación, también se le conoce con el nombre de *control proporcional con retroalimentación tacométrica*. La ecuación del controlador Proporcional con retroalimentación de velocidad viene dada por:

$$\tau = \ k_p \, \tilde{q} - k_d \, \dot{q} \tag{3.1}$$

Donde k_p, $k_d \in \Re^{nxn}$ son matrices simétricas definidas positivas selecionadas por el diseñador

y denominadas ganancia de posición y de velocidad (o derivativa), respectivamente. El vector $q_d \in \Re^n$ es la posición articular deseada, y el vector $\tilde{q} = q_d - q \in \Re^n$ se denomina error de posición.[12]

El control Proporcional Derivativo (PD) es una extensión inmediata del control Proporcional con retroalimentación de velocidad 3.1. Como su nombre indica, la ley de control está formada no solo por un término proporcional al error de posición $\tilde{q}$ como el controlador Proporcional con retroalimentación de velocidad, sino también por otro término proporcional a su derivada, es decir, al error de velocidad $\dot{\tilde{q}}$. La ley de control PD viene dada por:

$$\tau = k_p \tilde{q} - k_d \dot{\tilde{q}} \tag{3.2}$$

Donde también k_p, $k_d \in \Re^{nxn}$ son matrices simétricas definidas positivas seleccionadas por el diseñador. La Figura 3.2 presenta el diagrama de bloques correspondiente al sistema de control formado por el control PD y un robot.

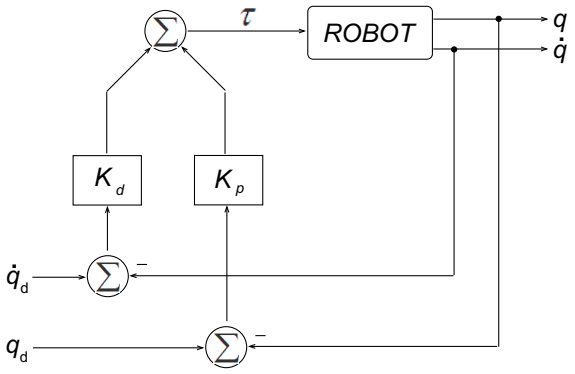

Figura 3.2: Digrama a bloques del control PD.[12]

Hasta este punto, para las definiciones de los controladores Proporcional con retroalimentación de velocidad y PD, no se ha impuesto ninguna restricción al vector de posiciones articulares deseadas q_d. Esto es natural, ya que el nombre dado a un controlador debe caracterizar únicamente a su estructura y no debe depender de la clase de referencias.[12]

Pese al comentario anterior, en la bibliografía sobre control de robots se denotan indistintamente las leyes de control (3.1) y (3.2) como la ley de control PD. La razón comúnmente argumentada para ello es que en el caso particular donde el vector de posición deseada q_d se restringe a ser constante, a partir de la definición de $\tilde{q}$ como $\tilde{q} = q_d - q$ se tiene que $\dot{\tilde{q}} = \dot{q}$, por lo cual, ambas leyes de control (3.1) y (3.2) resultan idénticas, de ahí que se justifique el empleo del mismo nombre para ambos controladores.[12]

En aplicaciones reales, el control PD es local en el sentido que el par o fuerza determinado por dicho controlador, y a ser aplicado en una articulación, sólo depende de la posición y velocidad de dicha articulación y no de las demás articulaciones. Este hecho se traduce en una selección

diagonal de las matrices de diseño k_p y k_d.[12]

El controlador PD dado por la ecuación (3.2) requiere la medición de las posiciones q y $\dot{q}$, así como la especificación de la posición articular deseada q_d (véase la Figura 3.2). Nótese que no es necesario especificar la velocidad ni la aceleración deseadas: $\dot{q}_d$ y $\ddot{q}_d$.

A continuación, se presenta el análisis del control PD de robots manipuladores de n g.d.l.

El comportamiento en malla cerrada de un robot de n g.d.l. bajo control PD se obtiene combinando el modelo (2.21) con la ley de control (3.2),

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + F(\dot{q}) = k_p\tilde{q} - k_d\dot{q}, \tag{3.3}$$

o equivalente en términos del vector de estado $\left[\tilde{q}^T\ \dot{\tilde{q}}^T\right]^T$:

$$\frac{d}{dt}\begin{bmatrix}\tilde{q}\\ \dot{\tilde{q}}\end{bmatrix} = \begin{bmatrix}\dot{\tilde{q}}\\ \ddot{q}_d - M(q)^{-1}\left[k_p\tilde{q} - k_d\dot{q} - C(q,\dot{q})\dot{q} - g(q) - F(\dot{q})\right]\end{bmatrix},$$

que es una ecuación diferencial no lineal y no autónoma. En lo que resta del presente epígrafe se supondrá que el vector de posiciones articulares deseadas q_d es constante. Bajo esta condición, la ecuación de malla cerrada puede escribirse en términos del nuevo vector de estado $\left[\tilde{q}^T\ \dot{\tilde{q}}^T\right]^T$ como:

$$\frac{d}{dt}\begin{bmatrix}\tilde{q}\\ \dot{q}\end{bmatrix} = \begin{bmatrix}-\dot{q}\\ M(q)^{-1}\left[k_p\tilde{q} - k_d\dot{q} - C(q,\dot{q})\dot{q} - g(q) - F(\dot{q})\right]\end{bmatrix} \tag{3.4}$$

Ahora la ecuación diferencial de malla cerrada es no lineal pero autómata. La autonomía se debe a que q_d es constante. Naturalmente, si el modelo del manipulador no posee el término de pares gravitacionales $g(q)$, entonces el único equilibrio será el origen.[12]

Para los robots que cuyos modelos dinámicos no poseen el término gravitacional $g(q)$:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + F(\dot{q}) = \tau$$

La clase de robots descrita por tales modelos son aquellos cuyos movimientos se realizan únicamente en el plano horizontal, así como los diseñados mecánicamente de forma conveniente.

Suponiendo que la posición articular deseada q_d es constante, la ecuación de malla cerrada (3.4) se simplifica en ($g(q) = 0$):

$$\frac{d}{dt}\begin{bmatrix}\tilde{q}\\ \dot{q}\end{bmatrix} = \begin{bmatrix}-\dot{q}\\ M(q_d - \tilde{q})^{-1}\left[k_p\tilde{q} - k_d\dot{q} - C(q_d - \tilde{q},\dot{q})\dot{q} - F(\dot{q})\right]\end{bmatrix} \tag{3.5}$$

la cual, en razón de q_d constante, representa una ecuación diferencial autónoma. El origen $\left[\tilde{q}^T\ \dot{q}^T\right]^T = 0$ es el único equilibrio de la ecuación[12].

3.2.2. Control PID

El control PD es capaz de satisfacer el objetivo de control de posición pura para el caso de robots modelados sin término de gravedad (g(q)=0). En este caso, el procedimiento de sintonía del controlador PD es trivial ya que basta con seleccionar sus matrices de diseño k_p y k_d como simétricas y definidas positivas.

Una alternativa para este controlador es la introducción del componente *Integral* para intentar llevar a cero el error de posición. De este razonamiento surge una justificación para la aplicación del controlador Proporcional-Integral-Derivativo (PID) en el control de robots manipuladores.[12]

La ley de control PID puede expresarse de la siguiente manera:

$$U(t) = \; k_p\,\tilde{q} + k_d\,\dot{\tilde{q}} + k_i \int \tilde{q}\,dt \tag{3.6}$$

donde e es el error de posición articular y las matrices de diseño k_p, k_d, $k_i \in \Re^{nxn}$, llamadas respectivamente las ganancias proporcional, derivativa e integral, son matrices simétricas y definidas positivas convenientemente elegidas. La Figura 3.3 muestra el diagrama de bloques del control PID de robots manipuladores.

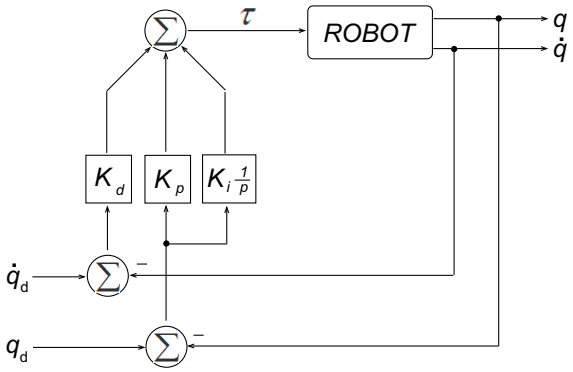

Figura 3.3: Diagrama a bloques del control PID.[12]

La mayoría de los robots manipuladores actuales son controlados mediante controladores PID. La evidencia cotidiana del uso de robots manipuladores pone de manifiesto el buen servicio que se logra en una amplia variedad de aplicaciones cuando se emplea el control PID. No obstante, en contraste con el control PD, el procedimiento de sintonía del control PID, la elección de las matrices definidas positivas k_p, k_d y k_i, dista mucho de ser fácil[12].

3.3. Control moderno

3.3.1. Control PID no lineal (modos deslizantes de 2º orden)

El controlador PID ha sido una alternativa viable para tareas de más baja precisión industrial. Pero últimamente se ha puesto en práctica uno de los mejores controladores, el control PID no lineal, su ley de control puede expresarse de la siguiente manera:

$$U(t) = k_d\ s + k_i \int \mathrm{sgn}(s)\, ds \tag{3.7}$$

en donde s es el *Manyfold de error*, posición y velocidad articular

$$s = \alpha\, \tilde{q} + \dot{\tilde{q}} \tag{3.8}$$

$$\alpha = \frac{Wn}{2} \tag{3.9}$$

Como se puede dar cuenta, en la parte integral se encuentra la función $sgn(s)$, esta función al ser integrada genera atractores terminales para hacer que el error de estado estable llegue a cero. La Figura 3.4 muestra la función sgn(x), para valores mayores a 0 de s la función es igual a 1, para valores menores es igual a -1 y cuando es igual a 0 cae en una indeterminación como se muestra a continuación.

$$\mathrm{sgn(s)} \begin{cases} +1 & s > 0 \\ \text{Indeterminado} & s = 0 \\ -1 & s < 0 \end{cases}$$

Esta función que se encuetra en la parte integral, es muy costosa en cuanto a lineas de programación debido a que se encuentra inmersa en la integral y cuando es igual a cero es indeterminada, pero otra solución propuesta muy aproximada sería con la tangente hiperbólica de la siguiente forma:

$$\lim_{\beta \to \infty}\ \tanh(\beta\, s) \approx \mathrm{sgn}(s) \tag{3.10}$$

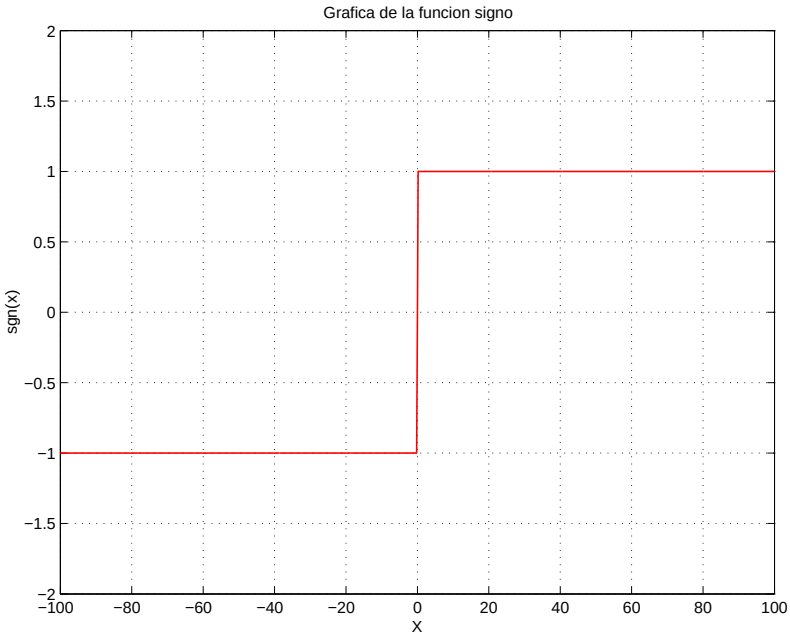

Figura 3.4: Gráfica de la función sgn(x) de MATLAB.

$$\int \text{sgn}(s)\, ds \approx \int \tanh(\beta\, s)\, ds \Big|_{\beta=100} \tag{3.11}$$

Luego entonces la ley de control de la ecuación (3.7) se modifica quedando una aproximación de la siguiente forma:

$$U(t) = k_d\ s + k_i \int \tanh(\beta\, s)\, ds \tag{3.12}$$

en donde se propone a $\beta = 100$ ya que al graficar la $tanh(\beta\, x)$ conforme se incrementa la constante β como lo dice la ecuación (3.10), la pendiente se hace cero aproximándose bastante a la funcion $sgn(x)$ pero sin caer en indeterminación.

A continuación se muestra en la Figura 3.5 la gráfica de la $tanh(\beta\, x)$ con valores de β igual a 1, 25, 50 y 100, en donde se puede observar que con una $\beta = 100$ es suficiente para modificar la pendiente de tal manera que se aproxima bastante a la función sgn.

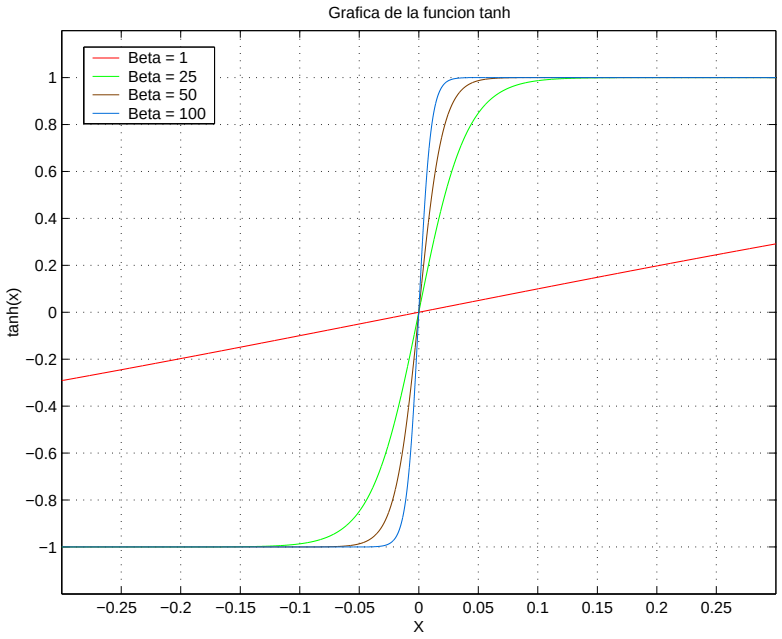

Figura 3.5: Gráfica de la función $tanh(\beta\, x)$.

3.4. Conclusiones

En este capítulo fueron estudiados y presentados de manera breve y concreta las estrategias de control lineal, como el PD y PID, y un controlador PID no lineal o control por modos deslizantes de segundo orden, se presentaron sus contribuciones para la regulación a una coordenada operacional, en el caso del control no lineal se hace una aproximación de la función de saturación a una tangente hiperbólica para que el cálculo numérico sea suave. No se presentan pruebas de estabilidad y métodos estrictos de sintonización de las ganancias de cada control debido a que en esta tesis se reporta el diseño e integración de una plataforma experimental R2SG, dejando como perspectiva la aplicación de controles más rápidos y robustos.

Capítulo 4

Diseño e integración de un robot planar de 2 grados de libertad

4.1. Introducción

La era de la información impulsada por los avances en computación, telecomunicaciones y electrónica está siendo testigo del crecimiento explosivo experimentado por la robótica y tecnologías a fines como la cibernética y la mecatrónica.

Por este motivo, en los últimos años el concepto de robot ha pasado a ser una visión de ciencia ficción como dispositivo mecánico super-humano a ser una realidad como máquina autónoma sorprendentemente animada, con un gran potencial de aplicaciones en un sinfín de actividades cotidianas. Estas máquinas se integran de elementos mecánicos, electrónicos, de control automático, eléctricos y de sistemas de cómputo. A esta clase de máquinas pertenecen los robots manipuladores industriales.

Los robots manipuladores actuales están constituidos físicamente de eslabones mecánicos interconectados por medio de articulaciones, formando un "*brazo*" y una "*mano*" para tomar objetos y herramientas, pudiendo realizar una amplia gama de operaciones físicas en el medio ambiente.

El presente capítulo trata de cómo se fue construyendo el robot R2SG, empezando por el diseño mecánico de cada una de sus piezas y posteriormente por el ensamblaje de las mismas realizado en Mechanical Desktop 6. Una vez diseñado el robot se prosigue a contruirlo físicamente eligiendo antes el material.

Una vez construido, se simula la interface electrónica de potencia adecuada para los servomotores que se van a utilizar, esto se hace para manejar el sentido de cada servomotor mediante un puente H descrito en su momento y posteriormente se realiza el impreso; para ello se utiliza el programa llamado Eagle 4.11 con el cual se hace el ruteo de las pistas correspondientes para así llevarlo a la placa.

Respecto al acondicionamiento de señales, se realizó tanto la simulación como el impreso de

un circuito electrónico digital para interpretar las señales de los codificadores ópticos que poseen los servomotores, se utilizaron circuitos integrados digitales TTL así como también microcontroladores PIC de Microchip que contienen características especiales necesarias, además de ser fáciles de programar.

Igualmente en este capítulo se describen las características de los servomotores a utilizar como también se desarrolla el modelo matemático del robot de 2 grados de libertad y se explican los distintos controles que se experimentaron, desde controles clásicos hasta un control no lineal con seguimiento de trayectorias basado en la teoría de modos deslizantes.

4.2. Diseño y construcción mecánica

La mecánica es rama de la física que se ocupa del movimiento de los objetos y de su respuesta a las fuerzas. Las descripciones modernas del movimiento comienzan con una definición cuidadosa de magnitudes como el desplazamiento, el tiempo, la velocidad, la aceleración, la masa y la fuerza.

El software que se utilizó para el diseño de las piezas mecánicas del robot fue Mechanical Desktop 6. Mechanical Desktop es un programa de modelado paramétrico 3D potente y fácil de usar empleado en aplicaciones de diseño mecánico. Diseñado sobre AutoCAD 2002, el paquete de software de diseño Mechanical Desktop 6 incluye:

- AutoCAD Mechanical 6 con Power Pack (piezas y cálculos 2D)
- Mechanical Desktop 6 con power pack (Mechanical Desktop 6, piezas y cálculos 3D)
- AutoCAD 2002

Al iniciar Mechanical Desktop 6, se tiene la posibilidad de ejecutarlo con o sin power pack. Es un paquete integrado con avanzadas herramientas de modelado tridimensional y funciones de dibujo bidimensional que nos permitirá conceptualizar, diseñar y documentar las piezas mecánicas.

El software Mechanical Desktop dispone de herramientas de diseño para

- Crear piezas a partir de operaciones de boceto y predefinidas.
- Combinar piezas externas y piezas auxiliares.
- Generar ensamblajes y subensamblajes.
- Definir escenas para vistas de dibujo.
- Configurar hojas y vistas de dibujo.
- Anotar dibujos para la documentación final.

- Administrar y volver a utilizar datos de diseño.
- Migrar y modificar datos de sólidos heredados.

Mechanical Desktop 6 funciona sobre la base de AutoCAD 2002 y se sirve de muchas de sus herramientas. Puesto que Mechanical Desktop es un programa de modelado paramétrico, debemos tener cuidado cuando utilicemos los comandos estándar de AutoCAD.

En la etapa de bocetos se pueden utilizar los comandos de AutoCAD para crear la geometría de un boceto. También se pueden utilizar las herramientas de dibujo y edición de AutoCAD para editar la geometría de un boceto después de que haya sido consumido por una operación.

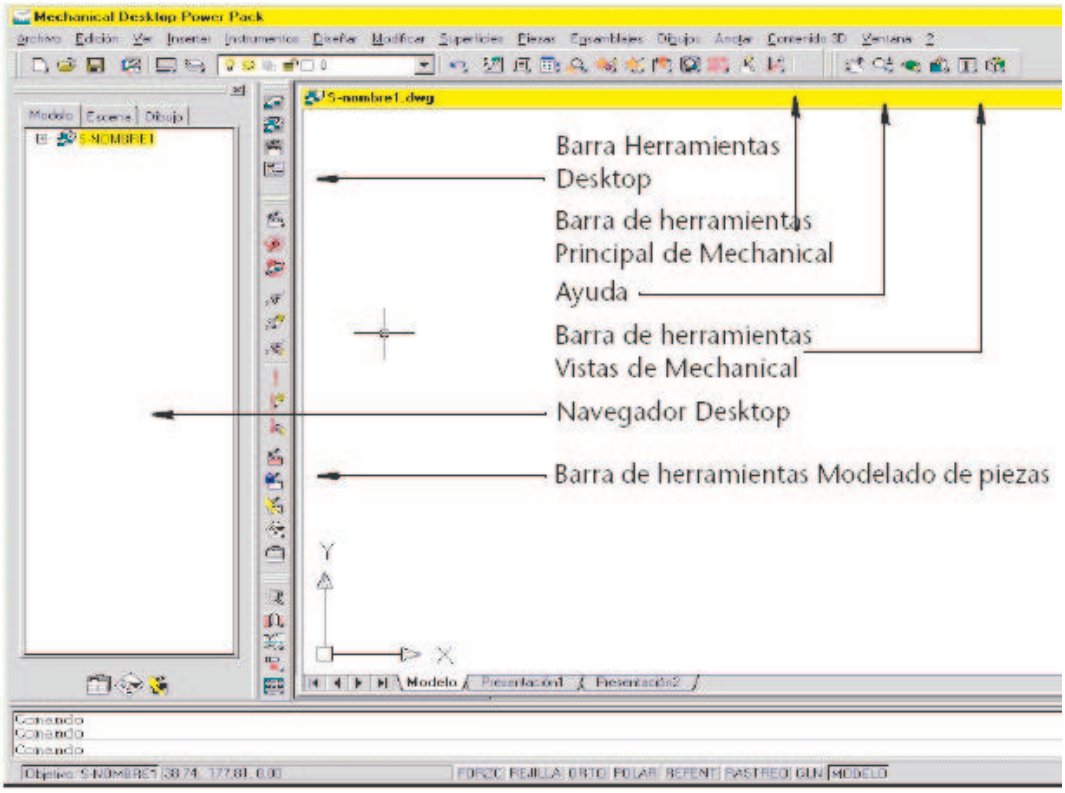

Figura 4.1: Vista del ambiente de Mechanical Desktop 6.

Con la ayuda de este software se diseñaron las distintas piezas necesarias para la construcción del robot R2SG, desde la base, eslabones, casquillos hasta los precisos tornillos, cada una de estas piezas será parte importante para el funcionamiento adecuado del mismo. Una de las cosas a evitar, es la fricción en cada una de las articulaciones, para esto se llegó a la conclusión de utilizar baleros, ya que con estos se obtiene un mínimo de fricción a parte de ser ligeros y pequeños, esto con la finalidad de implementar aplicaciones en interfaces hápticas, en el que la dinámica de fricción debe ser mínima.

La base

La base es una de las piezas fundamentales del robot, es aquella que sostiene a los eslabones del robot R2SG y a los servomotores de las articulaciones, es donde se encuentra sujeto el primer servomotor para la primera articulación y que mueve a todo el mecanismo de eslabones articulado, la base debe ser firme y sólida por ser la que sostiene todo el peso del robot, de forma tal que induzca estabilidad al sistema. El punto donde se encuentra la primera articulación debe ser concéntrico ya que es el punto de equilibrio del robot. La base se sujeta a una plataforma plana por medio de cuatro tornillos con tuerca para darle estabilidad y firmeza a la misma. En

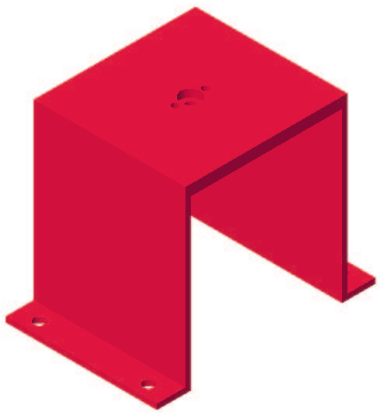

Figura 4.2: Vista isométrica de la base del robot.

el punto concéntrico de la base, en la primer articulación, se ahoga el primero de los baleros en donde entra uno de los casquillos (de los cuales se detalla más adelante) para eliminar así la mayor fricción posible. A un lado del balero, se atornillan los tornillos que sujetan a la base al servomotor de la articulación 1. En el pequeño pero suficiente espacio que hay entre el servomotor y la plataforma a la que va sujeta la base, se alojará la placa electrónica que nos sirve de interfaz entre la computadora y el robot R2SG. En la Figura 4.2 y 4.3 se muestra el diseño de la base del robot R2SG.

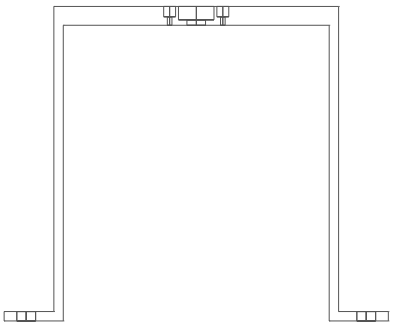

Figura 4.3: Vista de perfil de la base del robot.

Eslabones

Se propusieron dos eslabones de aluminio de diferente longitud, se escogieron de aluminio por que es un metal sólido y ligero. El aluminio resulta ser una aleación cuyas propiedades mecánicas permiten un desempeño óptimo en una plataforma experimental como el robot R2SG. El primer

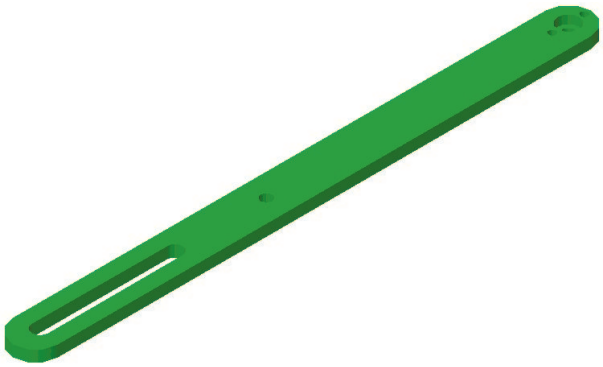

Figura 4.4: Diseño del primer eslabón del robot planar.

eslabon se pensó de 17cm de la primera articulación a la segunda ya que el eslabón completo mide 32cm y el segundo de 15cm para tener un alcanse máximo de 32cm de radio y un mínimo de 2cm. En un extremo del primer eslabón se le hizo una ranura de 9cm para poder mover y equilibrar al robot por medio de un contrapeso. En la articulación uno, que se encuentra a 4.5cm después de la ranura, se perfora para que pueda entrar el primer casquillo que se menciona más adelante, en el otro extremo, se ahogará el segundo balero como se realizó en la base, ambos baleros son de la misma medida. Al igual que la base, se atornillaron los tornillos que sostendrán al segundo servomotor en la articulación dos. La figura 4.4 y 4.5 muestran cómo queda el primer eslabón del robot planar.

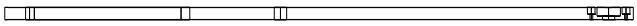

Figura 4.5: Vista de perfil del primer eslabón.

El segundo y último eslabón, simplemente lleva una perforación en la cuál entra el segundo casquillo de la articulación dos y que por medio de tuercas se fija al casquillo. En el otro extremo de la misma forma se la ha hecho otra perforación para cualquier herramineta en específico o en algún momento aumentar un grado de libertad poniendole una articulación más con otro eslabón. La Figura 4.6 muestra como queda el segundo eslabón.

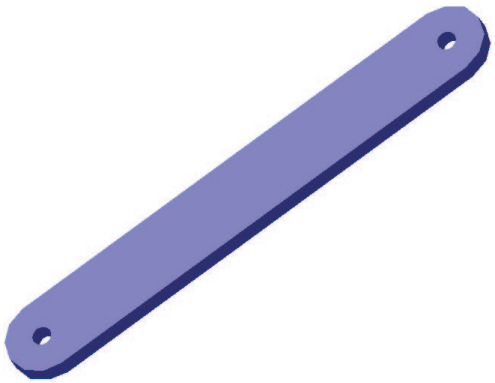

Figura 4.6: Diseño del segundo eslabón del robot R2SG.

Casquillos

Las únicas partes que fueron torneadas son pricisamente los casquillos, para la construcción del robot, son las piezas más importantes ya que tienen que ser muy precisas en cuanto a sus medidas, claro que con el torno es posible lograrlo.

El casquillo consta de dos pequeños cinturones, en la parte de superior, a partir del cinturón, se le hizo rosca para poder sujetar a los eslabones y en la parte inferior un orificio donde entra el eje del servomotor sujetado por un tornillo prisionero. La Figura 4.7 muestra una vista isométrica de los casquillos. En la parte superior del casquillo es donde primero entra una rondana de presión

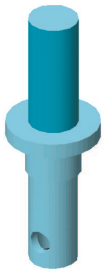

Figura 4.7: Diseño de los casquillos en las articulaciones.

seguido por el eslabón, posteriormente por otra rondana de presión y por último se sujetan con una tuerca que ajusta perféctamente con la rosca del casquillo, las perforaciones tanto de las

rondanas como de los eslabones son poco más grande sus diámetros que el diámetro de la rosca del casquillo, de tal manera que posan sin mayor problema en la base del primer cinturón de mayor diámetro, esta base fue diseñada precisamente para establecer los eslabones y que estos permanezcan firmes. El segundo de los cinturones del casquillo es el que se recuesta ahora en el balero para evitar el rozamiento del primer cinturon con la base del robot, cabe mencionar que la parte inferior del segundo cinturón del casquillo entra en el balero ligeramente a presión para eliminar la fricción entre el casquillo y el balero y que este cumpla con su función.

Como ya se mencionó antes, en la parte inferior del casquillo tiene una pequeña perforación en donde entra el eje del servomotor oprimiéndolo un prisionero para que gire en conjunto con el casquillo. En la Figura 4.8 se dán a conocer las medidas de los casquillos.

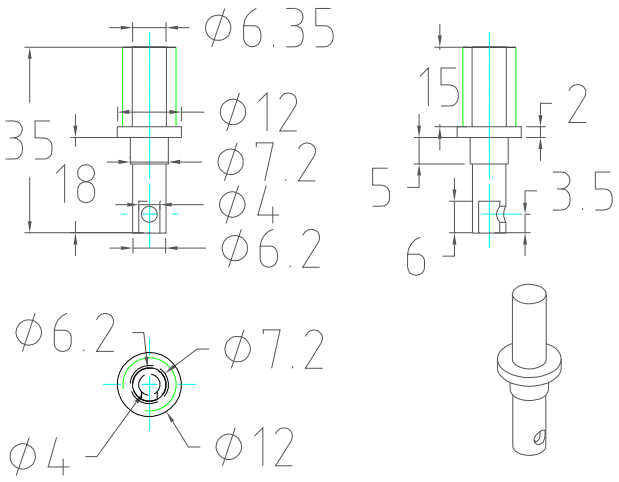

Figura 4.8: Medidas en milímetros de los casquillos.

Ensamblaje

Se empezó por armar el robot desde la base con la primer articulación y posteriormente la segunda articulación. Una vez que se fija el balero ahogado en la base, se introduce el casquillo y se fija el motor por medio de los tornillos aterrajados en la base, además del prisionero que fija al eje del servomotor como se muestra en la Figura 4.9 y 4.10.

Arriba del casquillo, en la parte de la rosca, se introduce el primer eslabón en medio de dos rondanas de presión que ayudan a que no se barra el eslabón al momento de alguna perturbación o constantes cambios de giro del servomotor, sujetas con una tuerca de la misma medida de rosca que el casquillo, obteniendo firmeza en el eslabón y que de igual forma gracias al cinturón que les sirve de base en donde se establecen. En las Figuras 4.11 y 4.12 se muestra el ensamblaje

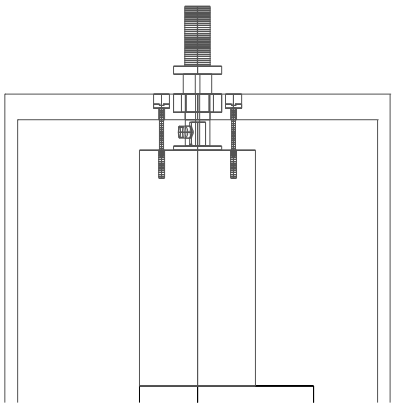

Figura 4.9: Ensamblaje sin texturas de la base, balero, casquillo y servomotor.

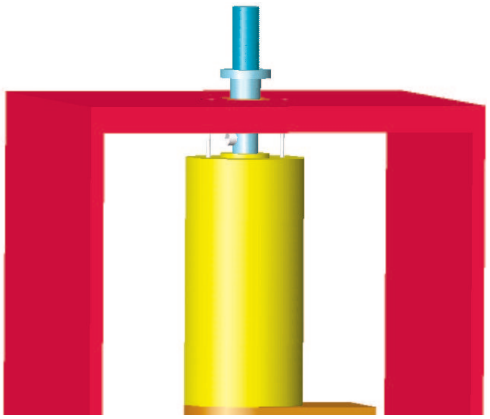

Figura 4.10: Vista isométrica del ensamblaje de la base, balero, casquillo y servomotor.

completo de la primera articulación.

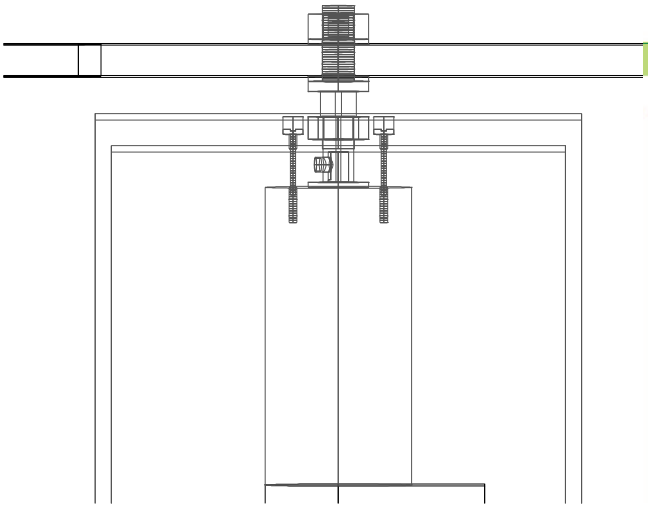

Figura 4.11: Ensamblaje sin texturas de la primera articulación.

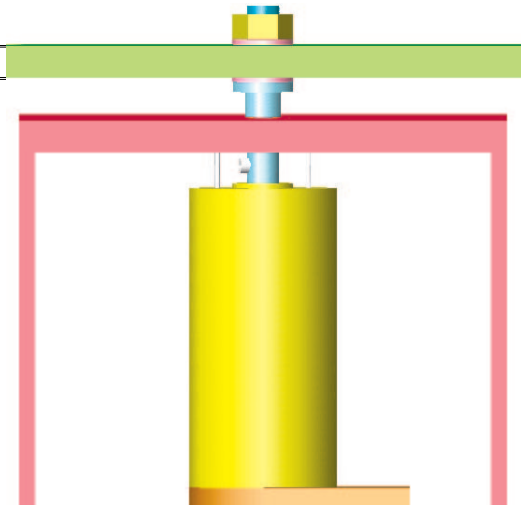

Figura 4.12: Vista isométrica del ensamblaje de la primera articulación.

Así como se construyó la primera articulación se construye de igual forma la segunda. En el extremo contrario a la segunda articulación del primer eslabón se colocó un contrapeso cilíndrico de acero con una perforación con cuerda en el centro del mismo para poder sostenerlo por medio de un tornillo el cual se podrá ajustar hacia adelante o hacia atrás deslizándolo por la ranura del eslabón hasta lograr equilibrar aproximadamente el peso y no forzar la primera articulación debido a la fuerza de gravedad que existe en los eslabones por su mismo peso y el del segundo

servomotor, si no se encuentra equilibrado, lo que ocaciona es que exista mayor fricción en el balero y no pueda girar de manera sencilla consiguiendo que el par del motor sea mayor del que se nesecita. En seguida se muestra por medio de las Figuras 4.13 y 4.14 como queda terminado el robot.

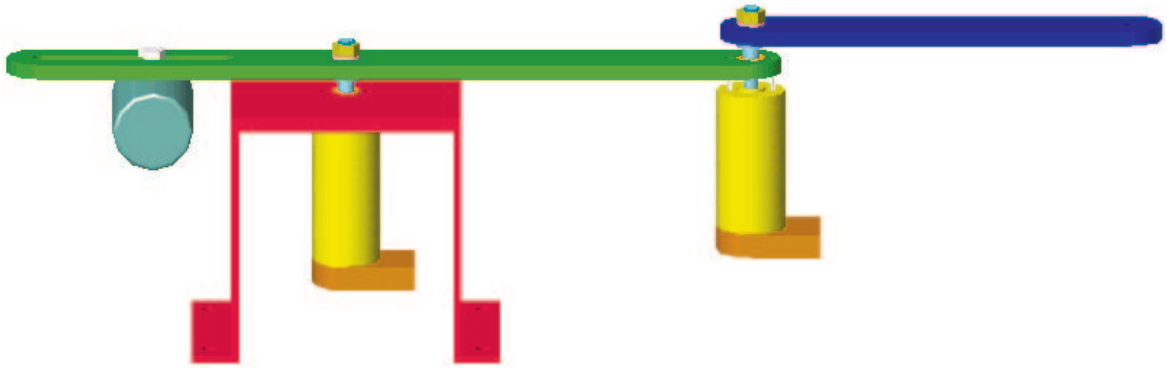

Figura 4.13: Vista isométrica del robot R2SG.

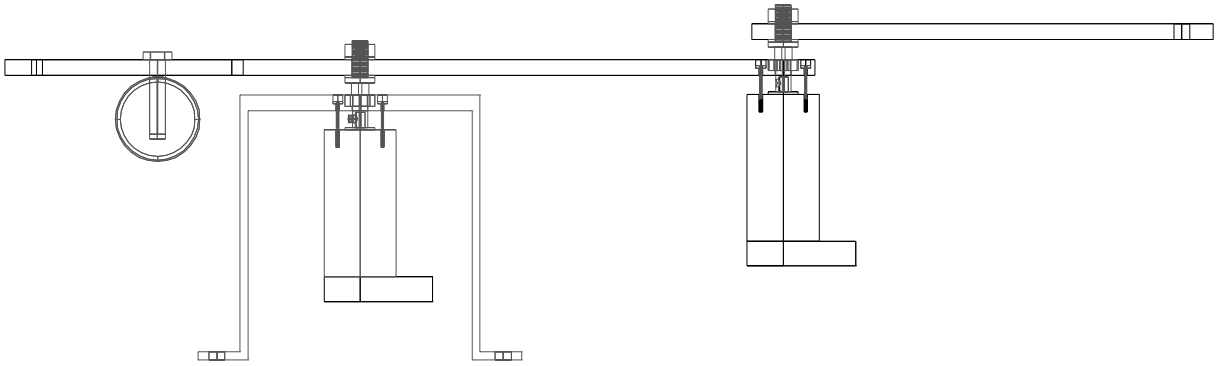

Figura 4.14: Piezas ensambladas del robot R2SG.

Una vez diseñado el robot R2SG se procede a la construcción física la cual se requiere de una buena presición ya que puede tener consecuencias para la estrategia de control utilizada así como ciertos esfuerzos mecánicos en determinados puntos dentro del espacio de trabajo del robot. La Figura 4.15 muestra la fotografía del robot R2SG ya terminado en base al diseño mecánico expuesto anteriormente.

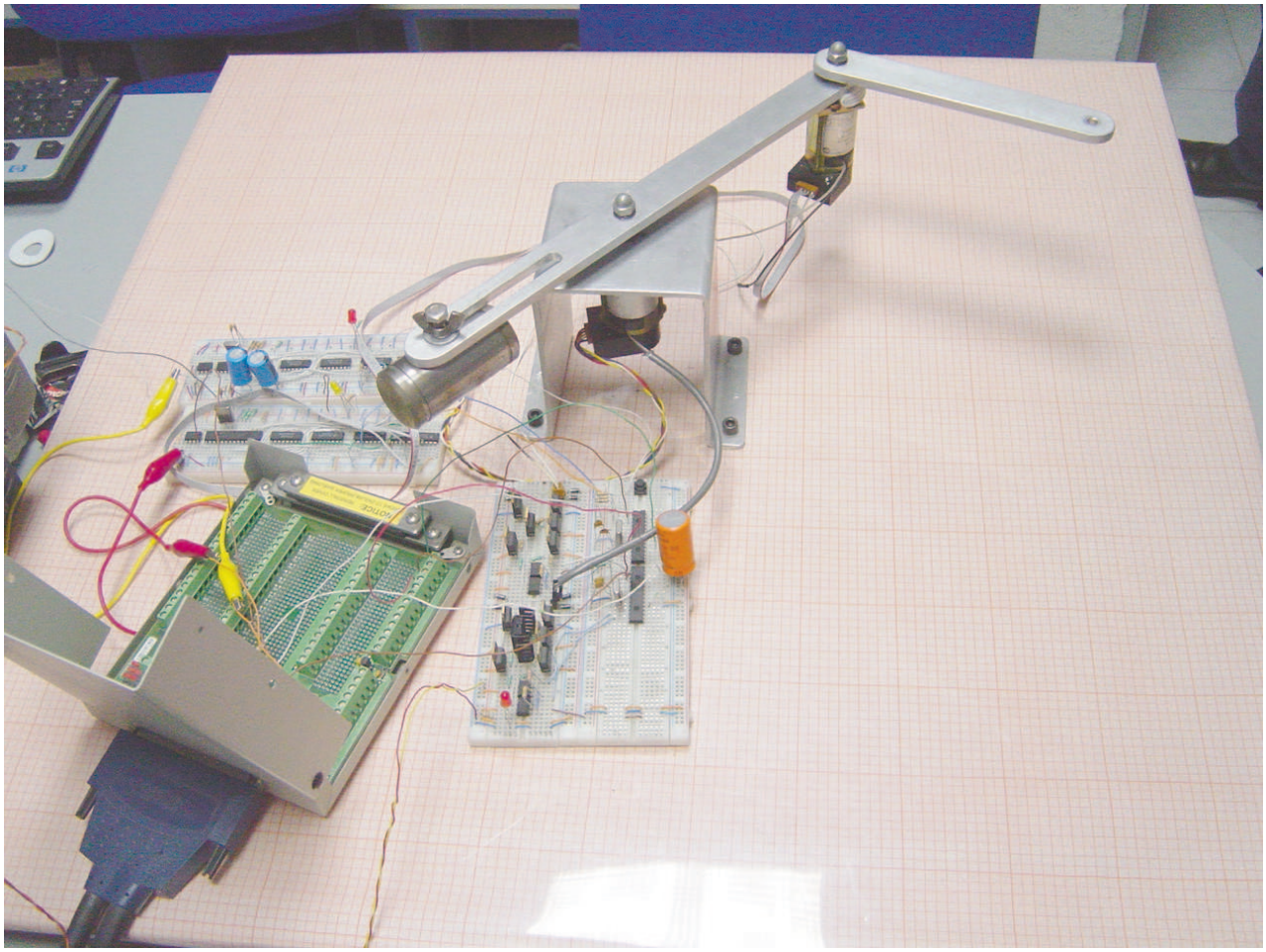

Figura 4.15: Fotografía del robot R2SG (vista superior).

4.3. Servomotores

Los servomotores son un tipo especial de motor que se caracterizan por su capacidad para posicionarse de forma inmediata en cualquier posición dentro de su rango de operación. Para ello, el servomotor espera un tren de pulsos que se corresponden con el movimiento a realizar. Están generalmente formados por un motor y un sensor regularmente acoplado al eje del motor.[19]

4.3.1. Motores de cd

Los motores eléctricos pueden clasificarse en cuatro grandes familias atendiendo a su principio de funcionamiento. Estas familias son:

1. Motores síncronos
2. Motores asíncronos o de inducción
3. Motores de corriente continua (dc)

4. Motores de colector

De todos estos tipos, el que tradicionalmente ha sido más utilizado para funcionar en régimen de velocidad variable es el motor de cd, ya que sus características permiten variar su velocidad de una forma relativamente sencilla manteniendo unas prestaciones satisfactorias (hoy en día, los avances en electrónica de potencia y microelectrónica permiten a los motores síncronos y asíncronos competir en el campo de la variación de velocidad con los motores de cd).

Los motores y generadores eléctricos son un grupo de aparatos que se utilizan para convertir la energía mecánica en eléctrica, o a la inversa, con medios electromagnéticos. A una máquina que convierte la energía mecánica en eléctrica se le denomina generador, alternador o dinamo, y a una máquina que convierte la energía eléctrica en mecánica se le denomina motor.

Si una armadura gira entre dos polos magnéticos fijos, la corriente en la armadura circula en un sentido durante la mitad de cada revolución, y en el otro sentido durante la otra mitad. Para producir un flujo constante de corriente en un sentido, o corriente continua, en un aparato determinado, es necesario disponer de un medio para invertir el flujo de corriente fuera del generador una vez durante cada revolución. En las máquinas antiguas esta inversión se llevaba a cabo mediante un conmutador, un anillo de metal partido montado sobre el eje de una armadura. Las dos mitades del anillo se aislaban entre sí y servían como bornes de la bobina. Las escobillas fijas de metal o de carbón se mantenían en contacto con el conmutador, que al girar conectaba eléctricamente la bobina a los cables externos. Cuando la armadura giraba, cada escobilla estaba en contacto de forma alternativa con las mitades del conmutador, cambiando la posición en el momento en el que la corriente invertía su sentido dentro de la bobina de la armadura. Así se producía un flujo de corriente de un sentido en el circuito exterior al que el generador estaba conectado. Los generadores de corriente continua funcionan normalmente a voltajes bastante bajos para evitar las chispas que se producen entre las escobillas y el conmutador a voltajes altos. El potencial más alto desarrollado para este tipo de generadores suele ser de 1,500 voltios. En algunas máquinas más modernas esta inversión se realiza usando aparatos de potencia electrónica, como por ejemplo rectificadores de diodo.

Los generadores modernos de corriente continua utilizan armaduras de tambor, que suelen estar formadas por un gran número de bobinas agrupadas en hendiduras longitudinales dentro del núcleo de la armadura y conectadas a los segmentos adecuados de un conmutador múltiple. Si una armadura tiene un solo circuito de cable, la corriente que se produce aumentará y disminuirá dependiendo de la parte del campo magnético a través del cual se esté moviendo el circuito. Un conmutador de varios segmentos usado con una armadura de tambor conecta siempre el circuito externo a uno de cable que se mueve a través de un área de alta intensidad del campo, y como resultado la corriente que suministran las bobinas de la armadura es prácticamente constante. Los campos de los generadores modernos se equipan con cuatro o más polos electromagnéticos que aumentan el tamaño y la resistencia del campo magnético. En algunos casos, se añaden interpolos más pequeños para compensar las distorsiones que causa el efecto magnético de la armadura en el flujo eléctrico del campo.

El campo inductor de un generador se puede obtener mediante un imán permanente (magneto) o por medio de un electroimán (dínamo). En este último caso, el electroimán se excita por una corriente independiente o por autoexcitación, es decir, la propia corriente producida en la dínamo sirve para crear el campo magnético en las bobinas del inductor. Existen tres tipos de dinamo según sea la forma en que estén acoplados el inductor y el inducido: en serie, en derivación y en combinación.

En general, los motores de corriente continua son similares en su construcción a los generadores. De hecho podrían describirse como generadores que funcionan al revés. Cuando la corriente pasa a través de la armadura de un motor de corriente continua, se genera un par de fuerzas debido a la acción del campo magnético, y la armadura gira. La función del conmutador y la de las conexiones de las bobinas del campo de los motores es exactamente la misma que en los generadores. La revolución de la armadura induce un voltaje en las bobinas de ésta. Este voltaje es opuesto al voltaje exterior que se aplica a la armadura, y de ahí que se conozca como voltaje inducido o fuerza contraelectromotriz. Cuando el motor gira más rápido, el voltaje inducido aumenta hasta que es casi igual al aplicado. La corriente entonces es pequeña, y la velocidad del motor permanecerá constante siempre que el motor no esté bajo carga y tenga que realizar otro trabajo mecánico que no sea el requerido para mover la armadura. Bajo carga, la armadura gira más lentamente, reduciendo el voltaje inducido y permitiendo que fluya una corriente mayor en la armadura. Debido a que la velocidad de rotación controla el flujo de la corriente en la armadura, deben usarse aparatos especiales para arrancar los motores de corriente continua. Cuando la armadura está parada, ésta no tiene realmente resistencia, y si se aplica el voltaje de funcionamiento normal, se producirá una gran corriente, que podría dañar el conmutador y las bobinas de la armadura. El medio normal de prevenir estos daños es el uso de una resistencia de encendido conectada en serie a la armadura, para disminuir la corriente antes de que el motor consiga desarrollar el voltaje inducido adecuado. Cuando el motor acelera, la resistencia se reduce gradualmente, tanto de forma manual como automática. La velocidad a la que funciona un motor depende de la intensidad del campo magnético que actúa sobre la armadura, así como de la corriente de ésta. Cuanto más fuerte es el campo, más bajo es el grado de rotación necesario para generar un voltaje inducido lo bastante grande como para contrarrestar el voltaje aplicado. Por esta razón, la velocidad de los motores de corriente continua puede controlarse mediante la variación de la corriente del campo.

Tradicionalmente, la máquina de corriente continua ha sido la más utilizada en aquellas aplicaciones que requieren variación de velocidad (regulación de procesos) ya que resultan muy sencillas de controlar incluso con métodos clásicos.

Los motores de cd del robot R2SG son marca Pittman, a continuación se enumeran sus características mas importantes.

1. Modelo LO-COG 9391E001 R5
2. 12v a 1.545A

3. Resolución del codificador 96 a 2048

4. 6500 rpm

Figura 4.16: Fotografía del servomotor Pittman LO-COG 9391E001 R5.

4.3.2. Sensores internos

Para conseguir que el robot R2SG realice su tarea con la adecuada precisión y velocidad, será necesario que tenga conocimiento tanto de su propio estado como del estado de su entorno. La información relacionada con su estado (fundamentalmente la posición de sus articulaciones) la consigue con los denominados sensores internos, mientras que la que se refiere al estado de su entorno, se adquiere con los sensores externos. El robot R2SG posee un sensor interno en cada articulación, el Cuadro 4.1 muestra algunos tipos de sensores internos.[25]

Para el control de posición angular del robot R2SG se emplearon fundamentalmente los denominados encoders incrementales. Se propuso otro tipo de sensor como los potenciomtros analógicos pero se descartaron debido a que dan bajas prestaciones como se muestra en el Cuadro 4.2.

Los codificadores ópticos o encoders incrementales constan, en su forma más simple, de un disco transparente con una serie de marcas opacas colocadas radialmente y equidistantes entre sí, de un sistema de iluminación en el que la luz es colimada de forma adecuada, y de un elemento fotorreceptor; aunque de otra manera, el disco suele ser metálico con perforaciones de igual forma

Sensores internos		
Posición	Analógicos	Poteciómetros Resolver Sincro Inductosyn LVDT
	Digitales	Encoders absolutos Encoders incrementales Regla óptica
Velocidad		Taco-generatriz
Presencia		Inductivo Capacitivo Efecto Hall Inductivo Célula Reed Óptico Ultrasónico Contacto

Cuadro 4.1: Tipos de sensores internos de robots.[25]

Tabla comparativa				
	Robustez mecánica	Rango dinámico	Resolución	Estabilidad térmica
Codificador O. (Encoder)	mala	media	buena	buena
Potenciómetro	regular	mala	mala	mala

Cuadro 4.2: Comparación entre dos sensores de posición angular[25].

equidistantes. El eje cuya posición se quiere medir va acoplado al disco transparente o metálico. Con esta disposición, a medida que el eje gire se irán generando pulsos en el receptor o receptores cada vez que la luz atraviese cada marca o perforación, y llevando una cuenta de estos pulsos es posible conocer la posición del eje.[25]

Existe, sin embargo, el problema en el que se desconoce en un momento dado de si se está realizando un giro en un sentido o en el opuesto, con el peligro que supone no estar contando adecuadamente. Una solución a este problema consiste en disponer de otra franja de marcas, desplazada de la anterior de manera que el tren de pulsos que con ella se genere esté desplazado 90° eléctricos con respecto al generado por la primera franja o colocar otro fotorreceptor en el que la distancia entre fotorreceptores sea menor que la medida de las marcas opacas o perforaiones en el disco. De esta manera, con un circuito relativamente sencillo, es posible obtener una señal adicional que indique cuál es el sentido de giro y que actúe sobre el contador correspondiente indicandole que incremente o reduzca la cuenta que se está realizando. Es necesario además disponer de una marca de referencia sobre el disco que indique que se ha dado una vuelta completa y que, por tanto, se ha de empezar la cuenta de nuevo. Esta marca sirve también para poder comenzar a contar tras recuperarse de una caída de tensión.[25]

La resolución de este tipo de sensores depende directamente del número de marcas que se pueden poner físicamente en el disco. Un método relativamente sencillo para aumentar esta resolución es, no solamente contabilizar los flancos de subida de los trenes de pulsos, sino contabilizar también los de bajada, incrementando así la resolución del captador, pudiéndose llegar, con ayuda de circuitos adicionales como el antes mencionado LS7184, hasta 4 veces la resolución del codificador.[25]

En comparación con los codificadores o *encoders* absolutos, el funcionamiento básico de los *encoders* absolutos es similar al de los incrementales. Se tiene una fuente de luz con las lentes de adaptación correspondientes, un disco graduado y unos fotorreceptores. En este caso, el disco transparente se divide en un número determinado de sectores (potencia de 2), codificándose cada uno de ellos según un código binario cíclico (normalmente código de Gray) que queda representado por zonas transparentes y opacas dispuestas radialmente. No es necesario ahora ningún contador o electrónica adicional para detectar el sentido del giro, pues cada posición (sector) es codificado de forma absoluta. Su resolución es fija, y vendrá dada por el número de anillos que posea el disco graduado. Las resoluciones habituales van desde 2^8 a 2^{19} bits (desde 256 a 524288 posiciones distintas).[25]

Normalmente los sensores de posición se acoplan al eje del motor. Considerando que en la mayor parte de los casos entre el eje del motor y el de la articulación se sitúa un reductor de relación N, cada movimiento de la articulación se verá multiplicado por N al ser medido por el sensor. Éste aumentara así su resolución, multiplicándola por N.[25]

Algunos encoders absolutos utilizan otro encoder absoluto más pequeño conectado por un engranaje reductor al principal, de manera que cuando éste gire una vuelta completa, el codificado

adicional avanzará una posición. Son los denominados encoder absolutos multivuelta.[25]

Esta misma circunstancia originará que en el caso de los codificadores incrementales la señal de referencia o marca de cero, sea insuficiente para detectar el punto origen para la cuenta de pulsos, pues habrá N posibles puntos de referencia para un giro completo de la articulación. Para distinguir cual de ellos es el correcto se suele utilizar un detector de presencia denominado sincronismo, acoplado directamente al eslabón del robot que se considere. Cuando se conecta el robot desde una situación de apagado, es preciso, ejecutar un procedimiento de búsqueda de referencias para los sensores (sincronizado). Durante su ejecución se leen los detectores de sincronismo que detectan la presencia o ausencia de eslabón del robot. Cuando se detecta la presencia o ausencia de pieza, o viceversa, se atiende al encoder incremental, tomándose como posición de origen la correspondiente al primer pulso de marca de cero que aquél genere.[25]

Los encoders pueden presentar problemas mecánicos debido a la gran precisión que se debe tener en su fabricación. La contaminación ambiental puede ser una fuente de interferencias en la transmisión óptica. Son dispositivos particularmente sensibles a golpes y vibraciones, estando su margen de temperatura de trabajo limitado por la presencia de componentes electrónicos.[25]

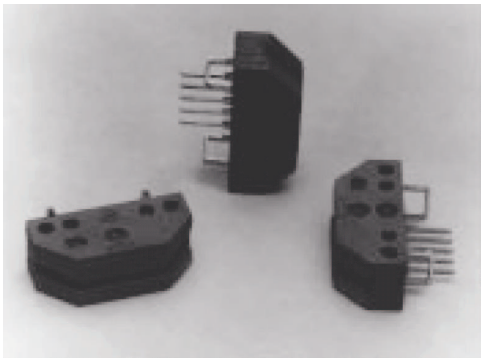

Figura 4.17: Foto de los codificadores ópticos internos del robot R2SG.[1]

Los codificadores ópticos del robot R2SG que se encuentran perfectamente acoplados a los motores son de la serie HEDS-9040, los cuales tienen tres canales ópticos incrementales, sus principales características se enumeran a continuación:[1]

1. 2 canales de salida en cuadratura con 1 pulso de reinicio
2. No requiere de ajuste de señal
3. Opera de -40^0 C a 100^0 C
4. Compatible con la tecnología TTL

La Figura 4.18 muestra el diagrama a bloques de los codificadores ópticos incrementales de la serie HEDS-9040. Constan de modulos emisores y receptores, estos modulos acoplados al disco, traducen el movimiento rotatorio en tres señales de salida digital. Como se ve en el diagrama a bloques, un solo LED (diodo emisor de luz) emite luz pasando por una lente de policarbonato a través de las perforaciones del disco llegando a los fotodetectores, después, por medio de circuitería, se hace el proceso necesario para producir las tres señales de salida digital.[1]

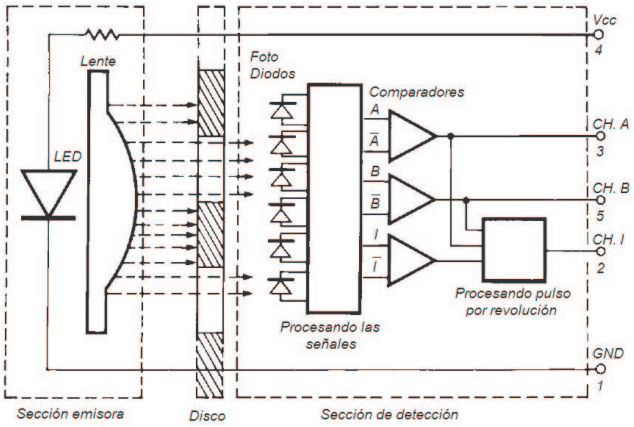

Figura 4.18: Diagrama a bloques de los codificadores ópticos de la serie HEDS-9040.[1]

Las tres señales de salida digitales requieren de resistesias de pull-up de 2.7kΩ (±10 %) como se muestra en la Figura 4.19, con esta configuración, cada una de las tres salidas digitales soporta una sola entrada TTL.[1]

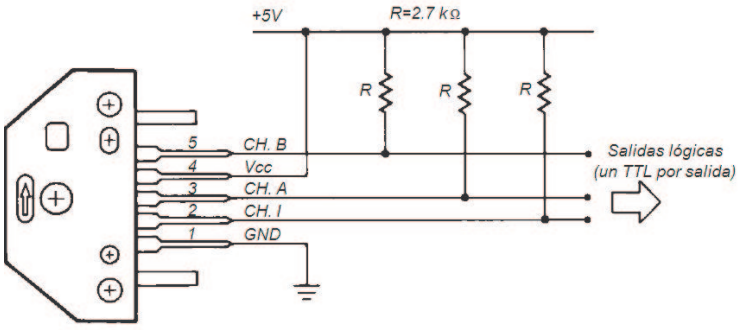

Figura 4.19: Interfaz eléctrica del codificador óptico.[1]

4.4. Interfaz electrónica de potencia

Electrónica, campo de la ingeniería y de la física aplicada relativo al diseño y aplicación de dispositivos, por lo general circuitos electrónicos, cuyo funcionamiento depende del flujo de electrones para la generación, transmisión, recepción y almacenamiento de información. Esta información puede consistir en voz o música (señales de voz) en un receptor de radio, en una imagen en una pantalla de televisión, o en números u otros datos en un ordenador o computadora.

Los circuitos electrónicos ofrecen diferentes funciones para procesar esta información, incluyendo la amplificación de señales débiles hasta un nivel utilizable; la generación de ondas de radio; la extracción de información, como por ejemplo la recuperación de la señal de sonido de una onda de radio (demodulación); el control, como en el caso de la superposición de una señal de sonido a ondas de radio (modulación), y operaciones lógicas, como los procesos electrónicos que tienen lugar en las computadoras.

La introducción de los tubos de vacío a comienzos del siglo XX propició el rápido crecimiento de la electrónica moderna. Con estos dispositivos se hizo posible la manipulación de señales, algo que no podía realizarse en los antiguos circuitos telegráficos y telefónicos, ni con los primeros transmisores que utilizaban chispas de alta tensión para generar ondas de radio. Por ejemplo, con los tubos de vacío se pudieron amplificar las señales de radio y de sonido débiles, y además podían superponerse señales de sonido a las ondas de radio. El desarrollo de una amplia variedad de tubos, diseñados para funciones especializadas, posibilitó el rápido avance de la tecnología de comunicación radial antes de la II Guerra Mundial, y el desarrollo de las primeras computadoras, durante la guerra y poco después de ella.

Hoy día, el transistor, inventado en 1948, ha reemplazado casi completamente al tubo de vacío en la mayoría de sus aplicaciones. Al incorporar un conjunto de materiales semiconductores y contactos eléctricos, el transistor permite las mismas funciones que el tubo de vacío, pero con un coste, peso y potencia más bajos, y una mayor fiabilidad. Los progresos en la tecnología de semiconductores, atribuible en parte a la intensidad de las investigaciones asociadas con la iniciativa de exploración del espacio, llevó al desarrollo, en la década de 1970, del circuito integrado. Estos dispositivos pueden contener centenares de miles de transistores en un pequeño trozo de material, permitiendo la construcción de circuitos electrónicos complejos, como los de los microordenadores o microcomputadoras, equipos de sonido y vídeo, y satélites de comunicaciones.

Los transistores se componen de semiconductores. Se trata de materiales, como el silicio o el germanio, dopados (es decir, se les han incrustado pequeñas cantidades de materias extrañas), de manera que se produce un exceso o una carencia de electrones libres. En el primer caso, se dice que el semiconductor es del tipo **n**, y en el segundo, que es del tipo **p**. Combinando materiales del tipo **n** y del tipo **p** se puede producir un diodo. Cuando este se conecta a una batería de manera tal que el material tipo **p** es positivo y el material tipo **n** es negativo, los electrones son repelidos desde el terminal negativo de la batería y pasan, sin ningún obstáculo, a la región **p**, que carece de electrones. Con la batería invertida, los electrones que llegan al material **p** pueden pasar sólo con muchas dificultades hacia el material **n**, que ya está lleno de electrones libres, en

cuyo caso la corriente es prácticamente cero.

El transistor bipolar fue inventado en 1948 para sustituir al tubo de vacío triodo. Está formado por tres capas de material dopado, que forman dos uniones **pn** (bipolares) con configuraciones **pnp** o **npn**. Una unión está conectada a la batería para permitir el flujo de corriente (polarización negativa frontal, o polarización directa), y la otra está conectada a una batería en sentido contrario (polarización inversa). Si se varía la corriente en la unión de polarización directa mediante la adición de una señal, la corriente de la unión de polarización inversa del transistor variará en consecuencia. El principio se puede utilizar para construir amplificadores en los que una pequeña señal aplicada a la unión de polarización directa provocará un gran cambio en la corriente de la unión de polarización inversa.

Otro tipo de transistor es el de efecto de campo (FET, acrónimo inglés de Field-Effect Transistor), que funciona sobre la base del principio de repulsión o de atracción de cargas debido a la superposición de un campo eléctrico. La amplificación de la corriente se consigue de modo similar al empleado en el control de rejilla de un tubo de vacío. Los transistores de efecto campo funcionan de forma más eficaz que los bipolares, ya que es posible controlar una señal grande con una cantidad de energía muy pequeña.

Actualmente, la eletrónica de potencia, dedicada al estudio de los conversores electrónicos, es una de las especialidades electrónicas de mayor auge y dinamismo. Constantemente están desarrollándose nuevas topologías y métodos de control para los conversores electrónicos mejorando sus prestaciones[15].

La interfaz electrónica de potencia que requiere el robot R2SG, es el manejo de los motores de cd. Para el manejo de sentido del servomotor, se ha recurrido al puente H, que es básicamente un arreglo de cuatro interruptores (transistores) acomodados de la siguiente manera:

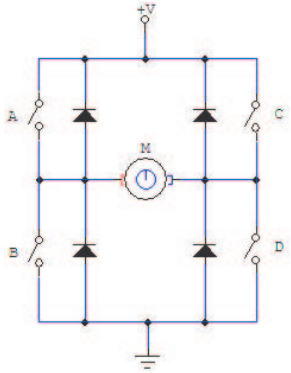

Figura 4.20: Circuito puente H.

Estos interruptores A, B, C y D que se muestran en la Figura 4.20 pueden ser de transistores

bipolares, mosfets, jfets, relés o de cualquier combinación de elementos, en este caso se utilizarán transistores Darlington.

El objetivo central es el de poder controlar el sentido de un motor de corriente continua sin la necesidad de aplicar voltaje negativo. Si se cierran solamente los contactos A y D la corriente circulará en un sentido a través del motor o del elemento conectado en la parte central como en la Figura 4.21.

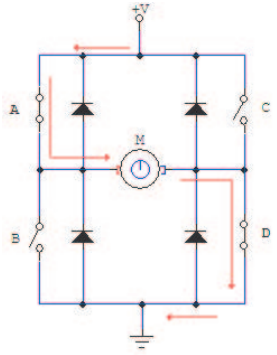

Figura 4.21: Cerrando los interruptores A y D (sentido 1).

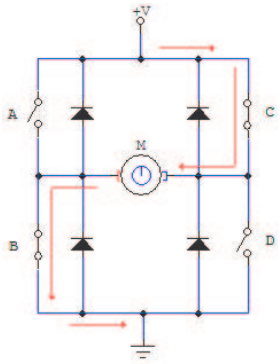

Figura 4.22: Cerrando los interruptores B y C (sentido 2) contrario a la figura 4.21.

Y si se cierran solamente los contactos B y C como en la Figura 4.22, la corriente circulará en sentido contrario.

Hay que observar también que un puente H necesita de cuatro diodos de protección para el motor. Un puente H tiene por lo general cuatro estados de operación:

Funciones	Interruptores		Salida	
	A D	C D	A B	C D
Motor en libertad de acción	0	0	0	0
Motor gira en un sentido	1	0	1	0
Motor gira en el otro sentido	0	1	0	1
Motor frenado	1	1	1	1

Cuadro 4.3: Modo de configuración del puente H.

Donde un 0 corresponde a un interruptor abierto o una salida sin alimentación y un 1 corresponde a un interruptor cerrado o una salida con alimentación. Como ya sabemos, el robot consta de dos articulaciones en las cuales se encuentran los servomotores que vamos a utilizar, esto quiere decir que se necesitó un puente H para cada uno de los servomotores. El circuito que

nos muestra la Figura 4.23 es el que se simuló y construyó para cada servomotor.

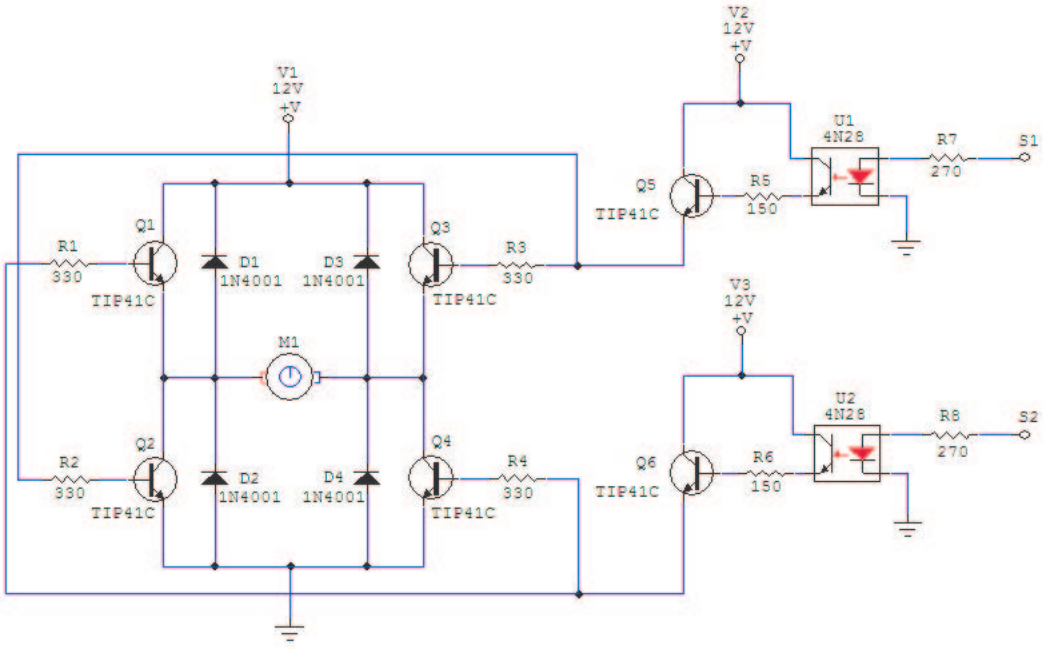

Figura 4.23: Interfaz de potencia para el manejo de cada motor de cd en las articulaciones.

Los llamados interruptores son transistores Darlington, una conexión Darlington consiste en dos transistores conectados en cascada cuya ganancia de corriente total es el producto de las ganancias de corriente individuales. Como la ganancia de corriente es mucho mayor, una conexión Darlington puede tener una impedancia de entrada muy alta y puede producir corrientes de salida muy grandes. Las conexiones Darlington se usan con reguladores de tensión y amplificadores de potencia.

La Figura 4.24*a* muestra un par Darlington. Como la coriente de emisor de Q_1 es la corriente de base de Q_2, el par Darlington tiene una corriente total de:

$$\beta = \beta_1 \beta_2 \tag{4.1}$$

En la Figura 4.24*b* se muestra otra conexión Darlington, denominada Darlington complementario, una conexión de transistores npn y pnp. La corriente de conexión de Q_1 es la corriente de base de Q_2. Si el transistor pnp tiene una ganancia de corriente de β_1 y el transistor npn de salida tiene una ganancia de corriente de β_2, el Darlington complementario actúa como un transistor único pnp con una ganancia de corriente de $\beta_1\beta_2$. El Darlington complementario se usa algunas veces en amplificadores de potencia en contrafase.

Los fabricantes de transistores Darlington pueden poner un par Darlington dentro de un solo encapsulado, como se ve en la Figura 4.24*c*. Este dispositivo con tres terminales, conocido como

transistor Darlington, actúa como si fuese un solo transistor con una ganancia de corriente extremadamente alta.[15]

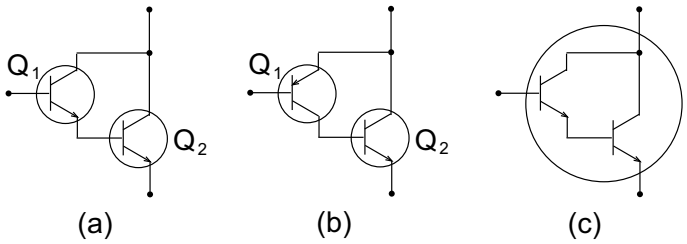

Figura 4.24: a)Par Darlington; b)Darlington complementario; c)transistor Darlington.[15]

Los transistores Darlington que se utilizaron en el circuito puente H de la Figura 4.23 son los TIP41C los cuales tienen una ganancia de corriente de 10,000 a 2A. Debido a que en la parte de potencia se manejan corrientes mucho mayores que en la parte de acondicionamiento de señales, es necesario separarlas, esto se hace mediante las circuitos integrados 4N28, los cuales internamente poseen un diodo emisor de luz y un fototransistor, la función del LED es exitar la base de un fototransitor y así lograr el opto-acoplamiento quedando como entradas digitales S1 y S2.

Para regular la velocidad de los servomotores, se utilizó la modulación por ancho de pulso (PWM), que se explicará más adelante, la cual se aplica en S1 y S2 de la Figura 4.23 según el sentido del servomotor.

4.5. Acondicionamiento de señales

El *diseño* digital se ocupa del diseño de circuitos electrónicos digitales. El tema se conoce también por otros nombres, como *diseño lógico*, *circuitos conmutadores*, *lógica digital* y *sistemas digitales*. Los circuitos digitales se emplean en el diseño de sistemas, por ejemplo computadores digitales, calculadoras electrónicas, dispositivos digitales de control, equipo de comunicación digital y muchas otras aplicaciones que requieren hardware digital electrónico.

Los componentes que se utilizan para construir sistemas digitales están encapsulados en paquetes de circuitos integrados. Los circuitos de integración a pequeña escala contienen varias compuertas o *flip-flops* en un solo paquete.

Los circuitos digitales combinacionales donde se emplean compuertas como la and, or, nor, nand, or exclusiva, etc. solamente dependen de las entradas presentes en ese tiempo. Aunque cualquier sistema digital es susceptible de tener circuitos combinacionales, la mayoría de los sistemas que se encuentran en la práctica también incluyen elementos de memoria, los cuales requieren que el sistema se describa en términos de *lógica secuencial*.

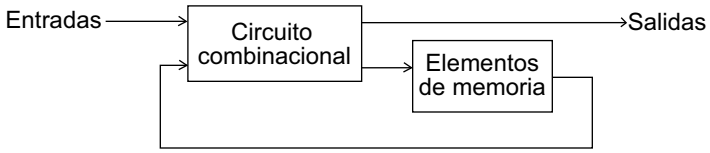

Figura 4.25: Diagrama de bloques de un circuito secuencial.[16]

Consta de un circuito combinacional al que se conectan elementos de memoria para formar una trayectoria de retroalimentación. Los elementos de memoria son dispositivos capaces de almacenar dentro de ellos información binaria. La información binaria almacenada en los elementos de memoria en cualquier momento dado define el *estado* del circuito secuencial. El circuito secuencial recibe información binaria de entradas externas. Estas entradas, junto con el estado presente de los elementos de memoria, determinan el valor binario de las entradas de salida. También determinan las condiciones para cambiar el estado de los elementos de memoria. El diagrama de bloques demuestra que las salidas externas en un circuito secuencial son funciones no solo de las entradas externas sino también del estado presente de los elementos de memoria. El siguiente estado de los elementos de memoria también es una función de las entradas externas y del estado presente. Por tanto un circuito secuencial está especificado por una secuencia de tiempo de entradas, salidas y estados internos.

Hay dos tipos de circuitos secuenciales. Su clasificación depende del temporizado de sus señales. Un circuito secuencial *síncrono* es un sistema cuyo comportamiento puede definirse por el conocimiento de sus señales en instantes discretos de tiempo. Los elementos de memoria que por lo común se utilizan en los circuitos secuenciales asíncronos son dispositivos de retardo de tiempo. La capacidad de memoria de un ciruito de retardo de tiempo se debe al hecho de que toma un tiempo finito para que la señal se propague a través del dispositivo.

Los elementos de memoria que se utilizan en los circuitos secuenciales de reloj se llaman *flip-flops*. Estos circuitos son celdas binarias capaces de almacenar un bit de información. Un circuito flip-flop tiene dos salidas, uno para el valor normal y otra para el valor complementario del bit almacenado en él. La información binaria puede entrar a un flip-flop en una gran variedad de formas, hecho que da lugar a diferentes tipos de flip-flops.

Un circuito flip-flop puede mantener un estado binario en forma definida (en tanto se suministre potencia al circuito) hasta que recibe la dirección de una señal de entrada para cambiar de estado. La diferencia principal entre los diversos tipos de flip-flops está en el número de entradas que poseen y en la manera en la cual las entradas afectan al estado binario.

Un circuito secuencial temporizado consta de de un grupo de flip-flops y compuertas combinacionales conectadas para formar una trayectoria de retroalimentación. Los flip-lops son esenciales porque, cuando estan ausentes, el circuito se reduce a un circuito combinacional puro (siempre que no haya trayectoria de retroalimentación). Un circuito solo con flip-flops se considera un

circuito secuencial incluso cuando están ausentes las compuertas combinacionales.

Un circuito de integración a media escala (MSI) que contiene celdas de almacenamiento en su interior es, por definición, un circuito secuencial. Los circuitos MSI que incluyen flip-flops u otras celdas de almacenamiento por lo común se clasifican según la función que realizan más que por el nombre "circuito secuencial". Estos circuitos MSI se clasifican en una de tres categorías: registros, contadores o memoria de acceso aleatorio.

Un registro es un grupo de celdas de almacenamiento binario adecuadas para mantener información binaria. Un grupo de flip-flops constituye un registro, ya que cada flip-flop es una celda binaria capaz de almacenar un bit de información. Un registro de n-bit tiene un grupo de n flip-flops y es capaz de almacenar cualquier información binaria que contenga n bits. Además de los flip-flops, un registro puede tener compuertas combinacionales que realicen ciertas tareas de procesamiento de datos. En su definición más amplia, un registro consta de un grupo de flip-flops y compuertas que efectúan su transición. Los flip-flops mantienen información binaria y las compuertas controlan cuando y como se transfiere información nueva al registro.

Un contador es un registro que pasa a través de una secuencia determinada de estados bajo la aplicación de pulsos de entrada. Las compuertas de un contador están conectadas de tal forma que producen una secuencia prescrita de estados binarios en el registro[16].

Los servomotores tienen codificadores ópticos los que proporcionan dos señales digitales (canal A y B) las cuales son dos trenes de pulsos desfasados 90° una con respecto a la otra. Para decodificar el sentido se introdujeron las dos señales A y B en el circuito integrado LS7184 el cual nos devuelve el pulso de reloj y el sentido para que de esta forma se introduzcan en un contador binario, en este caso el 74LS191.

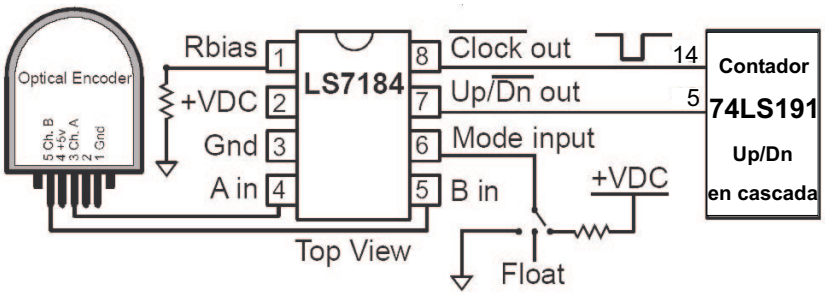

Figura 4.26: Configuración del circuito LS7184.[24]

El circuito integrado LS7184 es una interfaz entre el codificador óptico del servomotor y los contadores binarios para saber la posición, este circuito acepta de entrada dos canales comunes A y B en cuadratura y devuelve dos señales para contadores ascendentes/descendentes comunes, es compatible con la tecnología TTL y CMOS, proporciona una resolución de X1, X2 y X4, esto

quiere decir que puede aumentar la resolución del codificador óptico al doble y hasta cuatro veces más, es de bajo consumo, no requiere de un reloj externo y su rango de operación es de 3v a 5v.

La Figura 4.26 muestra la configuración del LS7184 teniendo como entrada el codificador óptico del servomotor y como salida a los contadores TTL 74LS191 en cascada.

RBias es una resistencia conectada entre el pin 1 y tierra que se escoje de acuerdo al Cuadro 4.4 para ajustar el ancho de pulso de salida del reloj y en el modo de entrada (Mode input) en el pin 6 se tienen 3 estados, para seleccionar una resolución de X1 se conecta a tierra, para X2 a la alimentación y para X4 se deja flotando[24].

RBias	Ancho de pulso	Max A,B Frec.(X1)	Max A,B Frec.(X2)	Max A,B Frec.(X4)
20 kOhm	500 ns	1 MHz	500 kHz	250 kHz
220 kOhm	3 μs	167 kHz	83 kHz	42 kHz
750 kOhm	9.5 μs	53 kHz	26 kHz	13 kHz
2 MOhm	28 μs	18 kHz	9 kHz	4.5 kHz
5.1 MOhm	65 μs	7.7 kHz	3.8 kHz	1.9 kHz
8.2 MOhm	119 μs	4.2 kHz	2.1 kHz	1.1 kHz
10 MOhm	142 μs	3.5 kHz	1.8 kHz	0.9 kHz

Cuadro 4.4: Selección de la resistencia RBias para el LS7184.[24]

La Figura 4.27 muestra el funcionamiento del circuito LS7184 en función del tiempo con cambio de sentido.

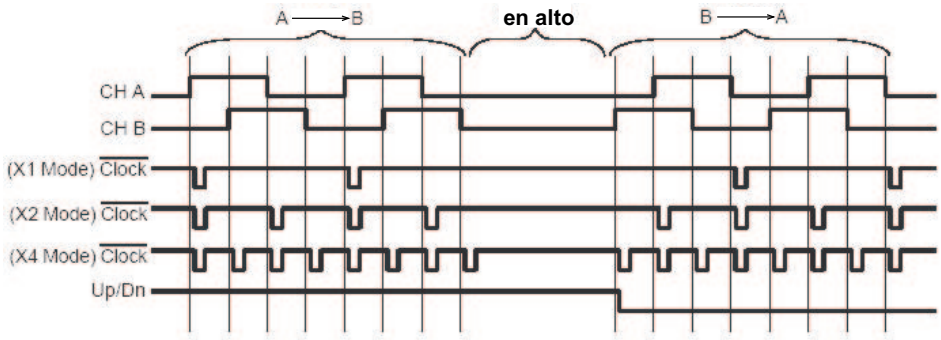

Figura 4.27: Diagrama en función del tiempo del circuito LS7184.[24]

El máximo tiempo de retardo leyendo los canales de entrada A ó B para cualquier salida es 270ns a 3v y 150ns a 5v de alimentación.

4.5.1. Microcontroladores

Los microcontroladores están conquistando el mundo. Están presentes en nuestro trabajo, en nuestra casa y en nuestra vida, en general. Se pueden encontrar controlando el funcionamiento de los ratones y teclados de las computadoras, en los teléfonos, hornos de microondas y los televisores de nuestro hogar. Pero la invación acaba de comenzar y el nacimiento del siglo XXI será testigo de la conquista masiva de estos diminutos computadores, que gobernarán la mayor parte de aparatos que fabricamos y usamos los humanos.

Las extensas áreas de aplicación de los microcontroladores, que se pueden considerar ilimitadas, exigirán un gigantesco trabajo de diseño y fabricación.

Aprender a manejar y aplicar microcontroladores sólo se consigue desarrollando prácticamente diseños reales. Sucede lo mismo que con cualquier intrumento musical, cualquier deporte y muchas actividades.

El microcontrolador es un circuito integrado programable que contiene todos los componentes de un computador. Se emplea para controlar el funcionamiento de una tarea determinada y, debido a su reducido tamaño, suele ir incorporado en el propio dispositivo al que gobierna. Esta última característica es la que le confiere la denominación de «controlador incrustado»(embedded controller). El microcontrolador es un computador dedicado. En su memoria solo reside un programa destinado a gobernar una aplicación determinda; sus líneas de entrada/salida soportan el conexionado de los sensores y actuadores del dispositivo a controlar y todos los recursos complementarios disponibles tienen como única finalidad atender los requerimientos. Una vez programado y configurado el microcontrolador solamente sirve para gobernar la tarea asignada.

El microcontrolador es un computador completo, aunque de limitadas prestaciones, que está contenido en el chip de un circuito integrado y se destina a gobernar una sola tarea.

La industria informática acapara gran parte de los mircrocontroladores que se fabrican. Casi todos los periféricos del computador, desde el ratón o el teclado hasta la impresora, son regulados por el programa de un microntrolador.

Los electrodomésticos de la línea blanca (lavadoras, hornos, lavavajillas, etc.) y la línea de marrón (televisores, vídeos, aparatos musicales, etc.) incorporan numerosos microcontroladores. Igualmente, los sistemas de supervisión, vigilancia y alarma en los edificios utilizan estos chips para optimizar el rendimiento de ascensores, calefacción, aire acondicionado, alarmas de incendio, robo, etc.

Las comunicaciones y sus sistemas de transferencia de información utilizan profusamente estos pequeños computadores incorporándolos en los grandes automatismos y en los modernos teléfonos.

La instrumentación y la electromedicina son dos campos idóneos para la implementación de

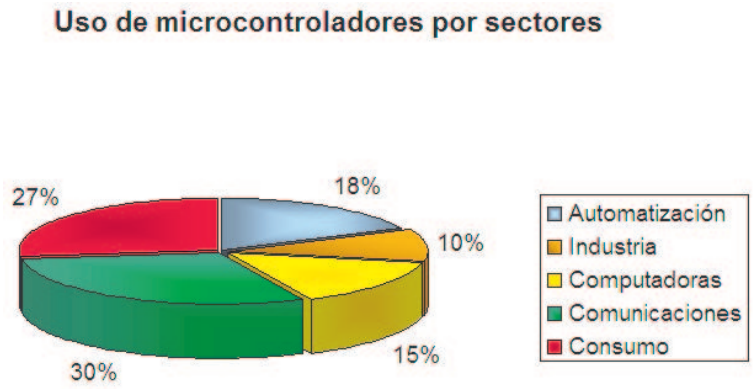

Figura 4.28: Gráfico que muestra la distribución de la producción mundial de microcontroladores en las diversas áreas de aplicación.[2]

estos circuitos integrados. Una importante industria consumidora de microcontroladores es la de automatización, que los aplica en el control de aspectos tan populares como la climatización, la seguridad y los frenos ABS.

Las comunicaciones y los productos de consumo general absorben más de la mitad de la producción de microcontroladores. El resto se distribuye entre el sector de la automatización, los computadores y la industria.

Diferencia entre microprocesador y microcontrolador

El microprocesador es un ciruito integrado que contiene la Unidad Central de Proceso (UCP), también llamada procesador, de un computador. La UCP está formada por la Unidad de Control, que interpreta las instrucciones, y el Camino de Datos, que las ejecuta.

Los pines de un microprocesador sacan al exterior las líneas de sus buses de direcciones, datos y control, para permitir conectarle con la Memoria y los Módulos de E/S y configurar una computadora implementada por varios circuitos integrados. Se dice que un microprocesador es un **sistema abierto** por que su configuración es variable de acuerdo con la aplicación a la que se destine.

Un microprocesador es un **sistema abierto** con el que puede contruirse una computadora con las características que se desee, acoplándole los módulos necesarios.

Un microcontrolador es un **sistema cerrado** que contiene una computadora completa y de prestaciones limitadas que no se pueden modificar.

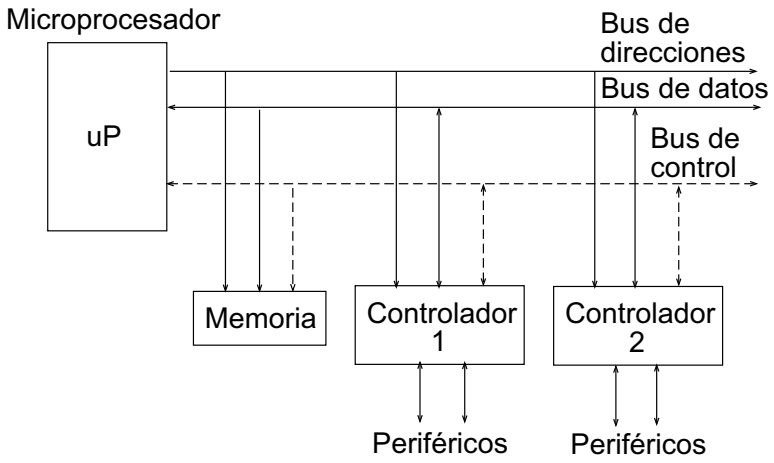

Figura 4.29: Estructura de un sistema abierto basado en un microprocesador.[2]

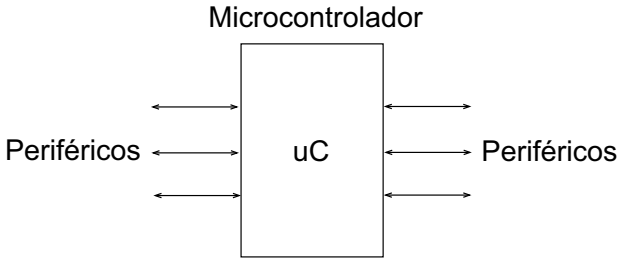

Figura 4.30: Estructura de un sistema cerrado de un microcontrolador.[2]

Arquitectura interna

Las partes principales de un microcontrolador son:

1. Procesador.
2. Memoria volátil para contener el programa.
3. Memoria de lectura y escritura para guardar los datos.
4. Líneas de E/S para los controladores de periféricos:
    a) Comunicación en paralelo.
    b) Comunicación serie.
    c) Diversas puertas de comunicación (bus I^2C, USB, RF, etc.).
5. Recursos auxiliares:
    a) Circuito de reloj.

b) Temporizadores.

c) Perro guardián («watchdog»).

d) Conversores de AD y DA.

e) Comparadores analógicos.

f) Protección ante fallos de alimentación.

g) Estado de reposo o de bajo consumo.

A continuación se pasa revista a las características más representativas de cada uno de los componentes del microcontrolador.

El procesador

La necesidad de conseguir elevados rendimientos en el procesamiento de las instrucciones ha desembocado en el empleo generalizado de procesadores de arquitetura **Harvard** frente a los tradicionales que seguían la arquitectura de **Von Neumann**. Esta última se caracterizaba porque la UCP se conectaba con una memoria única, donde coexistían datos e instrucciones, a través de un sistema de buses.

En la arquitectura **Harvard** son independientes la memoria de instrucciones y la memoria de datos y cada una dispone de su propio sistema de buses para el acceso. Esta dualidad, además de propiciar el paralelismo, permite la adecuación del tamaño de las palabras y los buses a los requerimientos específicos de las instrucciones y de los datos. También la capacidad de cada memoria es diferente.

El procesador de los modernos microcontroladores responde a la arquitectura RISC (Computadores de Juego de Instrucciones Reducido), que se identificó por poseer un repertorio de instrucciones máquina pequeño y simple, de forma que la mayor parte de las instrucciones se ejecuta en un ciclo de instrucción.

Otra aportación frecuente que aumenta el rendimiento del computador es el fomento del paralelismo implícito, que consiste en la segmentación del procesador (pipe-line), descomponiéndolo en etapas para poder procesar una instrucción diferente a cada una de ellas y trabajar con varias a la vez.

El alto rendimiento y elevada velocidad que alcanzan los modernos procesadores, como el que poseen los microcontroladores PIC, se debe a la conjunción de tres técnicas:

- Arquitectura Harvard
- Arquitectura RISC
- Segmentación

Memoria de programa

El microcontrolador esta diseñado para que en su memoria de programa se almacenen todas las instrucciones del programa de control. No hay posibilidad de utilizar memorias externas de ampliación.

Como el programa siempre es el mismo, debe de estar grabado de forma permanente. Los tipos de memoria adecuados para soportar esta función admiten cinco versiones diferentes:

1. ROM con máscara

   En este tipo de memoria el programa se graba en el chip durante el proceso de su fabricación mediante el uso de «máscaras». Los altos costes de diseño e instrumental sólo aconsejan usar este tipo de memoria cuando se precisan series muy grandes.[2]

2. EPROM

   La grabación de este tipo de memoria se reliza mediante un dispositivo físico gobernado desde la computadora personal, que recibe el nombre de grabador.[2]

3. OTP (programable una vez)

   Este modelo de memoria sólo se puede grabar una vez por parte del usuario, utilizando el mismo procedimiento que con la memoria EPROM. Posteriormente no se puede borrar.[2]

4. EEPROM

   La grabación es similar a las memorias OTP y EPROM, pero el borrado es mucho más sencillo al poderse efectuar de la misma forma que el grabado, o sea, eléctricamente. Por lo que puede ser borrada y programada tantas veces como se quiera.[2]

5. FLASH

   Se trata de una memoria no volátil, de bajo consumo, que se puede escribir y borrar el circuito al igual que las EEPROM, pero suelen disponer de mayor capacidad que estas últimas. El borrado sólo es posible con bloques completos y no se puede realizar sobre posiciones concretas.[2]

Memoria de datos

Los datos que manejan los programas varian continuamente, y esto exige que la memoria que les contiene debe ser de lectura y escritura, por lo que la memoria RAM estática (SRAM) es la más adecuada aunque sea volátil.

Hay microcontroladores que disponen como memoria de datos una de lectura y escritura no volátil, del tipo EEPROM. De esta forma, un corte en el suministro de la alimentación no ocaciona la pérdida de la información, que está disponible al reiniciarse el programa[2].

Por otro lado es necesario controlar la velocidad de los servomotores de cd, una manera de hacerlo es variando el voltaje aplicado mediante modulación por ancho de pulso o PWM. Este método consiste en aplicar un tren de pulsos cuadrados con un período fijo, en el cual se va variando el ancho del pulso. Este método también se puede entender como apagar y encender el motor a una tasa muy rápida de manera que se pueda lograr una velocidad menor. El período de la señal no debe ser muy largo ya que se quiere que este tenga un movimiento continuo y no avance a tirones.

Para el PWM se utilizó un microcontrolador PIC16F873 el cual cuenta como ya se mencionó con CAD (Convertidores de Analógico a Digital) y PWM que nos permitirán interpretar la señal analógica de control proveniente de la tarjeta de adquisición de datos. La Figura 4.31 muestra el diagrama de bloques del circuito requerido para la manipulación de la señal de control.

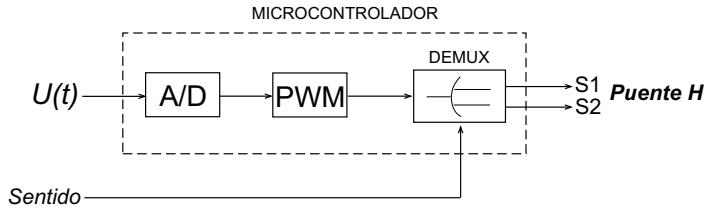

Figura 4.31: Manejo de la señal de control.

Como ya se sabe, el número de articulaciones del robot son dos, por lo tanto a continuación se muestra el programa en lenguaje ensamblador del microcontrolador que se utilizó para convertir las dos señales analógicas de control a modulación por ancho de pulso (S1 y S2).

```
    TITLE 'CAD - PWM'
    LIST P=16F873
    #INCLUDE <P16F873.INC>
    __CONFIG _XT_OSC&_WDT_OFF&_CP_OFF&_DEBUG_OFF&_LVP_OFF&_BODEN_ON&_PWRTE_ON&_WRT_ENABLE_OFF

    ORG 0
    GOTO INICIO

INICIO
        BSF STATUS,RP0      ;Banco 1.
        MOVLW B'00000100'   ;Se configuran como entradas
        MOVWF ADCON1        ;analógias RA0 y RA1.
        MOVLW 0XFF          ;Se fija la
        MOVWF PR2           ;Frecuencia del PWM.
```

```
        CLRF TRISC          ;Puerto C como salidas.
        CLRF TRISB          ;Puerto B como salidas.
        MOVLW B'00001011'   ;RA0, RA1 y RA3 se configuran
        MOVWF TRISA         ;como entradas.
        BCF STATUS,RP0      ;Banco 0.
        MOVLW B'00001111'   ;Se activa
        MOVWF CCP1CON       ;el PWM1
        MOVWF CCP2CON       ;y el PWM2.
        MOVLW B'00000100'   ;Se habilita
        MOVWF T2CON         ;el Timer2.
        CLRF PORTC          ;Se pone en cero el perto C.
        CLRF PORTB          ;Se pone en cero el perto B.
        CLRF PORTA          ;Se pone en cero el perto A.
        CLRF CCPR1L         ;Se pone en cero el PWM1.
        CLRF CCPR2L         ;Se pone en cero el PWM2.
CONV    MOVLW B'01000101'   ;Empieza la conversión
        MOVWF ADCON0        ;para RA0.
CPWM1   BTFSC ADCON0,GO     ;Pregunta si ya se realizó la
        GOTO CPWM1          ;conversión y salta si ya.
        MOVF ADRESH,W       ;Manda la conversión a W.
        MOVWF CCPR1L        ;Pasa el dato W al PWM1.
        MOVLW B'01001101'   ;Empieza la conversión
        MOVWF ADCON0        ;para RA1
CPWM2   BTFSC ADCON0,GO     ;Pregunta si ya se realizó la
        GOTO CPWM2          ;conversión y salta si ya.
        MOVF ADRESH,W       ;Manda la conversión a W.
        MOVWF CCPR2L        ;Pasa el dato W al PWM2.
        GOTO CONV           ;Regreza a la conversión RA0
        END                 ;Finaliza el programa
```

4.5.2. Convertidores de digital a analógico

Es un dispositivo para convertir los datos digitales en señales de corriente o de tensión analógica. También conocidos como CDA (Convertidor de Digital a Analógico) se utilizan profusamente en los reproductores de discos compactos (CD), en los reproductores de sonido y de cintas de vídeo digitales, y en los equipos de procesamiento de señales digitales de sonido y de vídeo. La mayoría de los DAC utilizan alguna forma de red reostática. Los datos digitales se aplican a los reóstatos en grupos de bits. Las resistencias varían en proporciones definidas; el flujo de corriente de cada uno está directamente relacionado con el valor binario del bit recibido.

El dispositivo LS7184 nos proporciona una resolución de 4 veces la resolución del codificador óptico de nuestro servomotor. Teniendo en cuenta que la resolución de nuestro codificador óptico, que se describe más adelante, es de 9 bits ó 512 ppr (pulsos por revolución), la resolución máxima proporcionada del LS7184 es de 11 bits ó 2048 ppr. Esta resolución obtenida lógicamente es por cada una de las articulaciones del robot planar, esto es, dos salidas digitales con una presición de 11 bits. Debido a esto, es necesario convertir de digital a analógico ya que la tarjeta de adquisición de datos P701E únicamente

posee un puerto de E/S de 8 bits.

El circuito integrado que se utilizó fue el MCP4921 el cual es un DAC con un desempeño de alta presición y bajo ruido utilizado para aplicaciones industriales en donde suele ser requerido para calibración o compensación de señales. Este dispositivo tiene como características importantes las siguientes:

. Resolución de 12 bits

. Interfaz SPI hasta 20MHz

. Ganancia de 1 ó 2x

. Voltaje de referencia externo

. Rango de temperatura: -40° C a +125° C

La siguiente figura muestra el diagrama a bloques del MCP492x

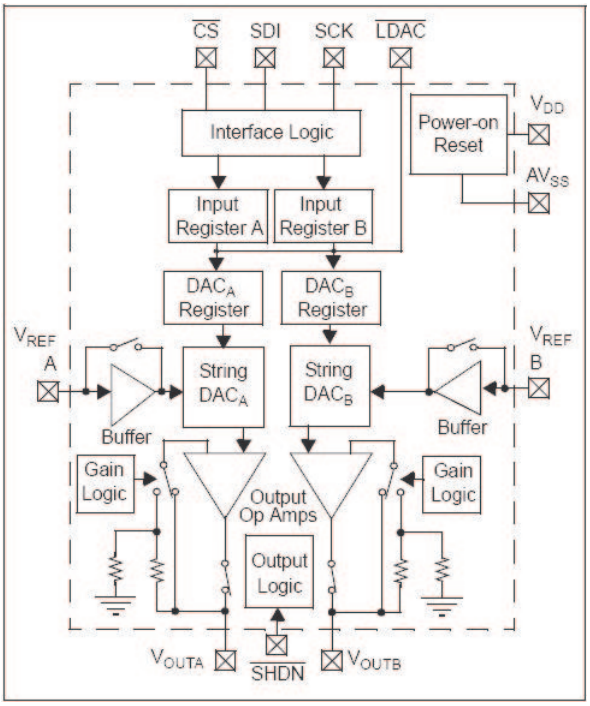

Figura 4.32: Diagrama a bloques del circuito integrado MCP492x.[23]

La familia MCP492x fue designada directamente para la interfaz periférica serial (SPI), que muchos de los microcontroladores soportan, como es el caso del PIC16F873A que es el que se utilizará para su manejo. La escritura consiste de 16 bits y es usada para configurar el control del DAC y el envio de datos, cualquier bit enviado de más, será ignorado. La Figura 4.33 muestra el envio de datos de escritura.

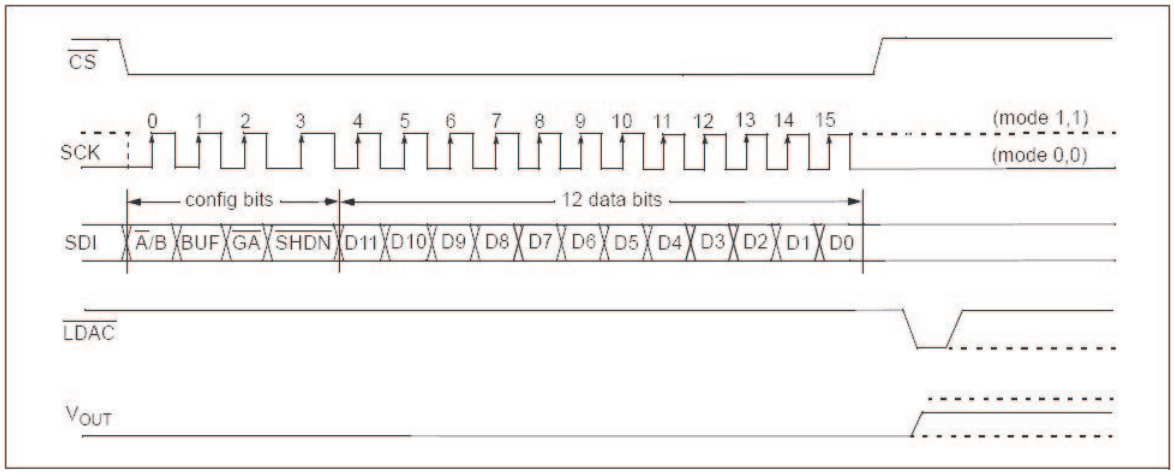

Figura 4.33: Escritura del MCP492x.[23]

Módulo master synchronous serial port

El módulo Master Synchronous Serial Port (MSSP) es una interfaz serial útil para la comunicación con otros dispositivos periféricos o microcontroladores. Estos dispositivos periféricos seriales pueden ser EEPROMs, display drivers, convertidores A/D, etc. El módulo MSSP puede operar de dos modos:

. Serial Peripheral Interface (SPI)

. Inter-Integrated Circuit (I^2C)

Como ya se mencionó antes, los CDA MCP4921 poseen la interfaz SPI en el cual se enfoca este tema. El diagrama de bloques siguiente muestra el modo SPI del microcontrolador PIC16F873A.

El modo SPI permite datos de 8 bits para ser síncronamente transmitidos y recibidos simultáneamente. El microcontrolador que se utilizará soporta los 4 modos SPI. Para lograr la comunicación, típicamente se usan 3 pines:

. Serial Data Out (SDO)

. Serial Data In (SDI)

. Serial Clock (SCK)

Adicionalmente, un cuarto pin puede usarse cuando esta en opración el modo esclavo:

. Slave Select ($\overline{SS}$)

Al inicializar el SPI, varias opciones necesitan ser especificadas. Esto se hace programando los bits de control apropiados (SSPCON$< 5 : 0 >$ and SSPSTAT$< 7 : 6 >$)[23]. Estos bits de control permiten especificar lo siguiente:

. Master Mode (SCK es el Clock de salida)

. Slave Mode (SCK es el Clock de entrada)

. Clock Polarity (estado inactivo de SCK)

. Data input sample phase (tiempo de salida de datos)

. Clock edge (flanco de subida o bajada)

. Clock rate (solo Master Mode)

. Slave Select Mode (solo Slave Mode)

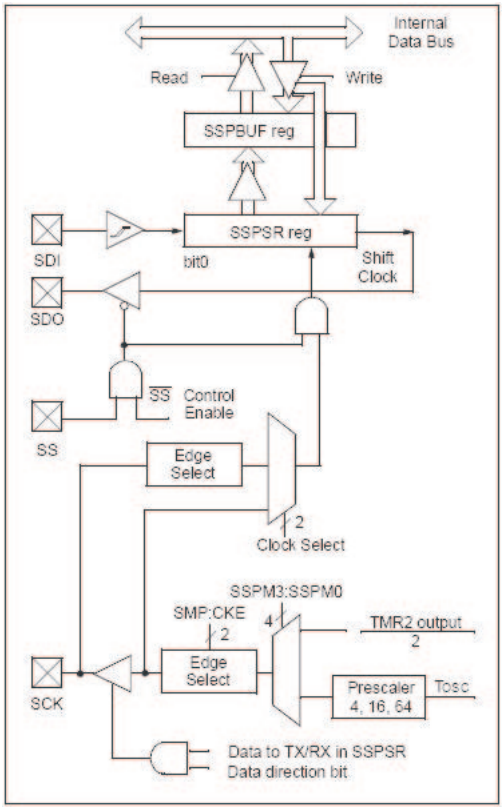

Figura 4.34: Muestra el diagrama de bloques del modo SPI.[23]

En este caso se utilizará el microcontrolador en modo maestro y el CDA será el esclavo. El maestro puede iniciar la transferencia de datos muchas veces, ya que este controla el SCK. La Figura 4.35 muestra la transmisión de datos en modo maestro.

El programa en ensamblador para el microcontrolador PIC16F873A que se utiliza para la comunicación con el dispositivo MCP4921 es el siguiente:

```
;*****************************************************************;
;   CODIGO PARA MANEJAR UN CONVERTIDOR DIGITAL - ANALOGICO        ;
;   DE 12BITS SERIAL, CON INTERFAZ SPI MANIPULADO CON UN          ;
;   PIC16F873A.                                                   ;
;                SILVIONEL VITE MEDÉCIGO                          ;
;                     25/07/2005                                  ;
;*****************************************************************;
;*****************************************************************;
;            CONFIGURACION MCP492X POR HARDWARE                   ;
;                                                                 ;
;   A/B  PORTC,7     SELECION DAC_A=0 O DAC_B=1                   ;
;   BUF  PORTC,6     BUFFER DEL Vref, BUFFERED=1 O UNBUFFERED=0   ;
;   GA   PORTA,5     GANANCIA 1x=1 O 2x=0                         ;
;   SHDN PORTA,4     SHUTDOWN, OUTPUT ENABLED=1 O DISABLED=0      ;
;*****************************************************************;

    TITLE 'Manejo del C.I. MCP492x con SPI, DAC 12 bits'
    LIST P=16F873A
    #INCLUDE <P16F873A.INC>
    __CONFIG _XT_OSC&_WDT_OFF&_CP_OFF&_DEBUG_OFF&_LVP_OFF&_BODEN_ON&_PWRTE_ON&_CP_OFF

    ORG 0X00
    GOTO INICIO

    CS EQU 2            ;CS = PORTC,2

INICIO
    BSF STATUS,RP0      ;CAMBIO BANCO 1.
    MOVLW B'11000000'   ;PUERTO C, BITS 7 y 6 COMO ENTRADAS
    MOVWF TRISC         ;Y LAS DEMAS SALIDAS.
    MOVLW 0X06          ;PUERTO A, SE CONFIGURA COMO
    MOVWF ADCON1        ;E/S DIGITALES.
    MOVLW B'11111111'   ;PUERTOS A Y B
    MOVWF TRISB         ;COMO ENTRADAS
    MOVWF TRISA         ;DIGITALES.
    CLRF SSPSTAT        ;SE BORRA REGISTRO SSPSTAT CONVENIENTEMENTE.
    BCF STATUS,RP0      ;CAMBIO BANCO 0.
    MOVLW B'00110000'   ;Fosc/4, SE HABILITAN SCK,
    MOVWF SSPCON        ;SDO, SDI Y SS.
    CLRF PORTC          ;SE BORRA PUERTO C.
    BSF PORTC,CS        ;CS=1, NO HAY COMUNICACION SPI
PROGRA                  ;
    MOVF PORTC,W        ;W = PUERTO C.
    ANDLW B'11000000'   ;DE W SE DISCRIMINAN LOS BITS 0..5.
    ADDWF PORTA,W       ;W = W + PUERTO A.
    BSF PORTC,CS        ;CS=1, TERMINAN DE ENVIARSE LOS 15 BITS
```

```
BCF PORTC,CS        ;CS=0, EMPIEZA LA COM SPI.
MOVWF SSPBUF        ;SSPBUF = W, SE ENVIA W POR SPI.
NOP                 ;RETARDO
NOP                 ;RETARDO
NOP                 ;RETARDO
NOP                 ;RETARDO
NOP                 ;RETARDO
NOP                 ;RETARDO
NOP                 ;RETARDO
MOVF PORTB,W        ;W = PUERTO B
MOVWF SSPBUF        ;SSPBUF = W, SE ENVIA W POR SPI.
NOP                 ;RETARDO
NOP                 ;RETARDO
NOP                 ;RETARDO
GOTO PROGRA         ;REGRESA A LA ETIQUETA PROGRA
                    ;
END                 ;FIN CODIGO
```

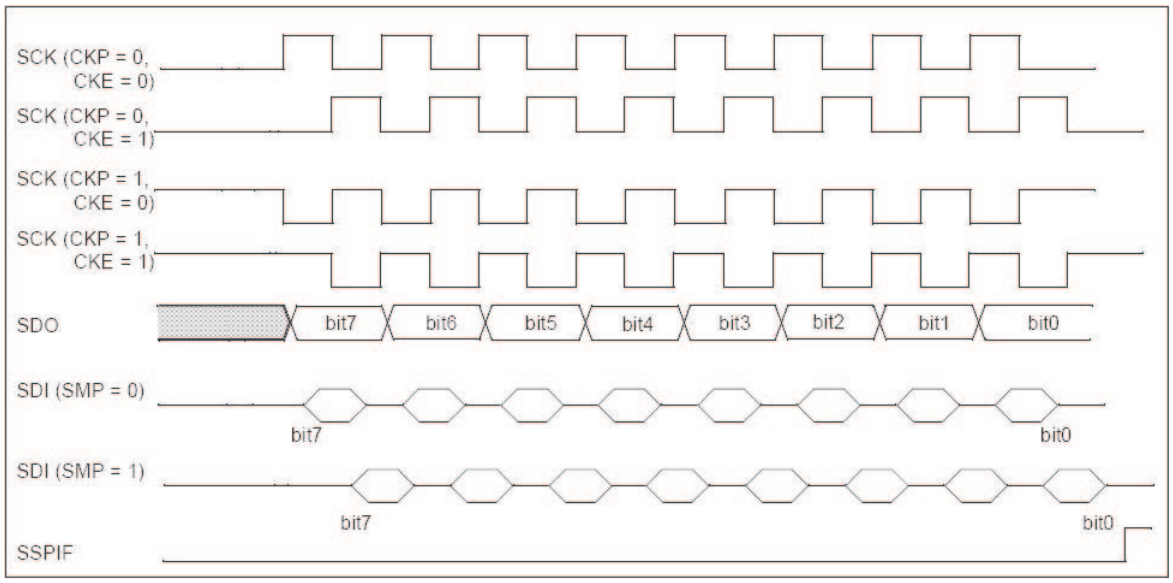

Figura 4.35: Transferencia de datos en modo maestro.[23]

Para que no se tenga problemas con retardos en cuanto a la visualización virtual, se trabaja con la mayor frecuencia que nos proporciona el microcontrolador para el modo SPI, esto es un cuarto de la frecuencia del cristal de cuarzo con el que trabajemos, que en este caso es de 4 MHz, por lo tanto asi como cada intrucción en el programa de un ciclo de reloj es de 1 μs, de igual forma cada pulso de reloj de SCK es de 1 μs.

La Figura 4.36 muestra la comunicación real del microcontrolador con el dispositivo MCP4921 obtenida mediante un osciloscopio.

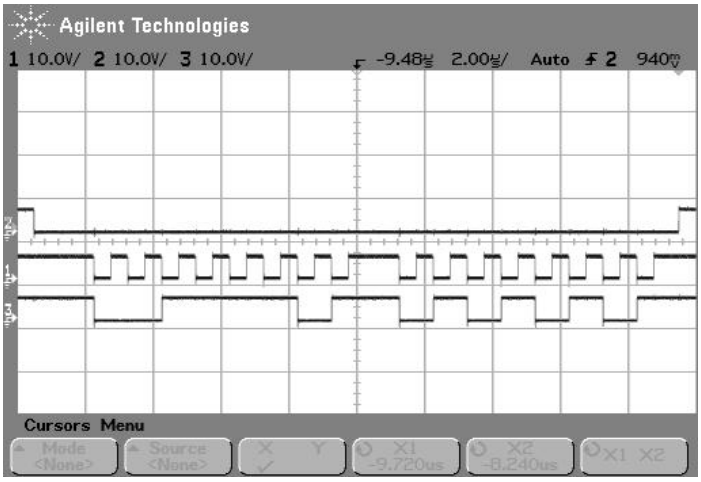

Figura 4.36: Trnasmisión SPI real entre el PIC16F873 y el MCP4921.

4.6. Modelo matemático

4.6.1. Modelo cinemático del robot R2SG

Modelo Cinemático Directo de Posición (MCDP)

El MCDP del robot R2SG como ya se mencionó antes, es la relación que permite determinar las coordenadas operacionales del robot en función de la configuración q del mismo, queda de la siguiente forma:

$$x = L_1 \cos(q_1) + L_2 \cos(q_1 + q_2) \tag{4.2}$$

$$y = L_1 \sin(q_1) + L_2 \sin(q_1 + q_2) \tag{4.3}$$

en donde $L_1 = 0{,}17m$ y $L_2 = 0{,}15m$ que son las longitudes de los eslabones y las variables articulares q_1 y q_2 representan la posición angular.

Modelo Cinemático Inverso de Posición (MCIP)

Los métodos de solución del modelo inverso de posición se clasifícan en numéricos y analíticos. Los métodos numéricos tienen la desventaja de que en caso de existencia de más de una solución el usuario no tiene la posibilidad de controlar la solución a la cual converge el método. Sin embargo los analíticos permiten resolver la mayoria de las arquitecturas de los robots industriales utilizados en práctica, aun no siendo de validez general. Dada la sencillez de la configuración del robot 2RSG en la que sus configuraciones se desenvuelven en un plano, se utiliza un método trigonométrico en base al esquema siguiente:

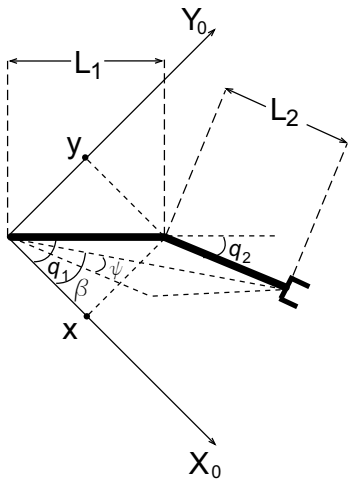

Figura 4.37: Diagrama base para la resolución de la tarea inversa del robot.

Analizando el triángulo oblicuángulo y aplicando la ley de cosenos se obtiene el ángulo q_2

$$X^2 + Y^2 = L_1^2 + L_2^2 - 2L_1L_2\cos(180^\circ + q_2) \tag{4.4}$$

despejando q_2 de (4.4) y aplicando la identidad $cos(180^\circ + q_2) = -cos(q_2)$ obtenemos

$$q_2 = \cos^{-1}\left[\frac{X^2 + Y^2 - L_1^2 - L_2^2}{2L_1L_2}\right] \tag{4.5}$$

El cálculo de ψ utilizando el triángulo obtuso inferior de la Figura 4.37 de igual forma con la ley de cosenos obtenemos

$$L_2^2 = X^2 + Y^2 + L_1^2 - 2L_1\sqrt{X^2 + Y^2}\cos(\psi) \tag{4.6}$$

despejando ψ de (4.6) nos queda

$$\psi = \cos^{-1}\left[\frac{X^2 + Y^2 + L_1^2 - L_2^2}{2L_1\sqrt{X^2 + Y^2}}\right] \tag{4.7}$$

y el cálculo de β se obtiene a partir del triángulo rectángulo que se genera en la Figura 4.37 teniendo como catetos a X y Y

$$\beta = atan2(X, Y) = tg^{-1}\left(\frac{Y}{X}\right) \tag{4.8}$$

Ya calculados β y ψ, se obtiene q_1

$$q_1 = \beta - \psi \tag{4.9}$$

Cabe aclarar que con este método existe una solución para q_1 y q_2 ya que para alcanzar cualquier punto dentro del espacio de trabajo del robot R2SG, existen dos posibles soluciones como se muestra en la Figura 4.38.

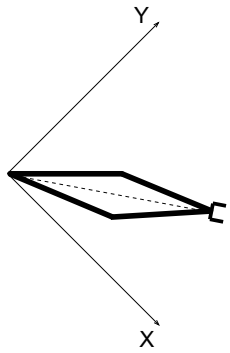

Figura 4.38: Posibles soluciones para llegar a un punto con el robot planar.

Modelo Cinemático Directo e Inverso de Velocidad (MCDV y MCIV)

Teniendo la ecuación (2.12) que describe el MCDV, para el robot R2SG queda de la siguinte manera:

$$\begin{aligned} \dot{x} &= J\dot{q} \\ J &= \begin{bmatrix} -L_1S_1 - L_2S_{12} & -L_2S_{12} \\ L_1C_1 + L_2C_{12} & L_2C_{12} \end{bmatrix} \qquad (4.10) \\ \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} &= \begin{bmatrix} -L_1S_1 - L_2S_{12} & -L_2S_{12} \\ L_1C_1 + L_2C_{12} & L_2C_{12} \end{bmatrix} \begin{bmatrix} \dot{q} \\ \dot{q} \end{bmatrix} \qquad (4.11) \end{aligned}$$

al igual que el MCIV:

$$\begin{aligned} \dot{q} &= J^{-1}\dot{x} \\ J^{-1} &= \frac{J_{adj}}{|J|} = \begin{bmatrix} \frac{C_{12}}{L_1(C_1S_{12}-C_{12}S_1)} & \frac{S_{12}}{L_1(C_1S_{12}-C_{12}S_1)} \\ \frac{L_1C_1+L_2C_{12}}{L_1L_2(C_{12}S_1-C_1S_{12})} & \frac{L_1S_1+L_2S_{12}}{L_1L_2(C_{12}S_1-C_1S_{12})} \end{bmatrix} \\ \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} &= \begin{bmatrix} \frac{C_{12}}{L_1(C_1S_{12}-C_{12}S_1)} & \frac{S_{12}}{L_1(C_1S_{12}-C_{12}S_1)} \\ \frac{L_1C_1+L_2C_{12}}{L_1L_2(C_{12}S_1-C_1S_{12})} & \frac{L_1S_1+L_2S_{12}}{L_1L_2(C_{12}S_1-C_1S_{12})} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} \qquad (4.12) \end{aligned}$$

Para la comprobación de la matriz jacobiana tenemos $I = JJ^{-1}$, en donde I es la matriz identidad de 2x2 como se muestra a continuación

$$\begin{bmatrix} -L_1S_1 - L_2S_{12} & -L_2S_{12} \\ L_1C_1 + L_2C_{12} & L_2C_{12} \end{bmatrix} \begin{bmatrix} \frac{C_{12}}{L_1(C_1S_{12}-C_{12}S_1)} & \frac{S_{12}}{L_1(C_1S_{12}-C_{12}S_1)} \\ \frac{L_1C_1+L_2C_{12}}{L_1L_2(C_{12}S_1-C_1S_{12})} & \frac{L_1S_1+L_2S_{12}}{L_1L_2(C_{12}S_1-C_1S_{12})} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.13}$$

4.6.2. Índice de manipulabilidad cinemática

Dentro del espacio de trabajo de cualquier robot se puede alcanzar cualquier punto, pero es posible que en alguno de ellos el robot sufra de esfuerzos mecánicos, es decir, que en ciertos puntos a los que puede llegar el robot, las articulaciones puedan dañarse debido a que el robot aplica demasiada fuerza para alcanzarlos, a estos puntos se les denomina *singulares*.

El *índice de manipulabilidad* (IM) nos ayuda a monitorear cuando el robot se aproxima a esos puntos singulares a los que no se quiere alcanzar evitando que se corrompa alguna de las articulaciones del robot. El IM se describe de la siguiente manera:

$$IM = \sqrt{|J\,J^T|} \tag{4.14}$$

en donde J es la matriz jacobiana descrita en la ecuación (2.12) y J^T su transpuesta. Esta ecuación nos dice que cuando el IM se aproxima a cero es cuando el robot se acerca a los denominados puntos singulares.

Los puntos singulares del robot R2SG que se mustran en la Figura 4.39, estos son los límites de su espacio de trabajo debido a que es únicamente de 2 grados de libertad. Por la medida de los eslabones L_1 y L_2 el diámetro de la circunferencia interior es de 4cm y de la circunferencia superior es de 64cm.

De la ecuación (4.14) que define al IM, para el robot R2SG queda de la manera siguiente:

$$\begin{aligned} J_{11} &= -L_1S_1 - L_2S_{12} \\ J_{12} &= -L_2S_{12} \\ J_{21} &= L_1C_1 + L_2C_{12} \\ J_{22} &= L_2C_{12} \\ \\ J^T &= \begin{bmatrix} J_{11} & J_{21} \\ J_{12} & J_{22} \end{bmatrix} \\ JJ^T &= \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{bmatrix} J_{11} & J_{21} \\ J_{12} & J_{22} \end{bmatrix} \\ &= \begin{bmatrix} J_{11}J_{11} + J_{12}J_{12} & J_{11}J_{21} + J_{12}J_{22} \\ J_{11}J_{21} + J_{12}J_{22} & J_{21}J_{21} + J_{22}J_{22} \end{bmatrix} \\ JJ_{11}^T &= J_{11}J_{11} + J_{12}J_{12} \end{aligned} \tag{4.15}$$

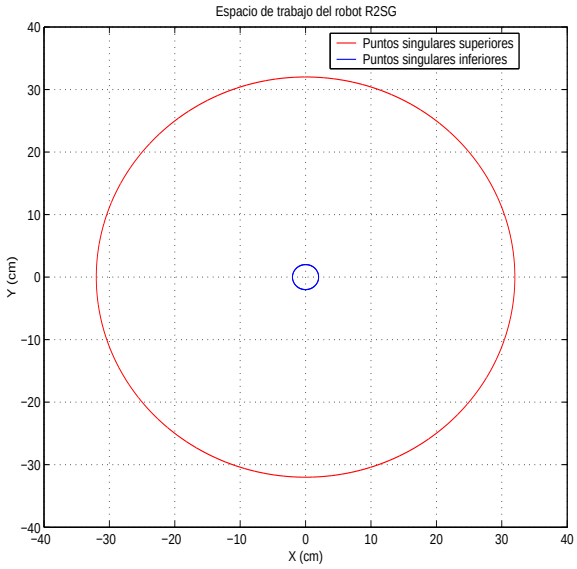

Figura 4.39: Espacio de trabajo del robot R2SG.

$$
\begin{aligned}
JJ_{12}^T &= J_{11}J_{21} + J_{12}J_{22} \\
JJ_{21}^T &= JJ_{12}^T \\
JJ_{22}^T &= J_{21}J_{21} + J_{22}J_{22}
\end{aligned}
$$

$$
\begin{aligned}
\left|JJ^T\right| &= JJ_{11}^T JJ_{22}^T - JJ_{12}^T JJ_{12}^T \\
IM &= \sqrt{JJ_{11}^T JJ_{22}^T - JJ_{12}^T JJ_{12}^T}
\end{aligned} \tag{4.16}
$$

Las Figuras 4.40 y 4.41 muestran el monitoreo del IM del robot R2SG.

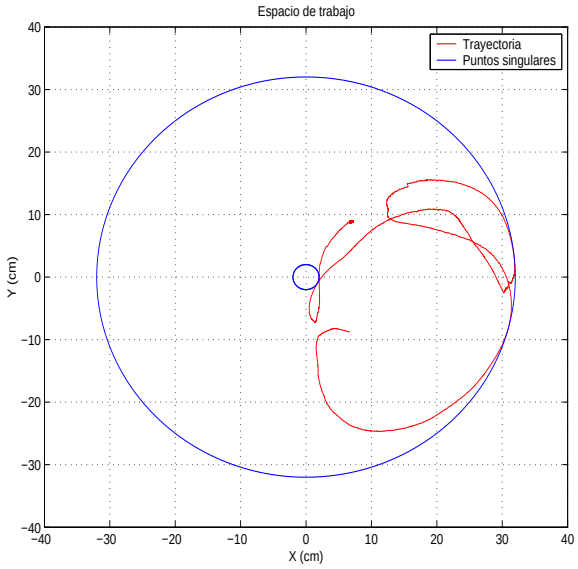

Figura 4.40: Trayectoria del índice de manipulabilidad en el espacio de trabajo.

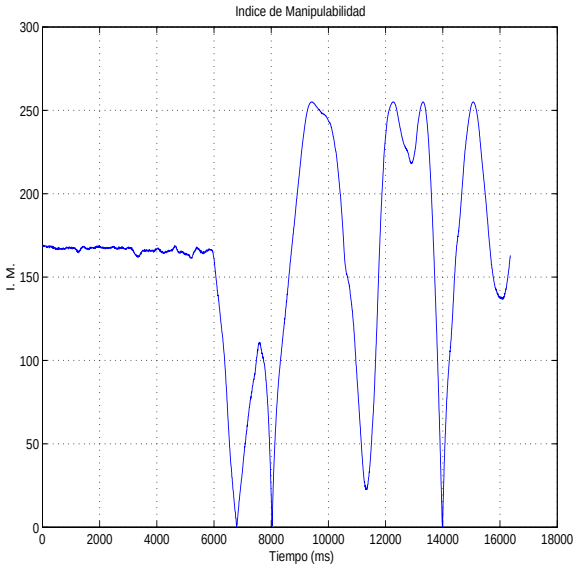

Figura 4.41: Índice de manipulabilidad cinemática del robot R2SG.

4.6.3. Modelo dinámico del robot R2SG

El comportamiento dinámico de un mecanismo de eslabones articulados es descrito en términos de la razón de cambio de la configuración del dispositivo en relación a los pares ejercidos por los eslabones.

El uso del método de Euler-Lagrange, para la obtención del modelo dinámico del robot R2SG es simple y sistemático, proporciona las ecuaciones de estado en forma explícita y estas pueden ser utilizadas para analizar y diseñar estrategias de control. A partir de la ecuación (2.17) del capítulo 2 tenemos que:

$$\tau_i = \frac{d}{dt}\frac{\partial}{\partial \dot{q}_i}L - \frac{\partial}{\partial q_i}L + b_i\dot{q}_i + K_i sgn(\dot{q}_i) \qquad i = 1, 2, ..., n$$

en donde n es el número de grados de libertad y que en este caso $n = 2$, $b_i\dot{q}_i$ representa la frición biscosa y $K_i sgn(\dot{q}_i)$ la fricción seca, para τ_1 queda de la siguiente forma

$$\tau_1 = \frac{d}{dt}\frac{\partial}{\partial \dot{q}_1}L - \frac{\partial}{\partial q_1}L + b_1\dot{q}_1 + K_1 sgn(\dot{q}_1) \tag{4.17}$$

y para τ_2

$$\tau_2 = \frac{d}{dt}\frac{\partial}{\partial \dot{q}_2}L - \frac{\partial}{\partial q_2}L + b_2\dot{q}_2 + K_2 sgn(\dot{q}_2) \tag{4.18}$$

El balance de energías (Lagrangiano) representado en la ecuación (2.18) para el robot de R2SG sería

$$L = k_1 + k_2 - P_1 - P_2 \tag{4.19}$$

en donde las energías cinética (k) y potencial (P) estan dadas por la ecuación (2.19) y (2.20), para el robot R2SG

$$\begin{aligned} k_1 &= \frac{1}{2} m_1 V_1^2 && (4.20)\\ k_2 &= \frac{1}{2} m_2 V_2^2 && (4.21)\\ P_1 &= m_1 g h_1 && (4.22)\\ P_2 &= m_2 g h_2 && (4.23) \end{aligned}$$

recordando, m es la masa de cada eslabón en donde $m_1 = 0{,}81kg$ y $m_2 = 0{,}112kg$, V la velocidad, g es la constante de gravedad y h es la altura al centro de gravedad de los eslabones.

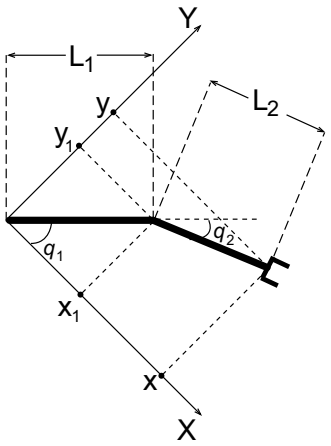

Figura 4.42: Diagrama base para la solución del modelo dinámico del robot R2SG.

Siguiendo la Figura 4.42 y con la ecuación (4.20) se obtiene k_1 en términos de $\dot{q}_1$

$$\begin{aligned} x_1 &= L_1 C_1 && (4.24)\\ y_1 &= L_1 S_1 && (4.25)\\ \dot{x}_1 &= -L_1 S_1 \dot{q}_1 && (4.26)\\ \dot{y}_1 &= L_1 C_1 \dot{q}_1 && (4.27)\\ V_1^2 &= \dot{x}_1^2 + \dot{y}_1^2 && (4.28)\\ &= (-L_1 S_1 \dot{q}_1)^2 + (L_1 C_1 \dot{q}_1)^2 \end{aligned}$$

$$
\begin{aligned}
&= L_1^2 S_1^2 \dot{q}_1^2 + L_1^2 C_1^2 \dot{q}_1^2 \\
&= L_1^2 \dot{q}_1^2 (S_1^2 + C_1^2) \\
&= L_1^2 \dot{q}_1^2 \\
k_1 &= \frac{1}{2} m_1 L_1^2 \dot{q}_1^2 \qquad (4.29)
\end{aligned}
$$

en donde x_1 y y_1 son las coordenadas operacionales del eslabón 1, C_1 es el coseno de q_1, S_1 es el seno de q_1, $\dot{x}_1$ y $\dot{y}_1$ son las velocidades operacionales.

De igual forma se deja a k_2 en términos de $\dot{q}_1$ y $\dot{q}_2$ siguiendo la ecuación (4.21)

$$
\begin{aligned}
x &= L_1 C_1 + L_2 C_{12} \\
y &= L_1 S_1 + L_2 S_{12} \\
\dot{x} &= -L_1 S_1 \dot{q}_1 - L_2 S_{12} (\dot{q}_1 + \dot{q}_2) \qquad (4.30) \\
\dot{y} &= L_1 C_1 \dot{q}_1 + L_2 C_{12} (\dot{q}_1 + \dot{q}_2) \qquad (4.31) \\
V_2^2 &= \dot{x}^2 + \dot{y}^2 \qquad (4.32) \\
&= \{-L_1 S_1 \dot{q}_1 - L_2 S_{12} (\dot{q}_1 + \dot{q}_2)\}^2 \\
&\quad + \{L_1 C_1 \dot{q}_1 + L_2 C_{12} (\dot{q}_1 + \dot{q}_2)\}^2 \\
&= L_1^2 S_1^2 \dot{q}_1^2 + 2 L_1 L_2 S_1 S_{12} \dot{q}_1 (\dot{q}_1 + \dot{q}_2) \\
&\quad + L_2^2 S_{12}^2 (\dot{q}_1 + \dot{q}_2)^2 + L_1^2 C_1^2 \dot{q}_1^2 \\
&\quad + 2 L_1 L_2 C_1 C_{12} \dot{q}_1 (\dot{q}_1 + \dot{q}_2) + L_2^2 C_{12}^2 (\dot{q}_1 + \dot{q}_2)^2 \\
&= L_1^2 \dot{q}_1^2 (S_1^2 + C_1^2) + L_2^2 (\dot{q}_1 + \dot{q}_2)^2 (S_{12}^2 + C_{12}^2) \\
&\quad + 2 L_1 L_2 \dot{q}_1 (\dot{q}_1 + \dot{q}_2) \underbrace{(S_1 S_{12} + C_1 C_{12})} \\
S_1 S_{12} + C_1 C_{12} &= S_1 (S_1 C_2 + S_2 C_1) + C_1 (C_1 C_2 + S_1 S_2) \\
&= S_1^2 C_2 + S_1 S_2 C_1 + C_1^2 C_2 - S_1 S_2 C_1 \\
&= C_2 (S_1^2 + C_1^2) \\
&= C_2 \\
V_2^2 &= L_1^2 \dot{q}_1^2 + L_2^2 (\dot{q}_1 + \dot{q}_2)^2 + 2 L_1 L_2 C_2 \dot{q}_1 (\dot{q}_1 + \dot{q}_2) \\
k_2 &= \frac{1}{2} m_2 \{L_1^2 \dot{q}_1^2 + L_2^2 (\dot{q}_1 + \dot{q}_2)^2 + 2 L_1 L_2 C_2 \dot{q}_1 (\dot{q}_1 + \dot{q}_2)\} \qquad (4.33)
\end{aligned}
$$

x y y son las coordenadas operacionales del efector final del robot R2SG, S_{12} es el seno de $(q_1 + q_2)$ y C_{12} es el coseno de $(q_1 + q_2)$.

De acuerdo con la Figura 4.43 para determinar las energias potenciales (P_1 y P_2) mediante las ecuaciones (4.22) y (4.23), $P_1 = 0$ debido a que $h_1 = 0$ ya que las coordenadas (0,0,0) se encuentran en la primera articulación y no hay altura alguna en relación al centro de gravedad del eslabón 1. Para h_2 la

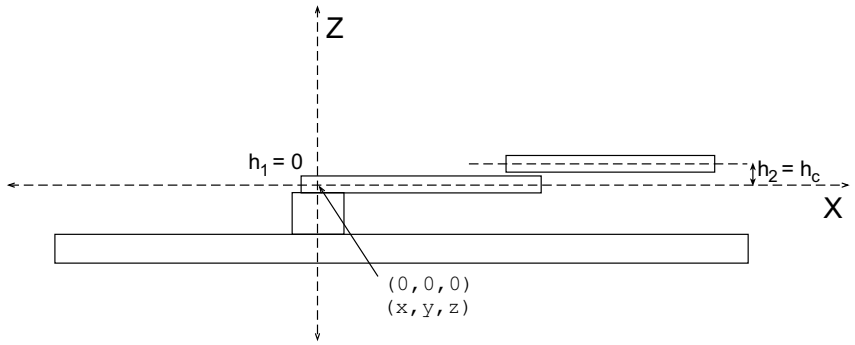

Figura 4.43: Diagrama base para la solución de P_1 y P_2.

altura al centro de gravedad es constante (h_c), debido a esto, puede discriminarse quedando representado en la Figura 4.44, a parte de no tener influencia en el lagrangiano. Las ecuaciones de las energias potenciales para el robot R2SG son las siguientes:

$$\begin{aligned} P_1 &= 0 \\ P_2 &= m_2 g h_c \end{aligned}$$

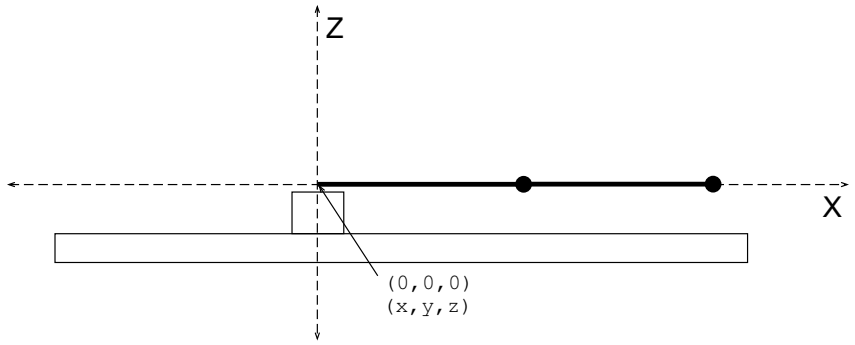

Figura 4.44: Representación del robot sin término de gravedad.

Teniendo tanto las energias cinéticas como potenciales

$$\begin{aligned} L &= k_1 + k_2 - P_2 \\ &= \frac{1}{2} m_1 L_1^2 \dot{q}_1^2 + \frac{1}{2} m_2 \{ L_1^2 \dot{q}_1^2 + L_2^2 (\dot{q}_1 + \dot{q}_2)^2 + 2 L_1 L_2 C_2 \dot{q}_1 (\dot{q}_1 + \dot{q}_2) \} - m_2 g h_c \\ &= \frac{1}{2} (m_1 + m_2) L_1^2 \dot{q}_1^2 + m_2 L_1 L_2 C_2 \dot{q}_1 (\dot{q}_1 + \dot{q}_2) + \frac{1}{2} m_2 L_2^2 (\dot{q}_1 + \dot{q}_2)^2 - m_2 g h_c \\ &= \frac{1}{2} (m_1 + m_2) L_1^2 \dot{q}_1^2 + m_2 L_2 (\dot{q}_1 + \dot{q}_2) \{ L_1 C_2 \dot{q}_1 + \frac{1}{2} L_2 (\dot{q}_1 + \dot{q}_2) \} - m_2 g h_c \end{aligned}$$

Una vez obtenido el Lagrangiano, se logra τ_1 de la ecuación (4.17)

$$\begin{aligned}
\tau_1 &= \frac{d}{dt}\frac{\partial}{\partial \dot{q}_1}L - \frac{\partial}{\partial q_1}L + b_1\dot{q}_1 + K_1 sgn(\dot{q}_1) \\
\frac{\partial}{\partial \dot{q}_1}L &= \frac{\partial}{\partial \dot{q}_1}\left\{\frac{1}{2}(m_1+m_2)L_1^2\dot{q}_1^2 + m_2L_2(\dot{q}_1+\dot{q}_2)\{L_1C_2\dot{q}_1 + \frac{1}{2}L_2(\dot{q}_1+\dot{q}_2)\} - m_2gh_c\right\} \\
&= (m_1+m_2)L_1^2\dot{q}_1 + 2m_2L_1L_2C_2\dot{q}_1 + m_2L_1L_2C_2\dot{q}_2 + m_2L_2^2\dot{q}_1 + m_2L_2^2\dot{q}_2 \\
\frac{d}{dt}\frac{\partial}{\partial \dot{q}_1}L &= (m_1+m_2)L_1^2\ddot{q}_1 + 2m_2L_1L_2[C_2\ddot{q}_1 - S_2\dot{q}_1\dot{q}_2] + m_2L_1L_2[C_2\ddot{q}_2 - S_2\dot{q}_2^2] + m_2L_2^2(\ddot{q}_1+\ddot{q}_2) \\
&= [(m_1+m_2)L_1^2 + 2m_2L_1L_2C_2 + m_2L_2^2]\ddot{q}_1 + [m_2L_1L_2C_2 + m_2L_2^2]\ddot{q}_2 \\
&\quad -[2m_2L_1L_2S_2]\dot{q}_1\dot{q}_2 - [m_2L_1L_2S_2]\dot{q}_2^2 \\
\frac{\partial}{\partial q_1}L &= 0
\end{aligned}$$

$$\begin{aligned}
\tau_1 &= [(m_1+m_2)L_1^2 + 2m_2L_1L_2C_2 + m_2L_2^2]\ddot{q}_1 + [m_2L_1L_2C_2 + m_2L_2^2]\ddot{q}_2 \\
&\quad -[2m_2L_1L_2S_2]\dot{q}_1\dot{q}_2 - [m_2L_1L_2S_2]\dot{q}_2^2 + b_1\dot{q}_1 + K_1\tanh(\beta_1\dot{q}_1)
\end{aligned}$$

para τ_2 a partir de la ecuación (4.18)

$$\begin{aligned}
\tau_2 &= \frac{d}{dt}\frac{\partial}{\partial \dot{q}_2}L - \frac{\partial}{\partial q_2}L + b_2\dot{q}_2 + K_2 sgn(\dot{q}_2) \\
\frac{\partial}{\partial \dot{q}_2}L &= \frac{\partial}{\partial \dot{q}_2}\left\{\frac{1}{2}(m_1+m_2)L_1^2\dot{q}_1^2 + m_2L_2(\dot{q}_1+\dot{q}_2)\{L_1C_2\dot{q}_1 + \frac{1}{2}L_2(\dot{q}_1+\dot{q}_2)\} - m_2gh_c\right\} \\
&= m_2L_1L_2\dot{q}_1 + m_2L_2^2(\dot{q}_1+\dot{q}_2) \\
\frac{d}{dt}\frac{\partial}{\partial \dot{q}_2}L &= m_2L_1L_2[C_2\ddot{q}_1 - S_2\dot{q}_1\dot{q}_2] + m_2L_2^2(\ddot{q}_1+\ddot{q}_2) \\
\frac{\partial}{\partial q_2}L &= -m_2L_1L_2S_2\dot{q}_1(\dot{q}_1+\dot{q}_2)
\end{aligned}$$

$$\tau_2 = [m_2L_1L_2C_2 + m_2L_2^2]\ddot{q}_1 + [m_2L_2^2]\ddot{q}_2 + [m_2L_1L_2S_2]\dot{q}_1^2 + b_2\dot{q}_2 + K_2\tanh(\beta_2\dot{q}_2)$$

La representación del modelo dinámico del robot R2SG sería

$$\tau : M(q)\ddot{q} + C(q,\dot{q})\dot{q} + F(\dot{q}) \tag{4.34}$$

representado en forma matricial

$$\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} b_1\,\dot{q}_1 + k_1\,\mathrm{sgn}(\dot{q}_1) \\ b_2\,\dot{q}_2 + k_2\,\mathrm{sgn}(\dot{q}_2) \end{bmatrix} \tag{4.35}$$

para el robot R2SG

$$\begin{aligned} \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} &= \begin{bmatrix} (m_1+m_2)L_1^2 + 2m_2L_1L_2C_2 + m_2L_2^2 & m_2L_1L_2C_2 + m_2L_2^2 \\ m_2L_1L_2C_2 + m_2L_2^2 & m_2L_2^2 \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} \\ &+ \begin{bmatrix} -2m_2L_1L_2S_2\dot{q}_2 & -m_2L_1L_2S_2\dot{q}_2 \\ m_2L_1L_2S_2\dot{q}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} b_1\,\dot{q}_1 + k_1\tanh(\beta\dot{q}_1) \\ b_2\,\dot{q}_2 + k_2\tanh(\beta\dot{q}_2) \end{bmatrix} \end{aligned} \tag{4.36}$$

4.6.4. Índice de manipulabilidad dinámica

Al igual que el índice de manipulabilidad cinemática, ahora se monitorean las singularidades en base a las propiedades dinámicas del robot R2SG. A continuación se describen las propiedades dinámicas uno y dos (P-I y P-II) del robot R2SG a partir de la ecuación (4.36) en donde:

$$\begin{aligned} M_{11} &= (m_1+m_2)L_1^2 + 2m_2L_1L_2C_2 + m_2L_2^2 \\ M_{12} &= m_2L_1L_2C_2 + m_2L_2^2 \\ M_{21} &= M_{12} \\ M_{22} &= m_2L_2^2 \end{aligned}$$

$$\begin{aligned} C_{11} &= -2m_2L_1L_2S_2\dot{q}_2 \\ C_{12} &= -m_2L_1L_2S_2\dot{q}_2 \\ C_{21} &= m_2L_1L_2S_2\dot{q}_1 \\ C_{22} &= 0 \end{aligned}$$

P-I:

La matriz de inercias es simétrica $M(q) = M(q)^T$ y definida positiva $x^TM(q)x > 0$, en donde $x \in \Re^{nx1}$ y es cualquier vector.

$$\begin{aligned} x &= \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T \\ x^TM(q)x &= \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ &= \begin{bmatrix} x_1M_{11} + x_2M_{21} & x_1M_{12} + x_2M_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ &= x_1^2M_{11} + 2x_1x_2M_{12} + x_2^2M_{22} > 0 \end{aligned} \tag{4.37}$$

P-II:

La matriz de inercias es antisimétrica $x^T \left\{ \dot{M}(q) - 2C(q,\dot{q}) \right\} x \equiv 0$, en donde $x \in \Re^{nx1}$ y es cualquier vector.

$$\begin{aligned}
\dot{M}_{11} &= -2L_1L_2S_2\dot{q}_2 \\
\dot{M}_{12} &= -m_2L_1L_2S_2\dot{q}_2 \\
\dot{M}_{21} &= \dot{M}_{12} \\
\dot{M}_{22} &= 0
\end{aligned}$$

$$\begin{aligned}
\dot{M}(q) - 2C(q,\dot{q}) &= \begin{bmatrix} -2m_2L_1L_2S_2\dot{q}_2 + 4m_2L_1L_2S_2\dot{q}_2 & -m_2L_1L_2S_2\dot{q}_2 + 2m_2L_1L_2S_2\dot{q}_2 \\ -m_2L_1L_2S_2\dot{q}_2 - 2m_2L_1L_2S_2\dot{q}_1 & 0 \end{bmatrix} \\
&= \begin{bmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{bmatrix}
\end{aligned}$$

$$\begin{aligned}
E_{11} &= 2m_2L_1L_2S_2\dot{q}_2 \\
E_{12} &= m_2L_1L_2S_2\dot{q}_2 \\
E_{21} &= -(2\dot{q}_1 + \dot{q}_2)m_2L_1L_2S_2 \\
E_{22} &= 0
\end{aligned}$$

$$\begin{aligned}
x^T \left\{ \dot{M}(q) - 2C(q,\dot{q}) \right\} x &= \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\
&= x_1^2E_{11} + x_1x_2(E_{12} + E_{21})
\end{aligned}$$

$$\begin{aligned}
EE &= E_{12} + E_{21} \\
&= -2m_2L_1L_2S_2\dot{q}_1
\end{aligned}$$

$$x^T \left\{ \dot{M}(q) - 2C(q,\dot{q}) \right\} x = x_1^2E_{11} + x_1x_2EE \equiv 0 \tag{4.38}$$

El índice de manipulabilidad dinámica se monitorea mediante la propidad uno (P-I). Siguiendo con el mismo experimento que muestra la Figura 4.40 para el IM cinemática, en las Figuras 4.45 y 4.46 se comprueban experimentalmente las propiedades dinámicas expuestas.

Cabe mencionar que en el IM dinámica para el robot R2SG a difrencia del IM cinemática, se reconocen los puntos singulares tanto del límite de la circunferencia inferior como los de la circunferencia superior como se muestran en la Figura 4.45, es decir, que en este caso se puede distinguir cuando el robot se acerca a la zona singular inferior manifestándoce el IM aproximadamente a cero y a la zona singular superior como un valor máximo.

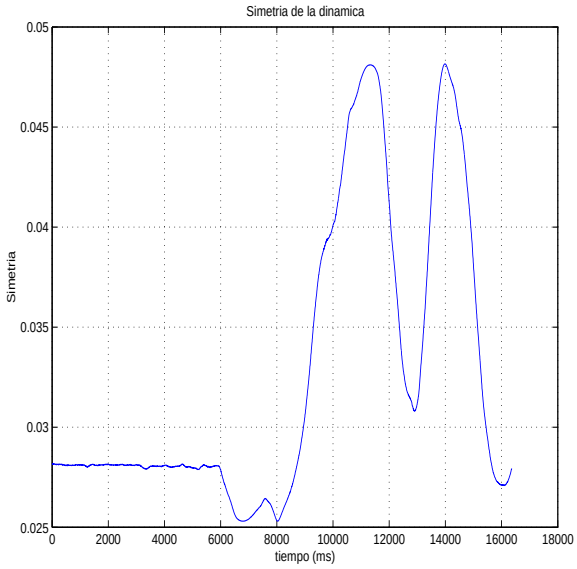

Figura 4.45: Índice de manipulbilidad dinámica del robot R2SG, basado en la propiedad definido positiva.

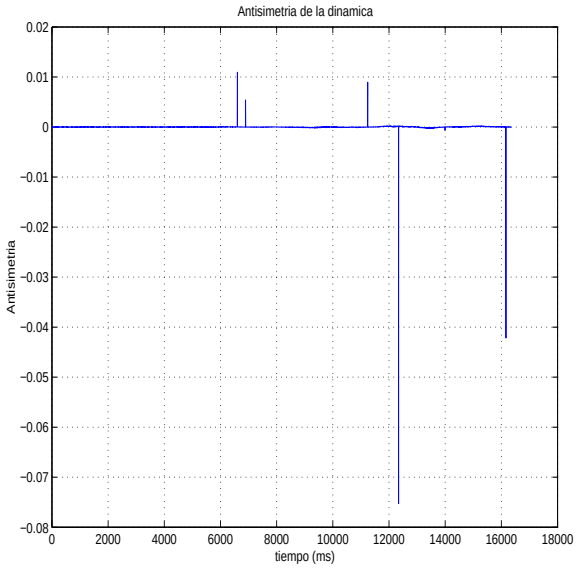

Figura 4.46: Propiedad dinámica de la matriz antisimétrica del robot R2SG.

4.7. Control

Se evaluaron experimentalmente algunas estrategias de control clásica y moderna para regulación a una coordenada operacional, regulación basada en seguimiento a una coordenada operacional y seguimiento de una trayectoria cerrada.

Dentro de las estrategias de control clásicas se evaluó el PD siendo su ley de control la ecuación (3.1) en donde la constante proporcional y derivativa teniendo en cuenta que el coeficiente de amortiguamiento $\xi = 1$ para ser críticamente amortiguado, quedan de la siguiente forma:

$$k_p = Wn^2 \tag{4.39}$$

$$k_d = 2Wn \tag{4.40}$$

Siguiendo con el control PID en donde la ecuación (3.6) describe su ley de control, la constante de integración

$$k_i \ll k_p k_d \tag{4.41}$$

El método de integración que se llevó a cabo para los experimentos es el Runge-Kutta de 4° orden para ecuaciones diferenciales ordinarias. Este método es uno de los mejores y más utilizados el cual describe la siguiente ecuanción

$$Y = (k_1 + 2k_2 + 2k_3 + k_4)\frac{h}{6} \tag{4.42}$$

en donde h es el incremento

$$k_1 = f(x_i) \tag{4.43}$$

$$k_2 = f(x_i + \tfrac{h}{2}) \tag{4.44}$$

$$k_3 = f(x_i + \tfrac{h}{2}) \tag{4.45}$$

$$k_4 = f(x_i + h) \tag{4.46}$$

y x_i es el valor inicial. Debido a que k_2 y k_3 son iguales, la ecuación (4.42) se simplifica a

$$Y = (k_1 + 4k_2 + k_4)\frac{h}{6} \tag{4.47}$$

Para el control PID no lineal de igual forma se integra con el método de Runge-Kutta de la siguiente manera:

$$k_1 = \tanh(\beta x_i)$$

$$k_2 = \tanh(\beta x_i + \beta\tfrac{h}{2})$$

$$k_3 = \tanh(\beta x_i + \beta\tfrac{h}{2})$$

$$k_4 = \tanh(\beta x_i + \beta h)$$

4.7.1. Seguimiento de trayectorias

El *seguimiento de trayectorias* es cuando ahora la posición articular deseada q_d ya no es constante, sino que se encuentra en función del tiempo dada por una ecuación (spline). El uso del seguimiento de

trayectorias se debe a que lo ideal es romper el estado en reposo del robot de una forma lenta para evitar inercia llegando a un punto medio máximo de velocidad y finalizar en la posición deseada lentamente de igual forma.

La ecuación que se tomó es un polinomio de 5° orden que se muestra a continuación:

$$\xi(t) = a_3\frac{(t-t_0)^3}{(t_b-t_0)^3} - a_4\frac{(t-t_0)^4}{(t_b-t_0)^4} + a_5\frac{(t-t_0)^5}{(t_b-t_0)^5} \tag{4.48}$$

siendo su derivada

$$\dot{\xi}(t) = 3a_3\frac{(t-t_0)^2}{(t_b-t_0)^3} - 4a_4\frac{(t-t_0)^3}{(t_b-t_0)^4} + 5a_5\frac{(t-t_0)^4}{(t_b-t_0)^5} \tag{4.49}$$

donde t_0 es el tiempo inicial de la trayectoria y t_b es el tiempo en el que se quiere que llegue a la posición deseada. Las constantes a_3, a_4 y a_5 se determinarán a continuación:

Para

$$\begin{aligned}\xi(t_0) &= 0\\ \xi(t_b) &= 1\\ \dot{\xi}(t_0) &= 0\\ \dot{\xi}(t_b) &= 0\end{aligned}$$

tenemos que

$$\begin{aligned}a_3 - a_4 + a_5 &= 1\\ 3a_3 - 4a_4 + 5a_5 &= 0\end{aligned} \tag{4.50}$$

proponiendo a a_3 resulta una matriz cuadrada de 2x2

$$\left._{a_3=10}\right| \begin{aligned}a_4 &= 15\\ a_5 &= 6\end{aligned} \tag{4.51}$$

donde las ecuaciones quedan de la siguiente forma

$$\xi(t) = 10\frac{t^3}{t_b^3} - 15\frac{t^4}{t_b^4} + 6\frac{t^5}{t_b^5} \tag{4.52}$$

$$\dot{\xi}(t) = 30\frac{t^2}{t_b^3} - 60\frac{t^3}{t_b^4} + 30\frac{t^4}{t_b^5} \tag{4.53}$$

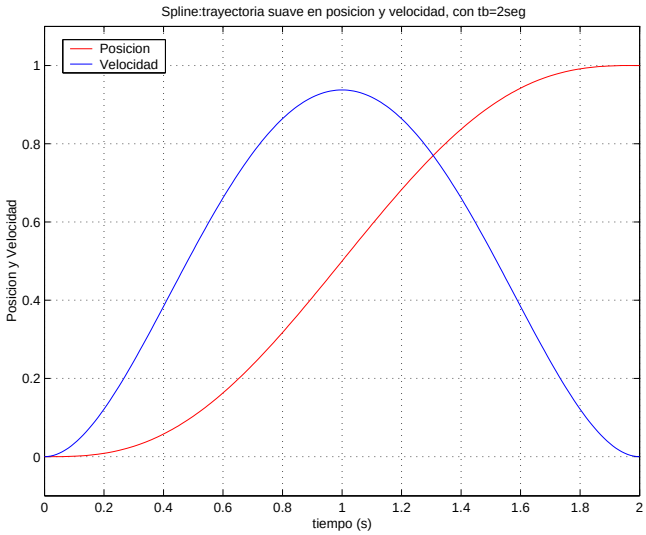

Figura 4.47: Generador de base de tiempo $\xi(t)$ y derivada $\dot{\xi}(t)$ con t_b=2 segundos.

La trayectoria se diseña de tal forma que tenga un desempeño suave de 0 a 1 en un tiempo finito arbitrario $t = t_b > 0$ y $\dot{\xi}(t)$ sea una campana tal que $\dot{\xi}(t_0) = \dot{\xi}(t_b) \equiv 0$ como se muestra en la Figura 4.47.

4.8. Conclusiones

En este capítulo se detalló el diseño y construción del robot R2SG,en el que la parte mecánica fue diseñada y visualizada en tres dimensiones mediante el software Mechanical Desktop 6, perimitiendo auxiliar de manera conveniente en la construcción del prototipo, este fue elaborado en aluminio con rodamientos y compensadores dinámicos de acero, esto con la finalidad de acoplar de manera conveniente a los servomotores seleccionados para el control de movimiento, ambos servomotores son de las mismas características por lo que la interfaz electrónica de potencia y el correspondiente acondicionamiento de señales corresponden al mismo diseño. Debido a ciertas limitaciones de la tarjeta de adquisición de datos y control empleada y su acoplamiento a los codificadores ópticos, fue necesario emplear circuitería especializada para aprovechar la máxima resolución para el control de movimiento. En este mismo capítulo se reportan las síntesis cinemática y dinámica del robot R2SG con los parámetros cinemáticos y dinámicos correspondientes definidos a apartir de pruebas experimentales y mediciones directas, cuya comprobación experimental está definida por las propiedades analíticas y la manipulabilidad en ambos criterios. Para evaluar estrategias de control, se propone la regulación basada en seguimiento, es decir el empleo de splines definidas por polinomios de quinto grado tal que sea posible vencer los efectos inerciales por el estado de reposo y de movimiento del robot, y simultáneamente lograr la convergencia en tiempo finito. Estas contribuciones permiten lograr los estudios experimentales reportados en el capítulo 6.

Capítulo 5

Visualización virtual

5.1. Introducción

El modelado en 3D (tres dimensiones) se ha convertido en uno de los recursos más utilizados en la actualidad en distintas áreas, como es el caso de la realidad virtual, estas dos disciplinas van de la mano. Como su nombre lo dice, el modelado 3D no es otra cosa más que la de crear objetos en 3D, pero en este caso se hará en una computadora, es decir que realmente el objeto no existirá, sino que será un código binario que a través de un software, una tarjeta de video y un monitor podremos observar como es que el código binario dibuja una forma.

Gracias al software, el modelado en 3D es cada vez más sencillo de realizar, y de una forma mas intuitiva, aunque claro, es necesario que se tengan ciertos conocimientos sobre lo que son las gráficas en tercera dimensión y sus propiedades, también son importantes los conocimientos matemáticos relacionados con la tercera dimensión. Aunque actualmente, el modelado con software ha caído en dominio de diseñadores gráficos, es decir, como si de un arte se tratara, ya que son ellos los que mejor modelos presentan al mundo, lo que nos quiere decir que el modelado puede ser desarrollado por distintas personas y que se necesita más imaginación y creatividad que conocimientos técnicos.

Para el desarrollo del sofware que se utilizó en la visualización virtual y control del robot R2SG, se acudió al lenguaje de programación Visual Pascal (Delphi), el cual soporta OpenGL, esto es una librería que sirve de herramienta para la construcción de gráficos tridimencionales.

Para el manejo de la tarjeta de adquisición de datos National Instrument P701E desde Delphi, se instalarón los componentes ActiveX en Delphi, los cuales se pueden adquirir después de instalar la tarjeta de adquisición de datos. ActiveX no es nada realmente nuevo, en esencia llamamos hoy en día ActiveX lo que antes se agrupaba en torno a OLE, OCX y COM, el desarrollo de Microsoft de un stándar binario para la comunicación entre dos componentes de sofware.

5.2. Realidad virtual en robots manipuladores

La rápida curva de crecimiento tanto del software como del hardware que han sufrido las computadoras en los últimos años, se debe en gran medida al perfeccionamiento de las interfaces hombre-máquina, las cuales cada vez son más "amigables"para la interacción humana, lo cual a permitido obtener una considerable popularización de estas máquinas dentro del ambiente informático operativo. Es así que la realidad virtual (RV) se considera como el nuevo modo de interacción entre el hombre y las computadoras, con el objetivo de simular la presencia de entornos reales en mundos virtuales abstractos y sintetizados. Con lo cual se dice que el objetivo primordial de la realidad virtual es lograr que se lleve a cabo la unión Hombre-Máquina de una manera más estrecha.

La realidad virtual podemos concebirla como todo aquel tipo de simulación lograda a partir de un conjunto de datos concretos y complejos tomados del mundo real e integrados tridimensionalmente mediante una computadora, para conformar modelos gráficos que interactuen en tiempo real con el usuario, y donde tal interacción se efectúe a través de sofisticados dispositivos que incorporan información multisensorial al modelo, para alimentar no solo la visión del ser humano sino también su tacto y oído, e inclusive eventualmente hasta el gusto y el olfato.

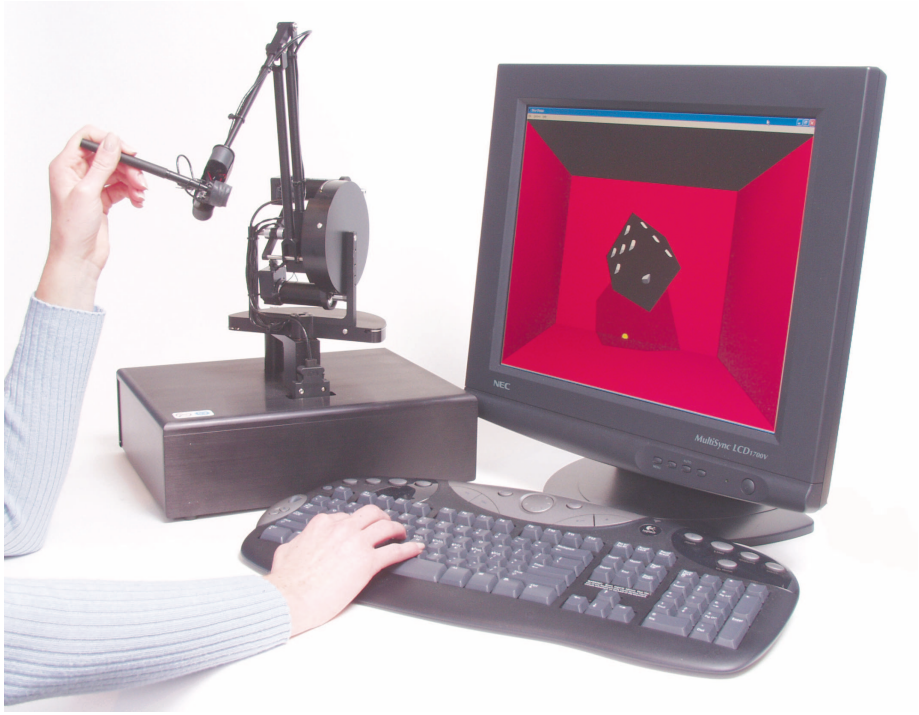

Figura 5.1: Realidad virtual con el robot PHANToM.

La realidad virtual la podemos comprender de una forma aun más genérica como aquella experiencia que nos sumerge a un mundo virtual generada por una computadora, haciendo uso de tecnología capaz de recrear mundos reales en entornos virtuales sintetizados tan realista que el usuario crea que realmente se encuentra ahí.

Los sistemas de realidad virtual estan compuesto de varios sistemas como es el hardware, software y electrónica; estos sistemas independientes se han desarrollado para producir efectos visuales, auditivos y táctiles que son utilizados en entornos virtuales, así como también refuerza un aspecto de la ilusión del usuario durante su inmersión en el mundo virtual.

Un sistema de realidad virtual como cualquier otro tipo de sistema, se encuentra constituido por una serie de patrones y características que lo hacen ser particularmente distinto a otros. La realidad virtual básicamente la caracterizan tres elementos esenciales, que es inmersiva, interactiva y en tiempo real.

En primer lugar se dice que es *inmersiva* puesto que trata de utilizar la mayor cantidad de los sentidos para crear la sensación de inmersión coloca así al usuario en el interior de un mundo creado por la computadora donde éste se puede mover ahí, ver hacia arriba, abajo, derecha o izquierda como en el mundo real. También es *interactiva*, lo que le permite al usuario manipular los objetos del universo virtual e interactuar con sus habitantes. Y a diferencia de otras tecnologías como la animación, la realidad virtual es en *tiempo real*, esto significa que lo que pasa se calcula exactamente en ese momento. permitiendo una libertad realista que no puede dar la animación.

La aplicación de técnicas de realidad virtual que se han venido generando en diferentes áreas de investigación puede ser concebidas como un proceso de exploración de descubrimientos, de observación y de construcción de nuestra visión del conocimiento (siendo la actividad humana por preferencia).

Dentro de este marco la realidad virtual nos permite aprender sirviendo como una herramienta más para la ejecución de nuestras tareas en diferentes áreas de aplicación como: en hospitales, facultades de arquitectura, centros de tecnología educativa, laboratorios virtuales de la NASA, laboratorios de física virtual, etc.

5.3. Integración de un mundo virtual activo del robot R2SG

Desde el punto de vista del programador, las aplicaciones que se ejecutan bajo Windows se caracterizan sobre todo porque utilizan ventanas como interfaz de comunicación con el usuario y porque se controlan a través de eventos.

Para reducir el exceso de trabajo que acompaña el desarrollo de nuevas aplicaciones, la mayoria de compiladores con los que se crean las aplicaciones Windows proporcionan diversos tipos de ayuda, desde bibliotecas de clases, en las que estan encapsuladas las funciones API de Windows o programas especializados en la creación de componentes hasta editores que permiten programar de forma visual las aplicaciones. Como precursor de estas ayudas al programador es preciso mencionar Visual Basic, creado precisamente para desarrollar aplicciones para windows.

Delphi combina la idea del desarrollo visual de aplicaciones con Pascal, un lenguaje de programación extremadamente versátil y potente. El resultado de esta relación se puede calificar perfectamente de extraordinario. Delphi ofrece al usuario poco iniciado en elementos de programa orientados a objetos y una programación visual por componentes, de tal forma que pasado un cierto tiempo de aclimatación

cualquier usuario podrá crear cómodamente aplicaciones Windows de calidad.[8] En la Figura 5.2 se visualiza el entorno de programación Delphi.

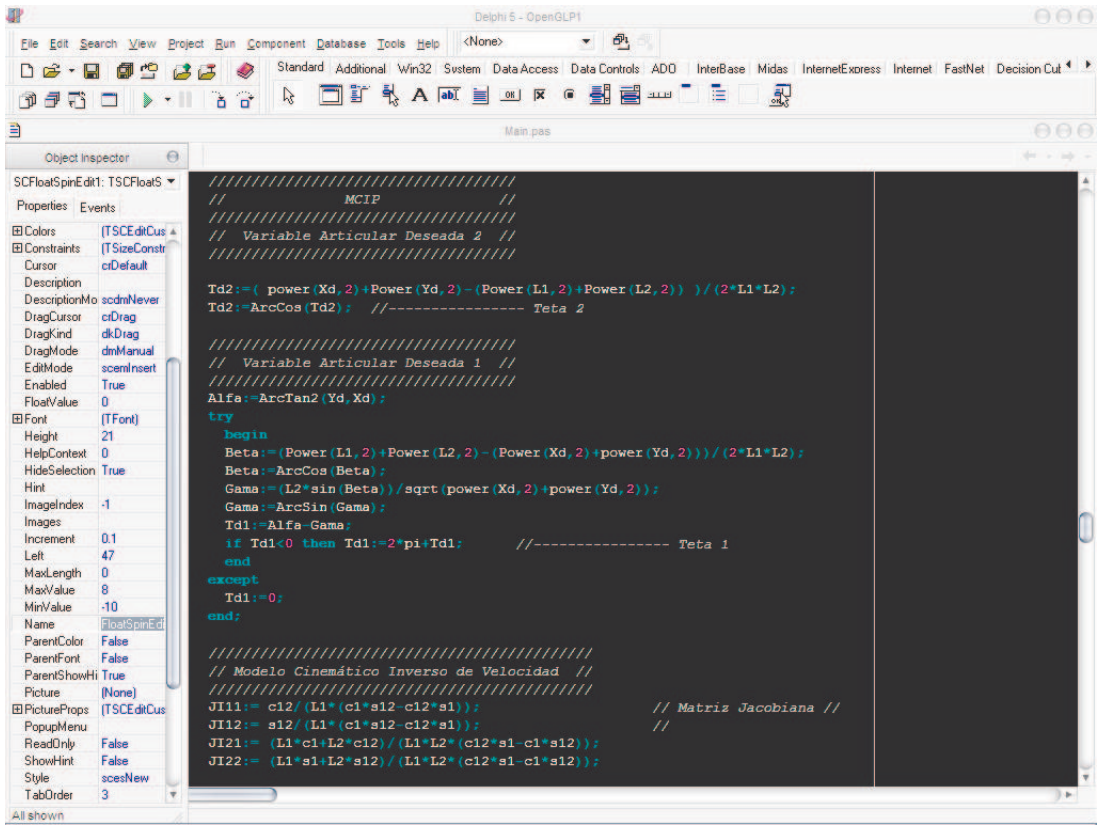

Figura 5.2: Visualización del entorno de programación Delphi.

En 1971 Nikolaus Wirth desarrolló el lenguaje Pascal. Aunque en un principio se diseñó como un lenguaje de aprendizaje para programación estructurada, actualmente todavía goza de mucha popularidad y encuentra aplicación en ámbitos profesionales y semiprofesionales. Además de las prpiedades positivas del lenguaje, este desarrollo le debe también mucho al compilador de Pascal que Borland introdujo en el mercado en los años ochenta. Lo que hace Delphi tan interesante para los programdores es el desarrollo rápido y visual de aplicaciones Windows.[8]

Para la visualización en tres dimenciones, Delphi posee soporte para OpenGL, la cual es una herramineta de gráficos tridimencionales y rendering abierta a todos sistemas desarrollada por Silicon Graphics, pero esta no se incluye en la distribución de Delphi, la situación tampoco es muy trágica ya que probablemente ya se tiene instalada en la computadora(desde Windows95b OpenGL se distribuye junto con el sistema operativo). La Figura 5.3 muestra un mundo virtual creado en Delphi utilizando las librerias OpenGL, el cual es un laberinto en 3D que puede ser manipulado con el robot R2SG o con el teclado de la computadora.

OpenGL es una librería gráfica escrita originalmente en C que permite la manipulación de gráficos

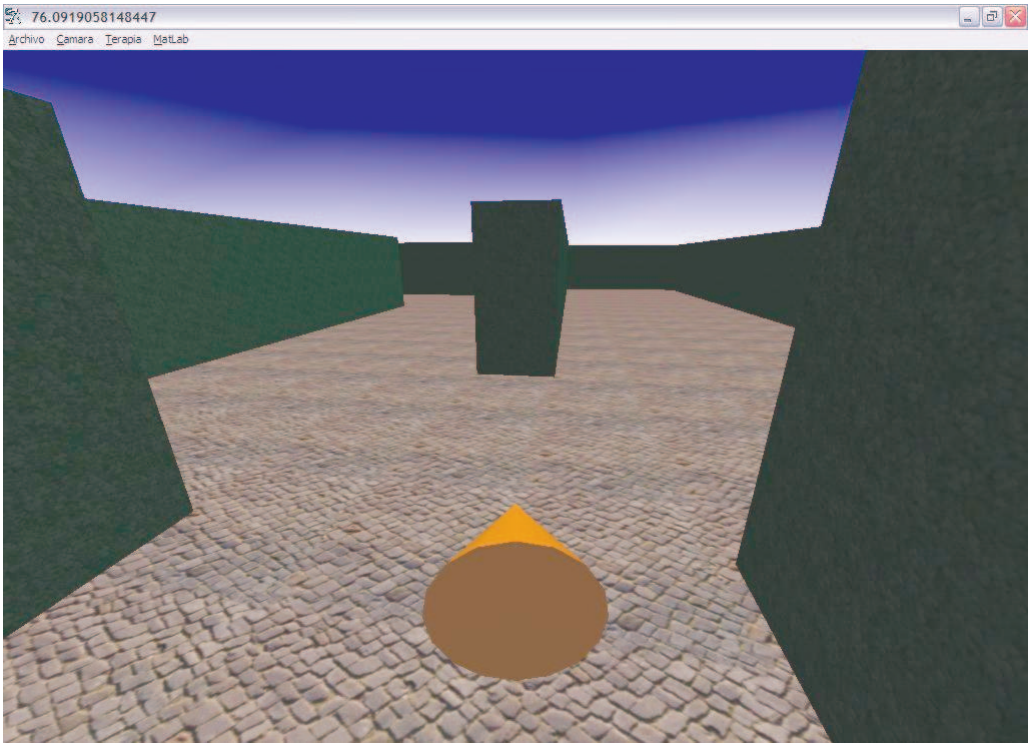

Figura 5.3: Mundo virtual creado con Delphi y OpenGL.

3D a todos los niveles. Esta librería se concibió para programar en máquinas nativas Silicon Graphics bajo el nombre de GL (Graphics Library). Posteriormente se consideró la posibilidad de extenderla a cualquier tipo de plataforma y asegurar así su portabilidad y extensibilidad de uso con lo que se llegó al término Open Graphics Library, es decir, OpenGL.

La librería se ejecuta a la par con nuestro programa independientemente de la capacidad gráfica de la máquina que usamos. Esto significa que la ejecución se dará por software a no ser que contemos con hardware gráfico específico en nuestra máquina. Si contamos con tarjetas aceleradoras de video, tecnología MMX, aceleradoras 3D, pipelines gráficos implementados en placa, etc., gozaremos por supuesto de una ejecución más rápida en tiempo real suave.

Así esta librería puede usarse bajo todo tipo de sistemas operativos e incluso usando una gran variedad de lenguajes de programación. Se pueden encontrar variantes de OpenGL para Windows 95/NT, Unix, Linux, Iris, Solaris, Java e incluso lenguajes de programación visuales como Visual Basic, Borland C++ Builder y por supuesto Delphi.

OpenGL funciona a través de una serie de librerías DLL que suelen tener variados nombres, pero que a fin de cuentas cubren funciones muy específicas y son muy sencillas de identificar, las cuales son estas:

OpenGL.DLL (Open Graphics Library, Librería de Gráficos Abierta), esta se puede encontrar también con el nombre de OpenGL32.DLL (para el caso de los sistemas operativos de 32 bits) o bién

simplemente como GL.DLL. Esta es la librería principal, que contiene la mayoría de las funciones que se utilizarán. Las funciones contenidas en esta librería inician con las letras gl.

GLU.DLL (Graphics Utility Library, Librería de Utilerías de Gráficos), también se puede encontrar como GLU32.DLL. Contiene funciones para objetos comunes a dibujar como esferas, donas, cilindros, cubos, etc., varias figuras ya predefinidas que pueden llegar a ser útiles en ciertos casos, así como funciones para el manejo de la cámara entre muchas otras. Se pueden identificar las funciones que pertenecen a esta librería porque llevan antepuestas en su nombre las siglas glu.

GLUT.DLL (GL Utility Toolkit, Equipo de Herramientas de Utilería para el desarrollo de Gráficos), esta también permite crear objetos complejos como GLU, aunque la principal función de esta librería es permitir que los programas se vuelvan interactivos, o sea que posibilita la libre creación de ventanas, así como el acceso al ratón y al teclado. Se podría llegar a prescindir en ese aspecto de GLUT, ya que Delphi ofrece de manera natural poder crear ventanas y responder a los diferentes eventos del manejo del ratón y el teclado; sin embargo, muchos programadores (principalmente programadores de C y C++) piensan que programar aplicaciones basadas en GLUT tiene muchísimas ventajas, ya que esto ofrece independencia del sistema operativo, pues no es lo mismo crear una ventana basándose en las APIs (Application Programming Interface) de Windows que crear una ventana en algún ambiente gráfico de Unix o Linux, por esto, si se usa GLUT como manejador de ventanas, no se tendría que reescribir el código para migrar la aplicación de plataforma, ya que solo habría que conseguir la versión de GLUT para el ambiente gráfico deseado y volver a compilar nuestro programa en el nuevo ambiente.

Estas librerías se encuentran disponibles en Internet, y se distribuyen de manera gratuita en diferentes sitios, principalmente se pueden encontrar en el sitio oficial de OpenGL[27].

Otro componente es el llamado Frame Buffer, que no es otra cosa que el área de memoria donde se construyen los gráficos antes de mostrarlos al usuario, es decir, que el programa de OpenGL escribe en esta área de memoria, y automáticamente envía su contenido a la pantalla una vez que la escena está completamente construida. La Figura 5.4 muestra gráficamente como esta constituida la estructura de abstracción de OpenGL.

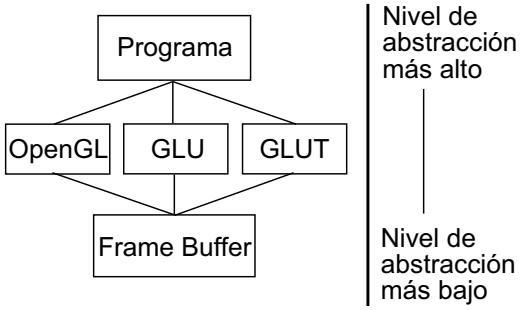

Figura 5.4: Niveles de abstracción en OpenGL.

Algunas de las principales razones para la utilización de OpenGL son que no se debe de preocupar por el hardware a la hora de programar, por lo tanto, la programación no se hace tan complicada, de

igual manera el código que se escribe con OpenGL será independiente de la plataforma y del sistema operativo, esto es algo importante por que en un laboratorio de realidad virtual se manejan distintos sistemas operativos, entre los cuales se encuentran Windows 2000, Irix, e inclusive linux.

A fin de obtener un despliegue de una escena tridimensional que se modela en coordenadas mundiales, primero se debe de establecer una referencia de coordenadas para la "cámara". Esta referencia de coordenadas define la posición y orientación para el plano de la película de la cámara, que es el palno que se desea utilizar para desplegar una vista de los objetos de una escena. De esta manera se transfieren las descripciones de los objetos a las coordenadas de referencia de la cámara y se proyectan sobre el plano de despliegue que se selecciona. Entonces se pueden desplegar los objetos en forma de armazón o podemos aplicar técnicas de iluminación y presentación de superficie para sombrear las áreas visibles.

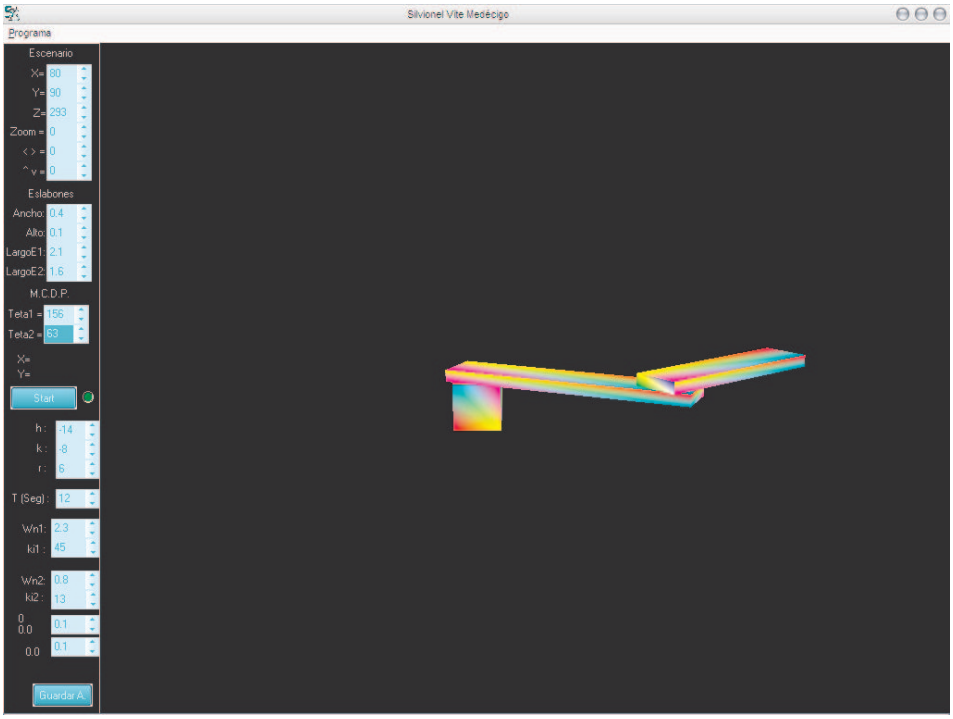

Figura 5.5: Vista del ambiente virtual del robot R2SG.

Dibujar en un monitor de computadora es diferente a dibujar en un papel. En una computadora, la memoria que mantiene el dibujo es usualmente llenado con el último dibujo que se dibujo, así que siempre se necesita limpiar el dibujo con algún color de fondo antes de empezar a dibujar la nueva escena. El color que se usa de fondo dependerá de la aplicación, por ejemplo, para un procesador de palabras, generalmente utilizamos el color blanco como fondo, ya que es el color de una hoja de papel real, en el caso del programa para el robot R2SG se propuso el color negro. En muchos tipos de diferentes de computadoras, el hardware gráfico consiste de multiples buffers aparte del buffer que contiene los colores de los píxeles que serán desplegados en la pantalla. La Figura 5.5 muestra el programa desarrollado para la visualización virtual del robot R2SG.

El prgrama se divide en dos ventanas, en la ventana derecha se visualiza la posición del robot R2SG de forma tridimensional y en la parte izquierda se encuenran los controles, en donde se encuentran los botones y las variables a utilizar; para cada experimento como la trayectoria sinusoidal, la circunferencia, el índice de manipulabilidad, etc. se desarrolló un programa distinto en el que realmente no difieren tanto, la diferencia entre un programa y otro son las distintas variables a modificar para el control del robot R2SG así como el tipo de control que se está experimentando, el código de la visualzación virtual del robot por lo regular queda de la misma forma.

Las variables comunes entre los distintos programas son las que modifican la perspectiva del escenario virtual (posición de la cámara en x, y y z dentro del espacio tridimensional), el largo, ancho y alto de los eslabones y la posición angular de cada eslabón de forma manual.

En cuanto a los botones, por lo regular se deja un botón para comenzar o detener la visualización virtual en tiempo real suave, ya sea aplicandole un control al robot R2SG o experimentando de forma manual. Y otro botón para guardar en un archivo de extensión .mat los resultados obtenidos del experimento para posteriormente procesarlos o visualizarlos con la ayuda del programa MATLAB de forma gráfica.

Una de las problemáticas con la que se cuenta por el hecho de utilizar una plataforma como Windows, es que windows no deja de de ser un sistema multitareas, es decir, mientras windows atiende al programa del robot R2SG de igual manera puede estar atendiendo a otro programa de tal forma que se encuentra utilizando recursos del sistema como memoria RAM, memoria caché, memoria virtual, etc. Es por ello que llamamos tiempo real suave.

Windows nos brinda la opción de darle prioridad a los programas en ejecución, pero no deja de atender a otros programas; debido a esto, el tiempo de muestreo no es constante y lo ideal es que el tiempo de muestreo de nuestro programa sea constante para lograr un control óptimo. El programa

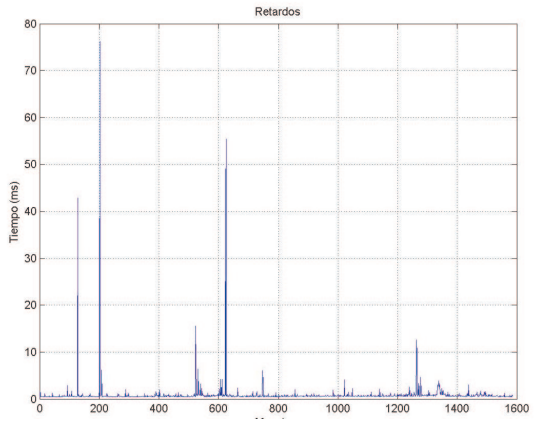

Figura 5.6: Retardos de hasta 76ms ejecutando el programa Acrobat Reader.

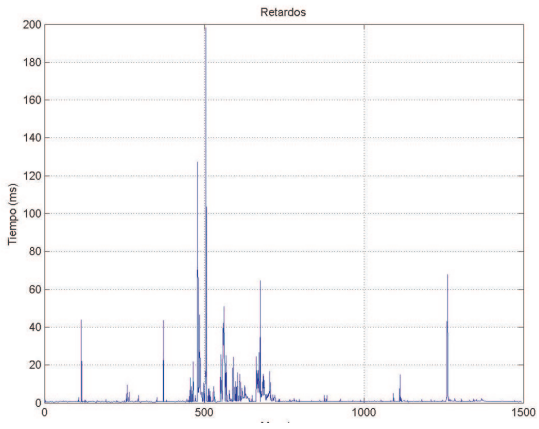

Figura 5.7: Retardos de hasta 200ms ejecutando el programa Acrobat Reader y Word.

para el robot R2SG monitorea en tiempo de ejecucion los retardos en el ciclo de control, de tal manera

que se pueda realizar un análisis posterior del control. Las Figuras 5.6 y 5.7 muestran las gráficas de los retardos ocacionados a propósito ejecutando otros programas al mismo tiempo. Los dispositivos de salida de datos del mundo virtual se complementan con sistemas de entrada de datos, que permiten desarrollar una actividad en el entorno digital. El dispoitivo que permite estas entradas y salidas de datos es una tarjeta de adquisición de datos (TAD) de la National Instrument modelo P701E. Para poderla utilizar de interfaz entre la circuitería electrónica del robot y la computadora, además de ser instalada, es necesario instalar los componentes Active X en el lenguaje de programacón con el que se desea manipularla que en este caso es Delphi. La Figura 5.8 muestra los componentes utilizados para el manejo de la tarjeta de adquisición de datos y componentes que se utilizan para el ambiente virtual.

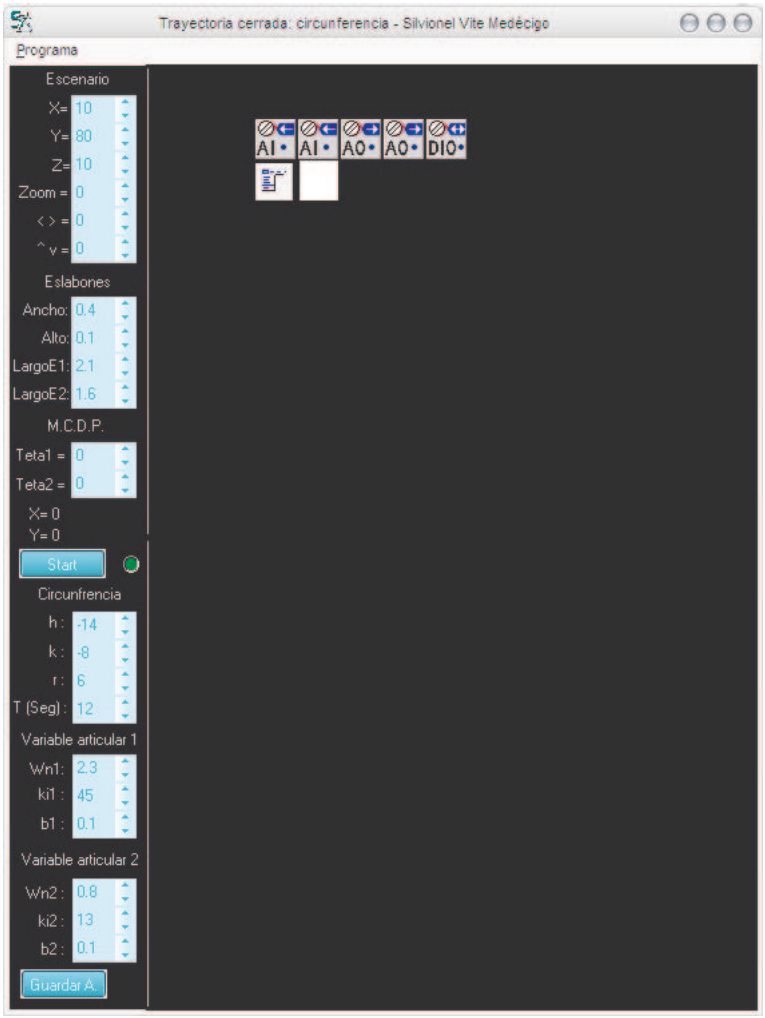

Figura 5.8: Visualizaión de componentes en el ambiente Delphi.

Se necesitó de dos componentes que manipulan las entradas analógicas de la TAD para cada señal proveniente del acondicionamiento de señales que revela la posición de cada articulación, de otros dos componentes para las salidas analógicas las cuales son las señales de control y de un componente de

entradas/salidas digitales del cual únicamente se manipulan dos bits como salidas, uno para cada articulación, esto es para controlar el sentido de los motores.

5.4. Conclusiones

Este capítulo tuvo como meta la construcción del ambiente virtual del robot R2SG para su visualización en línea. Para ello se utilizó el lenguaje de programación visual como Delphi, el cual es un lenguaje de programación potente y demasiado versátil con el cual es posible manejar librerías OpenGL para lograr un buen diseño, pero como es cierto que Delphi opera bajo la plataforma Windows, se tiene una viasualización en tiempo real suave, es decir, existen pequeños retardos poco visibles ya que Windows es un sistema operativo multitareas; esto se podría resolver llevándolo al sistema operativo Linux mediante el lenguaje Kylix, el problema es que es menos comercial. En este capítulo se presenta la metodología OpenGL-Delphi para integrar un ambiente virtual dinámico en línea del robot, tal que los movimientos virtuales corresponden a los movimientos reales mediante el uso de los modelos cinemáticos sintetizados en el capítulo 4, la visualización virtual en línea permite conocer el desempeño del robot, sobre todo cuando la estación de trabajo es remota y el operador humano no tiene conocimiento explícito de lo acontecido en el robot, el ambiente virtual le brinda un panorama del desempeño real del robot tal que pueda tomar decisiones de operación. Esta contribución se realiza como un principio de visualización científica para trabajos futuros, sin embargo y como inicialmente se aclara, los retardos afectan el tiempo de muestreo en los experimentos de control, y la visualización contribuye en este problema, por lo que se recomienda hacer la visualización virtual de manera independiente a la tarea de control, los estudios de retardo de tiempo presentados en este capítulo permiten verificar lo anteriormente expuesto.

Capítulo 6

Experimentos de control

6.1. Introducción

En este capítulo se encuentran todos los resultados experimentales de las estrategias de control clásica y moderna para regulación a una coordenada operacional, regulación basada en seguimiento a una coordenada operacional y seguimiento de una trayectoria cerrada.

6.2. Configuración inicial de los experimentos

A continuación se reportan las configuraciones iniciales de todos los controles experimentados. Es necesario sintonizar cada una de las constantes ya sea la k_p, k_d, k_i o α según sea el caso y se logra esto al modificar la frecuencia natural (Wn) para cada articulación y control a experimentar obteniendo un óptimo control.

6.2.1. Control PD (regulación)

La configuración inicial del control proporcional derivativo se muestra en el Cuadro 6.1 siguiente:

Constantes control PD		
	q_1	q_2
Wn	1.8	0.6
Ángulo inicial	0º	0º
Ángulo deseado	90º	90º

Cuadro 6.1: Configuración inicial del control PD.

Las Figuras 6.1 y 6.2 muestran la visualización virtual inicial y final del robot R2SG con un control PD.

Figura 6.1: Posición inicial del robot R2SG (control PD).

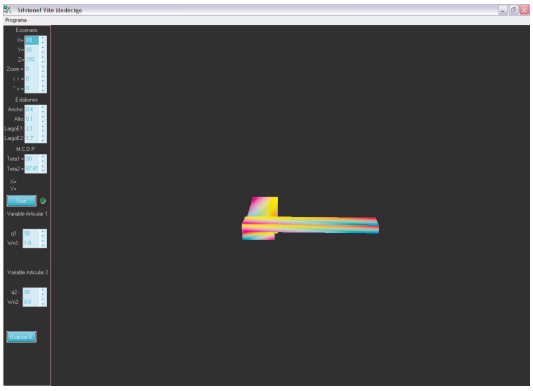

Figura 6.2: Posición final del robot R2SG (control PD).

6.2.2. Control PID (regulación)

La configuración inicial del control proporcional integral derivativo se muestra en el Cuadro 6.2 siguiente:

Constantes control PID		
	q_1	q_2
Wn	1.7	0.55
Ángulo inicial	0º	0º
Ángulo deseado	90º	90º

Cuadro 6.2: Configuración inicial del control PID.

Las Figuras 6.3 y 6.4 muestran la visualización virtual inicial y final del robot R2SG con un control PID.

Figura 6.3: Posición inicial del robot R2SG (control PID).

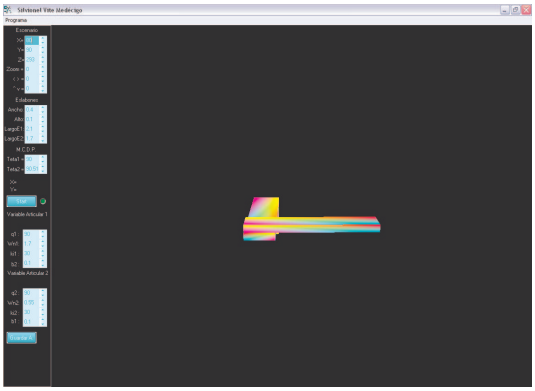

Figura 6.4: Posición final del robot R2SG (control PID).

6.2.3. Control PID no lineal (regulación)

La configuración inicial del control proporcional integral derivativo no lineal por modos deslizantes de 2º orden es la siguiente:

Constantes control PIDNL		
	q_1	q_2
Wn	1.9	0.9
Ángulo inicial	0º	0º
Ángulo deseado	90º	90º

Cuadro 6.3: Configuración inicial del control PIDNL.

Las Figuras 6.5 y 6.6 muestran la visualización virtual inicial y final del robot R2SG con un control PIDNL.

Figura 6.5: Posición inicial del robot R2SG (control PIDNL).

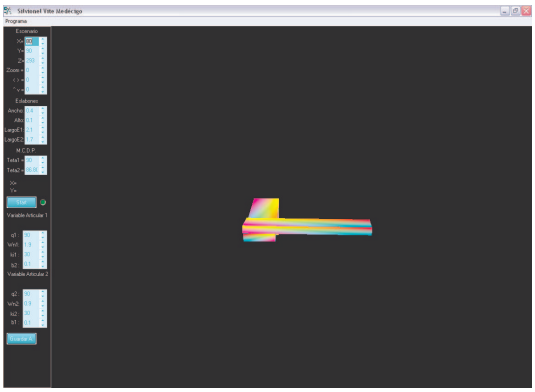

Figura 6.6: Posición final del robot R2SG (control PIDNL).

6.2.4. Seguimiento de trayectorias articulares: polinomio de 5º

La configuración inicial del control proporcional integral derivativo no lineal por modos deslizantes de 2º orden con seguimiento de trayectoria mediante los polinomios de las ecuaciones (4.52) y (4.53) es la siguiente:

Constantes control PIDNLST polinomio de 5º		
	q_1	q_2
t_b	12s	7s
Wn	2.7	1
Ángulo inicial	0º	0º
Ángulo deseado	90º	90º

Cuadro 6.4: Configuración inicial del control PIDNLST: polinomio de 5º.

Las Figuras 6.7 y 6.8 muestran la visualización virtual inicial y final del robot R2SG con un control PIDNL con regulación basada en seguimiento de trayectorias.

Figura 6.7: Posición inicial del robot R2SG (control PIDNLST).

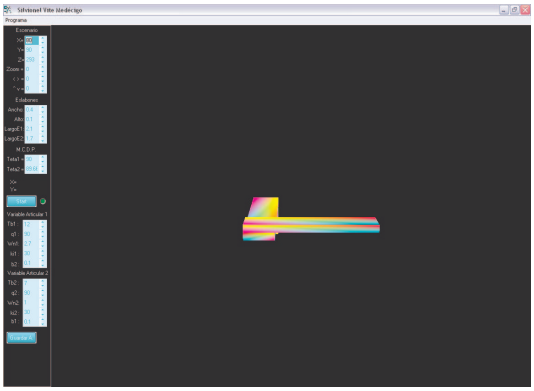

Figura 6.8: Posición final del robot R2SG (control PIDNLST).

6.2.5. Seguimiento de trayectorias articulares: sinusoidal

La configuración inicial del control proporcional integral derivativo no lineal por modos deslizantes de 2º orden es la siguiente:

Constantes control PIDNLST sinusoidal		
	q_1	q_2
Wn	1.2	0.4
Amplitud	0.6	0.6
Periodo	8s	7s

Cuadro 6.5: Configuración inicial del control PIDNLST: sinusoidal.

6.2.6. Seguimiento de una trayectoria cerrada operacional: circunferencia

La configuración inicial del control proporcional integral derivativo no lineal por modos deslizantes de 2º orden es la siguiente:

Constantes control PIDNLST circunferencia		
	q_1	q_2
Wn	2.3	0.9
radio	6cm	
h	-14cm	
k	-8cm	
Periodo	12s	

Cuadro 6.6: Configuración inicial del control PIDNLST cerrada: circunferencia.

6.3. Gráficas de los experimentos

6.3.1. Control PD (regulación)

Las siguientes gráficas muestran experimentos reales del control PD con el robot R2SG.

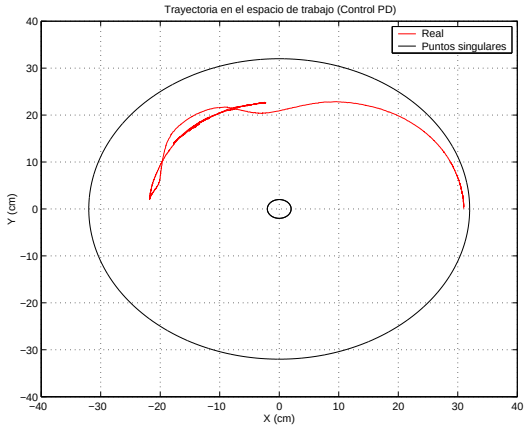

Figura 6.9: Trayectoria en el espacio de trabajo (control PD)

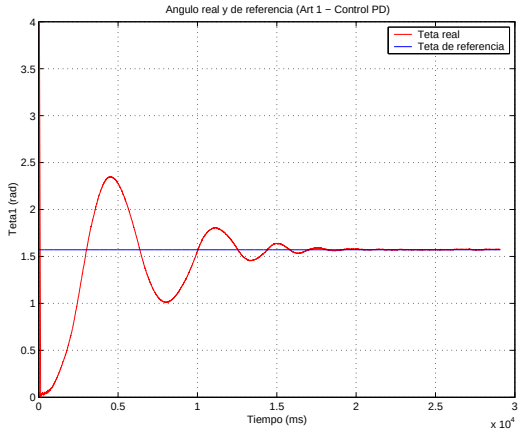

Figura 6.10: Coordenada generalizada q_1 real y deseada (control PD).

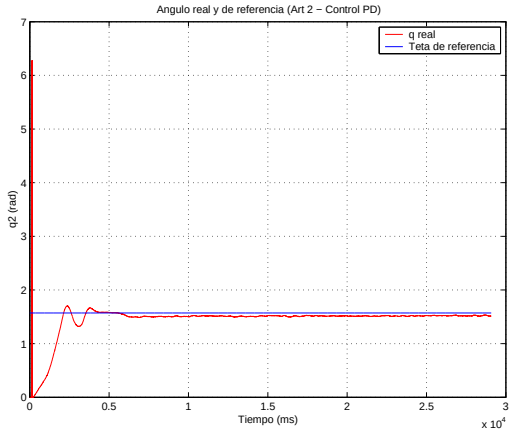

Figura 6.11: Coordenada generalizada q_2 real y deseada (control PD).

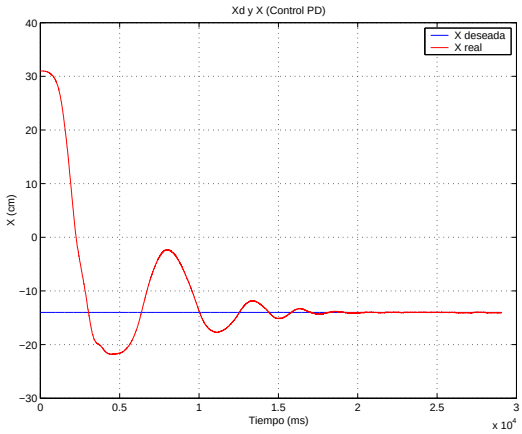

Figura 6.12: Coordenada operacional X real y deseada (control PD).

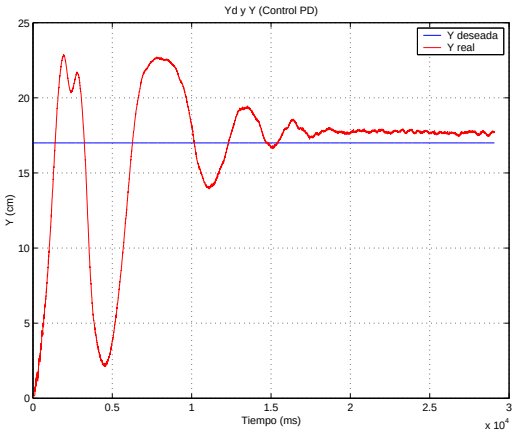

Figura 6.13: Coordenada operacional Y real y deseada (control PD).

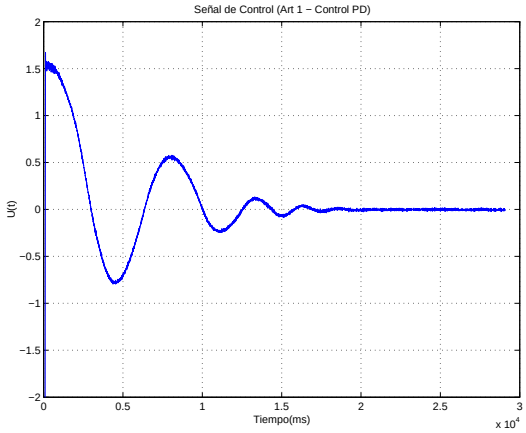

Figura 6.14: Señal de control q_1 (control PD).

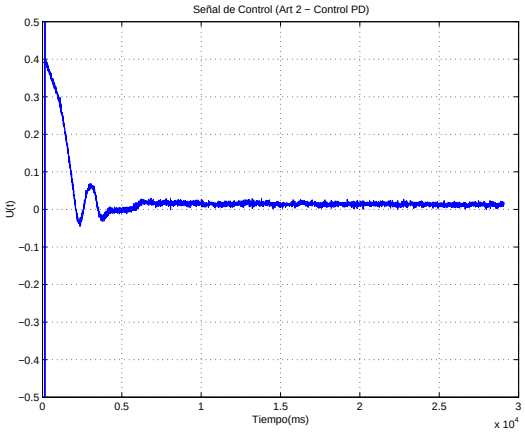

Figura 6.15: Señal de control q_2 (control PD).

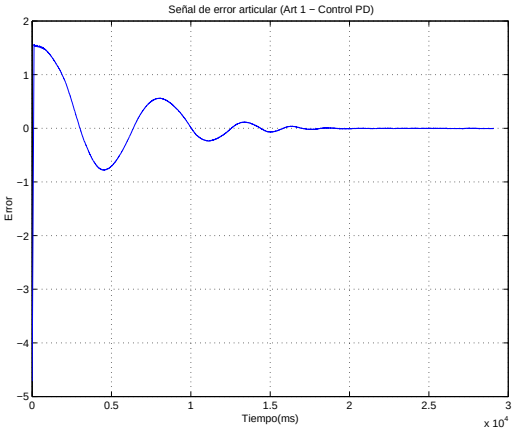

Figura 6.16: Error articular q_1 (control PD).

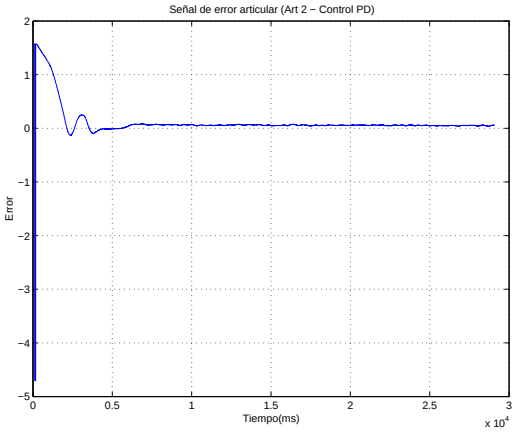

Figura 6.17: Error articular q_2 (control PD).

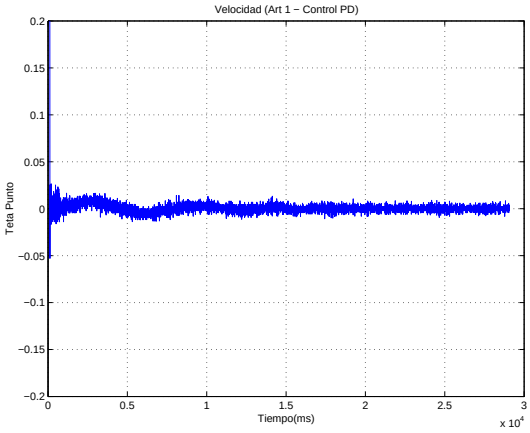

Figura 6.18: Velocidad articular q_1 (control PD).

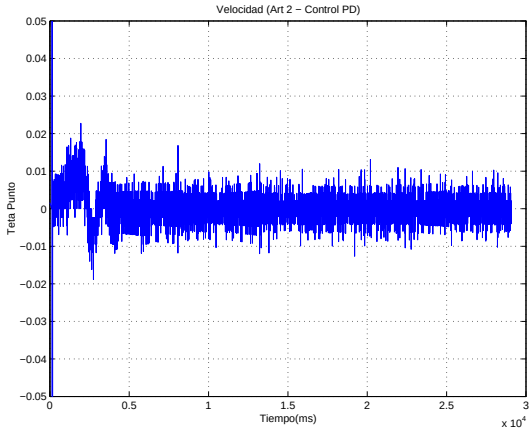

Figura 6.19: Velocidad articular q_2 (control PD).

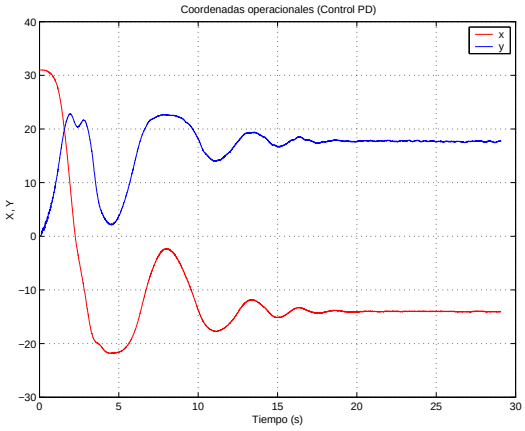

Figura 6.20: Coordenadas operacionales x y y (control PD).

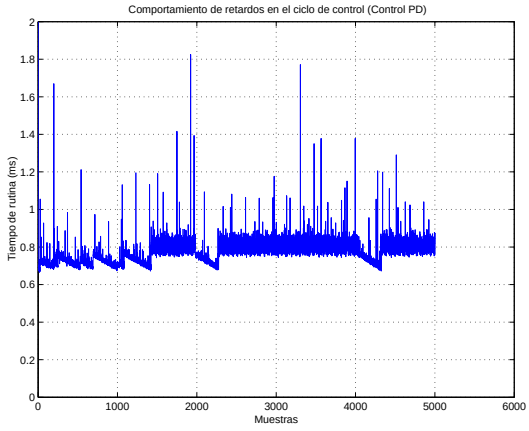

Figura 6.21: Retardos en el ciclo de control (control PD).

6.3.2. Control PID (regulación)

Las siguientes gráficas muestran experimentos reales del control PID con el robot R2SG.

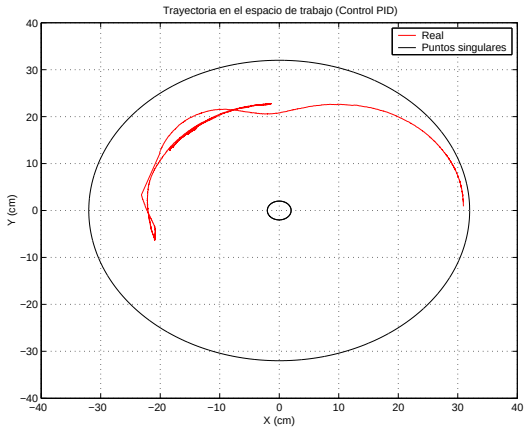

Figura 6.22: Trayectoria en el espacio de trabajo (control PID)

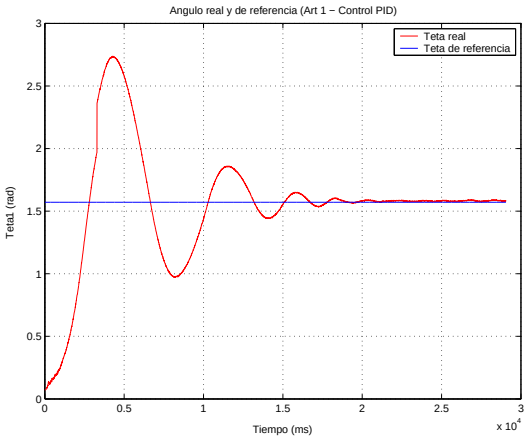

Figura 6.23: Coordenada generalizada q_1 real y deseada (control PID).

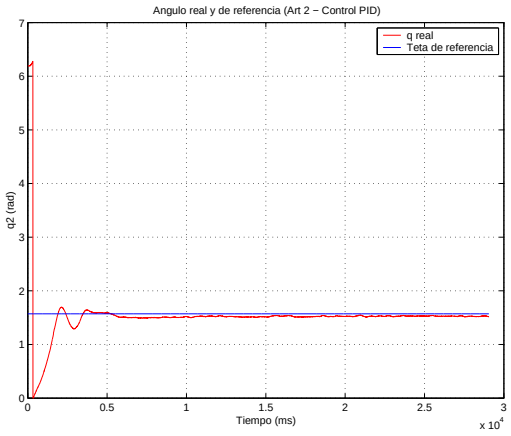

Figura 6.24: Coordenada generalizada q_2 real y deseada (control PID).

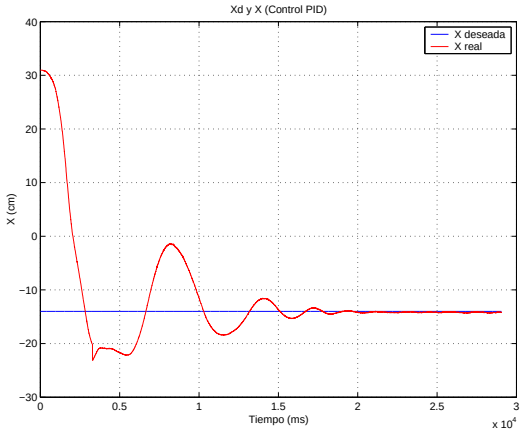

Figura 6.25: Coordenada operacional X real y deseada (control PID).

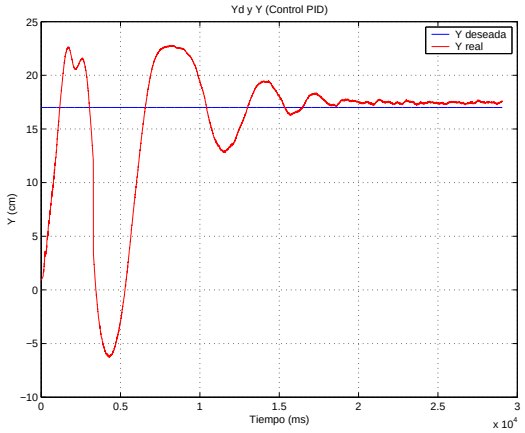

Figura 6.26: Coordenada operacional Y real y deseada (control PID).

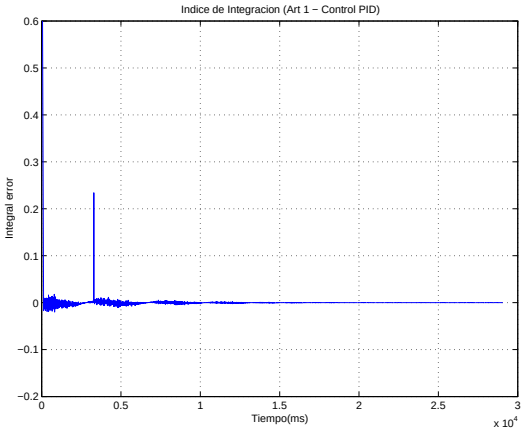

Figura 6.27: Índice de integración q_1 (control PID).

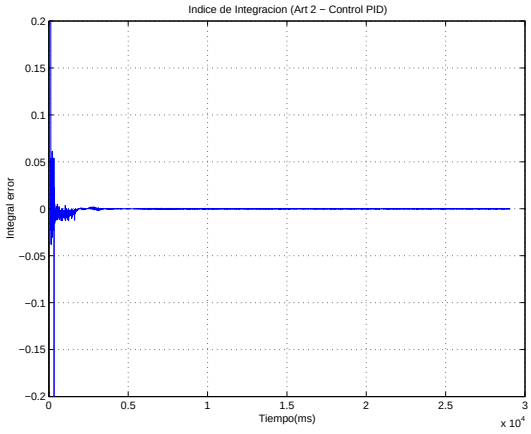

Figura 6.28: Índice de integración q_2 (control PID).

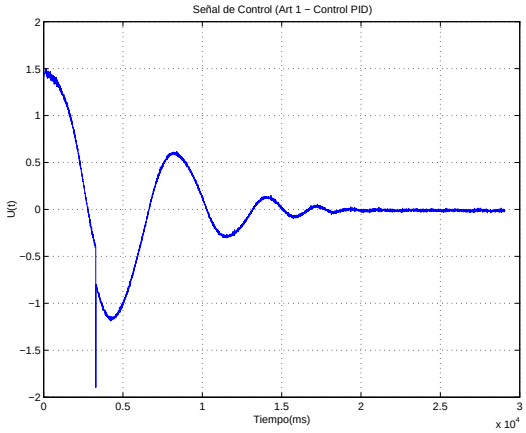

Figura 6.29: Señal de control q_1 (control PID).

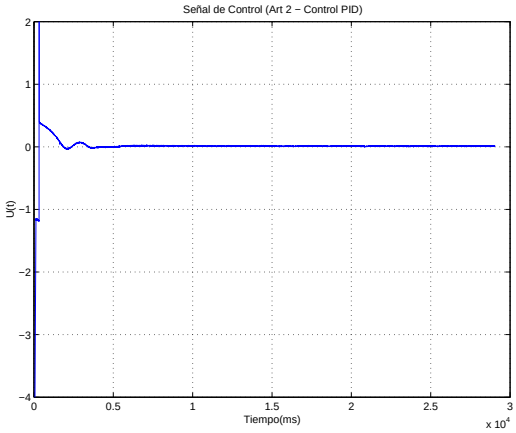

Figura 6.30: Señal de control q_2 (control PID).

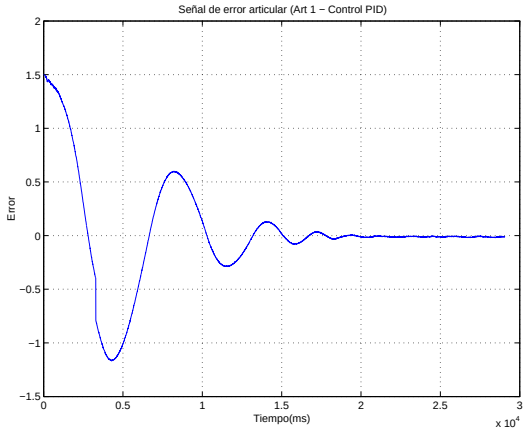

Figura 6.31: Error articular q_1 (control PID).

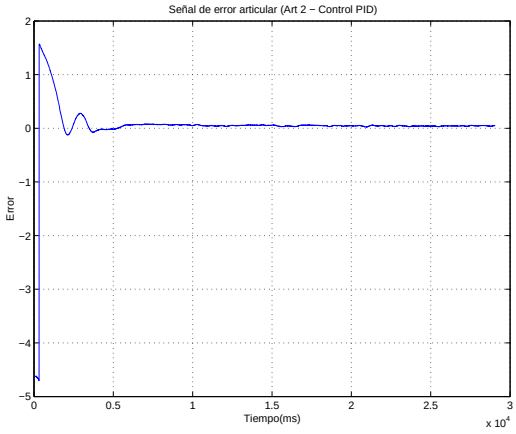

Figura 6.32: Error articular q_2 (control PID).

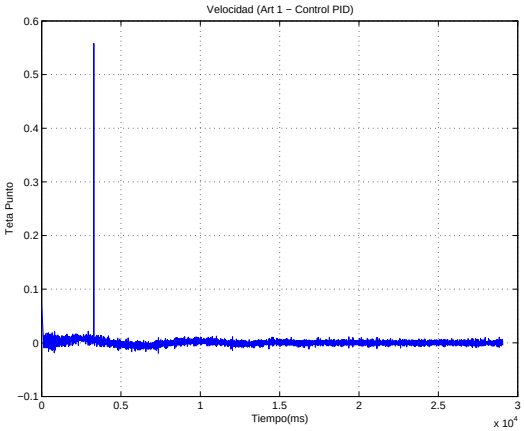

Figura 6.33: Velocidad articular q_1 (control PID).

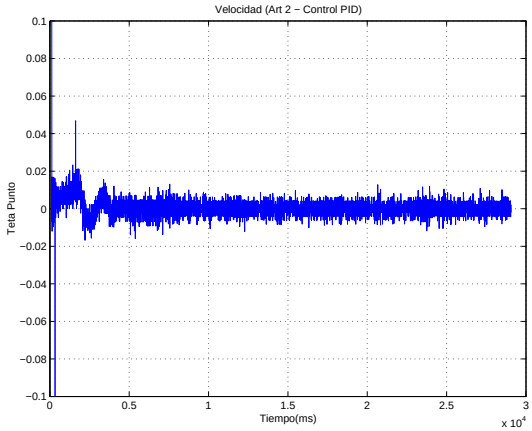

Figura 6.34: Velocidad articular q_2 (control PID).

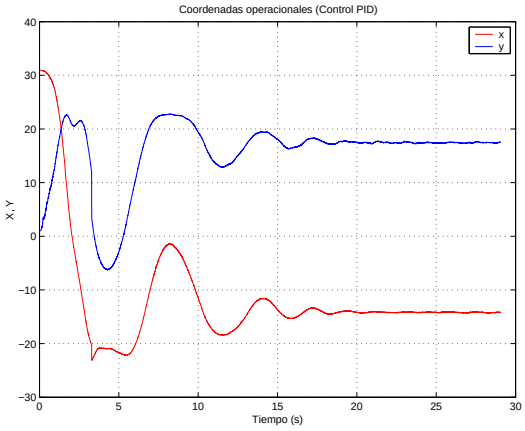

Figura 6.35: Coordenadas operacionales x y y (control PID).

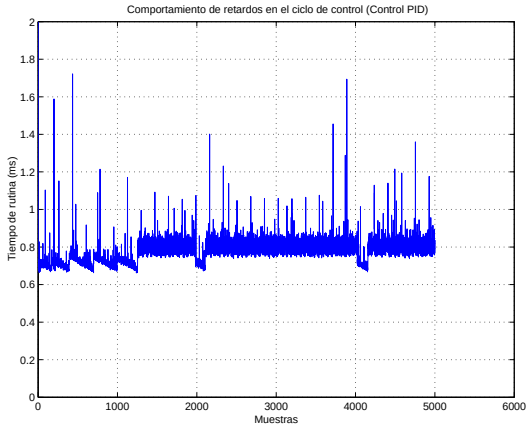

Figura 6.36: Retardos en el ciclo de control (control PID).

6.3.3. Control PID no lineal (regulación)

Las siguientes gráficas muestran experimentos reales del control PIDNL con el robot R2SG.

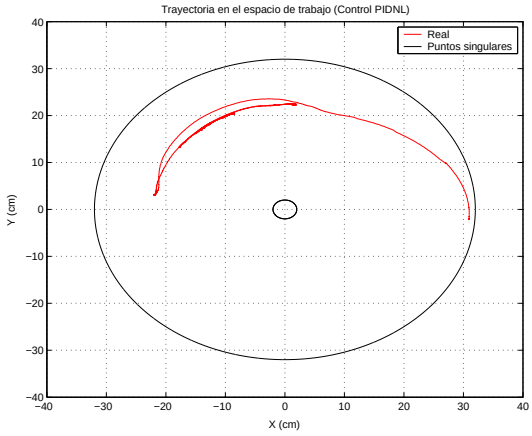

Figura 6.37: Trayectoria en el espacio de trabajo (control PIDNL)

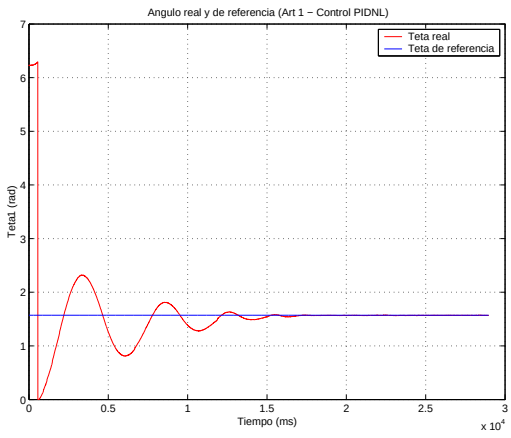

Figura 6.38: Coordenada generalizada q_1 real y deseada (control PIDNL).

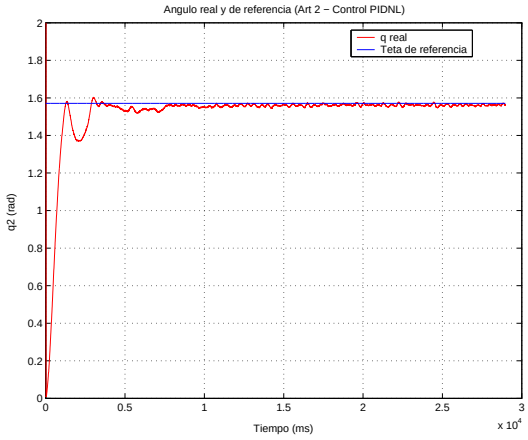

Figura 6.39: Coordenada generalizada q_2 real y deseada (control PIDNL).

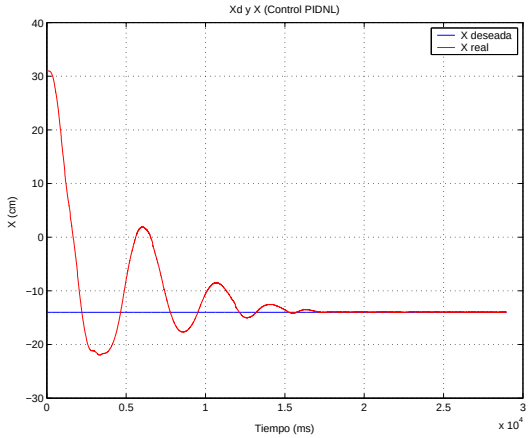

Figura 6.40: Coordenada operacional X real y deseada (control PIDNL).

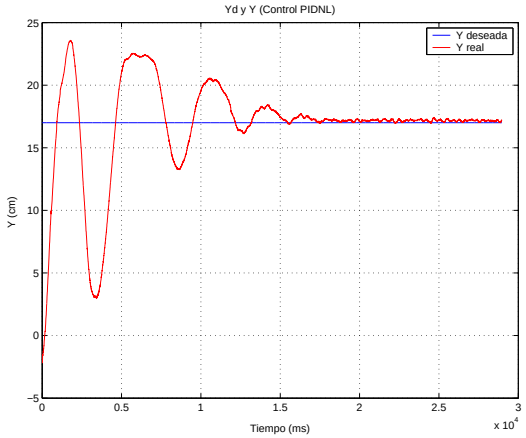

Figura 6.41: Coordenada operacional Y real y deseada (control PIDNL).

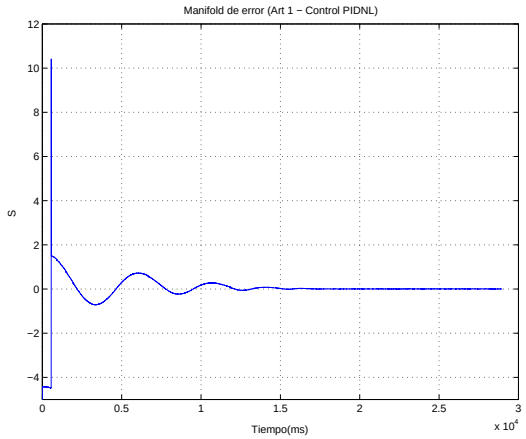

Figura 6.42: Error extendido q_1 (control PIDNL).

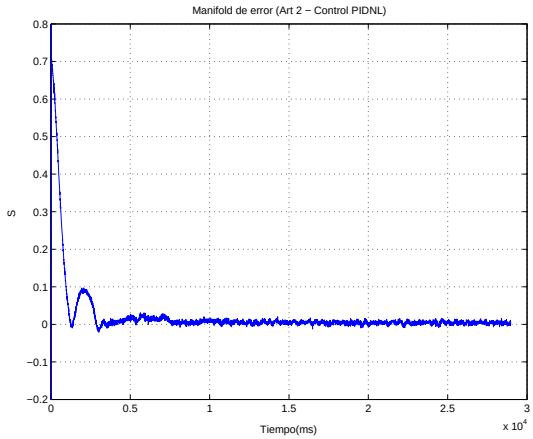

Figura 6.43: Error extendido q_2 (control PIDNL).

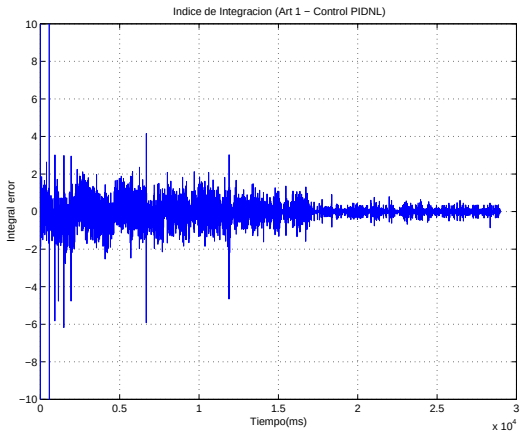

Figura 6.44: Índice de integración q_1 (control PIDNL).

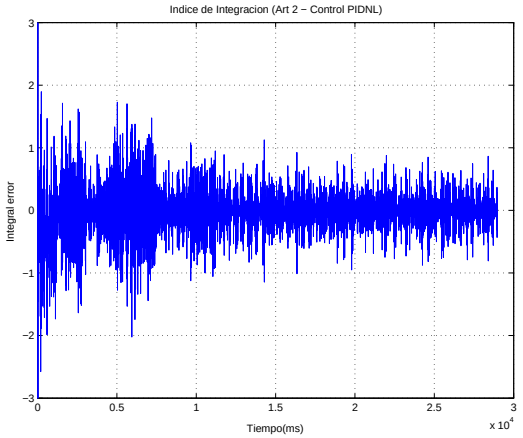

Figura 6.45: Índice de integración q_2 (control PIDNL).

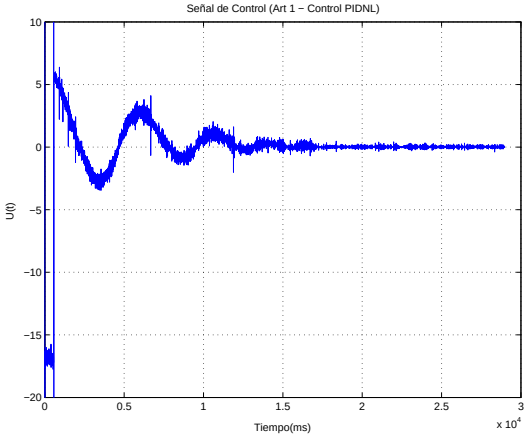

Figura 6.46: Señal de control q_1 (control PIDNL).

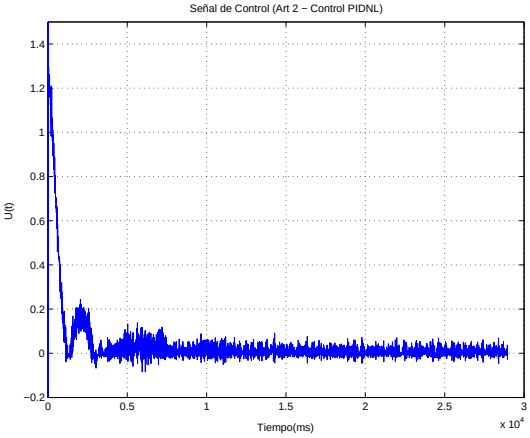

Figura 6.47: Señal de control q_2 (control PIDNL).

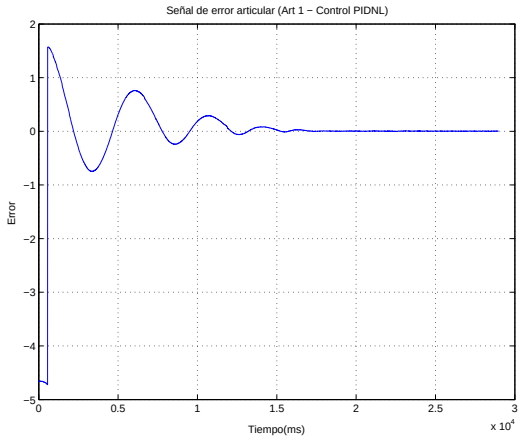

Figura 6.48: Error articular q_1 (control PIDNL).

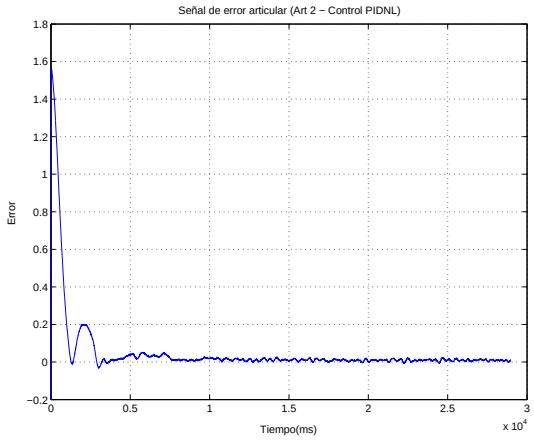

Figura 6.49: Error articular q_2 (control PIDNL).

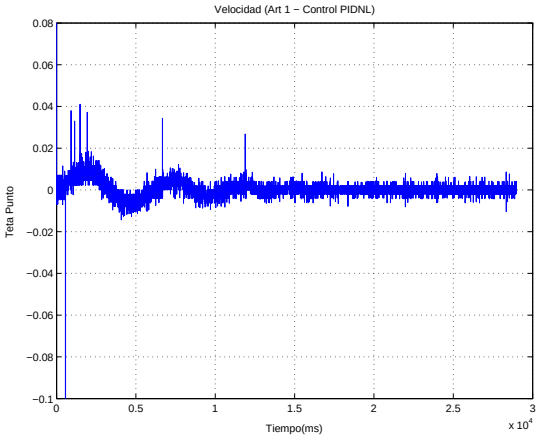

Figura 6.50: Velocidad articular q_1 (control PIDNL).

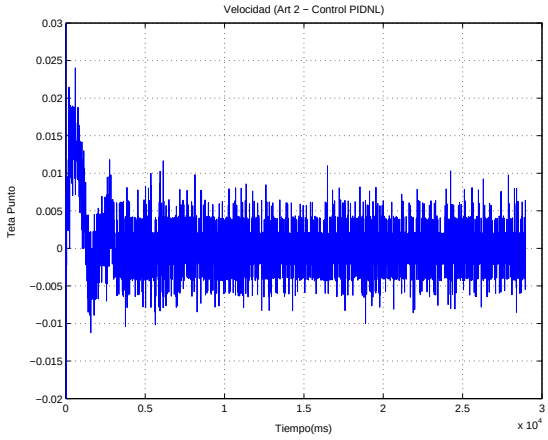

Figura 6.51: Velocidad articular q_2 (control PIDNL).

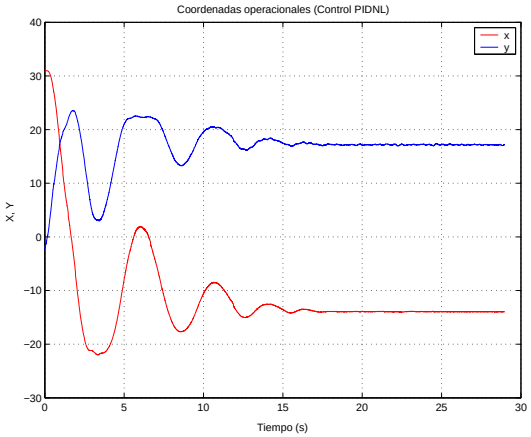

Figura 6.52: Coordenadas operacionales x y y (control PIDNL).

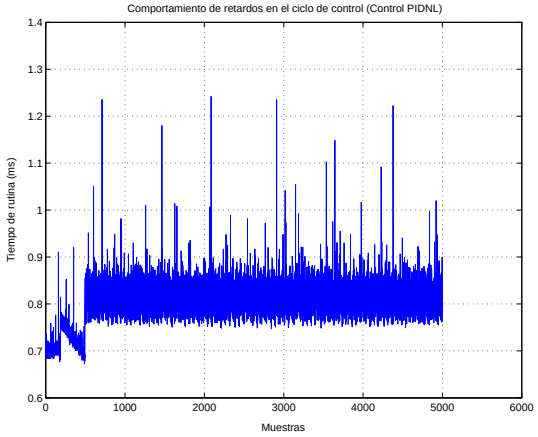

Figura 6.53: Retardos en el ciclo de control (control PIDNL).

6.3.4. Estudio comparativo (PD, PID y PIDNL)

Las siguientes gráficas muestran las comparaciones experimentales reales de los controles PD, PID y PIDNL con el robot R2SG.

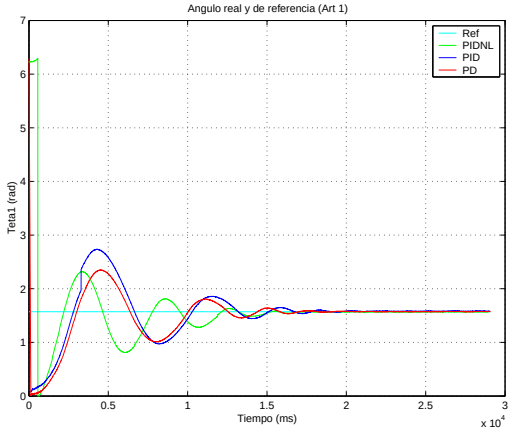

Figura 6.54: Comparación de la coordenada generalizada q_1 real y deseada (control PD, PID y PIDNL).

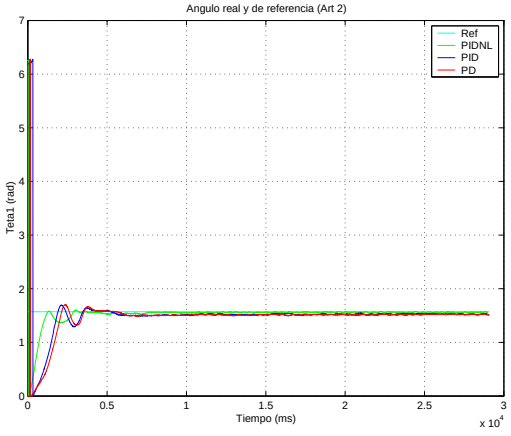

Figura 6.55: Comparación de la coordenada generalizada q_2 real y deseada (control PD, PID y PIDNL).

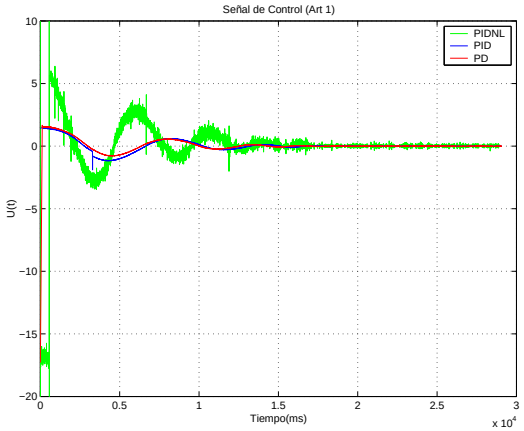

Figura 6.56: Comparación de la señal de control q_1 (control PD, PID y PIDNL).

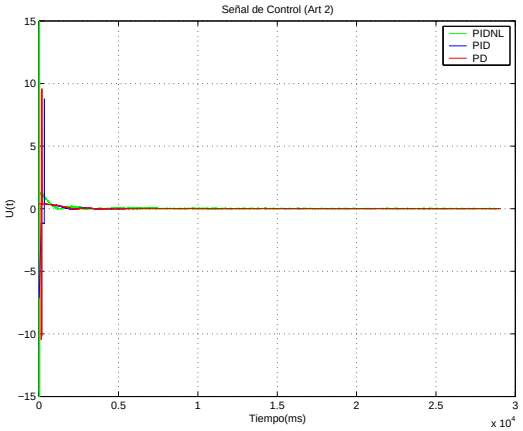

Figura 6.57: Comparación de la señal de control q_2 (control PD, PID y PIDNL).

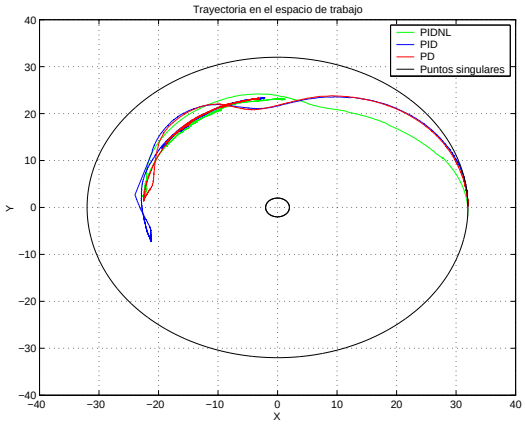

Figura 6.58: Comparación de trayectorias en el espacio de trabajo (control PD, PID y PIDNL).

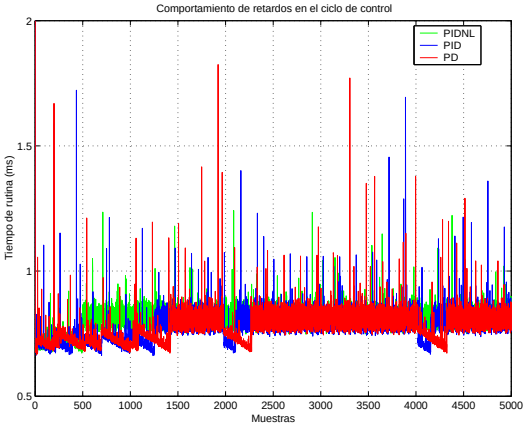

Figura 6.59: Comparación de retardos en el ciclo de control (control PD, PID y PIDNL).

6.3.5. Seguimiento de trayectorias articulares: polinomio de 5º

Las siguientes gráficas muestran experimentos reales del control PIDNL con generador de tiempo base (TBG) con el robot R2SG.

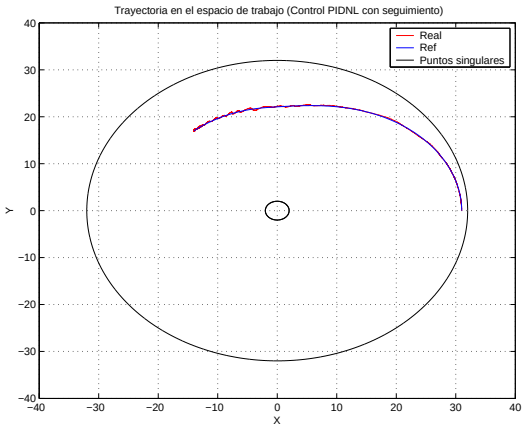

Figura 6.60: Trayectoria en el espacio de trabajo (control PIDNL con TBG)

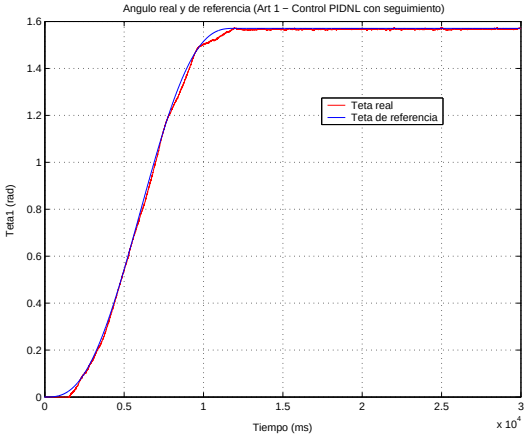

Figura 6.61: Coordenada generalizada q_1 real y deseada (control PIDNL con TBG).

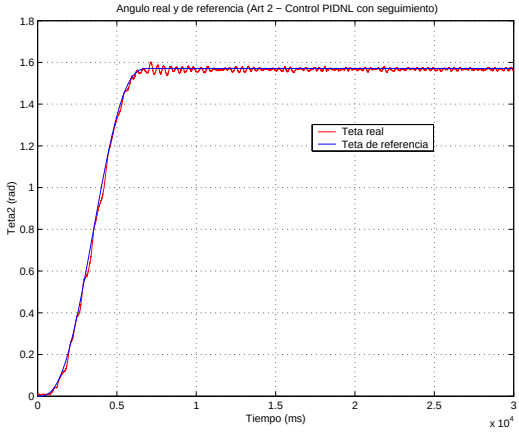

Figura 6.62: Coordenada generalizada q_2 real y deseada (control PIDNL con TBG).

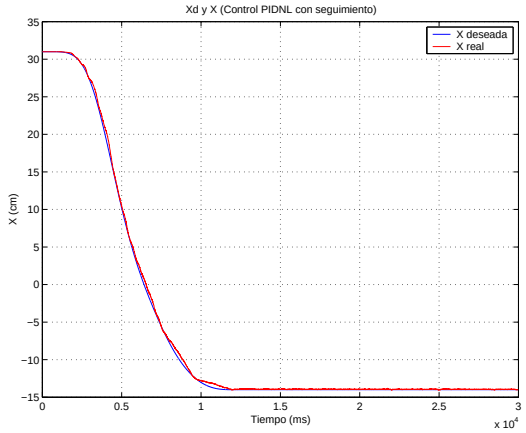

Figura 6.63: Coordenada operacional X real y deseada (control PIDNL con TBG).

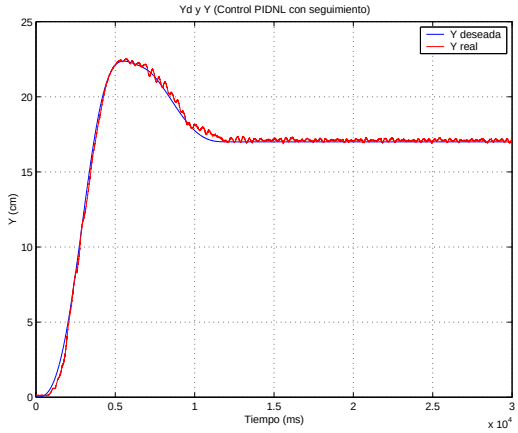

Figura 6.64: Coordenada operacional Y real y deseada (control PIDNL con TBG).

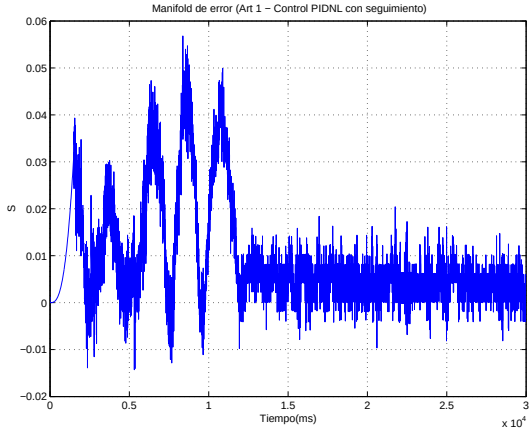

Figura 6.65: Error extendido q_1 (control PIDNL con TBG).

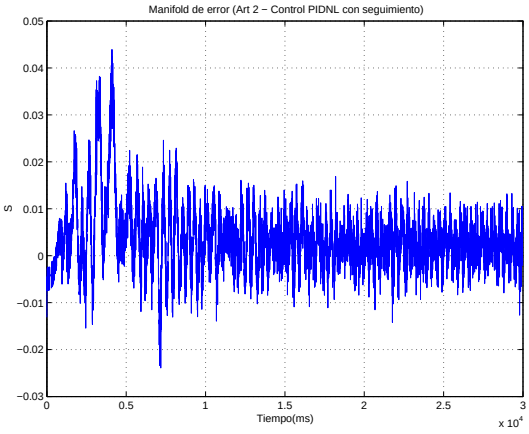

Figura 6.66: Error extendido q_2 (control PIDNL con TBG).

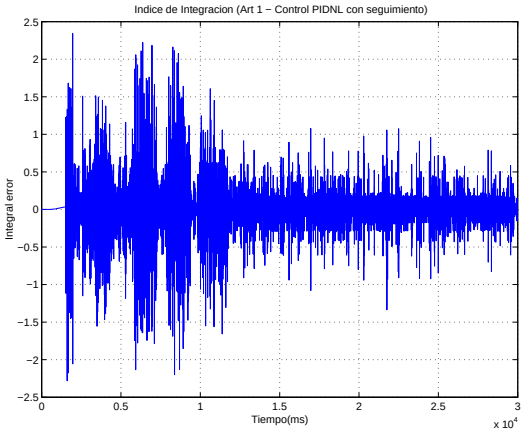

Figura 6.67: Índice de integración q_1 (control PIDNL con TBG).

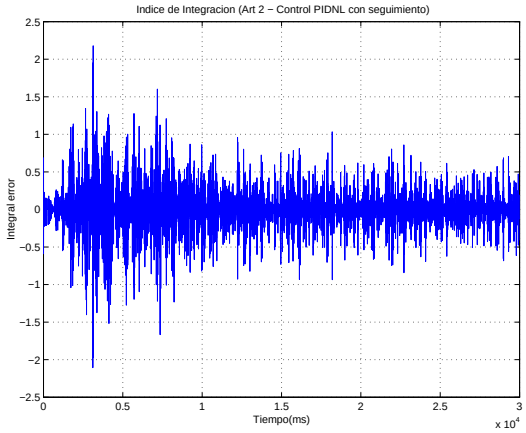

Figura 6.68: Índice de integración q_2 (control PIDNL con TBG).

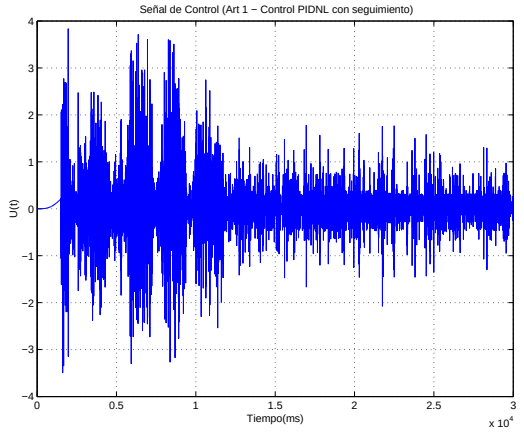

Figura 6.69: Señal de control q_1 (control PIDNL con TBG).

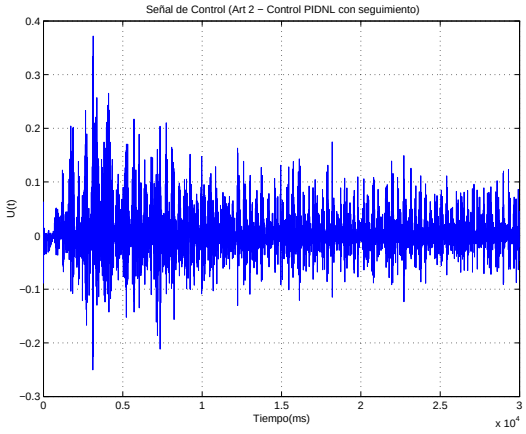

Figura 6.70: Señal de control q_2 (control PIDNL con TBG).

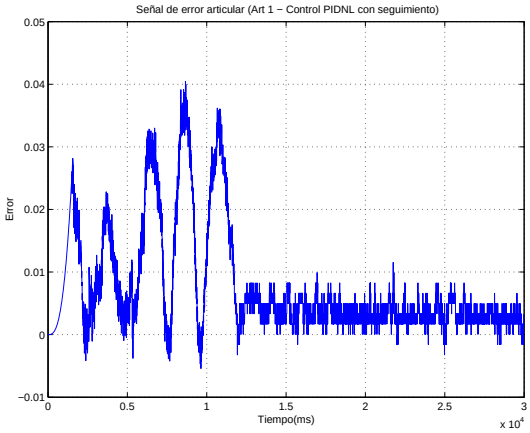

Figura 6.71: Error articular q_1 (control PIDNL con TBG).

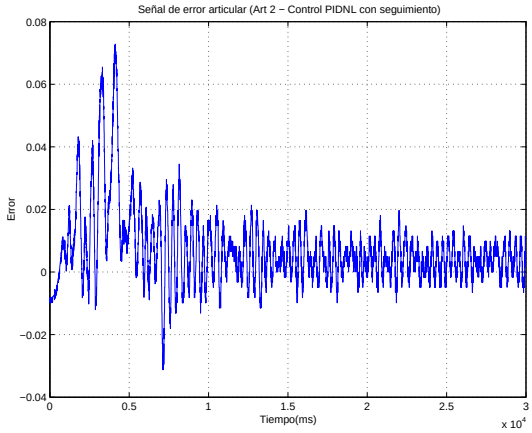

Figura 6.72: Error articular q_2 (control PIDNL con TBG).

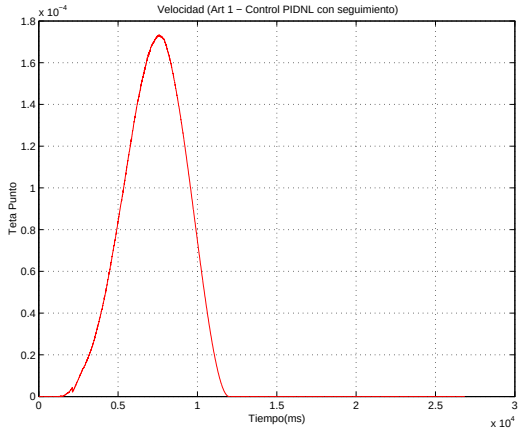

Figura 6.73: Velocidad articular q_1 (control PIDNL con TBG).

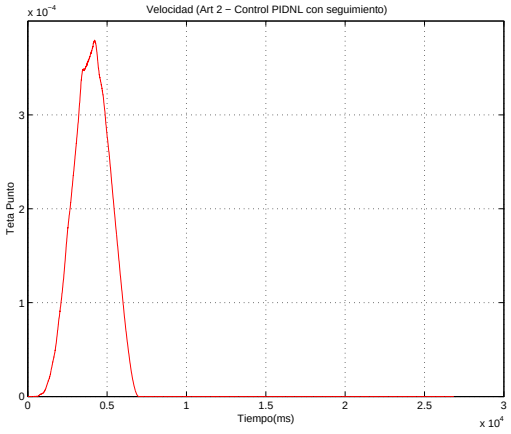

Figura 6.74: Velocidad articular q_2 (control PIDNL con TBG).

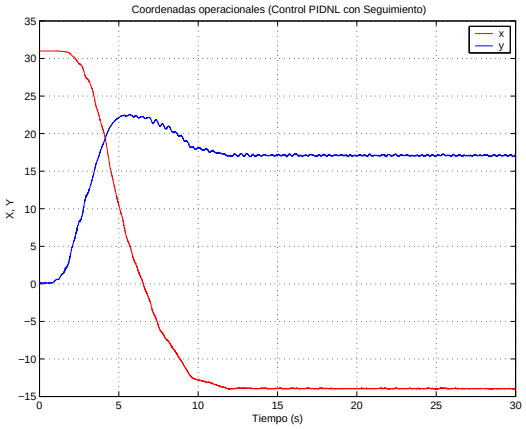

Figura 6.75: Coordenadas operacionales x y y (control PIDNL con TBG).

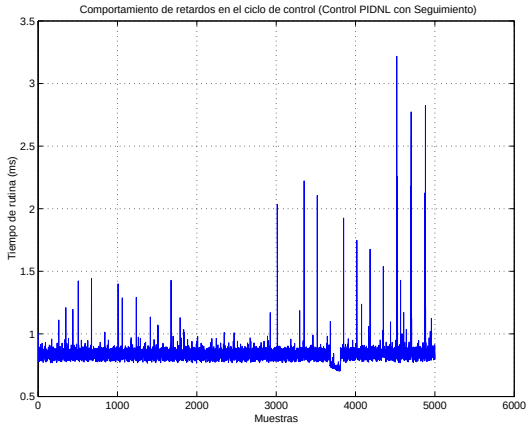

Figura 6.76: Retardos en el ciclo de control (control PIDNL con TBG).

6.3.6. Seguimiento de trayectorias articulares: sinusoidal

El robot R2SG sigue una trayectoria definida continua sinusoidal utilizando un controlador PIDNL.

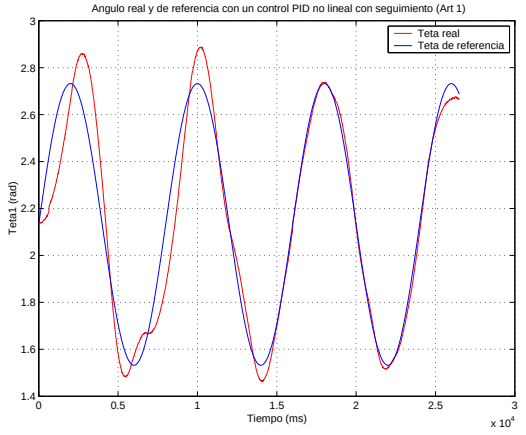

Figura 6.77: Coordenada generalizada q_1 real y deseada (trayectoria sinusoidal articular).

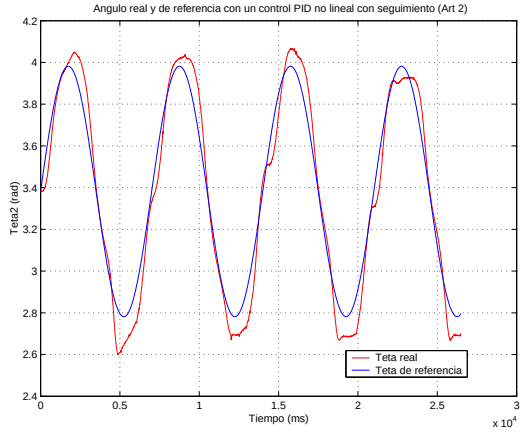

Figura 6.78: Coordenada generalizada q_2 real y deseada (trayectoria sinusoidal articular).

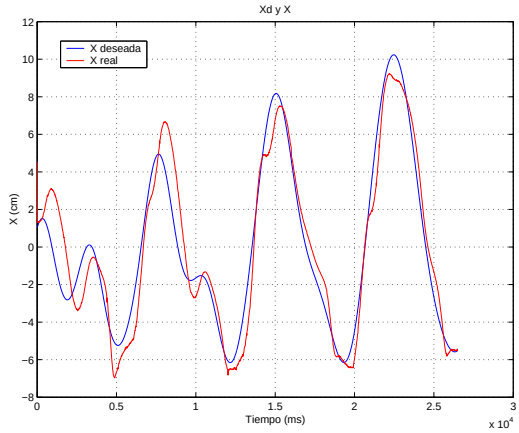

Figura 6.79: Coordenada operacional X real y deseada (trayectoria sinusoidal articular).

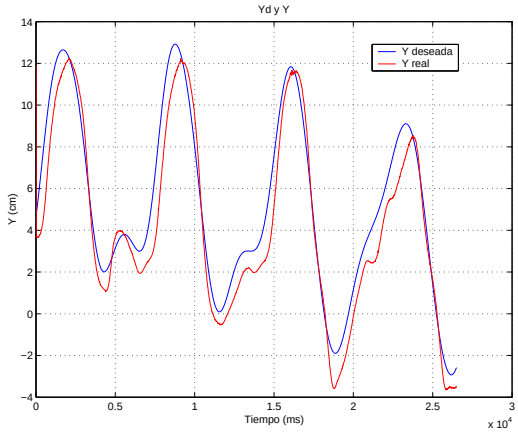

Figura 6.80: Coordenada operacional Y real y deseada (trayectoria sinusoidal articular).

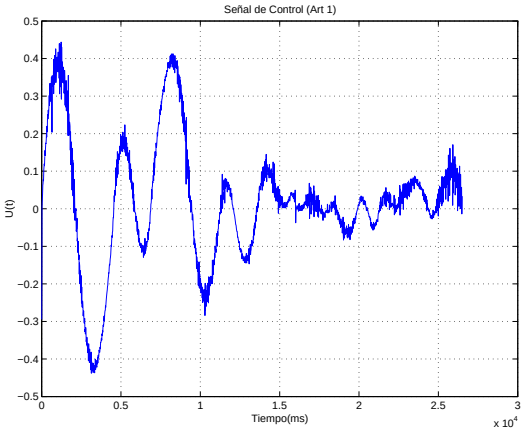

Figura 6.81: Señal de control q_1 (trayectoria sinusoidal articular).

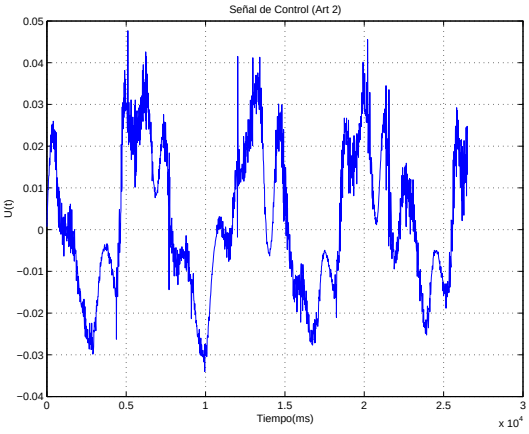

Figura 6.82: Señal de control q_2 (trayectoria sinusoidal articular).

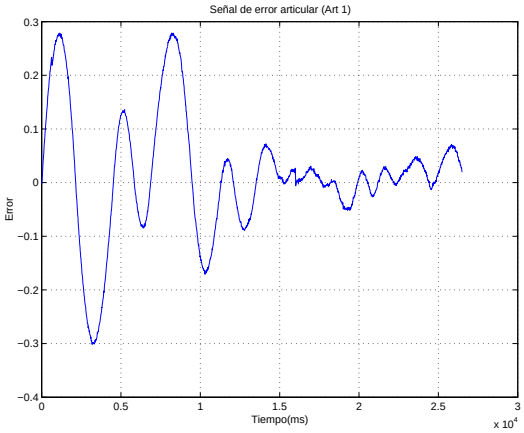

Figura 6.83: Error articular q_1 (trayectoria sinusoidal articular).

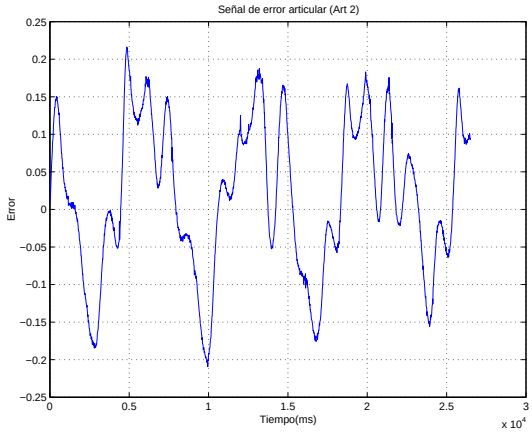

Figura 6.84: Error articular q_2 (trayectoria sinusoidal articular).

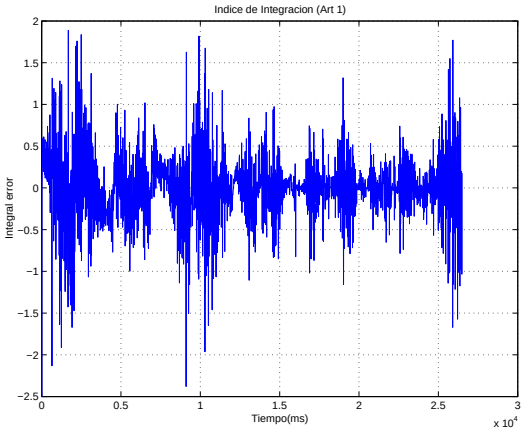

Figura 6.85: Índice de integración q_1 (trayectoria sinusoidal articular).

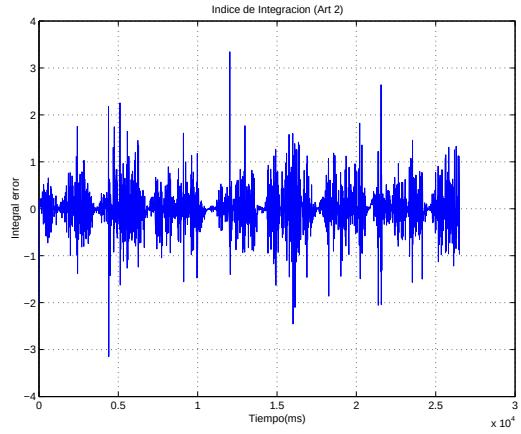

Figura 6.86: Índice de integración q_2 (trayectoria sinusoidal articular).

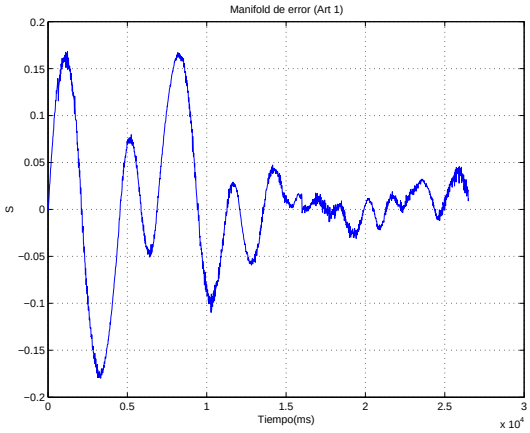

Figura 6.87: Manyfold de error q_1 (trayectoria sinusoidal articular).

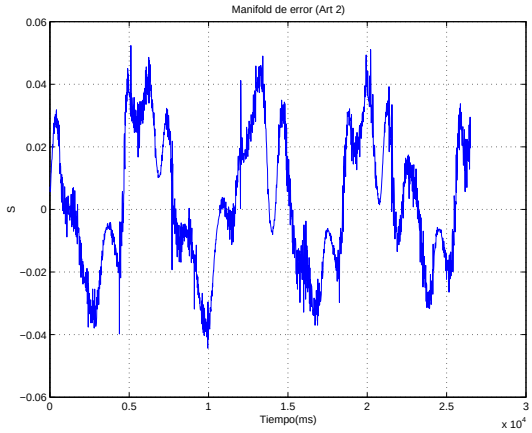

Figura 6.88: Manyfold de error q_2 (trayectoria sinusoidal articular).

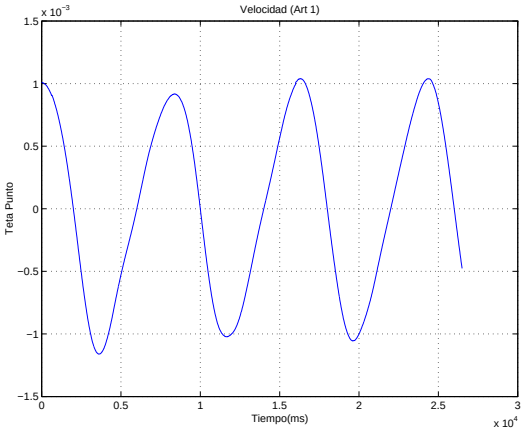

Figura 6.89: Velocidad articular q_1 (trayectoria sinusoidal articular).

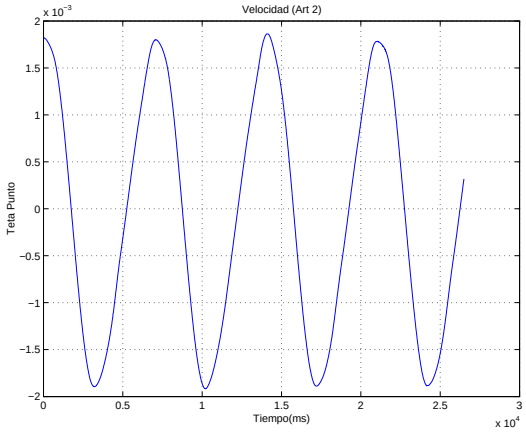

Figura 6.90: Velocidad articular q_2 (trayectoria sinusoidal articular).

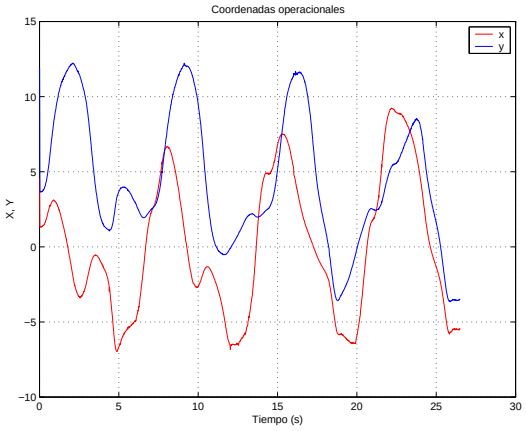

Figura 6.91: Coordenadas operacionales (trayectoria sinusoidal articular).

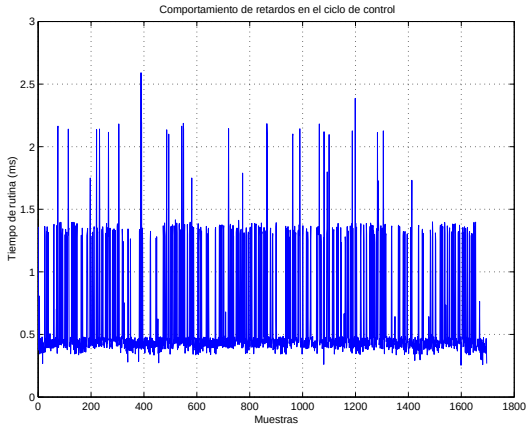

Figura 6.92: Retardos en el ciclo de control (trayectoria sinusoidal articular).

6.3.7. Seguimiento de una trayectoria cerrada operacional: circunferencia

El robot R2SG sigue una trayectoria cerrada, esta es una circunferencia utilizando un controlador PIDNL, para ello se requirió del MCIV.

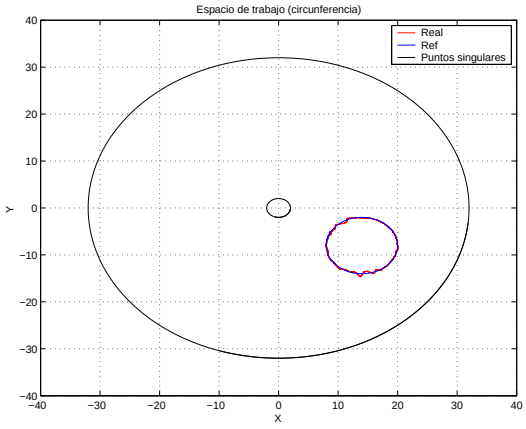

Figura 6.93: Circunferencia real y deseada en el espacio operacional.

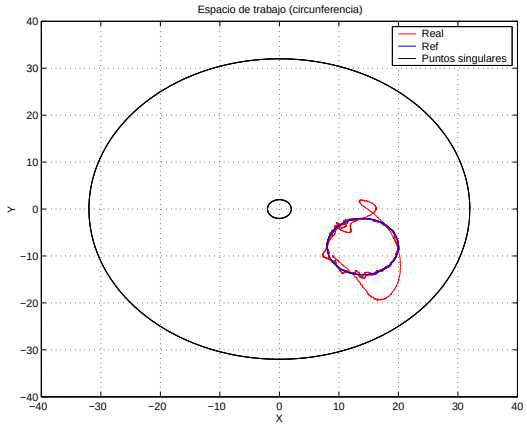

Figura 6.94: Circunferencia real y deseada completa en el espacio operacional.

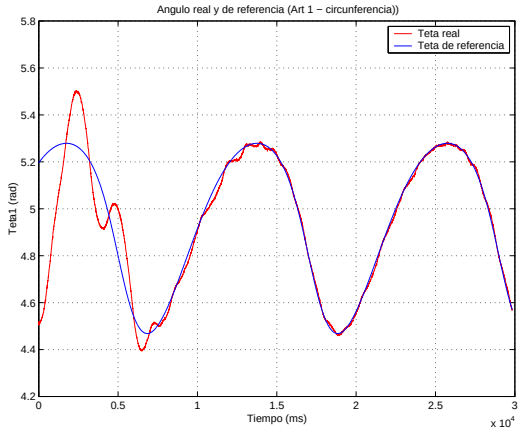

Figura 6.95: Coordenada generalizada q_1 real y deseada (circunferencia).

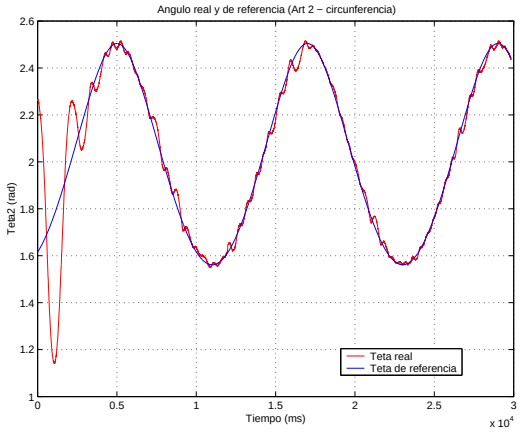

Figura 6.96: Coordenada generalizada q_2 real y deseada (circunferencia).

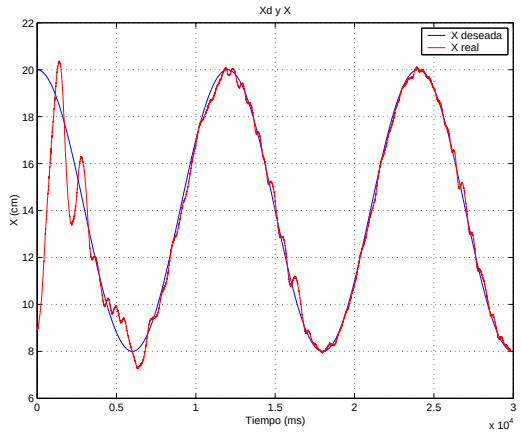

Figura 6.97: Coordenada operacional X real y deseada (circunferencia).

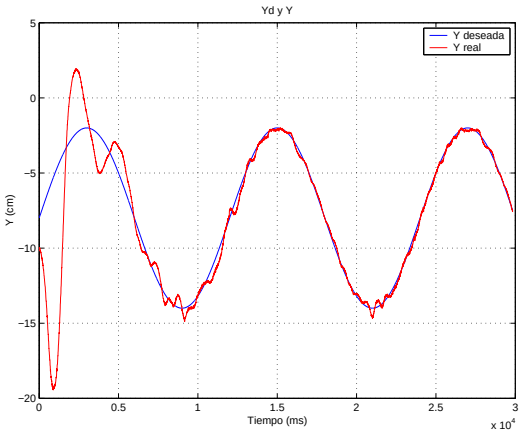

Figura 6.98: Coordenada operacional Y real y deseada (circunferencia).

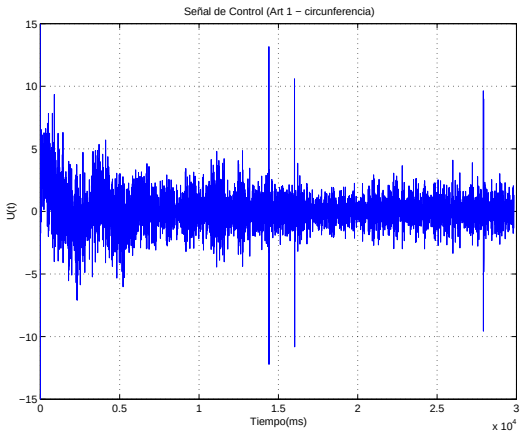

Figura 6.99: Señal de control q_1 (circunferencia).

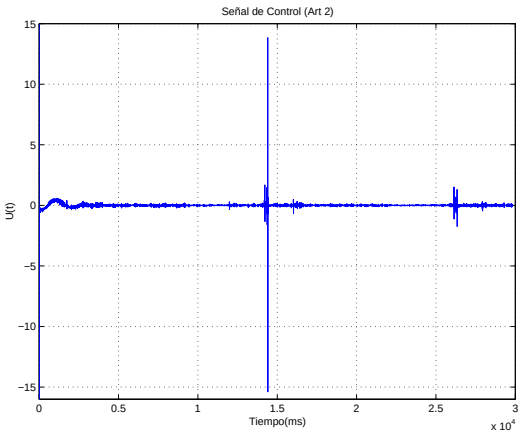

Figura 6.100: Señal de control q_2 (circunferencia).

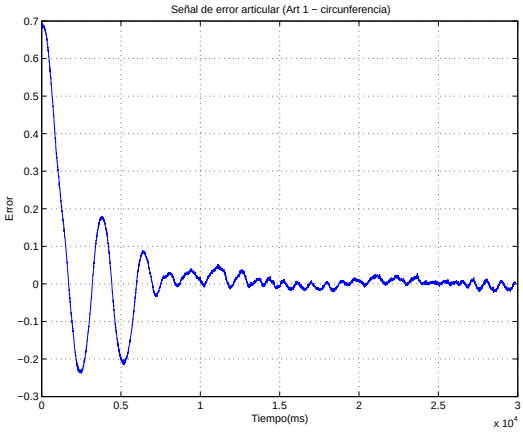

Figura 6.101: Error articular q_1 (circunferencia).

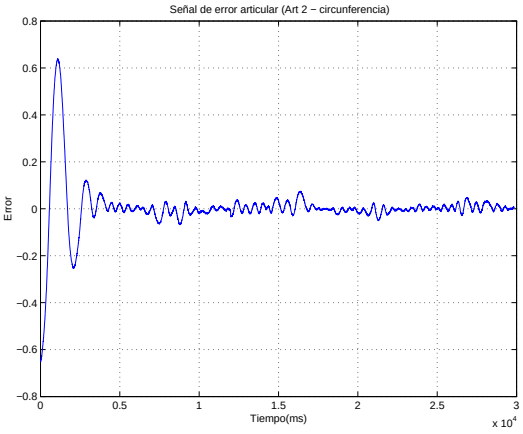

Figura 6.102: Error articular q_2 (circunferencia).

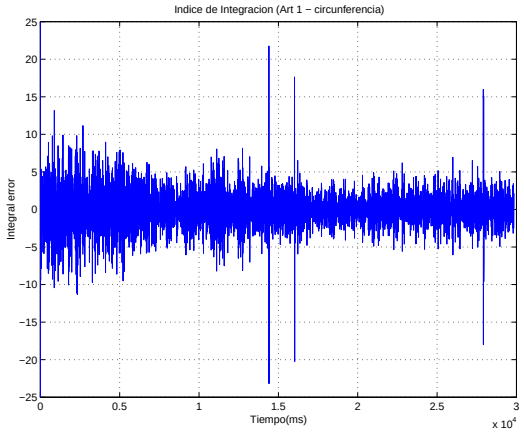

Figura 6.103: Índice de integración q_1 (circunferencia).

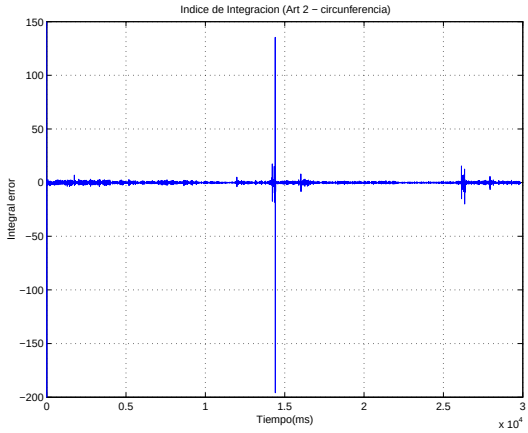

Figura 6.104: Índice de integración q_2 (circunferencia).

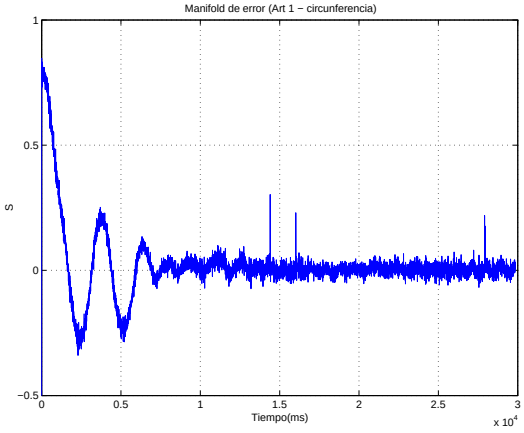

Figura 6.105: Manyfold de error q_1 (circunferencia).

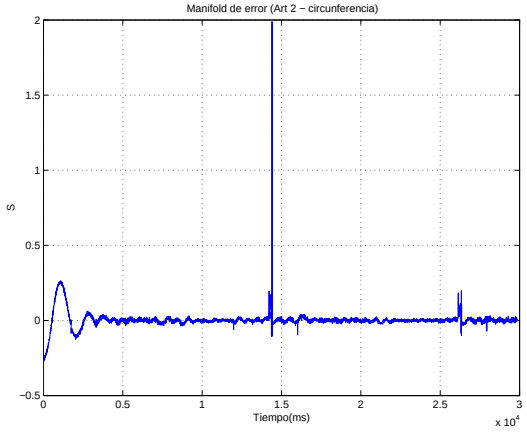

Figura 6.106: Manyfold de error q_2 (circunferencia).

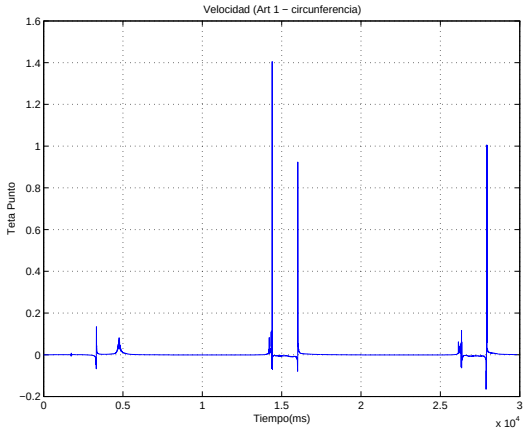

Figura 6.107: Velocidad articular q_1 (circunferencia).

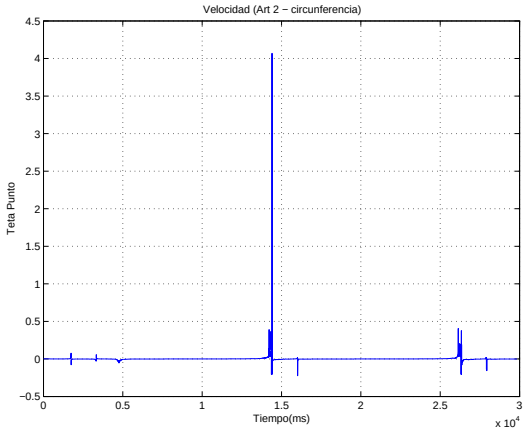

Figura 6.108: Velocidad articular q_2 (circunferencia).

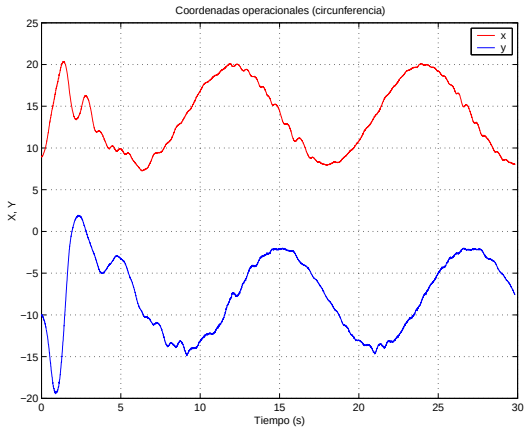

Figura 6.109: Coordenadas operacionales (circunferencia).

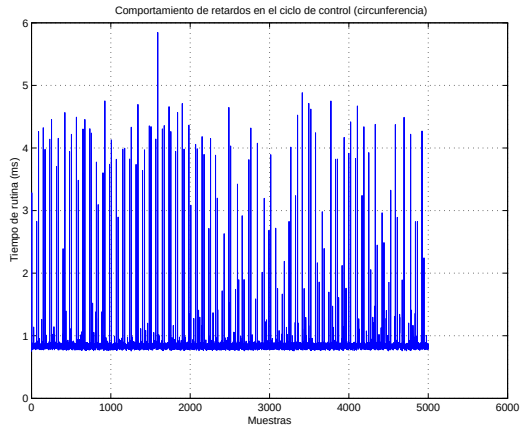

Figura 6.110: Retardos en el ciclo de control (circunferencia).

6.4. Comentarios de los experimentos

6.4.1. Control PD (regulación)

Este experimento consiste en planificar a una coordenada articular de referencia para ambas articulaciones del robot R2SG, este control lineal es diseñado para estabilizar a las variables articulares reales en una referencia admisible, como es el caso de $q_1 = 90^o$ y $q_2 = 90^o$. La gráfica de la Figura 6.9 muestra el espacio operacional y la trayectoria real del robot R2SG. Podemos apreciar en las Figuras 6.10 y 6.11 el desempeño real de las coordenadas generalizadas q_1 y q_2 con relación a las de referencia, es posible apreciar el error de estado estable, y el efecto transitorio en 2 segundos para la articulación 1 y 0.8 segundos para la articulación 2, influenciadas por las no linealidades del robot. Las Figuras 6.12 y 6.13 corresponden al desempeño del robot en el espacio de trabajo, es decir, las correspondientes coordenadas operacionales del efector final o extremo final del robot R2SG, en las gráficas de las Figuras 6.14 y 6.15 se aprecia el desempeño del control articular, amortiguándose a medida que se tiende a las coordenadas operacionales de referencia y cuyos cambios de amplitud dependen de la condición inicial y de la dinámica no lineal del robot. Vale la pena comentar que la articulación 1 tiene un error casi cero, sin embargo para la articulación 2 existe un error de estado estable. El comportamiento de la velocidad articular deja apreciar altas frecuencias debido a que esta es estimada a partir de la posición articular instantánea y del tiempo de muestreo en el ciclo de control. La Figura 6.21 representa la gráfica de los retardos en los ciclos de control extendiéndose hasta un máximo de 1.8 milisegundos. Podemos concluir que este control PD, a pesar de ser muy utilizado en la industria por su fácil implementación, no es el que resuelva de mejor forma el problema de regulación, es decir cuando un robot lleva a un objeto de una región de sus espacio de trabajo a otro, ya que carece de repetibilidad y precisión, así como baja influencia sobre la dinámica inercial y de fricción. Es importante mencionar que el control PD evaluado en un robot planar de dos grados de libertad sin efecto gravitatorio actúa como un PD plus (Arimoto y Takegaki) que implica un control PD con compensación gravitatoria, es decir, ofrece estabilidad asintótica local, es por ello que se aprecian experimentos relativamente eficientes de este control.

6.4.2. Control PID (regulación)

En relación a los experimentos del control PD, el control PID ofrece un error de estado estable menor, debido a la acción de integración durante el régimen permanente, sin embargo y debido al desempeño planar (sin gravedad) se apreció muy poca diferencia dado que existe estabilidad asintótica local como en los experimentos del PD. Podemos apreciar un error articular de estado estable sin embargo este puede reducirse con la resolución de la retroalimentación de posición articular (Figuras 6.23 y 6.24). Con el uso del modelo cinemático directo de posición podemos apreciar el desempeño operacional del robot y su relación con las referencias operacionales. Las Figuras 6.27 y 6.28 corresponden al índice de integración e implica los instantes en el tiempo en que el controlador corrige el error de estado estable debido a su acción integral. Podemos apreciar que en las gráficas de control (Figuras 6.29 y 6.30) con relación a las del control PD, implican menores frecuencias de menor amplitud en régimen estable, debido a la acción del integrador situación que beneficia a los servomotores del robot y evita saturación y calentamiento de la fuente de voltaje de alimentación. Las Figuras 6.31 y 6.32 corresponden al error articular y comparativamente con los correspondientes al control PD se aprecia el beneficio en estado estable. Las respuestas de velocidad articular son estimadas a partir de posición y con desplazamientos relativamente lentos. Podemos apreciar en la gráfica de la Figura 6.36 el desempeño de los retardos, influenciados por el sistema multitareas de windows y la demanda del controlador bajo la acción integral, situación que permite asegurar un mejor desempeño de control en plataformas experimentales de tiempo

real.

6.4.3. Control PID no lineal (regulación)

El control PID no lineal conocido en la literatura como control por modos deslizantes de segundo orden, como su nombre lo indica tiene la capacidad de compensar las dinámicas inherentes al robot, situación que no es posible ejecutar con las estrategias de control clásicas reportadas experimentalmente en las secciones anteriores. La Figura 6.37 representa el espacio de trabajo del robot R2SG y la trayectoria operacional ejcutada para llevar al robot a una referencia articular $q_1 = q_2 = 90^o$ a partir de una condición inicial de $q_1 = q_2 = 0^o$. Las gráficas que corroboran la regulación a estas referencias son las Figuras 6.38 y 6.39, en ellas podemos verificar el buen desempeño para la coordenada articular q_1 y q_2, sin embargo es apreciable un error de estado estable a una frecuencia de estado estable, cuya razón es la estimación de la velocidad a partir de la posición instantánea. Las gráficas de las Figuras 6.40 y 6.41 corresponden a las coordenadas operacionales instantáneas, equivalentes en el extremo final del robot, y que son definidas mediante el modelo cinemático directo de posición del robot R2SG, transmitiendo las respuesta en estado estable ocurrida en el espacio articular. Las gráficas de las Figuras 6.42 y 6.43 representan a los errores articulares de posición y velocidad simultáneos, tal que el control logra convergencia en ambos casos, integrados mediante un manifold de error. Las Figuras 6.44 y 6.45 corresponden a las respuestas del índice de integración de la función de saturación, en cada controlador articular, tal que representa la actividad no lineal por compensar el error dinámico en estado estable. Las gráficas de las Figuras 6.46 y 6.47 exhiben la respuesta de las señales de control en ambas articulaciones, es posble apreciar en ellas la frecuencia instantánea debido a la estimación de la velocidad a partir de posición y a la integración de la función de saturación. Las Figuras 6.48, 6.49, 6.50 y 6.51 corresponden a las señales de error de posición y velocidad articular, relacionados en el manifold de error o error extendido descrito en las Figuras 6.42 y 6.43. La Figura 6.52 corresponde a las coordenadas operacionales en régimen del tiempo y que describen la tarea en el espacio de trabajo, y finalmente la gráfica de la Figura 6.53 es relativa al comportamiento de los retardos en el ciclo de control, señal que sugiere la operación del sistema en tiempo real suave, como la prestación que Windows ofrece, traduciendose como limitaciones en la plataforma experimental, se aprecian picos de hasta 1.2 milisegundos, cuando el tiempo de muestreo buscado a partir del diseño es a un milisegundo.

6.4.4. Estudio comparativo (PD, PID y PIDNL)

Haciendo una comparación entre los controles clásicos PD y PID, y el control moderno PID no lineal o control por modos deslizantes de segundo orden, podemos apreciar que para la coordenada generalizada q_1 y q_2, que se muestran en las gráficas de las Figuras 6.54 y 6.55, el control PIDNL se estabiliza con mayor rapidez que los controles clásicos, esto ocurre en 16.5 segundos para q_1 y 3.7 segundos para q_2, con las mismas condiciones inciales para las variables articulares, la diferencia entre estas dos se debe a la fuerza inercial existente debido al peso que soporta cada una de las articulaciones. Otra de las cosas que se logra observar en estas gráficas es que en los controles clásicos no existe tanta diferencia en el factor como la que se aprecia con el control PID no lineal ya que ésta es más pronunciada que la de los controles clásicos como se muestra en las gráficas de las Figuras 6.56 y 6.57, una de las razones es en la sintonización empleada directamente por la frecuencia natural Wn para cada control. En la Figura 6.58 se muestran las trayetorias de los controles en el espacio de trabajo operacional en donde podemos apreciar un mejor desempeño del control PID no lineal con relación a los controles clásicos. Cabe hacer mención que los beneficios de un control PID clásico en el control de movimiento de un robot se aprecian más cuando existe la presencia del efecto gravitatorio, no siendo el caso de estudio. La influencia de la

dinámica inercial y de fricción, con la ausencia del efecto gravitatorio contribuyen a la estabilidad en lazo cerrado. Por último, en la Figura 6.59 se puede observar que el mayor consumo de recursos del sistema es del control PD seguido por el PID y como menor consumidor el control PID no lineal, debido a los retardos en el ciclo de control para cada uno de los controles, obteniendo retardos como máximo del control PD de aproximadamente 1.8 milisegundos, el control PID con 1.7 milisegundos y finalmente el control PID no lineal con 1.24 milisegundos. Se puede concluir que el control PID no lineal supera a los controles clásicos que suelen ser más utilizados en la industria, dado que logra estabilidad asintótica global, compensando la dinámica inherente al robot y requiere de menos recursos del sistema verificados en el estudio experimental de retardos de tiempo.

6.4.5. Seguimiento de trayectorias articulares: spline basada en un polinomio de 5º

Este experimento se realizó con el propósito de proponer la regulación basada en seguimiento, es decir, el empleo de una spline definida por un polinomio de quinto grado tal que sea posible vencer los efectos inerciales por el estado de reposo y de movimiento del robot R2SG, y simultáneamente lograr la convergencia en tiempo finito. La Figura 6.60 muestra la trayectoria real y de referencia del control PID no lineal con seguimiento de trayectoria en el espacio de trabajo operacional del robot R2SG, en el cual se aprecia un margen mínimo de error entre la real y la de referencia. Esta propuesta de llevar al robot a una posición a través de una trayectoria de posición y velocidad suaves y controladas permite aprovechar los beneficios del control PID no lineal utilizado, sobre los controles clásicos PD y PID, por lo que los resultados incluidos en este reporte están basados exclusivamente en el PID no lineal. La gráfica de la Figura 6.60 representa el desempeño en el espacio de trabajo, y comparando con el desempeño experimental en regulación de la Figura 6.58 de los tres controladores apreciamos significativamente el beneficio de emplear la regulación a partir de seguimiento, en el que la influencia inercial desaparece debido a que la velocidad inicial y final son cero. Las Figuras 6.61 y 6.62 permiten apreciar el comportamiento de las variables articulares real y deseadas utilizando el concepto de regulación a partir de seguimiento, para la variable articular q_2 se aprecia la existencia de un chattering en estado estable situación inherente a este tipo de control y a la dinámica relativamente inestable en el segundo eslabón debido a la baja inercia y fricción. Las Figuras 6.63 y 6.64 corresponden a las señales operacionales del extremo final del robot empleando el modelo cinemático directo de posición. Las Figuras 6.65 y 6.66 corresponden a el comportamiento articular del error extendido que involucra de manera simultánea al error de posición y velocidad verificados de manera independiente en las gráficas de las Figuras 6.71, 6.72. Las gráficas de las Figuras 6.73 y 6.74 muestran la trayectoria de velocidad en forma de campana representada de forma ideal mediante la ecuación (4.53) y que se observa en la gráfica de la Figura 4.47. Las respuestas de la señales de control, verificadas en las Figuras 6.69 y 6.70 necesarias para lograr la tarea planificada, son de baja amplitud y relativa alta frecuencia, cuyo ancho de banda no excede al ancho de banda de la plataforma experimental empleada. La Figura 6.75 corresponde a las coordenadas operacionales del robot y finalmente los retardos en el ciclo de control verificados en la Figura 6.76, con un retardo máximo de 3.3 milisegundos, comparativamente con los experimentos de los controles clásicos para regulación basada en seguimiento no reportados en esta tesis, son considerablemente menores.

6.4.6. Seguimiento de trayectorias articulares: sinusoidal

En este experimento se establece como trayectoria de referencia a una señal sinusoidal para ambas articulaciones, con el propósito de evaluar funciones trascendentes que permitan establecer trayectorias cerradas en el espacio de trabajo. En las Figuras 6.77 y 6.78 se prsentan las trayectorias reales y de referencia, se puede apreciar que a medida que el experimento transcurre, se obtiene más convergencia

articular. En las Figuras 6.79 y 6.80 se aprecia el desempeño operacional definido a partir del modelo cinemático directo de posición, es decir las coordenadas operacionales del extremo final del robot. en las Figuras 6.81 y 6.82 se aprecia la señal de control con una frecuencia instantánea debido a que el control busca estabilizar al robot en todos los valores instantáneos de la trayectoria de referencia, sin embargo y en términos del ancho de banda de la plataforma experimental, esta frecuencia es despreciable. Las Figuras 6.83 y 6.84 corresponden a los errores articulares instantáneos, cuyo comportamiento es definido por el factor de forma de la trayectoria de referencia y su baja amplitud por la eficiencia del control no lineal empleado. La operación del integrador Runge-Kutta de la función de saturación que define la compensación no lineal de la dinámica del robot, es apreciada mediante el índice de integración en las Figuras 6.85 y 6.86, acción que repreenta la corrección de estado estable. El error extendido que permite visualizar la eficiencia del control en convergencia de posición y velocidad simultáneos son definidos por cada una de las variables articulares en las Figuras 6.87 y 6.88. Las Figuras 6.89 y 6.90 corresponden a las velocidades articulares, cuyas referencias están definidas por la primera derivada temporal de las rayectorias de posición articular. La Figura 6.91 representa a las coordenadas operacinales del extremo final del robot, para ello se empleo el modelo cinemático directo de posición. Finalmente la Figura 6.92 corresponde al comportamiento de los retardos sobre la plataforma de Windows, teniendo como máximo retardo 2.6 milisegundos.

6.4.7. Seguimiento de una trayectoria cerrada operacional: circunferencia

Este experimento consolida a la plataforma experimental, ya que el seguimiento de una trayectoria cerrada en el espacio de trabajo u operacional implica precisión articular, certeza en modelos matemáticos y ancho de banda en el sistema. En este caso se planifica una circunferencia en el espacio de trabajo, a partir de sus ecuaciones paramétricas, cuya configuración fue descrita anteriormente. La Figura 6.93 representa la rayectoria real y deseada en el espacio de trabajo cuando el robot R2SG logra la convergencia, sin embargo y desde la condición inicial, la rayectoria real sobre la referencia es definida en la Figura 6.94. En las Figuras 6.95 y 6.96 se presenta el seguimiento articular real y deseado definido mediante la cinemática inversa de posición, las Figuras 6.97 y 6.98 representa la respuesta de las coordenadas operacionales real y deseada definidas por las ecuaciones paramétricas de la circunferencia. Las Figuras 6.99 y 6.100 corresponde a la respuesta de la señal de control que describe el PID no lineal en lazo cerrado con la planta y una trayectoria cerrada planificada en el espacio de trabajo, las frecuencias y amplitudes son toleradas por el ancho de banda de la plataforma experimental. Los errores articulares instantáneos son presentados en las Figuras 6.101 y 6.102, es posible apreciar que durante la convergencia a la circunferencia los errores son cercanos a cero, es decir hay convergencia asintótica. La influencia de la integración no lineal instantánea repercute en la señal de control, esto puede apreciarse en las Figuras 6.103 y 6.104, cuyos comportmientos intentan la corrección del error de estado estable. En las Figuras 6.105 y 6.106 se aprecian los errores extendidos articulares o manifolds de error, que representan la convergencia del error de posición y velocidad articular a cero. En las Figuras 6.107 y 6.108 se dan a conocer las velocidades articulares pequeñas y casi constantes. La Figura 6.109 corresponde a la trayectoria operacional del robot R2SG, definida por la señal real de las articulaciones sustituida en el modelo cinemático diecto de posición, permiten apreciar la forma de onda de las ecuaciones paramétricas de la circunferencia real trazada por el robot durante el experimento. El retardo promedio en el ciclo de control esta en 4.8 milisegundos, esto debido a que el area exige operación en tiempo real, sin embargo los resultados obtenidos aún en una plataforma como Windows son satisfactorios, ver Figura 6.110.

6.5. Conclusiones

En este capítulo fueron reportados los experimentos realizados con el robot R2SG utilizando estrategias de control lineales como el PD y el PID, y controladores no lineales como el control por modos deslizantes de segundo orden o PID no lineal. Realizandose los estudios comparativos en cuanto al error articular, al error operacional, a la ley de control, a los retardos en el ciclo de control, etc. Se dan a conocer las configuraciones de cada experimento como las condiciones iniciales de cada uno, los valores de las constantes utilizadas y la visualización virtual inicial y final del robot R2SG. De igual forma se reportan las gráficas de todos los experimentos realizados que posteriormente son comentadas. Se planificó una trayectoria sinusoidal articular, y una circunferencia en el espacio operacional o de trabajo empleando la cinemática directa e inversa de posición y velocidad. En estas últimas tareas planficadas se hicieron estudios comparativos de los controladores considerados en la regulación con los mismos tópicos de comparación mencionados anteriormente.

Capítulo 7

Conclusiones y perspectivas

7.1. Conclusiones

Este trabajo de tesis tuvo como propósito esencial el de constituir una plataforma experimental robótica a partir de un diseño fundamentado en el modelo matemático del robot (modelo cinemático y dinámico). El diseño mecánico estuvo basado en un robot planar de dos grados de libertad, sin efecto gravitatorio, para ello se utilizaron materiales como aluminio para los eslabones, y aceros en articulaciones y compensadores dinámicos. Se realizaron diversas pruebas mecánicas para constituir la versión final del robot, esto debido a fenómenos inherentes al mecanismo de eslabones articulados como los esfuerzos cortantes y torsionales en articulaciones y eslabones respectivamente, así como los relativos a los momentos inerciales. Se realizaron estudios en simulación digital de las propiedades cinemáticas y dinámicas, utilizando MATLAB 6.5, y se corroboraron los resultados de manera experimental, describiendo movimientos aleatorios dentro del espacio de trabajo del robot evaluandose de manera instantánea los modelos cinemáticos directo e inverso de posición, velocidad y aceleración, así como la manipulabilidad de Yoshikawa definida explícitamente de la matriz Jacobiana del robot, y que define su operación dentro del espacio de trabajo y su tendencia a operar en regiones cercanas a las singulares; se evaluaron las propiedades dinámicas del robot para validar a la matriz de fuerzas inerciales, y a la matriz de fuerzas de Coriolis y centrípetas, adicionalmente se cálculo en tiempo real suave la manipulabilidad dinámica del robot encontrando resultados interesantes al definir las regiones singulares existentes en el mismo mecanismo de eslabones articulados cercanas al eslabón fijo de la cadena. Esto último define perspectivas de investigación y desarrollo en supervisión de robots y replanificación de trayectorias tal que motiven un desempeño de mínimo esfuerzo. Estos resultados motivaron a la aplicación de las primeras leyes de control de movimiento clásicas, con el propósito de cerrar el lazo con los codificadores ópticos acoplados mecánicamente a los actuadores del robot. Los primeros resultados fueron de regulación a una coordenada operacional en el espacio de trabajo realizándose estudios comparativos entre los controles PD y PID, posteriormente se propuso llevar al robot a una coordenada operacional de una manera suave, es decir que no implique señales de control con alta frecuencia y amplitud, para ello se propone la regulación a partir de seguimiento de trayectorias descrita por una spline definida por un polinomio de quinto grado, tal que venza a los esfuerzos inerciales debido al estado de reposo durante la condición inicial, y de movimiento durante la condición final; y adicionalmente definir un comportamiento de velocidad simétrico y suave descrito por la primera derivada temporal de la trayectoria propuesta. Los experimentos resultaron convincentes tal que al comparar con los trabajos realizados de regulación sin seguimiento, las señales de control se suavizaron en amplitud y frecuencia de manera importante lo que constituye una mejor operación de la plataforma. Esta tarea proporciona una bondad adicional que es

la convergencia en tiempo finito a una coordenada operacional, cuyos límites inferiores están descritos por el ancho de banda del sistema. A partir de estos resultados se aprecian las limitantes de los controles clásicos, debido a que estos operan en regiones lineales, y un robot manipulador es un sistema electromecánico altamente no lineal cuyo modelo es apreciado por la formulación de Euler-Lagrange, por tal razón se propuso diseñar y evaluar experimentalmente a un control no lineal basado en modos deslizantes de segundo orden, conocido también en la literatura como control PID No Lineal obteniendo resultados relevantes en régimen transitorio y estacionario de las variables articulares del robot. El resultado final de este trabajo de investigación y desarrollo es el de planificar una trayectoria cerrada en el espacio operacional definida por una circunferencia, utilizando la cinemática inversa de posición y velocidad para el mapeo de las variables de control al espacio articular, debido a que el control propuesto es articular. Los resultados de seguimiento articular y operacional permiten validar de manera convincente los beneficios de compensar mediante un control no lineal a la dinámica del robot. Cabe hacer mención que otras contribuciones importantes de este trabajo fue el desarrollo de la interfaz electrónica de potencia y el acondicionamiento de señales para lograr la correspondencia tecnológica entre la tarjeta de adquisición de datos y control con los servomotores empleados. Finalmente todos los experimentos fueron desarrollados con el auxilio de una interfaz de usuario diseñada para este propósito y que es parte de la integración de esta plataforma experimental, esta interfaz permite la visualización virtual del robot en línea y sintonizar las ganancias del controlador, así como tiempo de convergencia antes de cada experimento. Este trabajo define perspectivas en el desarrollo de aplicaciones mecatrónicas de distinta índole, planteadas en la siguiente sección.

7.2. Perspectivas

Este trabajo de investigación y desarrollo, a través de su arquitectura abierta, permite realizar investigación aplicada en:

. Control de robots manipuladores

. Interfaces hápticas

. Realidad virtual aumentada

. Interacción hombre-máquina

Algunos trabajos realizados con esta plataforma, y que van encaminados a los rubros mencionados anteriormente, son el desarrollo de una interfaz háptica activa para interacción con ambientes virtuales dinámicos, de éste se desprenden aplicaciones de guiado háptico local con propósitos de diagnóstico y rehabilitación de pacientes con discapacidades motrices como consecuencia de enfermedades neurológicas y neuropsicológicas, proyecto que se desarrolla de manera conjunta con la Facultad de Neurociencias de la Universidad Nacional Autónoma de México y el Instituto Nacional de Rehabilitación. Otro trabajo relativo al uso de esta plataforma experimental es en telepresencia y teleoperación de sistemas electromecánicos, y finalmente en el diseño de estrategias de control avanzado de robots manipuladores.

Acrónimos, unidades y glosario

Lista de acrónimos

2D : Dos dimensiones.

3D : Tres dimensiones.

CAD : Convertidor de analógico a digital.

cd : Corriente directa.

CDA : Convertidor de digital a analógico.

CI : Circuito integrado.

CMOS : Semiconductor complementario de óxido metálico.

E-L : Euler - Lagrange.

E/S : Entrada-salida.

EPROM : Memoria borrable y programable de solo lectura.

EEPROM : Memoria eléctricamente borrable y programable de solo lectura.

f : Frecuencia.

IM : Índice de manipulabilidad.

LED : Diodo emisor de luz.

J : Matriz Jacobiana.

K : Energía cinética.

L : Lagrangiano.

MCDP : Modelo cinemático directo de posición.

MCIP : Modelo cinemático inverso de posición.

MCDV : Modelo cinemático directo de velocidad.

MCIV : Modelo cinemático inverso de velocidad.

MCDA : Modelo cinemático directo de aceleración.

MCIA : Modelo cinemático inverso de aceleración.

M : Masa o inercia.

MSI : Integración a media escala.

N-E : Newton - Euler.

PDH : Parámetros Denavit-Hartenberg.

P : Energía potencial.

PC : Computadora personal.

PD : Control proporcional-derivativo.

PIC : Circuito integrado programable.

PID : Control proporcional-integral-derivativo.

PIDNL : Control proporcional-integral-derivativo no lineal.

PIDNLST : Control proporcional-integral-derivativo no lineal con seguimiento de trayectoria.

PWM : Modulación por ancho de pulso.

R : Robot.

R2SG : Robot de 2 grados de libertad sin término de gravedad.

RAM : Memoria de acceso aleatorio.

RISC : Computadoras de juego de instruciones reducido.

R-K : Runge-Kutta.

ROM : Memoria de solo lectura.

rpm : Revoluciones por minuto.

RV : Realidad virtual.

SPI : Interfaz periférica serial.

T : Periodo.

t : Tiempo.

TAD : Tarjeta de adquisición de datos.

tb : Tiempo de convergencia.

TBG : Generador de tiempo base.

TTL : Lógica de transistor-transistor.

uC : Microcontrolador.

uP : Microprocesador.

μ**s** : Microsegundos.

UCP : Unidad central de proceso.

V : Velocidad de desplazamiento.

w : Velocidad angular.

Wn : Frecuencia natural.

Lista de unidades

C : Grados centígrados.

A : Amperios.

cm : Centímetros.

Hz : Hertz.

kΩ : Kiloohms.

KHz : Kilohertz.

Kg : Kilogramo.

m : Metro.

Mb : Megabytes.

MHz : Megahertz.

mm : Milímetros.

ms : Milisegundos.

ns : Nanosegundos.

rad : Radian.

s : Segundos.

v : Volts.

Glosario

Aceleración angular: Cantidad que expresa el cambio de la velocidad angular con respecto al tiempo.

Aceleración articular: Cantidad que expresa el cambio de la velocidad articular con respecto al tiempo.

Aceleración operacional: Cantidad que expresa el cambio de la velocidad de cualquier coordenada x, y ó z con respecto al tiempo.

Actuador: Es el dispositivo que proporciona la fuerza motriz real para las articulaciones de un robot. El actuador suele obtener su energía a partir de una de estas tres fuentes: aire comprimido, fluido por presión o electricidad; por lo que reciben el nombre de actuadores neumáticos, hidráulicos o eléctricos.

Los actuadores tales como los reles, solenoides y motores, posibilitan que la computadora, controle la operacion de los sistemas del vehiculo.

Articulación: Una articulación es la conexión que existe entre dos o más eslabones, la cual se encuentra en los nodos de los eslabones y permite algún movimiento o movimiento potencial, entre los eslabones conectados. Cada articulación provee al robot de al menos un "grado de libertad". En otras palabras, las articulaciones permiten al manipulador realizar movimientos

Cadena cinemática: Conjunto de elementos mecánicos que soportan la herramienta o útil del robot (base, armadura, muñeca, etcétera).

Centro de masa: Las fuerzas se distribuyen sobre una línea, un área o un volumen. Por lo común, no es muy difícil encontrar una resultante de estas fuerzas distribuidas. Para tener el mismo efecto esta resultante debe actuar sobre el centroide del cuerpo, el cual es un punto en el que se puede considerar que un sistema de fuerzas distribuidas está concentrado, con el mismo efecto que la fuerza distribuida.

Chattering: Señales de alta frecuencia en estado estable.

Cinemática: Parte de la mecánica, que estudia el movimiento prescindiendo de las fuerzas que lo producen.

Codificador óptico: Aquel que cuenta el número de pulsos producido por un disco óptico giratorio y relaciona ese número con la cantidad de movimiento del eje.

Control: Es el proceso de hacer que una variable o sistema de variables conforme a lo que se desea.

Controlador: Dispositivo con el que cuenta el robot, para el manejo de circuitos encargados del movimiento eléctrico.

Dinámica: Se denomina Dinámica a la parte de la Mecánica que estudia conjuntamente el movimiento y las fuerzas que lo originan.

Efecto de Coriolis: Efecto debido al movimiento rotacional de la tierra, que se manifiesta en todo cuerpo en movimiento, de tal forma que lo desvía de su trayectoria recta. En el hemisferio norte la desviación ocurre hacia la derecha de la dirección del cuerpo y mientras que en el hemisferio sur la desviación es hacia la izquierda.

Energía cinética: Es la energía que posee un cuerpo por razón de su movimiento.

Energia potencial: La que posee un cuerpo por el hecho de hallarse en un campo de fuerzas, se dice que esta en reposo.

Eslabón: El eslabón es un cuerpo rígido que posee por lo menos dos nodos, que son los puntos de unión con otros eslabones. Los eslabones se pueden nombrar de acuerdo al número de nodos que poseen, por ejemplo: Eslabón binario: Es el que contiene dos nodos. Eslabón ternario: Es el que contiene tres nodos. Eslabón cuaternario: Es el que contiene cuatro nodos.

Espacio de trabajo: Es la zona donde el robot puede posicionarse y está limitada por las dimensiones físicas del manipulador.

Fricción: La fricción aparece cuando dos superficies se frotan una en contra de la otra presentando desprendimiento de calor.

Fuerza centrífuga: Es la fuerza que hace que un cuerpo sometido a un movimiento circular uniforme tienda a alejarse del centro de rotación es de igual fuerza y magnitud que la centrípeta pero de dirección contraria.

Fuerza centrífuga: Es una fuerza hacia fuera para mantener el cuerpo en esa posición o en equilibrio.

Fuerza de coriolis: Efecto debido al movimiento rotacional de la tierra, que se manifiesta en todo cuerpo en movimiento, de tal forma que lo desvía de su trayectoria recta.

Fuerza de fricción: Fuerza aparente hacia el exterior experimentada por un cuerpo que gira alrededor de su eje

Fuerza de gravedad: Fuerza de atracción entre dos masas. La magnitud de esta fuerza es directamente proporcional al producto de ambas masas e inversamente proporcional al cuadrado de la distancia que las separa.

Grado de libertad: Son los posibles movimientos básicos independientes, ya sean giratorios o de desplazamiento, que el robot puede realizar.

Inercia: Propiedad de un cuerpo que tiende a oponerse a toda variación en su estado de reposo o de movimiento.

Jacobiano: Es la matriz que describe la relación entre las velocidades de las articulaciones y el efector final.

Jacobiano: Es una matriz que se puede ver como la versión vectorial de la derivada de una función escalar. Es importante en el análisis y control de movimiento de un robot (planificación de trayectorias suaves, determinación de configuraciones singulares, ejecución de movimientos coordinados, derivación de ecuaciones dinámicas).

Permite conocer las velocidades del extremo del robot a partir de los valores de las velocidades de cada articulación.

Límite operacional: Son las fronteras alrededor del espacio de trabajo del robot.

Longitud: Dimensión que expresa el valor de una distancia.

Manipulabilidad: Es la posibilidad de moverse libremente en todas las direcciones del espacio de trabajo.

Manipulador: Es un mecanismo que usualmente consiste en una serie de eslabones articulados o desplazamiento relativo a otro eslabón, para agarrar y mover objetos, por lo regular tiene varios grados de libertad.

Matriz: Conjunto de mxn números distribuidos en m *filas* y n *columnas*. Cada número se designa con dos índices: el primero señala el número de filas contadas de arriba a abajo, y el segundo de la columna contando de izquierda a derecha.

Mecánica: Es la rama de la Física y de la ingeniería que se ocupa de las relaciones mutuas entre fuerza, materia y movimiento.

Mecatrónica: Mecatrónica es la integración cinegética de la ingeniería mecánica con la electrónica y con el control de computadores inteligentes para el diseño y la manufactura de productos y procesos.

Modelo matemático: Es simplemente una representación matemática de sistemas del mundo real. Este modelo se desarrolla a través de la aplicación, a los elementos de un sistema, de las reglas conocidas del comportamiento.

Radianes: Unidades angulares que corresponden a un arco de longitud igual a su radio.

Robot: Es un dispositivo mecánico que realiza acciones basadas en movimientos. Sus acciones más comunes son moverse autonomicamente entre otras manipulaciones demasiado precisas, pesadas, repetitivas o riesgosas para el humano.

Robot manipulador: Son esencialmente brazos articulados. De forma más precisa, un manipulador convencional es una cadena cinemática abierta formada por un conjunto de eslabones o elementos de la cadena interrelacionados mediante articulaciones o pares cinemáticos.

Robot móvil: Estos están provistos de patas, ruedas u orugas que los capacitan para desplazarse de acuerdo a su programación. Elaboran la información que reciben a través de sus propios sistemas de sensores.

Robot redundante: Son robots que a su estructura se les añaden más de 6 grados de libertad.

Robótica: Es uno de las tecnologías más cercanas a lo que es el núcleo de la bioingeniería del conocimiento. Se trata de una tecnología mecatrónica (mecánica más electrónica más cibernética) que diseña máquinas que realizan acciones que normalmente se hacen manualmente.

Servomotor: Aparato mecánico gobernado por la rueda del timón y que a su vez acciona la caña del timón en los buques grandes.

Tarea de un robot: Es cualquier trayectoria que sigue el robot de forma repetitiva para realizar un objetivo. Es una rutina que ha sido programada con el fin de que el robot la realice cuando sea ejecutada.

Trayectoria óptima: Es una trayectoria óptima, es decir, librando singularidades para que el robot no sufra de esfuerzos mecánicos.

Simulación digital: Representación del comportamiento de algo, expresado en magnitudes numéricas mediante el computador.

Singularidad: Puntos y zonas que no son admisibles dentro y fuera del espacio de trabajo respectivamente para el robot.

Variable articular: Conocida también como coordenada generalizada para robots cuyas articulaciones son de revolución y corresponde al ángulo entre los eslabones adyacentes.

Velocidad angular: Razón del cambio del desplazamiento angular con el tiempo transcurrido.

Apéndice A

Código del ambiente virtual

```
unit Main;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  GLControl, ExtCtrls, StdCtrls, ComCtrls, LED, Supertimer, Math, Menus,
  SCControl, SCEdits, SCMaskEdit, SCAdvEdits, OleCtrls,
  CWDAQControlsLib_TLB, SCStdControls, CWUIControlsLib_TLB;

type
  TMainForm = class(TForm)
    GLControl1: TGLControl;
    Panel1: TPanel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label1: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    MainMenu1: TMainMenu;
    Programa1: TMenuItem;
    Salir1: TMenuItem;
    Reiniciarvariables1: TMenuItem;
    Label9: TLabel;
    StaticText1: TStaticText;
    Label10: TLabel;
    StaticText2: TStaticText;
    StaticText3: TStaticText;
    StaticText4: TStaticText;
    StaticText5: TStaticText;
```

```
  StaticText6: TStaticText;
  StaticText7: TStaticText;
  SCIntSpinEdit1: TSCIntSpinEdit;
  SCIntSpinEdit2: TSCIntSpinEdit;
  SCIntSpinEdit3: TSCIntSpinEdit;
  SCFloatSpinEdit1: TSCFloatSpinEdit;
  SCFloatSpinEdit2: TSCFloatSpinEdit;
  SCFloatSpinEdit3: TSCFloatSpinEdit;
  SCButton1: TSCButton;
  CWCounter1: TCWCounter;
  LED2: TLED;
  SCFloatSpinEdit4: TSCFloatSpinEdit;
  CWDIO1: TCWDIO;
  SCFloatSpinEdit5: TSCFloatSpinEdit;
  SCFloatSpinEdit6: TSCFloatSpinEdit;
  SCFloatSpinEdit7: TSCFloatSpinEdit;
  SCFloatSpinEdit8: TSCFloatSpinEdit;
  SCFloatSpinEdit9: TSCFloatSpinEdit;
  SuperTimer2: TTimer;
  CWAOPoint1: TCWAOPoint;
  Label11: TLabel;
  Label12: TLabel;
  Label13: TLabel;
  Label14: TLabel;
  CWCounter2: TCWCounter;
  CWAOPoint2: TCWAOPoint;
  procedure GLControlRender(Sender: TObject);
  procedure FormCreate(Sender: TObject);
  procedure GLControlSetup(Sender: TObject);
  procedure Salir1Click(Sender: TObject);
  procedure Reiniciarvariables1Click(Sender: TObject);
  procedure SCIntSpinEdit1Change(Sender: TObject);
  procedure SCIntSpinEdit2Change(Sender: TObject);
  procedure SCIntSpinEdit3Change(Sender: TObject);
  procedure SCFloatSpinEdit1Change(Sender: TObject);
  procedure SCButton1Click(Sender: TObject);
  procedure SuperTimer2Timer(Sender: TObject);
  procedure SCFloatSpinEdit4Change(Sender: TObject);
  procedure SCFloatSpinEdit5Change(Sender: TObject);
  procedure SCFloatSpinEdit6Change(Sender: TObject);
  procedure SCFloatSpinEdit7Change(Sender: TObject);
  procedure SCFloatSpinEdit8Change(Sender: TObject);
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure SCFloatSpinEdit9Change(Sender: TObject);
private
  FNeedSetup: Boolean;
```

```
    FList: Cardinal;
    FFirstContext: HGLRC;
    procedure Construir;
    procedure Eslabon2(ancho:Single; alto:Single; largo:Single);
    procedure Circulo1(resolucion:Integer;radio:Single; plano:Short; inclinacion:Single);
  end;

var
  MainForm: TMainForm;
  teta1,teta2:Extended;
  rotx,roty,rotz,cambio,cambio2:Integer;
  largoEsl1:Double=1.9;
  largoEsl2:Double=1.5;
  CAncho:Double=0.4;
  CAlto:Double=0.1;
  posX,posY:Array[1..20000] of Single;
  varN:Integer=1;

//-----------------------------------------------------------------------------------------------

implementation

uses
  OpenGL12;

{$R *.DFM}

procedure TMainForm.Eslabon2(ancho:Single; alto:Single;
largo:Single);
  begin

  glTranslatef(-ancho/2,0,-CAncho/2);

  glBegin(GL_QUADS);
    //Cara chica frente
    glColor3f(1,1,1); glVertex3f(0,     0,      0);
    glColor3f(0,0,1); glVertex3f(0,     alto,   0);
    glColor3f(1,1,0); glVertex3f(ancho, alto,   0);
    glColor3f(0,1,0); glVertex3f(ancho, 0,      0);
    //Cara chica Atras
    glColor3f(0,1,1); glVertex3f(0,     0,      largo);
    glColor3f(1,0,0); glVertex3f(0,     alto,   largo);
    glColor3f(1,1,0); glVertex3f(ancho, alto,   largo);
    glColor3f(1,0,1); glVertex3f(ancho, 0,      largo);
    //Cara Izquierda
    glColor3f(0,1,1); glVertex3f(0,     0,      largo);
```

```
    glColor3f(1,0,0); glVertex3f(0,     alto,   largo);
    glColor3f(1,1,0); glVertex3f(0,     alto,   0);
    glColor3f(1,0,1); glVertex3f(0,     0,      0);
    //Cara Derecha
    glColor3f(0,1,1); glVertex3f(ancho, 0,      largo);
    glColor3f(1,0,0); glVertex3f(ancho, alto,   largo);
    glColor3f(1,1,0); glVertex3f(ancho, alto,   0);
    glColor3f(1,0,1); glVertex3f(ancho, 0,      0);
    //Cara Abajo
    glColor3f(0,1,1); glVertex3f(0,     0,      largo);
    glColor3f(1,0,0); glVertex3f(ancho, 0,      largo);
    glColor3f(1,1,0); glVertex3f(ancho, 0,      0);
    glColor3f(1,0,1); glVertex3f(0,     0,      0);
    //Cara Arriba
    glColor3f(0,1,1); glVertex3f(0,     alto,   largo);
    glColor3f(1,0,0); glVertex3f(ancho, alto,   largo);
    glColor3f(1,1,0); glVertex3f(ancho, alto,   0);
    glColor3f(1,0,1); glVertex3f(0,     alto,   0);

  glEnd;
  glTranslatef(ancho/2,0,CAncho/2);

  end;

procedure TMainForm.Construir; var px,py:Single;

function MyRound(Const X: Double): Double;
  begin
  Result := Trunc(X + (-2 * Ord(X < 0) + 1) * 0.5);
  end;

function Round_nDec( const Valor : Double; const NumDec : Smallint
): Double;
  begin
  Result:= MyRound( Valor * IntPower(10, NumDec)) / IntPower(10, NumDec);
  end;

  begin
  FNeedSetup := False;
  FList := glGenLists(1);
  glNewList(FList, GL_COMPILE);
  glEnable(GL_DEPTH_TEST);
  glFrontFace(GL_CCW);

  Eslabon2(-CAncho,-CAncho,-CAncho);
```

```
  glTranslatef(0,0,-CAncho);
  glRotatef(teta1,0,1,0);
  Eslabon2(CAncho,CAlto,largoEsl1);

  glTranslatef(0,Calto,largoEsl1-CAncho);
  glRotatef(teta2,0,1,0);
  Eslabon2(CAncho,CAlto,largoEsl2);

  px:=((largoEsl1-CAncho)*Round_nDec(cos(teta1*PI/180),5)
      +(largoEsl2-CAncho/2)*Round_nDec(cos((teta2+teta1)*PI/180),5)) *10;
  py:=((largoEsl1-CAncho)*Round_nDec(sin(teta1*PI/180),5)
      +(largoEsl2-CAncho/2)*Round_nDec(sin((teta2+teta1)*PI/180),5)) *10;

  posX[varN]:=px;
  posY[varN]:=py;
  inc(varN);

  Label9.Caption :='X= '+FloatToStr(px);

  Label10.Caption:='Y= '+FloatToStr(py);

  glEndList;
  end;

//---------------------------------------------------------------------------

procedure TMainForm.GLControlRender(Sender: TObject);

begin

  Construir;

  with Sender as TGLControl do
  begin
    gluLookAt(0, 0, 9, 0, 0, -10, 0, 1, 0);

    glTranslatef(SCFloatSpinEdit2.FloatValue,
                 SCFloatSpinEdit3.FloatValue,SCFloatSpinEdit1.FloatValue);

    glRotatef(rotx, 1, 0, 0);
    glRotatef(roty, 0, 1, 0);
    glRotatef(rotz, 0, 0, 1);

    glCallList(FList);
  end;
end;
```

```
//---------------------------------------------------------------------------

procedure TMainForm.FormCreate(Sender: TObject);

begin
  Reiniciarvariables1.Click;
end;

//---------------------------------------------------------------------------

procedure TMainForm.GLControlSetup(Sender: TObject);

begin

  if FNeedSetup then
  begin

    FFirstContext := TGLControl(Sender).RenderingContext;
  end
  else

    wglShareLists(FFirstContext, TGLControl(Sender).RenderingContext);

end;

//---------------------------------------------------------------------------

procedure TMainForm.Salir1Click(Sender: TObject); begin close; end;

procedure TMainForm.Reiniciarvariables1Click(Sender: TObject); begin
FNeedSetup := True; cambio:=0; rotx:=10; roty:=90; rotz:=0;
teta1:=0; teta2:=0; SCIntSpinEdit1.IntValue:=rotx;
SCIntSpinEdit2.IntValue:=roty; SCIntSpinEdit3.IntValue:=rotz;
SCFloatSpinEdit4.FloatValue:=teta1;
SCFloatSpinEdit9.FloatValue:=teta2; GLControl1.Invalidate; end;

procedure TMainForm.SCIntSpinEdit1Change(Sender: TObject); begin if
SCIntSpinEdit1.IntValue=-1 then SCIntSpinEdit1.IntValue:=359; if
SCIntSpinEdit1.IntValue=360 then SCIntSpinEdit1.IntValue:=0;
rotx:=SCIntSpinEdit1.IntValue; GLControl1.Invalidate; end;

procedure TMainForm.SCIntSpinEdit2Change(Sender: TObject); begin if
SCIntSpinEdit2.IntValue=-1 then SCIntSpinEdit2.IntValue:=359; if
SCIntSpinEdit2.IntValue=360 then SCIntSpinEdit2.IntValue:=0;
roty:=SCIntSpinEdit2.IntValue; GLControl1.Invalidate; end;
```

```
procedure TMainForm.SCIntSpinEdit3Change(Sender: TObject); begin if
SCIntSpinEdit3.IntValue=-1 then SCIntSpinEdit3.IntValue:=359; if
SCIntSpinEdit3.IntValue=360 then SCIntSpinEdit3.IntValue:=0;
rotz:=SCIntSpinEdit3.IntValue; GLControl1.Invalidate; end;

procedure TMainForm.SCFloatSpinEdit1Change(Sender: TObject); begin
GLControl1.Invalidate; end;

procedure TMainForm.SCButton1Click(Sender: TObject); begin
  // Start counting events
  if LED2.Lit then
    begin
    SuperTimer2.Enabled:= False;
    CWCounter1.Stop;
    CWAOPoint1.SingleWrite(0, True);
    CWCounter2.Stop;
    CWAOPoint2.SingleWrite(0, True);
    LED2.Lit:=False;
    SCButton1.Caption:='Start';
    teta1:=0;
    teta2:=0;
    SCFloatSpinEdit4.FloatValue:=teta1;
    SCFloatSpinEdit9.FloatValue:=teta2;
    GLControl1.Invalidate;
    end else
    begin
    Label11.Caption:='0';
    CWCounter1.Configure;
    CWCounter1.Start;
    CWCounter2.Configure;
    CWCounter2.Start;
    LED2.Lit:=true;
    SCButton1.Caption:='Stop';
    SuperTimer2.Enabled := True;
    end;
end;

procedure TMainForm.SuperTimer2Timer(Sender: TObject); var
   valCount,valCount2:Integer;
   sentido,sentido2: WordBool;
  begin

  CWCounter1.ReadCounter(valCount, sentido);
  CWCounter2.ReadCounter(valCount2, sentido2);
  CWDIO1.Update;
```

```
  sentido:= CWDIO1.Lines.Item(0).Value;
  CWDIO1.Update;
  sentido2:= CWDIO1.Lines.Item(2).Value;

  if valCount>cambio then
    if sentido then teta1:=teta1+(valCount-cambio)*0.703125//0.3515625
      else teta1:=teta1-(valCount-cambio)*0.703125;//0.3515625;

  if valCount2>cambio2 then
    if sentido2 then teta2:=teta2+(valCount2-cambio2)*0.703125//0.3515625
      else teta2:=teta2-(valCount2-cambio2)*0.703125;//0.3515625;

  // Teta 1
  if teta1>360 then teta1:=teta1-360;
  if teta1<0 then teta1:=360+teta1;
  // Teta 2
  if teta2>360 then teta2:=teta2-360;
  if teta2<0 then teta2:=360+teta2;

  cambio:=valCount;
  cambio2:=valCount2;

  GLControl1.Invalidate;
  end;

procedure TMainForm.SCFloatSpinEdit4Change(Sender: TObject); begin
if SCFloatSpinEdit4.FloatValue=-1 then
SCFloatSpinEdit4.FloatValue:=359; if SCFloatSpinEdit4.FloatValue=360
then SCFloatSpinEdit4.FloatValue:=0;
teta1:=SCFloatSpinEdit4.FloatValue; GLControl1.Invalidate; end;

procedure TMainForm.SCFloatSpinEdit5Change(Sender: TObject); begin
CAncho:=SCFloatSpinEdit5.FloatValue; GLControl1.Invalidate; end;

procedure TMainForm.SCFloatSpinEdit6Change(Sender: TObject); begin
CAlto:=SCFloatSpinEdit6.FloatValue; GLControl1.Invalidate; end;

procedure TMainForm.SCFloatSpinEdit7Change(Sender: TObject); begin
largoEsl1:=SCFloatSpinEdit7.FloatValue; GLControl1.Invalidate; end;

procedure TMainForm.SCFloatSpinEdit8Change(Sender: TObject); begin
largoEsl2:=SCFloatSpinEdit8.FloatValue; GLControl1.Invalidate; end;

procedure TMainForm.FormClose(Sender: TObject; var Action:
TCloseAction); begin if SuperTimer2.Enabled then SCButton1.Click;
end;
```

```
procedure TMainForm.SCFloatSpinEdit9Change(Sender: TObject); begin
if SCFloatSpinEdit9.FloatValue=-1 then
SCFloatSpinEdit9.FloatValue:=359; if SCFloatSpinEdit9.FloatValue=360
then SCFloatSpinEdit9.FloatValue:=0;
teta2:=SCFloatSpinEdit9.FloatValue; GLControl1.Invalidate; end;

end.
```

Apéndice B

Código del control PD

```
function CPD:Single;
  begin
  kp:= power(Wn.FloatValue,2);                                    //Constante Proporcional
  kd:= Wn.FloatValue*2;                                           //Constante Derivativa
  TetaR[varN]:= TetaRef.IntValue*pi/180;                          //Teta de referencia
  error[varN]:= TetaR[varN]-teta*pi/180;                          //Error

  result:=kp*error[varN]-kd*(teta*pi/180-tetaV*pi/180)/lapsoV; //Control PD
  end;

//Control PD con seguimiento de trayectoria

function CPDSeg:Single; var tao,TaoDer,Tb:Single;
  begin
  kp:= power(Wn.FloatValue,2);                          //Constante Proporcional
  kd:= Wn.FloatValue*2;                                 //Constante Derivativa

  Tb:=tb.FloatValue*1000;                               //Tiempo de trayectoria

  tao:=(10*power(vtiempo,3)/power(Tb,3)                 //Spline a seguir
      - 15*power(vtiempo,4)/power(Tb,4)
      + 6*power(vtiempo,5)/power(Tb,5));
  if vtiempo>Tb then tao:=1;                            //Condición para cuando llegue

  taoDer:=(30*power(vtiempo,2)/power(Tb,3)              //Derivada de la Splina
          -60*power(vtiempo,3)/power(Tb,4)
          +30*power(vtiempo,4)/power(Tb,5));
  if vtiempo>Tb then taoDer:=0;                         //Condición de la derivada para cuando llegue

  TetaR[varN]:= TetaRef.IntValue*pi/180*tao;            //Teta Referencia
  error[varN]:= TetaR[varN]-teta*pi/180;                //Error
  tetaPunto[varN]:=teta*taoDer;                         //Velocidad
```

```
result:= kp*error[varN]                              //Control PID
        +kd*(TetaRef.IntValue*pi/180*taoDer-(teta*tao*pi/180-tetaV*tao*pi/180)/lapsoV);
end;
```

Apéndice C

Código del control PID

```
function CPID2:Single; var ki:Single;
  begin
  kp:= power(Wn.FloatValue,2);                      //Constante Proporcional
  kd:= Wn.FloatValue*2;                             //Constante Derivativa
  ki:= kp*kd/cki.IntValue;                          //Constante de Integración

  TetaR[varN]:= TetaRef.IntValue*pi/180;            //Teta Referencia
  error[varN]:= TetaR[varN]-teta*pi/180;            //Error
  integralE[varN]:=Integral(errorV,error[varN]);
  errorV:=error[varN];

  result:= kp*error[varN]                           //Control PID
          -kd*(teta*pi/180-tetaV*pi/180)/lapsoV
          +ki*integralE[varN];
  end;

//Control PID con seguimiento de trayectoria

function CPIDSeg:Single; var ki,tao,TaoDer,Tb:Single;
  begin
  kp:= power(Wn.FloatValue,2);                  //Constante Proporcional
  kd:= Wn.FloatValue*2;                         //Constante Derivativa
  ki:= kp*kd/cki.IntValue;                      //Constante de Integración
  Tb:=tb.FloatValue*1000;                       //Tiempo de trayectoria

  tao:=(10*power(vtiempo,3)/power(Tb,3)         //Splina a seguir
      - 15*power(vtiempo,4)/power(Tb,4)
      + 6*power(vtiempo,5)/power(Tb,5));
  if vtiempo>Tb then tao:=1;                    //Condición para cuando llegue

  taoDer:=(30*power(vtiempo,2)/power(Tb,3)      //Derivada de la Splina
          -60*power(vtiempo,3)/power(Tb,4)
          +30*power(vtiempo,4)/power(Tb,5));
```

```
if vtiempo>Tb then taoDer:=0;                 //Condición para cuando llegue

TetaR[varN]:= TetaRef.IntValue*pi/180*tao;    //Teta Referencia
error[varN]:= TetaR[varN]-teta*pi/180;        //Error
tetaPunto[varN]:=teta*taoDer;                 //Velocidad
integralE[varN]:=Integral(errorV,error[varN]);
errorV:=error[varN];

result:= kp*error[varN]                       //Control PID
        +kd*(TetaRef.IntValue*pi/180*taoDer-(teta*tao*pi/180-tetaV*tao*pi/180)/lapsoV)
        +ki*integralE[varN];
end;
```

Apéndice D

Código del control PID no lineal

```
function CNoLineal:Single; var ki,alfa:Single;
  begin
  alfa:= Wn.FloatValue/2;                   //Constante prporcional
  kd:= Wn.FloatValue*2;                     //Constante Derivativa
  ki:= kp*kd/cki.IntValue;                  //Constante de Integración

  TetaR[varN]:= TetaRef.IntValue*pi/180;    //Teta1 Referencia
  error[varN]:= TetaR[varN]-teta*pi/180;    //Error
  s[varN]:= alfa*error[varN]-(teta*pi/180-tetaV*pi/180)/lapsoV;
  integralE[varN]:=integralTanh(sV,s[varN]);
  sV:=s[varN];

  Result:= kd*s[varN]+ki*integralE[varN];   //Control PID no lineal
  end;

//Control PID no lineal con seguimiento de trayectorias

function CNoLinealSeg:Single; var ki,alfa,tao,taoDer,Tb:Single;
  begin
  alfa:= Wn.FloatValue/2;       //Constante prporcional
  ki:= kp*kd/cki.IntValue;      //Constante de Integración
  Tb:=tb.FloatValue*1000;       //Tiempo de trayectoria

  tao:=(10*power(vtiempo,3)/power(Tb,3)        //Splina a seguir
      - 15*power(vtiempo,4)/power(Tb,4)
      +  6*power(vtiempo,5)/power(Tb,5));
  if vtiempo>Tb then tao:=1;                   //Condición para cuando llegue

  taoDer:=(30*power(vtiempo,2)/power(Tb,3)     //Derivada de la Splina
          -60*power(vtiempo,3)/power(Tb,4)
          +30*power(vtiempo,4)/power(Tb,5));
  if vtiempo>Tb then taoDer:=0;                //Condición para cuando llegue
```

```
{
tao:=(7*power(vtiempo,3)/power(vTb,3)
    - 9*power(vtiempo,4)/power(vTb,4)
    + 3*power(vtiempo,5)/power(vTb,5));
if vtiempo>vTb then tao:=1;

taoDer:=(21*power(vtiempo,2)/power(vTb,3)
        -36*power(vtiempo,3)/power(vTb,4)
        +15*power(vtiempo,4)/power(vTb,5));
if vtiempo>vTb then taoDer:=0;
}

TetaR[varN]:= TetaRef.IntValue*pi/180*tao;                    //Teta1 Referencia
error[varN]:= TetaR[varN]-teta*pi/180;                        //Error
s[varN]:= alfa*error[varN]+(TetaRef.IntValue*pi/180*taoDer    //Manyfold de error
          -(teta*tao*pi/180-tetaV*tao*pi/180)/lapsoV);
tetaPunto[varN]:=teta*taoDer;
integralE[varN]:=integralTanh(sV,s[varN]);                    //Integral R-K
sV:=s[varN];

Result:= kd*s[varN]+ki*integralE[varN];                       //Control PIDNLSeg
end;
```

Apéndice E

Código de la trayectoria sinusoidal (control PIDNL)

```
procedure TMainForm.SCButton1Click(Sender: TObject); var
vol1,vol2:OleVariant; begin

  if LED2.Lit then
    begin
    SuperTimer2.Enabled:= False;

    CWAOPoint1.SingleWrite(0, True);
    CWAOPoint2.SingleWrite(0, True);

    LED2.Lit:=False;
    SCButton1.Caption:='Start';
    end else
    begin
    // Variables
      // Teta 1
    CWAIPoint1.SingleRead(vol1,1000);
    teta1Ini:= (vol1*2*pi/4.673);
    T1ref:= TetaRef1.IntValue*pi/180;
    TG1:=T1ref-teta1Ini;

    if TG1>0 then
      begin
      if TG1>pi then TG1:=2*pi-TG1;
      end
    else
      begin
      if TG1<(-pi) then TG1:=(2*pi-abs(TG1))*-1;
      end;
```

```
    //Label13.Caption:=FloatToStr(TG1*180/pi);
    kp1:= power(Wn1.FloatValue,2);
    kd1:= Wn1.FloatValue*2;
    alfa1:= Wn1.FloatValue/2;            //Constante prporcional
    ki1:= kp1*kd1/cki1.IntValue;         //Constante de Integración
    Tb1:=Etb1.FloatValue*1000;            //Tiempo de trayectoria
      //
      //Teta 2
    CWAIPoint2.SingleRead(vol2,1000);
    teta2Ini:= (vol2*2*pi/4.668);
    {T2ref:= TetaRef2.IntValue*pi/180;

    TG2:=T2ref-teta2Ini;

    if TG2>0 then
      begin
      if TG2>pi then TG2:=2*pi-TG2;
      end
    else
      begin
      if TG2<(-pi) then TG2:=(2*pi-abs(TG2))*-1;
      end;
    //Label13.Caption:=FloatToStr(TG2*180/pi);
    }
    kp2:= power(Wn2.FloatValue,2);
    kd2:= Wn2.FloatValue*2;
    alfa2:= Wn2.FloatValue/2;       //Constante prporcional
    ki2:= kp2*kd2/cki2.IntValue;      //Constante de Integración
    Tb2:=ETb2.FloatValue*1000;      //Tiempo de trayectoria
    vtiempo:=0;
    varN:=1;
    //
    //
    GLControl1.Invalidate;
    LED2.Lit:=true;
    Label11.Caption:='0';
    SCButton1.Caption:='Stop';
    SuperTimer2.Enabled := True;
    QueryPerformanceCounter(Start1);
    //Timer1.Enabled := True;
    end;
end;

procedure TMainForm.SuperTimer2Timer(Sender: TObject);
var
    udt1,udt2,vol1,vol2: OleVariant;
```

```
    freq, start, stop: TLargeInteger;

const beta:Integer=100;

function integralTanh(sv,sn:Single):Single; var
xi,h,k1,k2,k4:Single;
  begin
  //result := (tanh(s[varN]*beta.IntValue)+tanh(sVieja*beta.IntValue))*lapso1/2;
  xi:= Sv*beta;
  h:= Sn*beta-xi;
  k1:=tanh(xi);
  k2:=tanh(xi+h/2);
  k4:=tanh(xi+h);
  result:=h*(k1+4*k2+k4)/6;
  end;

////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////
//Funciones para teta1
////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////

function CNoLinealSeg1:Single; var TG,tao,taoDer:Single;
    sentido:Boolean;
const W:Single=2*pi/8000;
  begin

  tao:= 0.6*cos(W*vtiempo);
  taoDer:=-0.6*W*sin(W*vtiempo);

  TetaR1[varN]:= tao+teta1Ini;    //Teta1 Referencia
  TG:=TetaR1[varN]-teta1;
  if TG<0 then
    if TG<-pi then sentido:=False else sentido:=True
  else
    if TG>pi then sentido:=True else sentido:=False;
  //if sentido then Label13.Caption:='Iz' else Label13.Caption:='De';
  CWDIO1.Lines.Item(0).Value:=not sentido;
  CWDIO1.Update;
  error1[varN]:= TetaR1[varN]-teta1;       //Error
  s1[varN]:= alfa1*error1[varN]+(taoDer-(teta1-tetaV1)/lapsoV);
  tetaPunto1[varN]:=teta1*taoDer;
  integralE1[varN]:=integralTanh(sV1,s1[varN]);
  sV1:=s1[varN];

  Result:= kd1*s1[varN]+ki1*integralE1[varN];
```

```
  end;

////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////
//Funciones para teta2
////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////

function CNoLinealSeg2:Single; var TG,tao,taoDer:Single;
    sentido:Boolean;
const W:Single=2*pi/8000;
  begin

  tao:= 0.4*sin(W*vtiempo);
  taoDer:= 0.4*W*cos(W*vtiempo);

  TetaR2[varN]:= tao+teta2Ini;    //Teta1 Referencia
  TG:=TetaR2[varN]-teta2;
  if TG<0 then
    if TG<-pi then sentido:=False else sentido:=True
  else
    if TG>pi then sentido:=True else sentido:=False;
  CWDIO1.Lines.Item(1).Value:=not sentido;
  CWDIO1.Update;

  error2[varN]:= TetaR2[varN]-teta2;       //Error
  s2[varN]:= alfa2*error2[varN]+(taoDer-(teta2-tetaV2)/lapsoV);
  //sd[varN]:=alfa*TetaR[varN]*(TetaRef.IntValue*pi/180)*power(2.7,(-k*vtiempo));
  tetaPunto2[varN]:=teta2*taoDer;
  integralE2[varN]:=integralTanh(sV2,s2[varN]);
  sV2:=s2[varN];

  Result:= kd2*s2[varN]+ki2*integralE2[varN];
  end;

////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////

  begin
  CWAIPoint1.SingleRead(vol1,1000);
  teta1:= (vol1*2*pi/4.66);//+230.2;
  CWAIPoint2.SingleRead(vol2,1000);
  teta2:= (vol2*2*pi/4.668);

  QueryPerformanceFrequency(freq);
  QueryPerformanceCounter(Start);
```

```
QueryPerformanceCounter(Stop1);
vtiempo:=(Stop1-Start1)/freq*1000;

//Controles [CPD, CPDSeg, CPID, CPIDSeg, CNoLineal, CNoLinealSeg]

udt1:=CNoLinealSeg1;
udt2:=CNoLinealSeg2;
//udt2:=0;

//Teta1
udtA1[varN]:=udt1;
udt1:=abs(udt1);
if udt1>5 then udt1:=5;
CWAOPoint1.SingleWrite(udt1, True);

//Teta2
udtA2[varN]:=udt2;
udt2:=abs(udt2);
if udt2>5 then udt2:=5;
CWAOPoint2.SingleWrite(udt2, True);

Label11.Caption:='Teta1:'+FloatToStr(teta1*180/pi);
Label12.Caption:='Teta2:'+FloatToStr(teta2*180/pi);

//Teta1
teta11[varN] := teta1;
tetaV1:=teta1;

//Teta2
teta12[varN] := teta2;
tetaV2:=teta2;

GLControl1.Invalidate;

QueryPerformanceCounter(Stop);
lapso[varN] := (Stop-Start)/freq*1000;  //en milisegundos
lapsoV:=lapso[varN];
tiempo[varN]:=vTiempo;
inc(varN);
end;
```

Apéndice F

Código de la trayectoria cerrada: circunferencia (control PIDNL)

```
procedure TMainForm.SCButton1Click(Sender: TObject); var
vol1,vol2:OleVariant;
    i:Integer;
begin
  // Start counting events
  if not LED2.Lit then
    begin
    // Variables
      // Teta 1
    CWAIPoint1.SingleRead(vol1,1000);

    kp1:= power(Wn1.FloatValue,2);
    kd1:= Wn1.FloatValue*2;
    alfa1:= Wn1.FloatValue/2;            //Constante prporcional
    ki1:= kp1*kd1/cki1.IntValue;         //Constante de Integración
      //
      //Teta 2
    CWAIPoint2.SingleRead(vol2,1000);

    kp2:= power(Wn2.FloatValue,2);
    kd2:= Wn2.FloatValue*2;
    alfa2:= Wn2.FloatValue/2;       //Constante prporcional
    ki2:= kp2*kd2/cki2.IntValue;      //Constante de Integración
    vtiempo:=0;
    varN:=1;
    GLControl1.Invalidate;
    LED2.Lit:=true;
    Label11.Caption:='0';
    SCButton1.Caption:='Stop';
```

```
    QueryPerformanceCounter(Start1);
    for i:=0 to 5000 do
      begin
      control;
      Application.ProcessMessages;
      if GetKeyState(VK_ESCAPE)and 128=128 then break;
      end;

    CWAOPoint1.SingleWrite(0, True);
    CWAOPoint2.SingleWrite(0, True);

    LED2.Lit:=False;
    SCButton1.Caption:='Start';
    end;
end;

procedure TMainForm.Control;
var   udt1,udt2,vol1,vol2: OleVariant;
      freq, start, stop: TLargeInteger;
      W,Xd,Yd,Xdp,Ydp,Td1,Td2,Tdp1,Tdp2,x,y:Single;

procedure MCDIPyV; var
JI11,JI12,JI21,JI22,L1,L2,Alfa,Beta,Gama,c1,s1,c12,s12:Single;
  begin
  ////////////////////////////////////////////
  //  Modelo Cinemático Directo de Posición  //
  ////////////////////////////////////////////

  L1:=(largoEsl1-CAncho)*10;
  L2:=(largoEsl2-CAncho/2)*10;
  c1:=  cos(teta1);
  c12:= cos(teta1+teta2);
  s1:=  sin(teta1);
  s12:= sin(teta1+teta2);

  x:= (L1*c1 + L2*c12);
  y:= (L1*s1 + L2*s12);

  posx1[varN]:=x;
  posy1[varN]:=y;

  ////////////////////////////////////
  //              MCIP              //
  ////////////////////////////////////
  //  Variable Articular Deseada 2  //
```

```
  ////////////////////////////////////

  Td2:=( power(Xd,2)+Power(Yd,2)-(Power(L1,2)+Power(L2,2)) )/(2*L1*L2);
  Td2:=ArcCos(Td2);  //---------------- Teta 2

  ////////////////////////////////////
  //  Variable Articular Deseada 1  //
  ////////////////////////////////////
  Alfa:=ArcTan2(Yd,Xd);
  try
    begin
    Beta:=(Power(L1,2)+Power(L2,2)-(Power(Xd,2)+power(Yd,2)))/(2*L1*L2);
    Beta:=ArcCos(Beta);
    Gama:=(L2*sin(Beta))/sqrt(power(Xd,2)+power(Yd,2));
    Gama:=ArcSin(Gama);
    Td1:=Alfa-Gama;
    if Td1<0 then Td1:=2*pi+Td1;      //---------------- Teta 1
    end
  except
    Td1:=0;
  end;

  ////////////////////////////////////////////
  // Modelo Cinemático Inverso de Velocidad  //
  ////////////////////////////////////////////
  JI11:= c12/(L1*(c1*s12-c12*s1));                    // Matriz Jacobiana //
  JI12:= s12/(L1*(c1*s12-c12*s1));                    //
  JI21:= (L1*c1+L2*c12)/(L1*L2*(c12*s1-c1*s12));
  JI22:= (L1*s1+L2*s12)/(L1*L2*(c12*s1-c1*s12));

  Tdp1:= JI11*Xdp + JI12*Ydp;
  Tdp2:= JI22*Ydp - JI21*Xdp;

  end;

function integralTanh(sv,sn:Single):Single; var
xi,h,k1,k2,k4:Single; const beta:Integer=100;
  begin
  xi:= Sv*beta;
  h:= Sn*beta-xi;
  k1:=tanh(xi);
  k2:=tanh(xi+h/2);
  k4:=tanh(xi+h);
  result:=h*(k1+4*k2+k4)/6;
  end;
```

```
////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////
//Funciones para teta1
////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////

function CNoLinealSeg1:Single; var TG,tao,taoDer,ed:Single;
    sentido:Boolean;
  begin
  tao:= Td1;
  taoDer:= Tdp1;

  TetaR1[varN]:= tao;    //Teta1 Referencia
  TG:=TetaR1[varN]-teta1;
  if TG<0 then
    if TG<-pi then sentido:=False else sentido:=True
  else
    if TG>pi then sentido:=True else sentido:=False;
  CWDIO1.Lines.Item(0).Value:=not sentido;
  CWDIO1.Update;

  error1[varN]:= TetaR1[varN]-teta1;
  ed:=(teta1-teta11[varN-1])/lapsoV;   //Error
  s1[varN]:= alfa1*error1[varN]+taoDer-ed*tao;

  tetaPunto1[varN]:=teta1*taoDer;
  integralE1[varN]:=integralTanh(sV1,s1[varN]);
  sV1:=s1[varN];

  Result:= kd1*s1[varN]+ki1*integralE1[varN]//-eb1.FloatValue*ed*tao;
  end;

////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////
//Funciones para teta2
////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////

function CNoLinealSeg2:Single; var TG,tao,taoDer,ed:Single;
    sentido:Boolean;
  begin

  tao:= Td2;
  taoDer:= Tdp2;

  TetaR2[varN]:= tao;    //Teta1 Referencia
```

```
  TG:=TetaR2[varN]-teta2;
  if TG<0 then
    if TG<-pi then sentido:=False else sentido:=True
  else
    if TG>pi then sentido:=True else sentido:=False;
  CWDIO1.Lines.Item(1).Value:=not sentido;
  CWDIO1.Update;

  error2[varN]:= TetaR2[varN]-teta2;       //Error
  ed:=(teta2-teta12[varN-1])/lapsoV;
  s2[varN]:= alfa2*error2[varN]+taoDer-ed*tao;
  //sd[varN]:=alfa*TetaR[varN]*(TetaRef.IntValue*pi/180)*power(2.7,(-k*vtiempo));
  tetaPunto2[varN]:=teta2*taoDer;
  integralE2[varN]:=integralTanh(sV2,s2[varN]);
  sV2:=s2[varN];

  Result:= kd2*s2[varN]+ki2*integralE2[varN]-eb2.FloatValue*ed*tao;
  end;

////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////

  begin
  QueryPerformanceFrequency(freq);
  QueryPerformanceCounter(Start);
  QueryPerformanceCounter(Stop1);
  vtiempo:=(Stop1-Start1)/freq*1000;

  W:=2*pi/(ET.FloatValue*1000);

  CWAIPoint1.SingleRead(vol1,1000);
  CWAIPoint2.SingleRead(vol2,1000);
  teta1:= (vol1*2*pi/4.66);//+230.2;
  teta2:= (vol2*2*pi/4.668);

  Xd:= Eh.IntValue+Er.IntValue*cos(w*vtiempo);
  Yd:= Ek.IntValue+Er.IntValue*sin(w*vtiempo);
  //xd1[varN]:=Xd;
  //yd1[varN]:=Yd;
  Xdp:= -w*Er.IntValue*sin(w*vtiempo);
  Ydp:=  w*Er.IntValue*cos(w*vtiempo);

  MCDIPyV;

  udt1:=CNoLinealSeg1;
  udt2:=CNoLinealSeg2;
```

```
//udt1:=0;

//Teta1
udtA1[varN]:=udt1;
udt1:=abs(udt1);
if udt1>5 then udt1:=5;
CWAOPoint1.SingleWrite(udt1, True);

//Teta2
udtA2[varN]:=udt2;
udt2:=abs(udt2);
if udt2>5 then udt2:=5;
CWAOPoint2.SingleWrite(udt2, True);

//Label11.Caption:='Teta1:'+FloatToStr(teta1*180/pi);
//Label12.Caption:='Teta2:'+FloatToStr(teta2*180/pi);

teta11[varN] := teta1;
teta12[varN] := teta2;

GLControl1.Invalidate;

QueryPerformanceCounter(Stop);
lapso[varN] := (Stop-Start)/freq*1000;  //en milisegundos
lapsoV:=lapso[varN];
tiempo[varN]:=vTiempo;
inc(varN);
end;
```

Apéndice G

Código del integrador Runge-Kutta de 4^o orden

```
function Integral(a,b:Single):Single; var h,k1,k2,k4:Single;
  begin
  h:= b-a;
  k1:=a;
  k2:=a+h/2;
  k4:=a+h;
  result:=h*(k1+4*k2+k4)/6;
  end;

function integralTanh(sv,sn:Single):Single; var
xi,h,k1,k2,k4:Single;
  begin
  xi:= Sv*beta;
  h:= Sn*beta-xi;
  k1:=tanh(xi);
  k2:=tanh(xi+h/2);
  k4:=tanh(xi+h);
  result:=h*(k1+4*k2+k4)/6;
  end;
```

Apéndice H

Códigos de MATLAB

H.1. Programa para controles sin seguimiento de trayectorias.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%       Silvionel Vite Medécigo          %
%               2005                     %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                                         %
clc                                      % Borra pantalla
clear all                                % Borra variables
close all                                % Cierra ventanas (Gráficas)
                                         %
load Archivo.mat -ascii                  % Lee archivo de datos
                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%      Longitudes de los eslabones       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                                         %
L1=17;                                   % Longitud eslabón 1 (cm)
L2=14;                                   % Longitud eslabón 2 (cm)
                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%            Variables                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                                         %
q1          = Archivo(:,3);              % Teta 1
q2          = Archivo(:,11);             % Teta 2
Time        = Archivo(:,1);              % Tiempo
muestras    = Archivo(:,2);              % Total de Muestras
lapsos      = Archivo(:,4);              % Tiempo de rutina
q1Ref       = Archivo(:,5);              % Teta de referencia 1
error1      = Archivo(:,6);              % Error en la articulación 1
control1    = Archivo(:,7);              % Control en la articulación 1
q1P         = Archivo(:,8);              % Teta Punto 1 (Velocidad)
```

```
integral1  = Archivo(:,9);              % Integral en la articulación 1
manyfold1  = Archivo(:,10);             % Manyfold de error 1 (Pos y Vel)
q2Ref      = Archivo(:,12);             % Teta de referencia 2
error2     = Archivo(:,13);             % Error en la articulación 2
control2   = Archivo(:,14);             % Control en la articulación 2
q2P        = Archivo(:,15);             % Teta Punto 2 (Velocidad)
integral2  = Archivo(:,16);             % Integral en la articulación 2
manyfold2  = Archivo(:,17);             % Manyfold de error 2 (Pos y Vel)
                                        %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Cinemática Directa de Posición     %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
X = L1*cos(q1)+L2*cos(q1+q2);           % MCDP para X
Y = L1*sin(q1)+L2*sin(q1+q2);           % MCDP para Y
                                        %
Xd = L1*cos(q1Ref)+L2*cos(q1Ref+q2Ref); % MCDP para X deseada
Yd = L1*sin(q1Ref)+L2*sin(q1Ref+q2Ref); % MCDP para Y deseada
                                        %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%          Espacio de trabajo           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Periodo=Time(length(Time));             %
% Limite superior                       %
xets= 32*cos(2*pi*Time/Periodo);        %
yets= 32*sin(2*pi*Time/Periodo);        %
% Limite inferior                       %
xeti= 2*cos(4*pi*Time/Periodo);         %
yeti= 2*sin(4*pi*Time/Periodo);         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%             Gráficas                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure(1)

plot( muestras , lapsos ) title('Comportamiento de retardos en el
ciclo de control (Control PIDNL)'); xlabel('Muestras');
ylabel('Tiempo de rutina (ms)'); grid

figure(2)

plot( Time/1000 , X ,'r', Time/1000 , Y ,'b') title('Coordenadas
operacionales (Control PIDNL)'); xlabel('Tiempo (s)'); ylabel('X,
Y'); grid legend('x','y');

figure(4)
```

```
plot( Time , error1 ) title('Señal de error articular (Art 1 -
Control PIDNL)'); xlabel('Tiempo(ms)'); ylabel('Error'); grid

figure(5)

plot( Time , control1 ) title('Señal de Control (Art 1 - Control
PIDNL)'); xlabel('Tiempo(ms)'); ylabel('U(t)'); grid

figure(6)

plot( Time , q1P ) title('Velocidad (Art 1 - Control PIDNL)');
xlabel('Tiempo(ms)'); ylabel('Teta Punto'); grid

figure(7)

plot( Time , integral1) title('Indice de Integracion (Art 1 -
Control PIDNL)'); xlabel('Tiempo(ms)'); ylabel('Integral error');
grid

figure(8)

plot( Time , manyfold1 ) title('Manifold de error (Art 1 - Control
PIDNL)'); xlabel('Tiempo(ms)'); ylabel('S'); grid

figure(9)

plot( Time , q1 ,'r', Time , q1Ref ,'b') title('Angulo real y de
referencia (Art 1 - Control PIDNL)'); xlabel('Tiempo (ms)');
ylabel('Teta1 (rad)'); legend('Teta real','Teta de referencia');
grid

figure(10)

plot( X , Y ,'r', Xd , Yd ,'b') title('Espacio de trabajo (Control
PIDNL)'); xlabel('X'); ylabel('Y'); legend('Real','Ref','Puntos
singulares'); grid

figure(11)

plot( Time , error2 ) title('Señal de error articular (Art 2 -
Control PIDNL)'); xlabel('Tiempo(ms)'); ylabel('Error'); grid

figure(12)

plot( Time , control2 ) title('Señal de Control (Art 2 - Control
PIDNL)'); xlabel('Tiempo(ms)'); ylabel('U(t)'); grid
```

```
figure(13)

plot( Time , q2P ) title('Velocidad (Art 2 - Control PIDNL)');
xlabel('Tiempo(ms)'); ylabel('Teta Punto'); grid

figure(14)

plot( Time , integral2 ) title('Indice de Integracion (Art 2 -
Control PIDNL)'); xlabel('Tiempo(ms)'); ylabel('Integral error');
grid

figure(15)

plot( Time , manyfold2 ) title('Manifold de error (Art 2 - Control
PIDNL)'); xlabel('Tiempo(ms)'); ylabel('S'); grid

figure(16)

plot( Time , q2 ,'r', Time , q2Ref ,'b') title('Angulo real y de
referencia (Art 2 - Control PIDNL)'); xlabel('Tiempo (ms)');
ylabel('Teta2 (rad)'); legend('Teta real','Teta de referencia');
grid

figure(17)

plot( Time , Xd ,'b', Time , X ,'r') title('Xd y X (Control
PIDNL)'); xlabel('Tiempo (ms)'); ylabel('X (cm)'); legend('X
deseada','X real'); grid

figure(18)

plot( Time , Yd ,'b', Time , Y ,'r') title('Yd y Y (Control
PIDNL)'); xlabel('Tiempo (ms)'); ylabel('Y (cm)'); legend('Y
deseada','Y real'); grid

figure(19)

plot( X , Y ,'r', Xd , Yd ,'b', xets , yets ,'k', xeti , yeti ,'k')
title('Espacio de trabajo (Control PIDNL)'); xlabel('X');
ylabel('Y'); legend('Real','Ref','Puntos singulares'); grid
```

H.2. Programa para controles con seguimiento de trayectorias.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%       Silvionel Vite Medécigo          %
%               2005                     %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                                         %
clc                                      % Borra pantalla
clear all                                % Borra variables
close all                                % Cierra ventanas (Gráficas)
                                         %
load Archivo.mat -ascii                  % Lee archivo de datos
                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%      Longitudes de los eslabones       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                                         %
L1=17;                                   % Longitud eslabón 1 (cm)
L2=14;                                   % Longitud eslabón 2 (cm)
                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%            Variables                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                                         %
q1         = Archivo(:,3);               % Teta 1
q2         = Archivo(:,11);              % Teta 2
Time       = Archivo(:,1);               % Tiempo
muestras   = Archivo(:,2);               % Total de Muestras
lapsos     = Archivo(:,4);               % Tiempo de rutina
q1Ref      = Archivo(:,5);               % Teta de referencia 1
error1     = Archivo(:,6);               % Error en la articulación 1
control1   = Archivo(:,7);               % Control en la articulación 1
q1P        = Archivo(:,8);               % Teta Punto 1 (Velocidad)
integral1  = Archivo(:,9);               % Integral en la articulación 1
manyfold1  = Archivo(:,10);              % Manyfold de error 1 (Pos y Vel)
q2Ref      = Archivo(:,12);              % Teta de referencia 2
error2     = Archivo(:,13);              % Error en la articulación 2
control2   = Archivo(:,14);              % Control en la articulación 2
q2P        = Archivo(:,15);              % Teta Punto 2 (Velocidad)
integral2  = Archivo(:,16);              % Integral en la articulación 2
manyfold2  = Archivo(:,17);              % Manyfold de error 2 (Pos y Vel)
                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Cinemática Directa de Posición      %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
X = L1*cos(q1)+L2*cos(q1+q2);            % MCDP para X
```

```
Y = L1*sin(q1)+L2*sin(q1+q2);            % MCDP para Y
                                         %
Xd = L1*cos(q1Ref)+L2*cos(q1Ref+q2Ref);  % MCDP para X deseada
Yd = L1*sin(q1Ref)+L2*sin(q1Ref+q2Ref);  % MCDP para Y deseada
                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%          Espacio de trabajo            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Periodo=Time(length(Time));              %
% Limite superior                        %
xets= 32*cos(2*pi*Time/Periodo);         %
yets= 32*sin(2*pi*Time/Periodo);         %
% Limite inferior                        %
xeti= 2*cos(4*pi*Time/Periodo);          %
yeti= 2*sin(4*pi*Time/Periodo);          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%             Gráficas                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure(1)

plot( muestras , lapsos ) title('Comportamiento de retardos en el
ciclo de control (Control PIDNL con Seguimiento)');
xlabel('Muestras'); ylabel('Tiempo de rutina (ms)'); grid

figure(2)

plot( Time/1000 , X ,'r', Time/1000 , Y ,'b') title('Coordenadas
operacionales (Control PIDNL con Seguimiento)'); xlabel('Tiempo
(s)'); ylabel('X, Y'); grid legend('x','y');

figure(4)

plot( Time , error1 ) title('Señal de error articular (Art 1 -
Control PIDNL con seguimiento)'); xlabel('Tiempo(ms)');
ylabel('Error'); grid

figure(5)

plot( Time , control1 ) title('Señal de Control (Art 1 - Control
PIDNL con seguimiento)'); xlabel('Tiempo(ms)'); ylabel('U(t)'); grid

figure(6)

plot( Time , V1real, 'r')%, Time , V1ref, 'b' )
%plot( Time , q1P)
```

```
title('Velocidad (Art 1 - Control PIDNL con seguimiento)');
xlabel('Tiempo(ms)'); ylabel('Teta Punto'); grid

figure(7)

plot( Time , integral1) title('Indice de Integracion (Art 1 -
Control PIDNL con seguimiento)'); xlabel('Tiempo(ms)');
ylabel('Integral error'); grid

figure(8)

plot( Time , manyfold1 ) title('Manifold de error (Art 1 - Control
PIDNL con seguimiento)'); xlabel('Tiempo(ms)'); ylabel('S'); grid

figure(9)

plot( Time , q1 ,'r', Time , q1Ref ,'b') title('Angulo real y de
referencia (Art 1 - Control PIDNL con seguimiento)'); xlabel('Tiempo
(ms)'); ylabel('Teta1 (grados)'); legend('Teta real','Teta de
referencia'); grid

figure(10)

plot( X , Y ,'r', Xd , Yd ,'b') title('Espacio de trabajo (Control
PIDNL con seguimiento)'); xlabel('X'); ylabel('Y');
legend('Real','Ref'); grid

figure(11)

plot( Time , error2 ) title('Señal de error articular (Art 2 -
Control PIDNL con seguimiento)'); xlabel('Tiempo(ms)');
ylabel('Error'); grid

figure(12)

plot( Time , control2 ) title('Señal de Control (Art 2 - Control
PIDNL con seguimiento)'); xlabel('Tiempo(ms)'); ylabel('U(t)'); grid

figure(13)

plot( Time , V2real, 'r')
%plot( Time , q2P )
title('Velocidad (Art 2 - Control PIDNL con seguimiento)');
xlabel('Tiempo(ms)'); ylabel('Teta Punto'); grid

figure(14)
```

```
plot( Time , integral2 ) title('Indice de Integracion (Art 2 -
Control PIDNL con seguimiento)'); xlabel('Tiempo(ms)');
ylabel('Integral error'); grid

figure(15)

plot( Time , manyfold2 ) title('Manifold de error (Art 2 - Control
PIDNL con seguimiento)'); xlabel('Tiempo(ms)'); ylabel('S'); grid

figure(16)

plot( Time , q2 ,'r', Time , q2Ref ,'b') title('Angulo real y de
referencia (Art 2 - Control PIDNL con seguimiento)'); xlabel('Tiempo
(ms)'); ylabel('Teta2 (rad)'); legend('Teta real','Teta de
referencia'); grid

figure(17)

plot( Time , Xd ,'b', Time , X ,'r') title('Xd y X (Control PIDNL
con seguimiento)'); xlabel('Tiempo (ms)'); ylabel('X (cm)');
legend('X deseada','X real'); grid

figure(18)

plot( Time , Yd ,'b', Time , Y ,'r') title('Yd y Y (Control PIDNL
con seguimiento)'); xlabel('Tiempo (ms)'); ylabel('Y (cm)');
legend('Y deseada','Y real'); grid

figure(19)

plot( X , Y ,'r', Xd , Yd ,'b', xets , yets ,'k', xeti , yeti ,'k')
title('Trayectoria en el espacio de trabajo (Control PIDNL con
seguimiento)'); xlabel('X'); ylabel('Y');
legend('Real','Ref','Puntos singulares'); grid
```

H.3. Programa para la comparación de los controles PD, PID y PIDNL sin seguimiento de trayectorias.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%       Silvionel Vite Medécigo          %
%               2005                     %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                                         %
clc                                      % Borra pantalla
clear all                                % Borra variables
close all                                % Cierra ventanas (Gráficas)
                                         %
load Comparaciones.mat -ascii            % Lee archivo de datos
                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%            Variables                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                                         %
TimePD       = Comparaciones(:,1);       % Tiempo PD
TimePID      = Comparaciones(:,2);       % Tiempo PID
TimePIDNL    = Comparaciones(:,3);       % Tiempo PIDNL
muestras     = Comparaciones(:,4);       % Total de Muestras
TRef         = Comparaciones(:,5);       % Teta de referencia
T1PD         = Comparaciones(:,6);       % Teta 1 PD
T1PID        = Comparaciones(:,7);       % Teta 1 PID
T1PIDNL      = Comparaciones(:,8);       % Teta 1 PIDNL
T2PD         = Comparaciones(:,9);       % Teta 2 PD
T2PID        = Comparaciones(:,10);      % Teta 2 PID
T2PIDNL      = Comparaciones(:,11);      % Teta 2 PIDNL
lapsosPD     = Comparaciones(:,12);      % Tiempo de rutina PD
lapsosPID    = Comparaciones(:,13);      % Tiempo de rutina PID
lapsosPIDNL  = Comparaciones(:,14);      % Tiempo de rutina PIDNL
C1PD         = Comparaciones(:,15);      % Control 1 PD
C1PID        = Comparaciones(:,16);      % Control 1 PID
C1PIDNL      = Comparaciones(:,17);      % Control 1 PIDNL
C2PD         = Comparaciones(:,18);      % Control 2 PD
C2PID        = Comparaciones(:,19);      % Control 2 PID
C2PIDNL      = Comparaciones(:,20);      % Control 2 PIDNL
                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%      Longitudes de los eslabones       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                                         %
L1=17;                                   % Longitud eslabón 1 (cm)
L2=15;                                   % Longitud eslabón 2 (cm)
                                         %
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%        Cinemática Directa de Posición             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                                                    %
XPD    = L1*cos(T1PD)+L2*cos(T1PD+T2PD);            % MCDP para XPD
YPD    = L1*sin(T1PD)+L2*sin(T1PD+T2PD);            % MCDP para YPD
                                                    %
XPID   = L1*cos(T1PID)+L2*cos(T1PID+T2PID);         % MCDP para XPID
YPID   = L1*sin(T1PID)+L2*sin(T1PID+T2PID);         % MCDP para YPID
                                                    %
XPIDNL = L1*cos(T1PIDNL)+L2*cos(T1PIDNL+T2PIDNL);   % MCDP para XPIDNL
YPIDNL = L1*sin(T1PIDNL)+L2*sin(T1PIDNL+T2PIDNL);   % MCDP para YPIDNL
                                                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%          Espacio de trabajo           %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Periodo=TimePD(length(TimePD));         %
% Limite superior                       %
xets= 32*cos(2*pi*TimePD/Periodo);      %
yets= 32*sin(2*pi*TimePD/Periodo);      %
% Limite inferior                       %
xeti= 2*cos(4*pi*TimePD/Periodo);       %
yeti= 2*sin(4*pi*TimePD/Periodo);       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%             Gráficas                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure(1)

plot( muestras , lapsosPIDNL,'g', muestras , lapsosPID,'b', muestras
, lapsosPD,'r' ) title('Comportamiento de retardos en el ciclo de
control'); xlabel('Muestras'); ylabel('Tiempo de rutina (ms)');
legend('PIDNL','PID','PD'); grid

figure(2)

plot( TimePIDNL , C2PIDNL ,'g', TimePID , C2PID ,'b', TimePD , C2PD
,'r') title('Señal de Control (Art 2)'); xlabel('Tiempo(ms)');
ylabel('U(t)'); legend('PIDNL','PID','PD'); grid

figure(3)

plot(TimePIDNL , C1PIDNL ,'g', TimePID , C1PID ,'b', TimePD , C1PD
,'r') title('Señal de Control (Art 1)'); xlabel('Tiempo(ms)');
ylabel('U(t)'); legend('PIDNL','PID','PD'); grid
```

```
figure(4)

plot( TimePD , TRef ,'c', TimePIDNL , T2PIDNL ,'g', TimePID , T2PID
,'b', TimePD , T2PD ,'r') title('Angulo real y de referencia (Art
2)'); xlabel('Tiempo (ms)'); ylabel('Teta1 (rad)');
legend('Ref','PIDNL','PID','PD'); grid

figure(5)

plot( TimePD , TRef ,'c', TimePIDNL , T1PIDNL ,'g', TimePID , T1PID
,'b', TimePD , T1PD ,'r') title('Angulo real y de referencia (Art
1)'); xlabel('Tiempo (ms)'); ylabel('Teta1 (rad)');
legend('Ref','PIDNL','PID','PD'); grid

figure(6)

plot( XPIDNL , YPIDNL ,'g', XPID , YPID ,'b', XPD , YPD ,'r', xets ,
yets ,'k', xeti , yeti ,'k') title('Trayectoria en el espacio de
trabajo'); xlabel('X'); ylabel('Y');
legend('PIDNL','PID','PD','Puntos singulares'); grid
```

Apéndice I

Artículos publicados del trabajo de Licenciatura

1. **Domínguez-Ramírez, O. A.** and Vite-Medécigo, S., *"Robot R2SG para Interacción Hombre-Máquina"*, Fundación Colosio 2005, Centro de Investigación en Tecnologías de Información y Sistemas, Universidad Autónoma del Estado de Hidalgo, México, Pachuca, Hgo., 2005.

Bibliografía

[1] Agilent Technologies Inc., 1999, *"Agilent Technologies"*, *www.semiconductor.agilent.com.*

[2] Angulo, Usategui José M. and I. Angulo Martínez, *"Microcontroladores PIC"*, Diseño Práctico de Aplicaciones, Segunda Edición, 1999.

[3] Autodesk Inc., *"Cyberspace Developer Kit Concepts and Components"*, Cytechnical Notes No.1, Sausalito, CA,10pp., June 1993.

[4] Baratoff, G., and Blanksteen, S., *"Tracking Devices"*, Encyclopedia of Virtual Environments, *http* : *//www.cs.umd.edu/projects/hcil/eve.restore/eve − articles/I.D*,1.*b.TrackingDevices.html.*

[5] Bejczy, A. and K. Salisbury, *"Kinematic Coupling Between Operator and Remote Manipulator"*, Advances in Computer Technology, Vol.1, ASME, New York, pp. 197-211, 1980.

[6] Burdea G. C., *"Force and Touch Feedback for Virtual Reality"*, John Wiley and Sons, Inc., Electrical and Computer Engineering Department, The State University of New Jersey, 1996.

[7] Denavit, J. and R. Hartenberg, *"A Kinematic Notation for Lower Pair Mechanisms Based on Matrices"*, Journal of Applied Mechanics, Vol. 77, pp. 215-221, 1955.

[8] Dirk Louis, *"Delphi 5"*, 2000 by Markt & Technik Verlag.

[9] Domínguez-Ramírez, O. A. and V. Parra-Vega, *"Haptic Remote Guided Exploration of Deformable Objects"*, Proceedings of IMECE2003, 2003 International Mechanical Engineering Congress and Exposition, ASME, DSC-14B, Section: Dynamics Systems and Control, Topic: Advances in Robot Dynamics and Control, IMECE2003-43894, ISBN: 0-7918-4664-4. Washington, D.C., USA, 2003.

[10] Domínguez-Ramírez, O. A. and V. Parra-Vega, *"Haptic Guided Exploration with Cartesian Control for Force-Position Tracking in Finite Time"*, Proceedings of ISRA2004, International Symposium on Robotics and Automation, ISBN: 970-9702-00-9. Querétaro, México, August, 25-27, 2004.

[11] Domínguez-Ramírez, O. A., *"Introducción a las Interfaces Hápticas"*, Universidad Autónoma del Estado de Hidalgo, Centro de Investigación en Tecnologías de Información y Sistemas, Pachuca, Hgo, México, 2004.

[12] Kelly, M. Rafael, *"Control de Movimiento de Robots Manipuladores"*, Notas de Curso, Centro de sistemas de Manufactura, Instituto Tecnológico y de Estudios Superiores de Monterrey, México.

[13] Rafael Kelly, Paul Shirkey and Mark W. Spong, *"Fixed-Camera Visual Servo Control for Planar Robots"*, División de Física Aplicada, Mexico and University of Illinois, USA.

[14] Erjen Lefeber, Rafael Kelly, Romeo Ortega and Henk Nijmeijer, *"Adaptive and Filtered Visual Servoing of Planar Robots"*, Faculty of Mathematical Sciences and Department of Systems Signals and Control, The Netherlands, División de Física Aplicada, CICESE, Mexico, Laboratoire des Signaux et Systemes, France, and Eindhoven University of Technology, The Netherlands.

[15] Malvino, Albert Paul, *"Principios de electrónica"*, West Balley College, Sexta edición, 2000.

[16] Morris, Mano M., *"Diseño Digital"*, California State University, Los Angeles, 1987.

[17] Parra-Vega, V., L. G. García-Valdovinos, A. Castillo-Tapia and O. A. Domínguez-Ramírez, *"Sliding PID Control for Tracking in Finite Time for Robot Arms"*, Proceedings of the 11th International Conference on Advanced Robotics, ICAR 2003, University of Coimbra, Coimbra, Portugal, ISBN 972-96889-9-0, Vol. 3, pp. 1526-1531, 2003.

[18] Parra-Vega, V., Omar A. Domínguez-Ramírez and J. A. Méndez-Iglesias, *"Haptic Guidance for Training Motor Skills"*, Memorias del Congreso Nacional de Control Automático 2003, Asociación de México de Control Automático A.C. AMCA 2003, Ensenada, Baja California, México, ISBN 970-32-1173-9, pp. 180-186, 2003.

[19] Sciavicco, L. and B. Siciliano, *"Modelling and Control of Robot Manipulators"*, Springer-Verlag, Second Edition, 2000.

[20] Slotine, J.J. and W. Li, *"On the Adaptive Control of Manipulators"*, International Journal of Robotics Research, Vol. 6, No. 3, 1987.

[21] Spong, M. and Vidyasagar, M., *"Robot Dynamics and Control"*, , John Wiley and Sons, 1989

[22] $http://www.itpuebla.edu.mx$

[23] $http://www.microchip.com$

[24] $http://www.usdigital.com$

[25] $http://www.cpr2valladolid.com/tecno/cyr_01/robotica/sistema/sensores.htm$

[26] $http://www.mathworks.com$

[27] $http://www.opengl.org$