Visualización Cartesiana de un robot antropomórfico virtual en lazo cerrado

Por

Alejandro Camargo Montaño
Ma. del Carmen Garcia Meneses e Ibeth Ortíz Robles

Director de tesis: Dr. Omar Arturo Domínguez Ramírez

Tesis para obtener el título de Licenciados en Sistemas Computacionales

en la

Universidad Autónoma del Estado de Hidalgo
Instituto de Ciencias Básicas e Ingeniería
Centro de Investigación en Tecnologías de Información y Sistemas
Licenciatura en Sistemas Computacionales

Pachuca, Hgo., Febrero del 2005

A nuestras familias que han sido y seran parte esencial en nuestras vidas, de quienes recibimos el apoyo incondicional en todo momento.

A nuestros profesores quienes han compartido sus conocimientos formandonos como profesionistas.

A nuestros amigos por compartir sus experiencias y ser parte de nuestras vidas.

Agradecimientos

A mis padres quienes sin escatimar esfuerzo alguno han sacrificado gran parte de su vida que me han formado y educado. Mil gracias por ser el mejor ejemplo que he podido tener, gracias por todos sus cuidados y porque siempre creyeron en mí.

Sus brazos siempre se abrieron cuando necesitaba un abrazo. Sus corazones saben comprender cuándo necesito un amigo. Sus ojos sensibles se endurecen cuando necesito una lección. Sus fuerzas y su amor me han dirigido por la vida y me han dado las alas que necesitaba para volar.

Son los mejores padres que conozco. Les dedico este trabajo porque es algo que sin sus desvelos no hubiera podido ser.

A mis hermanos por ser y estar, por compartir el espacio y los momentos significativos. Gracias por permitirme estar con ustedes.

A mis mejores amigos y compañeros de tesis por permitirme participar en este proyecto y hacer de mi una mejor persona, por su apoyo incondicional y por perdonar todos aquellos tropiezos. Gracias Carmen por tus consejos y por compartir conmigo tu valiosa amistad; Gracias Alejandro por tu apoyo como pareja y como amigo, por dejar que forme parte de tu vida.

Ibeth Ortiz Robles

Agradecimientos

Dedico este trabajo a todos ustedes que me han apoyado durante toda mi vida, a mis padres por darme la vida, ustedes me han dado el mejor ejemplo a seguir que con su amor y comprensión han hecho de mi una persona de bien con valores y muchas metas en las que estarán presentes siempre.

A mis hermanos que con su compañía han llenado mi memoria de incontables momentos de alegría, gracias por estar conmigo.

A mis amigos y compañeros que me han apoyado durante mi desarrollo estudiantil y personal, a Carmen quien ha sido mas que una amiga en la que se que puedo confiar, gracias por confiar en mi.

Gracias a mi mejor amiga y pareja que con su corazón, comprensión y alegría me ha hecho mejorar como persona así como ha ocupado un lugar muy importante en mi vida, y por formar parte de ella.

Su apoyo, comprensión, amistad, amor y tolerancia se las agradezco de corazón algún día se los he de pagar con creces mientras tanto sepan que en mi tienen a un amigo que los apoyara siempre y estará al pendiente de todos ustedes.

Alejandro Camargo Montaño

Agradecimientos

Gracias a la vida y a Dios, por permitirme cumplir esta meta, que será el inicio de muchas más. A mis padres Pedro Garcia e Isabel Meneses por estar siempre a mi lado, por enseñarme el valor de la vida, el trabajo, la perseverancia, confianza, por sus consejos y sobre todo por cariño y comprensión, hoy gracias a ustedes puedo ver realizado este sueño, por eso y mucho mas hoy quiero dedicarles este trabajo, como constancia de lo mucho que los aprecio y quiero, por que antes de ser padres son mis amigos.

Risas, buenos y malos momentos hemos compartido como hermanos, Tere, Toño y Mary, gracias a ustedes por que sin su apoyo y consejos mi vida no seria la misma, por que juntos hemos formado una gran familia que no cambiaria por nada.

A mis amigos de toda la vida, Juanita por que desde niñas hemos compartido los buenos y malos momentos, que aun con la distancia nuestra amistad permanece intacta. Angelica por ser más que una amiga, por que siempre hemos compartido los momentos importantes de la vida, por enseñarme que la vida es como tu quieres que sea.

Como olvidar a los amigos de la universidad Maru y Benjamín gracias por su amistad, confianza, sus consejos y su tiempo sin ustedes no habria sido lo mismo, a Laura por brindarme tu confianza y tu amistad , a mis amigos y compañeros de tesis Ale e Ibeth por que juntos hemos realizado un sueño, agradezco su apoyo, tolerancia, amistad, confianza y consejos que han sido y seran muy valiosos para mi.

Igualmente quiero agradecer a las personas que sin duda han colaborado en mi desarrollo profesional y personal, Tania, Norma, Reme y Fruebel, por que han compartido conmigo su amistad y experiencia profesional.

Al Doctor Omar por su apoyo, comprensión, entusiasmo, tiempo y esfuerzo, al M. C. Herbert por hacer posible la culminación de este trabajo, sin el apoyo de ustedes no habría sido posible.

Si alguien falto por mencionar me disculpo por ello, ya solo me resta decir mil gracias a todos y cada uno de ustedes.

Maria del Carmen Garcia Meneses

A todas las personas que hicieron posible alcanzar ésta meta permitiendo nuestro desarrollo profesional, mil gracias.

Al M. en C. Omar Arturo Domínguez Ramírez, por el apoyo incondicional durante la realización de esta tesis, por su entusiasmo para que en ningún momento claudicáramos en la realización de esta meta, por su confianza y amistad.

Al M.C.C. Herbert Lara Ordaz por brindarnos sus conocimientos durante la carrera, asi como su amistad y consejos, compartiendo con nosotros su experiencia en el diseño de la interfaz virtual.

Al L.S.C. Denis Oliver Luis Ramon Macias Pulido por su amistad, por su confianza y la oportunidad de participar en este proyecto.

Alejandro,Carmen e Ibeth

Resumen

Este trabajo de investigación, y cuyos propósitos son la obtención de la tesis de licenciatura de ciencias computacionales, toma como investigación inicial a un visualizador virtual para verificar el desempeño de controladores aplicados a robots manipuladores, con la importante ventaja de permitir la visualización global del robot. Sin embargo, el sistema esta desarrollado fuera de línea, es decir, el control en lazo cerrado es resuelto con el integrador ODE45 de MATLAB.

El presente trabajo permite la visualización virtual en línea, implementado la solución en lazo cerrado del robot con la estrategia de control utilizando como técnica de solución, del sistema de ecuaciones diferenciales en derivadas parciales, al método de Runge-Kutta de 4º Orden.

Se presenta un estudio comparativo de un controlador moderno basado en modos deslizantes de segundo orden y dos controladores clásicos. Una contribución adicional es la consola de entrada y salida, que permite definir las variables articulares de referencia, así como las condiciones de operación del método de integración involucrado. De igual forma, esta consola permite visualizar los resultados instantáneos de las variables articulares y las coordenadas operacionales que el robot adquiere durante el desempeño de una tarea. Todos los experimentos están basado en regulación a una posición de equilibrio.

Los modelos matemáticos involucrados están basados en la formulación de Euler-Lagrange para la dinámica y, Denavit-Hartemberg para la cinemática. El sistema es desarrollado con Visual C++ y OpenGL, y Matlab es empleado exclusivamente para corroborar los resultados numéricos del integrador programado.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

1.1. Objetivo general

Implementar un sistema de visualización virtual dinámica para robots manipuladores en línea empleando un integrador Runge-Kutta de 4º orden y que permita la evaluación de diferentes estrategias de control de movimiento para diferentes trayectorias de referencia.

1.2. Objetivos específicos

- Integración de la dinámica del robot en lazo cerrado en un ambiente virtual.
- Resolver modelo dinámico con Runge Kutta de 4to. orden en un lenguaje compatible con el visualizador para evitar el uso de programas externos.

1.3. Planteamiento del problema

Los simuladores virtuales de robots, en la actualidad, están basados en los modelos cinemáticos de posición y diferenciales (velocidad y aceleración), por lo que el comportamiento complejo en el ambiente depende exclusivamente de las variables articulares y sus derivadas temporales.

Sin embargo el representar a un sistema real en un mundo virtual implica considerar a todas sus propiedades y fenómenos dinámicos, tal es el caso de las fuerzas que contribuyen en el movimiento de un robot manipulador, situación que permitirá tener un juicio objetivo para el diseño de la estructura mecánica y de los algoritmos de control, siendo esto el problema esencial en la visualización virtual detectada en el estudio del estado del arte.

1.4. Hipótesis

Considerando la dinámica de Euler-Lagrange del robot manipulador en lazo cerrado con un control no lineal y obtener la solución numérica mediante un método de integración profesional como Runge-Kutta de 4° Orden en línea con la visualización virtual de la cadena del robot estudiado es posible conocer de manera objetiva la dinámica asociada al robot con propósitos de rediseño y el desempeño del control no lineal implicado para compensar las no linealidades inherentes a la dinámica tal que permitirá realizar una mejor sintonización de sus ganancias, o bien una reconsideración de su estructura.

1.5. Solución propuesta

La solución al problema de visualización virtual dinámica del robot manipulador se presenta de manera algorítmica a continuación:

1. Obtención del modelo dinámico inverso y directo de posición, velocidad y aceleración del robot de estudio.
2. Comprobación numérica de la cinemática obtenida mediante software de simulación digital, como MATLAB.
3. Obtención del modelo dinámico basado en la formulación de Euler-Lagrange del robot de estudio.
4. Comprobación numérica de las propiedades dinámicas mediante software de simulación digital, en este caso MATLAB.
5. Diseño y evaluación, a nivel simulación digital de un algoritmo de control no lineal.
6. Programación de un método de integración numérica de eficiente desempeño en la dinámica del robot en lazo cerrado con el control no lineal propuesto.
7. Integración de la plataforma virtual en visual C++ con OpenGL.
8. Desarrollo de una interfaz de usuario para entrada/salida de datos con propósitos de sintonización de controladores y definición de trayectoria de referencia.

1.6. Antecedentes

La realidad virtual a tenido un avance significativo en la última década, la gran diversidad de aplicaciones tienen sustento en desarrollo de videojuegos, visualización virtual, entrenadores para manejo y vuelo, entrenamiento remoto para cirugía y modelado de objetos virtuales con propósitos de manufactura. Sin embargo, la visualización virtual, como el resto de las aplicaciones, están limitadas a representaciones cinemáticas, situación que no corresponde a los fenómenos inherentes a un sistema real, al sistema que se desea representar en el mundo virtual [7].

El presente trabajo dio inicio con la integración de un laboratorio virtual de robótica en el cual operen robots manipuladores de tipo industrial con diversas actividades a efectuar cada uno de ellos, y cuyo propósito final es el de evaluar diferentes estrategias de control aplicadas a robots manipuladores. El desarrollo de este proyecto se lleva a cabo en el Centro de Investigación en Tecnologías de Información y Sistemas de esta Casa de Estudios, y cuyo equipo de trabajo esta integrado por alumnos universitarios de áreas afines a las Ciencias Computacionales, y de Electrónica y Telecomunicaciones.

La condición de inicio, para el desarrollo de este trabajo de investigación y desarrollo, parte de la existencia de un simulador virtual que permite la planificación de movimientos de un robot antropomórfico considerando la dinámica del mismo [13], [15]. Sin embargo las limitaciones de este corresponden a resolver el sistema de lazo cerrado, constituido entre la dinámica del robot y el controlador propuesto, con el integrador ODE45 de matlab [29], situación que demanda el uso de un software adicional, y consumo de tiempo para representar la dinámica de lazo cerrado en el robot virtual.

1.7. Justificación

La realidad virtual vino a revolucionar ideas de ciencia y tecnología, ya que es cada vez más frecuente observar simulaciones de experimentos con mas realismo en prácticamente todas las áreas de investigación, y la robótica no es la excepción.

Es por ello que se ha dedicado este tema de tesis a la simulación dinámica de un ambiente virtual , básicamente por que al experimentar estrategias de control de movimiento en un robot real, implica la adquisición de uno y de sus correspondientes equipos periféricos, siendo una limitación con propósitos de docencia. Por otro lado este recurso abre un paso adicional a la metodología para el control de robots industriales a partir del diseño, modelado matemático, simulación en lazo abierto, simulación en lazo cerrado y evaluación experimental, y es precisamente como paso anterior a la evaluación experimental en el que el ingeniero de diseño u operador de robots pueda visualizar cual será el desempeño físico del robot mediante la simulación virtual dinámica, de esta manera puede recurrir al rediseño del control o bien a la replanificación de la trayectoria sin poner necesariamente en operación al robot, lo que implica un riesgo en su integridad al querer evaluar estrategias con conocimiento previo (subjetividad en el desempeño espacial o espacio de trabajo del robot) descrito solamente por la simulación digital, definida por respuestas temporales obtenidas mediante un simulador digital [43], [13].

Así también, existen simuladores virtuales que incluyen la cinemática del robot y que representan su visualización [13],[15], es por ello este trabajo incluye la dinámica teniendo un importante impácto en el uso y manejo de robots industriales y la contribución de este trabajo es la de efectuar la simulación virtual en línea, es decir sin el uso de matlab, ya que se desarrollo un algoritmo de integración de la dinámica del robot en lazo cerrado basada en el método de Runge-Kutta de 4º orden [11], [12].

Adicionalmente se integro una terminal de usuario para entrada y salida de parámetros de la simulación como el conocimiento instantáneo de variables articulares y operacionales, la respuesta de control y las referencias de seguimiento que describen la tarea del robot, debido a que el visualizador no satisfacía las necesidades del usuario, ya que el cambio de trayectorias requería de cambios a nivel programación.

1.8. Un estudio del estado del arte

Dentro del trabajo de investigación desarrollado en esta tesis se han detectado diferentes antecedentes de trabajos similares que han sido desarrollados en laboratorios y centros de investigacion de diferentes países, así como de la Republica Mexicana.

1.8.1. Laboratorios virtuales de robótica

Laboratorio virtual de robótica y manufactura del ITESM [44]

La finalidad de este laboratorio es la de proveer educación a distancia generados a través de los mandos virtuales, debido a que se considera importante que el alumnado dependiente de esta institución tenga una aprendizaje continuo y no deje de practicar con este tipo de robot, claro sin dejar de lado la seguridad que proporciona al usuario y al equipo el uso de un ambiente virtual.

El diseño y la implementación del Laboratorio Virtual de Robótica y Manufactura (LVRM) en el campus Monterrey del ITESM, esta siendo en colaboración de los campus Morelos, Toluca, Edo. de México y Monterrey.

Componentes del LVRM La versión actual del LVRM cuenta con los siguientes componentes:

Robot cilíndrico: el robot SEIKO con 4 grados de libertad (DOF) y una pinza de 2 posiciones, fue conectada a una computadora. Esta conexión permite controlar y programar las operaciones del robot desde la red.

Simulador del robot: se construyó un modelador de sólidos, que permite la definición de objetos compuestos y de mecanismos articulados mediante la introducción de restricciones cinemáticas. Al definir las variables articulares para un robot simulado, el modelador genera simultáneamente los “controladores” para el robot real; de forma que al aplicar desplazamientos sobre el modelo simulado, estos desplazamientos se aplican simultáneamente sobre el robot real.

Reconocedor de objetos por forma y color: el modelo del objeto a reconocer es alimentado como una lista de segmentos en 2-D y una región para la búsqueda del color. Los segmentos son seleccionados y ordenados para guiar el reconocimiento.

El LVRM comprende dos entidades, residentes en diferentes lugares, comunicadas mediante una conexión Internet. El usuario demanda el servicio desde el "Invitado", el cual se compone de un conjunto de programas que brindan el control para la operación y programación de los mecanismos remotos; el simulador es uno de tales programas. Los mecanismos, a excepción del simulador, residen físicamente en el "Anfitrión" y están conectados a sus respectivas computadoras para su control y operación.

El Invitado envía, a los mecanismos, comandos para su ejecución; enseguida recibe e ilustra las respuestas de estos mecanismos, sobre la ejecución de los comandos.Los comandos se refieren a actividades tales como "mover robot", "adquirir y enviar secuencias de imágenes" o "reconocer objetos". Por su parte, el Anfitrión recibe los comandos y los distribuye a los mecanismos respectivos para su ejecución; además supervisa la ejecución de los comandos y recibe las respuestas para reenviarlas al solicitante. Por lo cual, el Invitado y el Anfitrión cuentan con controladores que registran los comandos solicitados para realizar un experimento, tomando en consideración la aplicación que envía el comando, el dispositivo al cual se dirige y los parámetros de ejecución. Mediante este registro, se supervisa la ejecución completa del experimento.

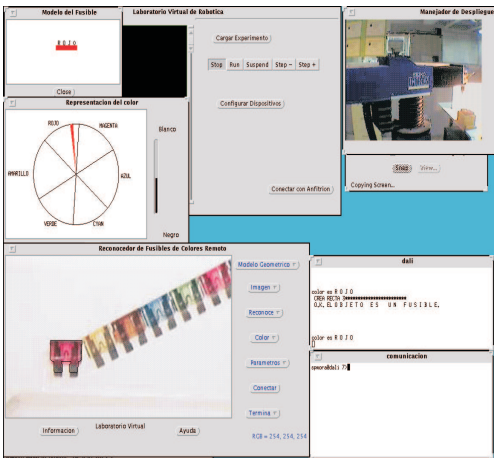

Figura 1.1: LVRM ITESM [44].

Robolab [45]

Robolab es un sistema que permite a estudiantes de asignaturas de robótica, practicar comandos de posicionamiento en un robot industrial simulado, aprendiendo aspectos básicos de robótica, cinemática y diseño de trayectorias. Además, los comandos se pueden enviar a un robot real, situado en el laboratorio del grupo AUROVA en la Universidad de Alicante, para ver los resultados de forma on-line, a través de Internet.

La primera versión de Robolab (Robolab I) ofrece una interfaz creada con Java y VRML, un software estándar de Internet para representar escenarios de realidad virtual. Posteriormente, se ha desarrollado una nueva versión (Robolab II) basada en Java y Java-3D. Los únicos medios requeridos por el estudiante son un ordenador personal conectado a Internet, un navegador web y los componentes de software de la máquina virtual de Java y de VRML en su caso.

Un estudiante puede usar las funciones que ofrece Robolab mediante un applet Java. En la primera versión de Robolab, se ofrece también una simulación de realidad virtual basado en VRML, que representa el entorno tridimensional con el robot (un Scorbot ER-IX de Intelitek) y su espacio de trabajo, como se puede ver en la siguiente figura.

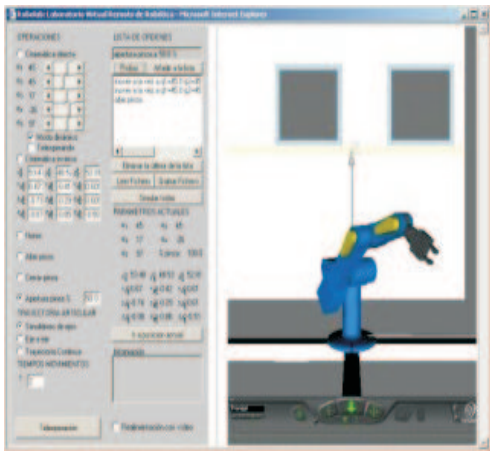

Figura 1.2: Robolab [45].

COSIMIR [46]

Es un antecedente más del desarrollo de laboratorios de robótica virtuales, este software fue diseñado por la compañía 3Festo, basándose en 3D.
Es un laboratorio virtual de robótica construido en Alemania para fines educativos e industriales dentro de cuyo ambiente se manipulan diferentes tipos de robots, así como las tareas que desempeñan.

COSIMIR utiliza los idiomas nativos robustos(MELFABASIC o comando de Movemaster) para programar dentro del ambiente de la simulación. Los paquetes de COSIMIR y de COSIROP son herramientas que permiten alcanzar la eficacia y costo eficiencia máximos en la configuración y la operación de las soluciones robustas apoyadas de la automatización, que permitan planear y que funcionen los sistemas con un grado muy alto de confianza. El software hace distinción entre las necesidades de los usuarios, para lo cual hace la siguiente clasificación:

Usuarios educativos: es altamente educacional de los conocimientos técnicos básicos y una funcionalidad extensa, culminando en el diseño, programando y operación de un sistema virtual robusto.

Los programadores: benefician la simulación esencial de funcionalidad, de 3D, y modelar la capacidad del paquete industrial de COSIMIR. Los usuarios pueden realizar fácilmente la programación off-line de los sistemas robustos de Mitsubishi, y hacen que "saque una representación gráfica funcional"de su trabajo.

El profesional de COSIMIR: con el software profesional 3D para modelar, simular y programar los diferentes sistemas de la robustez industrial, y un nuevo "ambiente al aprender"desarrollado por COSIMIR educativo.

El modelar con el profesional de COSIMIR se hace simple en virtud de las bibliotecas de robustez, efectores terminales y componentes de la automatización.

El programa también importa varios formatos de CAD. Sin embargo dicho software es de tipo comercial ya que tiene un costo de adquisición ya sea para fines educativos o comerciales.

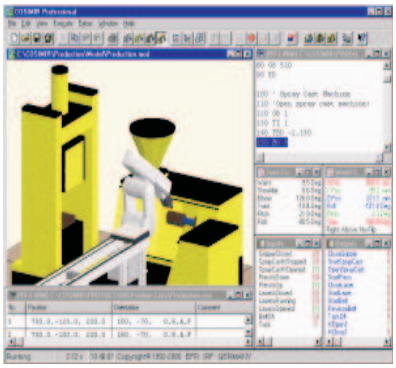

Figura 1.3: Cosimir [46].

1.9. Organización de la tesis

En el capítulo 2 de esta tesis se presenta una breve descripción de los conceptos importantes de la robótica, el modelo matemático de un robot 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 y el empleo de la metodología de Denavit-Hartenberg que han de proporcionar el movimiento al robot, mientras que el segundo modelo proporcionará las capacidades de inercia y movimiento del robot manipulador a través de la formulación de Euler-Lagrange, tomando en cuenta las propiedades dinámicas del robot.

En el capítulo 3 se hace un análisis general acerca de los métodos de control, revisando los conceptos básicos, métodos de sintonización de control, así como los controles clásicos tales como el control Proporcional y Derivativo (PD), control Proporcional, Integral, Derivativo (PID) y el control por modos deslizantes de segundo orden (PID no lineal). Los cuales brindán a los robots estabilidad en sus movimientos con diferente presición.

En el capítulo 4 se puede observar una revisión de las diferentes herramientas para la creación de ambientes virtuales, para lo cual se elaboró un estudio en el cual se pueden apreciar las principales ventajas y desventajas que nos proporcionan cada uno de ellos para definir cual ha de ser el lenguaje y herramienta utilizada en el presente trabajo para el modelado del robot, describiendo el funcionamiento del modelo cinemático y dinámico en el ambiente virtual, y las principales diferencias visuales entre ambos modelos.

En el capítulo 5 se muestra la solución numérica obtenida de la aplicación del método de Runge-Kutta implementado para proporcionar control en un robot manipulador, el modelo matemático para un robot antropomórfico de tres grados de libertad que es parte del caso de estudio en este trabajo de investigación y los resultados obtenidos de la implementación de dicho método en un control de lazo cerrado a través de una serie de gráficas que describen el comportamiento de la trayectoria del robot.

Capítulo 2

Generalidades del modelado matemático de robots manipuladores

2.1. Introducción

Cuando se habla de la robótica la mayoría de las veces se piensa en términos complejos y poco entendibles, es por ello que en este capítulo se presenta una breve descripción de los conceptos importantes de la robótica de forma simple y explícita, para mayor comprensión del presente trabajo.

Ahora bien se sabe que el modelo matemático de un robot se basa en la cinemática y dinámica, la primera esta compuesta por los modelos cinemáticos directos e inversos de posición, velocidad y aceleración definidos a través de la cadena cinemática y el empleo de la metodología de Denavit-Hartenberg que han de proporcionar el movimiento del robot, mientras que el segundo modelo proporcionará las capacidades de inercia y movimiento del robot manipulador a través de la formulación de Euler-Lagrange, tomando en cuenta las propiedades dinámicas del robot, es por ello que es importante entender la formulación general de cada uno de los modelos que sirven de base en el caso de estudio expuesto en el capítulo 5. Tomando en cuenta que dicha formulación matemática solamente describe las ecuaciones generales.

2.2. Conceptos preliminares de robótica

La palabra robot se emplea por primera vez en 1920 en una obra de teatro llamada " R.U.R" ó " Los Robots Universales de Rossum" escrita por el dramaturgo checo Karel Kapek. La trama era sencilla: el hombre fabrica un robot luego el robot mata al hombre [10]. Sin embargo Isaac Asimov un escritor y científico ruso [23] da otro sentido pues en su obra convierte a los robots en seres inofensivos para los humanos.

La palabra checa "Robota" significa servidumbre o trabajador forzado y cuando se tradujo a ingles se convirtió en el término robot [23]. De acuerdo con la Organización Internacional de Estandarización (ISO), se define a un robot como: "un manipulador automático reprogramable y multifuncional, que posee ejes capaces de manipular materiales, objetos, herramientas y mecanismos especializados a través de operaciones programadas para la ejecución de una variedad de tareas" [10]. Es claro que ambas concepciones nos indican la capacidad de estos dispositivos dentro del ámbito de trabajo del ser humano. Por ello, se debe tomar en cuenta que para que un robot sea capaz de realizar las tareas asignadas, es importante su selección acorde a la aplicación, su programación y al empleo de sensores que le brindan información del medio que le rodea y permiten la realización de su tarea de forma eficaz.

La robótica es aún un campo de reciente creación en la tecnología moderna ya que hoy en día esta ocupando un lugar destacado en la modernización de los diversos sectores industriales [7], por lo que se han realizado diversas clasificaciones de robots bajo diferentes normas, las cuales dependen ya sea de los países o de lo estándares, por

lo que en este trabajo solo se presentan algunas de las principales clasificaciones de los robots.

- **Propósito o función [10]**

  De acuerdo al propósito, los robots se clasifican en:

  - Industriales. Su uso se restringe dentro del ámbito de manufactura de productos de consumo tales como vehículos, empaque, etc.,un ejemplo claro de este tipo es el robot PUMA.
  - Personales / Educativos. Se emplean dentro de escuelas y laboratorios personales como: medios de desarrollo y prueba, así como capacitación de personal.
  - Militares. Son dispositivos especializados para una tarea como exploración, destrucción, vigilancia, seguimiento de blancos. Los rangos de operación y la precisión de estos dispositivos debe ser muy elevada.

- **Grado de inteligencia [10]**

  Un robot se puede clasificar por el grado de inteligencia y su capacidad de respuesta e integración con el medio ambiente, dentro de estos tenemos a:

  - Robots controlados por sensores, estos tienen un control en lazo cerrado de movimientos manipulados, y toman decisiones basados en datos obtenidos por sus sensores.
  - Robots controlados por visión, son aquellos robots que pueden manipular un objeto al utilizar información visual.
  - Robots controlados adaptativamente, estos pueden automáticamente reprogramar sus acciones sobre la base de los datos obtenidos por los sensores.
  - Robots con inteligencia artificial, ellos utilizan las técnicas de inteligencia artificial, para hacer sus propias decisiones y resolver problemas.

- **Tipo de controlador [10]**

  Con respecto al controlador del robot se pueden agrupar de acuerdo al nivel de control que realizan.

  - Nivel de inteligencia artificial, donde el programa aceptara un comando como “levantar el producto” y descomponerlo dentro de una secuencia de comandos de bajo nivel basados en un modelo estratégico de tareas.
  - Nivel de modo de control, donde los movimientos del sistema son modelados, para lo que se incluye la interacción dinámica entre los diferentes mecanismos, trayectorias planeadas, y puntos de asignación seleccionados.
  - Niveles de servosistemas, donde los actuadotes controlan los parámetros de los mecanismos con el uso de una retroalimentación interna de los datos obtenidos por los sensores, y la ruta es modificada sobre la base de los datos que se obtienen de sensores externos. Todas las detecciones de fallas y mecanismos de corrección son implementadas en este nivel.

- **Lenguaje de programación [10]**

  La clave para una aplicación efectiva de los robots en una amplia variedad de tareas, es el desarrollo de lenguajes de alto nivel. Los sistemas de programación de robots caen dentro de tres clases:

  - Sistemas guiados, en el cual el usuario conduce al robot a través de los movimientos a ser realizados.
  - Sistemas de programación de nivel-robot, en los cuales el usuario escribe un programa de computadora al especificar el movimiento y el sensado.
  - Sistemas de programación de nivel-tarea, el cual el usuario especifica las operaciones por las acciones sobre los objetos que el robot manipula.

2.3. Modelo cinemático de robots manipuladores

El modelo cinemático proporciona las características físicas del robot y se encuentra conformado por:

- El directo que puede ser de posicionamiento, velocidad y aceleración.
- Y el inverso igualmente de posicionamiento, velocidad y aceleración.

La diferencia entre ellos radica principalmente en que el primero es utilizado para dar al robot la capacidad de movimiento que requiere para realizar sus actividades, permite conocer su desempeño en el espacio de trabajo y el segundo será utilizado para planificar trayectorias realistas en el espacio de trabajo.

Para mejor entendimiento del modelo cinemático se propone un robot de un grado de libertad en el que se ha se ha realizado un análisis más profundo, el cual consta de:

- Una variable articular (también llamada coordenada generalizada o ángulo entre eslabones).
- La longitud que tendrá el eslabón.
- Y las coordenadas operacionales (x, y) ó también llamados puntos espaciales referidos al espacio de trabajo, estos puntos son los puntos que el efector final puede alcanzar durante el desempeño de una tarea.

2.3.1. Cadena cinemática

Se entiende por cadena cinemática a un mecanismo de eslabones articulados que pueden ser modelados como una cadena articulada. La cadena cinemática de un conjunto de eslabones, trata con el estudio analítico de la geometría del movimiento con respecto a un sistema de coordenadas de referencia fijas como una función del tiempo sin considerar las fuerzas que originan dicho movimiento [5].

2.3.2. Parámetros Denavit-Hartenberg (PDH)

El método utilizado para el manejo de las cadenas cinemáticas es el creado por Denavit y Hartenberg. Este método es una técnica sistemática que permite establecer matrices de desplazamiento para cada par adyacente de elementos dentro de una articulación de un mecanismo.

Denavit y Hartenberg propusieron un método sistemático para obtener y representar la geometría espacial de los elementos de una cadena cinemática, y en particular de un robot, con respecto a un sistema de referencia fijo. Este método utiliza una matriz de transformación homogénea para descubrir la relación espacial entre dos elementos rígidos adyacentes, reduciéndose el problema cinemático directo a encontrar una matriz de transformación homogénea 4 X 4 que relacione la localización espacial del robot con respecto al sistema de coordenadas de su base.

Según la representación PDH, escogiendo adecuadamente los sistemas de coordenadas asociados para cada eslabón, será posible pasar de uno al siguiente mediante 4 transformaciones básicas que dependen exclusivamente de las características geométricas del eslabón [4].

Donde:
Θ_i: es el ángulo que forman los ejes x_{i-1} y x_i medido en un plano perpendicular al eje z_{i-1}, utilizando la regla de la mano derecha. Se trata de un parámetro variable en articulaciones giratorias.

d_i: es la distancia a lo largo del eje z_{i-1} desde el origen del sistema de coordenadas i-esimo hasta la intersección del eje z_{i-1} con el eje x_i. Se trata de un parámetro variable en articulaciones prismáticas.

a_i: es la distancia a lo largo del eje x_i que va desde la intersección del eje z_{i-1} con el eje x_i hasta el origen del sistema i-esimo, en el caso de articulaciones giratorias. En el caso de articulaciones prismáticas, se calcula como la distancia mas corta entre los ejes z_{i-1} y z_i.

α_i : es el ángulo de separación del eje z_{i-1} y el eje z_i, medido en un plano perpendicular al eje x_i, utilizando la regla de la mano derecha.

De este modo, basta con identificar los parámetros q_i, a_i, d_i, para obtener matrices A y relacionar así todos y cada uno de los eslabones del robot, para así obtener los parámetros de un robot antropomorfico.

2.3.3. Matriz de transformación homogénea

Una matriz de transformación homogénea permite definir la posición y orientación de un marco de referencia respecto al otro.

De esta forma, para los marcos ortonormales de referencia definidos en la cadena cinemática, es posible deteminar una matriz de transformación homogénea general, que defina en función de los parámetros de Denavit-Hartemberg la posición y orientación de un marco de referencia i respecto al marco de referencia i-1.

Una matriz de transformación homogénea con estas características es la siguiente [5]:

$$ {}^{i}_{i-1}T = \begin{bmatrix} C\theta_i & -S\theta_i S\alpha_i & S\theta_i S\alpha_i & a_i C\theta_i \\ S\theta_i & C\theta_i C\alpha_i & -C\theta_i C\alpha_i & a_i C\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.1} $$

2.3.4. Modelo cinemático directo de posición(MCDP)

Se sabe que el conjunto de ecuaciones que determinan las coordenadas operacionales en términos de las variables articulares (para este caso serán x, y en términos angulares), el cual permite conocer la posición cartesiana instantánea durante el desempeño de una tarea [5].
Tal modelo desarrollado en términos matemáticos se encuentra en el capítulo 5.

2.3.5. Modelo cinemático inverso de posición(MCIP)

Es el conjunto de ecuaciones que permiten deteminar a las variables articulares en función de las coordenadas operacionales.

$$ \theta = F^{-1}(x) \tag{2.2} $$

2.3.6. Modelo cinemático directo de velocidad(MCDV)

Es un conjunto de ecuaciones que permiten definir la velocidad operacional de un robot durante su desempeño de una tarea.

Representación de la velocidad operacional:

$$ \frac{d}{dt}x = \bar{x} = f\left(\dot{\theta}, \dot{\theta}\right) = J\dot{\theta} \tag{2.3} $$

donde:
$\dot{\theta}$: velocidad articular

En el capítulo 5 se hará una descripción más detallada de formulas para obtención de la velocidad operacional.

2.3.7. Modelo cinemático inverso de velocidad(MCIV)

Es el conjunto de ecuaciones que permiten controlar a la velocidad articular en términos de la velocidad operacional y la posición angular.

$$\dot{\theta} = f^{-1}\left(\dot{x}, \dot{\theta}\right) = J^{-1}\,\dot{x} \tag{2.4}$$

2.3.8. Modelo cinemático directo de aceleración(MCDA)

Es el conjunto de ecuaciones que permiten determinar a la aceleración operacional en términos de la posición angular y sus derivadas. La ecuación generalizada representativa es [5]:

$$\ddot{X} = J\ddot{\theta} + \dot{J}\dot{\theta} \tag{2.5}$$

Donde:
$\ddot{X}$: Representa la aceleración articular.
$\dot{J}$: Es la derivada temporal de la matriz jacobiana, la cual se describe en el capítulo 5.

2.3.9. Modelo cinemático inverso de aceleración(MCIA)

Es el conjunto de ecuaciones que permite conocer la aceleración articular en términos de la aceleración operacional y de la velocidad articular, definida como [5]:

$$\ddot{\theta} = J^{-1}\left\{\ddot{X} - \dot{J}\,\dot{\theta}\right\} \tag{2.6}$$

2.4. Modelo dinámico de robots manipuladores

La dinámica es la rama de la mecánica la cual trata con el movimiento de los cuerpos bajo la acción de fuerzas, 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 actuadores. Durante el movimiento de un conjunto de eslabones articulados con el propósito de que el efector final describa una trayectoria deseada, el sistema de control tiene que producir pares en las articulaciones para equilibrar las fuerzas de los eslabones [5].

2.4.1. Conceptos preliminares de la dinámica de cuerpos rígidos

Cuando un eslabón es acelerado, el actuador debe proporcionar un par superior a la inercia.

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 centrípeto 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 esta cuantificada como la masa del material.

Los pares centrípetos surgen de las fuerzas centrípetas 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 vertiginosas derivadas de dos eslabones en rotación, estas fuerzas son similares a las fuerzas causadas en un vértice, y son proporcionales al producto de las velocidades articulares de esos eslabones.

El modelo dinámico de un robot, se puede obtener a partir de las leyes de la mecánica Newtoniana y Lagrangiana.

Existen diversos métodos convencionales como Euler-Lagrange (E-L) y Newton-Euler (N-E). El uso del método E-L, 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.

2.4.2. Formulación de Euler-Lagrange(E-L)

La formulación de Euler-Lagrange nos proporcionara las capacidades de inercia y movimiento que tiene el robot manipulador. El uso del método E-L, 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 [5].
Las ecuaciones de movimiento de E-L son:

$$\tau_i = \frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_i} - \frac{\partial L}{\partial \theta_i} \tag{2.7}$$

ı=1,2,.....n.
Donde el Lagrangiano es:

$$L = \sum_{i=1}^{n} (k_i - p_i) \tag{2.8}$$

Donde:
k_i: Es la energía cinética del i-ésimo eslabón.
p_i: la energía potencial del i-ésimo eslabón.

2.4.3. Propiedades dinámicas de un robot manipulador

Las propiedades del modelo dinámico son:

1. Simetría de la matriz de inercia

$$H(q) = H(q)^T \tag{2.9}$$

Elemento	Descripción
y	$y \in \Re^{3*1}$ cualquier vector y
y^T	transpuesta del vector y
$\dot{H}(q)$	derivada de la matriz de inercia
$C(q, \dot{q})$	Matriz de fuerzas de coriolis y centrípetas

Cuadro 2.1: Descripción de variables de las propiedades dinámicas de un robot manipulador

2. La matriz de inercia es definida positiva

$$\det \{H(q)\} > 0 \tag{2.10}$$

para todo vector $y \in \Re^{n*1}$,$y^T \dot{H}(q)\, y > 0$

3. Matriz antisimétrica

$$y^T \left(\dot{H}(q) - 2C(q, \dot{q}) \right) y \cong 0 \tag{2.11}$$

2.5. Conclusiones

El término de robot a sido símbolo de controversia y no tanto por la creación del término, si no lo que ello representa, aun en nuestros días, tiene distintos significados y son tan variados que no se ha definido una sola clasificación como se puede observar en este estudio.

Ahora bien las propiedades cinemáticas analizadas en este capítulo están dadas en términos bastante generales ya que se pueden aplicar en cualquier tipo de robots ya sea antropomórficos o no, la importancia de este modelo radica en las características que le proporciona al robot, pues le permite tener la capacidad de movimiento, mientras que la dinámica le asigna propiedades tales como gravedad, fricción, inercia por mencionar algunas que son factores que influyen en el comportamiento final del robot.

Una vez que se han definido los conceptos y formulas básicas del modelo tanto cinemático como dinámico de un robot manipulador, el lector esta preparado para comprender el desarrollo del modelo matemático presentado en el capítulo 5 para este caso de estudio.

Capítulo 3

Generalidades del control de robots manipuladores

3.1. Introducción

En la actualidad los sistemas de automatización industrial permiten incrementar la productividad sin sacrificar el índice de la calidad en el producto terminado, de ello depende estrictamente de las características del sistema involucrado en la automatización.

Un robot manipulador es un sistema electromecánico que permite resolver múltiples problemas de automatización, sin embargo y aun cuando existen diversos esquemas de control de robots, los que están involucrados en diversos casos reales en la industria mexicana implican estrategias de control clásico, situación que repercute como un bajo desempeño del robot y en la tarea encomendada. Las estrategias de control existentes han sido clasificadas por la comunidad de control y más aún por la comunidad de robótica, en control clásico, moderno e inteligente.

En este capítulo presentamos las estrategias de control clásico aplicadas comúnmente al control de movimiento en robots manipuladores industriales, y una estrategia de control moderna que compensa las no linealidades inherentes a la dinámica del robot, sin depender necesariamente de su modelo dinámico, y que asegura convergencia y estabilidad asintótica local, con un desempeño eficiente en regulación a una coordenada operacional, así como en el seguimiento de trayectorias dentro del espacio de trabajo.

El diseño de controladores para robots es una de las áreas de más interés tanto para los constructores de robot como para los centros de investigación. Por lo tanto el estudio del control es interesante por los grandes retos teóricos que ofrece al investigador, ya que esté es indispensable en aplicaciones especificas que pueden ser llevadas a cabo mediante los robots comerciales actuales .

Por lo tanto en este capítulo se abordaran las generalidad acerca de los métodos de control, revisando los conceptos básicos, métodos de sintonización de control, así como los controles clásicos tales como el control Proporcional Derivativo (PD), control Proporcional Integral Derivativo (PID) y el control por modos deslizantes de segundo orden (PID no lineal). Los cuales brindán a los robots, estabilidad en sus movimientos al igual que presición. Este estudio sirve de base para la comprensión de las pruebas realizadas en el capítulo 5.

3.2. Conceptos preliminares de control de robots

El control se define como un conjunto de mandos que regulan el funcionamiento del robot. La metodología de estudio de diseño de los sistemas de control puede resumirse a través de las siguientes etapas [7]:

- Familiarización con el sistema físico a controlarse
- Modelo
- Especificaciones de control
- Controlador

Familiarización con el sistema físico a controlarse:

En esta etapa deben considerarse las variables físicas del sistema cuyo comportamiento se desea controlar, como lo son velocidad, posición y aceleración, tales variables son llamadas salidas del sistema. Aunque tambien se debe tomar en cuenta las llamadas entradas de sistema que en este caso de estudio es la fuerza de par o Torque que ya han sido consideradas en el capítulo anterior.

Modelado:

Esté modelo matemático se obtine a través de dos técnicas la **analítica** (basada en ecuaciones de la física que rige el comportamiento del sistema) y la **experimental** (requiere de datos experimentales del sistema), en este caso en particular se ha realizado a través de la primera técnica.

Especificaciones de control:

En el se especifican las características que se desean para el sistema. Así se puede apreciar que el control es un tema sobresaliente de la intersección que se da entre la robótica y la ingeniería eléctrica y en especial en el área del control automático.

Controlador:

El controlador es el dispositivo que se encarga de regular el movimiento de los elementos del manipulador y de todo tipo de acciones, cálculos y procesado de información, que se realiza. La complejidad del control varía según los parámetros que se gobiernan, pudiendo existir las siguientes categorías:

1. Controlador de posición. Sólo interviene en el control de la posición del elemento terminal. Puede actuar en modo punto a punto, o bien, en modo continuo, en cuyo caso recibe el nombre de control continuo de trayectoria.
2. Control cinemático. Cuando además de la posición se regula la velocidad.
3. Control dinámico. Se tienen en cuenta, también, las propiedades dinámicas del manipulador, motores y elementos asociados.
4. Control adaptativo. Además de lo indicado en los anteriores controles, también se considera la variación de las características del manipulador al variar la posición.[7]

3.2.1. Sistema de control en un robot

Un robot puede realizar tareas monótonas y complejas sin errores en la operación, debe manejar partes mecánicas que tengan una forma y un peso determinados, para lo cual cuenta con algunos dispositivos sensores. Alos robots de nivel bajo, se les instala 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. Un robot de alto nivel utiliza un medio óptico para rastrear un objeto a fondo, primero reconoce el patrón y determina la presencia y orientación del objeto haciendo uso de una computadora para procesar las señales del proceso de reconocimiento de patrones. A continuación, el robot levanta la parte y la mueve de un lugar a otro dependiendo del tipo de actividad que realice, es por ello que se dice que una computadora digital bien programada funciona como computadora.

El tipo de controlador que se use debe decidirse con base en la naturaleza de la planta y condiciones operacionales, incluyendo consideraciones tales como seguridad, costo, disponibilidad, confiabilidad, precisión, peso y tamaño.

Sistema de control en lazo abierto

Los sistemas en los cuales la salida no afecta la acción del control se denominan sistemas de control en lazo abierto. En este tipo de control no se mide ni se realimenta para compararla con la entrada. Por lo tanto para cada entrada de referencia le corresponde una condición operativa fija; como resultado, la precisión del sistema depende de la calibración en consecuencia si llegan a existir perturbaciones el sistema no realizara la tarea deseada [17].

Este control es más fácil de desarrollar, por que la estabilidad del sistema no es un problema de importancia.

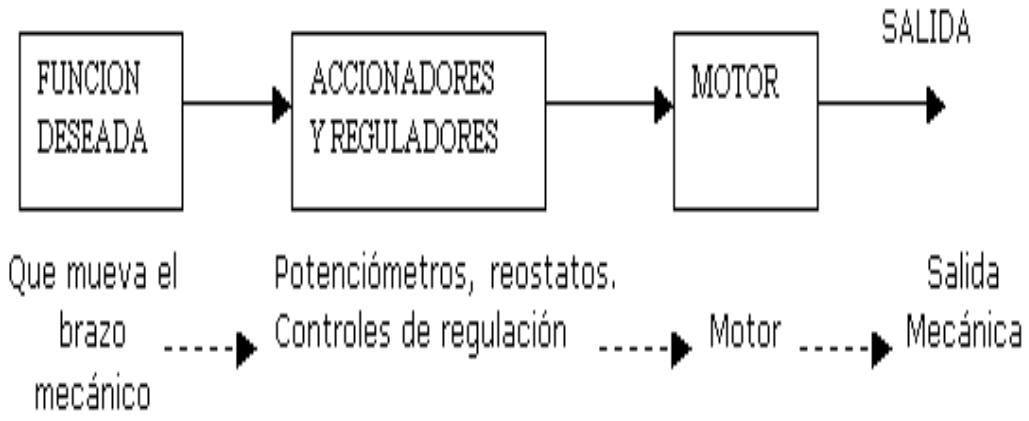

Figura 3.1: Esquema de un control de lazo abierto [17].

Sistema de control en lazo cerrado

Los sistemas de control realimentados se denominan sistemas de control en lazo cerrado. En este tipo de sistema se alimenta al controlador la señal de error de actuación, que es la diferencia entre la señal de entrada y la señal de realimentación (que puede ser la señal de la misma señal o una función de la señal de salida y sus derivadas y/o integrales), a fin de reducir el error y llevar la salida del sistema a un valor conveniente. El término de control en lazo cerrado siempre implica el uso de una acción de control realimentado para reducir el error del sistema.

En este tipo de control el uso de la realimentación vuelva la respuesta del sistema relativamente insensible a las perturbaciones externas y a las variaciones internas en los parámetros del sistema [17].

Por otro lado la estabilidad es una de las funciones mas importantes en un control de lazo cerrado, ya que puede conducir a corregir en exceso errores que producen oscilaciones de amplitud constante o cambiante.

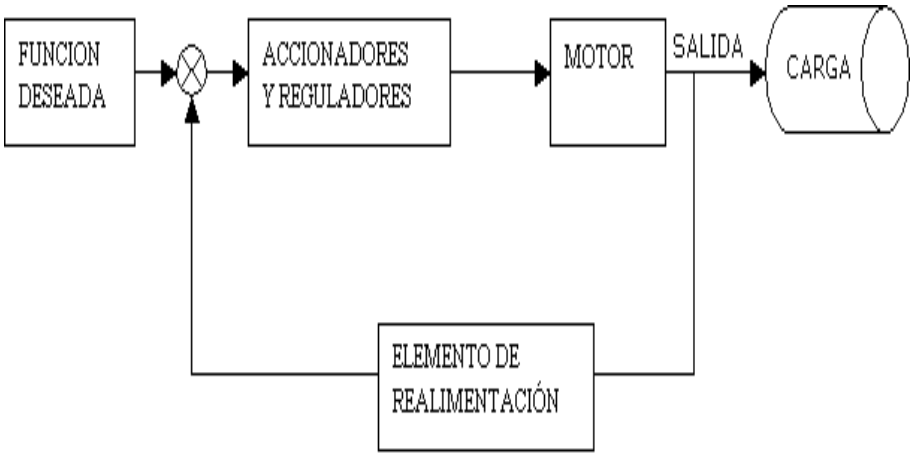

Figura 3.2: Esquema de un control de lazo cerrado[17].

3.3. Sintonización de controladores

Al proceso de seleccionar los parámetros del controlador que cumplan con las especificaciones de desempeño se conoce como sintonización del controlador. Ziegler y Nichols sugirieron reglas para sintonizar controles PID (lo cual significa establecer valores kp, ki y kd) basadas en el valor de kp que se produce en la estabilidad marginal cuando solo se usa la acción del control proporcional.

Las reglas de sintonización de Ziegler-Nichols se han usado ampliamente para sintonizar controladores PID en los sistemas de control de procesos en los que se conoce con precisión la dinámica del robot.

Existe el método denominado reglas de sintonización de Ziegler-Nichols, en el cual se pretende obtener un 25 % de sobrepaso máximo de la respuesta escalón como se muestra en la tabla 3.3.

Donde:

t: es el tiempo

λ: es el retardo de tiempo.

3.4. Control PD

Un controlador que funciona típicamente como un amplificador con una ganancia constante k se conoce formalmente como control proporcional (nos permite el control de posición pura de los robots manipuladores), ya

Tipo de controlador	k_p	k_i	k_d
P	t/λ	∞	0
PD	$0{,}9\,t/\lambda$	$\lambda/0{,}3$	0
PID	$1{,}2\,t/\lambda$	2λ	$0{,}5\lambda$

Cuadro 3.1: Tabla de regla de sintonización Ziegler-Nichols basada en la respuesta escalón[17].

que la señal de control a la salida del controlador está relacionada con la entrada del controlador mediante una constante proporcional.

La acción de control de un controlador proporcional-derivativa se define mediante:

$$\tau = k_p(t) + k_d\left(\frac{de(t)}{dt}\right) \tag{3.1}$$

Teniendo por condición:

$$k_d = 2\sqrt{k_p} \tag{3.2}$$

En donde k_p es la ganancia proporcional y k_d la ganancia derivativa. Tanto k_p como k_d son ajustables. La acción de control derivativa en ocasiones denominada control de velocidad, ocurre donde la magnitud de la salida del controlador es proporcional a la velocidad de cambio de la señal de error. El tiempo derivativo k_d es el intervalo de tiempo durante la cual la acción de la velocidad hace avanzar el efecto de la acción de control proporcional. La acción de control derivativa tiene un carácter de previsión pero ello trae como desventaja de amplificar las señales de ruido, lo cual puede provocar un efecto de saturación en el actuador, por lo tanto la acción de control derivativa no se usa nunca sola, debido a que solo es eficaz durante periodos transitorios.

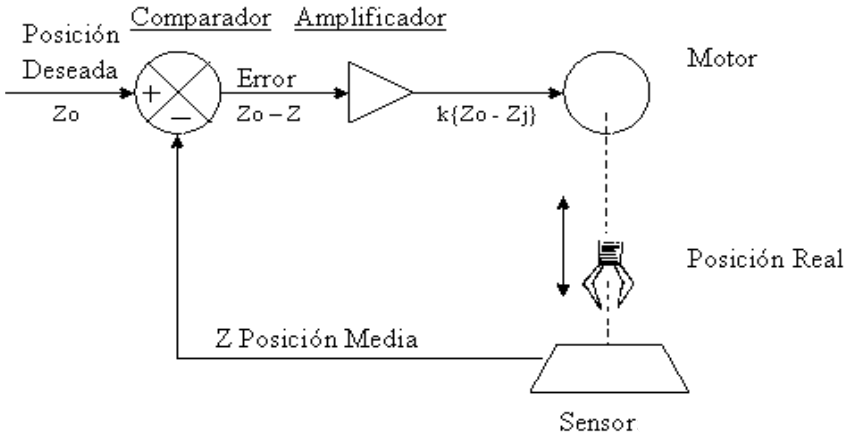

Figura 3.3: Controlador de tipo proporcional [42].

3.5. Control PID

La combinación de una acción de control proporcional, una acción de control integral y una acción derivativa se denomina acción de control proporcional-integral-derivativa (PID). Esta acción combinada tiene las ventajas

de que cada una de las acciones de control individuales elimina errores, perturbaciones y sobreoscilaciones, es decir, intenta mantener su salida en un nivel predeterminado.

3.5.1. Estructura PID

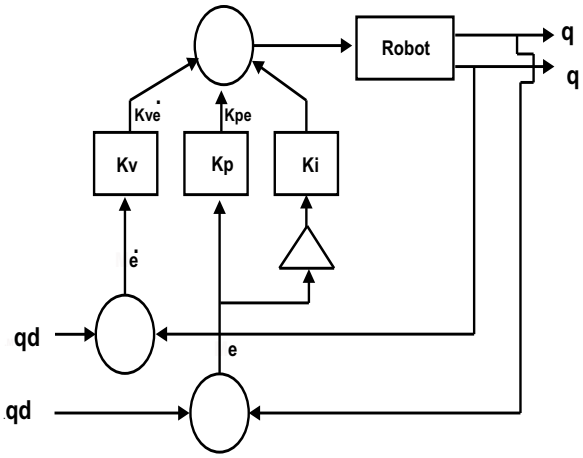

Figura 3.4: Control PID [42].

La formula estandar del controlador PID es:

$$\tau = k_p\, e(t) + k_i \int e(t)dt + k_d \frac{de}{dt} \tag{3.3}$$

Teniendo como condiciones:

k_d debe ser mucho menor que k_p y éste a su vez mucho menor a k_i

$$k_i << k_d << k_p$$

Siendo:

k_d: la Constante derivativa
k_p: la Constante proporcional

Por lo tanto se puede utilizar una proporción estandar como lo es la siguiente:

$$k_i << 2\sqrt{kp}$$
$$k_p = wn^2$$
$$k_d = 2 * wn$$

Siendo:

k_p: la constante proporcional
k_d: la constante derivativa
wn: la frecuencia natural

3.6. Control por modos deslizantes de segundo orden (PID no lineal) [14]

Para el diseño del control, es necesario diseñar una referencia nominal q_r tal que en lazo cerrado, el robot tenga un comportamiento estable, se propone la siguiente

$$\dot{q}_r = \dot{q}_d - \alpha \Delta q \tag{3.4}$$

sustituyendo (3.4) se obtiene el error extendido.

$$S = \dot{\Delta} q + \alpha \Delta q \tag{3.5}$$

donde $\Delta q = q - q_d$, y q_d representa la trayectoria o coordenada de referencia.

En aplicaciones de control de robots, con tareas de regulación a una coordenada no singular del espacio de trabajo del manipulador, el control PD con compensación de gravedad y el control PID lineal tienen excelente desempeño. Sin embargo, estos controladores no tienen eficiente desempeño en tareas de seguimiento de trayectorias. El control propuesto con eficiente desempeño en tareas de seguimiento, sin conocimiento de los parámetros dinámicos del robot, es el control PIDNL (Parra-Vega et al.).

Para el diseño del control, es necesario diseñar una referencia nominal q_r tal que en lazo cerrado, el robot tenga un comportamiento estable, se propone la siguiente

$$\dot{q}_r = \dot{q}_d - \alpha \Delta q + S_d - K_i \sigma \tag{3.6}$$
$$\dot{\sigma} = \operatorname{sgn}(S_q) \tag{3.7}$$

donde,

$$S_q = S - S_d \tag{3.8}$$
$$S_d = S(t_0) e^{-kt} \tag{3.9}$$

donde S es definido de forma similar a (3.5), $\alpha = \alpha^T \in \Re^{n \times n}$ es una matriz diagonal positiva definida, $K_i = K_i^T \in \Re^{n \times n}$ es una matriz diagonal positiva definida, $k > 0$, $\operatorname{sgn}(*)$ denota la función signo de $(*)$. Notese que $\ddot{q}_r = \ddot{q}_d - \alpha \dot{\Delta} q + \dot{S}_d - K_i \operatorname{sgn}(S_q)$ es discontinua, y que $S_q(t_o) = 0$ para cualquier condición inicial. Sustituyendo (3.6) se obtiene la coordenada del error extendido S_r.

$$S_r = S_q + K_i \sigma \tag{3.10}$$

El control PIDNL propuesto es el siguiente,

$$\tau = -K_d S_r \tag{3.11}$$

en términos de las anteriores definiciones,

$$\tau = -K_p \Delta q - K_v \dot{\Delta} q + K_d S_d - K_I I \tag{3.12}$$
$$I = \sigma = \int_{t_0}^{t} \operatorname{sgn}(S_q(\varsigma)) d\varsigma \tag{3.13}$$

donde $K_d = K_d^T \in \Re^{n \times n}$ es una matriz diagonal positiva definida, $K_p = K_d \alpha$, $K_v = K_d$ y $K_I = K_d K_i$ son matrices diagonales que representan las ganancias de control.

Las pruebas de estabilidad de este control se observan en el documento "Resultados Preliminares sobre Interacción Háptica en Laberintos Virtuales, con Propósitos de Diagnostico en Pacientes con Discapacidades Neuropsicológicas" [14].

3.7. Conclusiones

Los controles ayudan a que los robots se muevan de forma estable y como el operador humano lo desea, debido a que infinidad de robots son utilizados en la industria se necesita que éstos posean un comportamiento preciso de sus movimientos, debido a que en muchas ocasiones son utilizados en actividades complicadas que requieren de gran exactitud y por su dificultad no pueden ser realizados por operadores humanos.

Las acciones de control clásica (controles lineales) y el control moderno (control por modos deslizantes de segundo orden o control no lineal) fueron abordados en este capítulo dado que experimentalmente se aplicaron en el control de movimiento del robot antropomórfico, primero en su modelo dinámico y, posteriormente en el ambiente virtual, con el propósito de representar los movimientos dinámicos y tener una visualización tridimensional.

Los controladores clásicos compensan a la variable de control en regiones lineales, por lo que para sistemas altamente no lineales como un robot manipulador tienen bajo desempeño. El control no lineal descrito permite compensar las dinámicas no lineales inherentes al robot, como los efectos inerciales, tribológicos y gravitatorios, sin necesariamente depender de la dinámica de la planta. Esta situación lo hace merecedor de ser uno de los mejores controladores para sistemas industriales, incluyendo al caso de estudio.

Adicionalmente en este capítulo se presentó de manera concreta la estrategia de sintonización de los parámetros de los controladores empleados, recurriendo a métodos convencionales empleados por la comunidad de control.

Capítulo 4

Visualización virtual

4.1. Introducción

Una forma de representar las cosas en tiempo real es a través de una visión virtual en donde el usuario pueda interactuar en un ambiente y tener control de los movimientos ofreciendo así un realismo. Las diferentes herramientas que logran que esto pase son diferentes y que en complemento logran hacer que el usuario tenga una visualización más real.
En este capítulo se realizó un estudio comparativo de algunas herramientas y lenguajes utilizadas en la actualidad para el diseño de ambientes virtuales, en este estudio se pueden apreciar las principales ventajas y desventajas que nos proporcionan cada uno de ellos para definir cual ha de ser el lenguaje y herramienta utilizada en el presente trabajo para la construcción del robot.
Así también se describe el funcionamiento del modelo cinemático y dinámico en el ambiente virtual y las principales diferencias visuales entre ambos modelos.

4.2. Conceptos preliminares de realidad virtual

La realidad virtual es una representación de las cosas a través de medios electrónicos, que nos da la sensación de estar en una situación real en la que podemos interactuar con lo que nos rodea, consiste en simular todas las posibles percepciones de una persona, como los gráficos para la vista, sonido, tacto e incluso sensaciones de aceleración o movimiento [25].
Todas estas sensaciones diferentes deben ser presentadas de forma que uno se sienta inmerso en el universo generado por el ordenador, hasta el punto de dejar de percibir la realidad y ser engañado, sentirse transportado (al otro lado de la pantalla) como si se tratara de un universo nuevo. Definimos a la Realidad Virtual como un medio ambiente interactivo, tridimensional, generado por computadoras, en el cual se sumerge a una persona. Es la forma en que los humanos visualizan, manipulan e interactuan con computadoras y datos extremadamente complejos [3].
La realidad virtual puede ser de dos tipos:inmersiva y no inmersiva [26].

Los métodos inmersivos de realidad virtual con frecuencia se ligan a un ambiente tridimensional creado por computadora el cual se manipula a través de cascos, guantes u otros dispositivos que capturan la posición y rotación de diferentes partes del cuerpo humano. La realidad virtual no inmersiva utiliza medios como el que actualmente nos ofrece Internet en el cual podemos interactuar en tiempo real con diferentes personas en espacios y ambientes que en realidad no existen sin la necesidad de dispositivos adicionales a la computadora.

La realidad virtual no inmersiva ofrece un nuevo mundo a través de una ventana de escritorio. Este enfoque no inmersivo tiene varias ventajas sobre el enfoque inmersivo como: bajo costo , fácil y rápida aceptación de los usuarios. Los dispositivos inmersivos son de alto costo y generalmente el usuario prefiere manipular el ambiente virtual por medio de dispositivos familiares como son el teclado y el ratón en lugar de cascos pesados o guantes.

Existen diferentes tipos de inmersión como lo son: [47]

1. `Visual:`(foto-receptivo)
   - Estímulos: luz
   - Información externa: tamaño, forma, distancia, localización, color, textura y movimiento.
2. `Táctil:`(mecánico,térmico)
   - Estímulos: presiones, calor y humedad.
   - Información externa: formas, rugosidad, solidez, viscosidad, temperatura, etc..
3. `Orientador`(mecánico)
   - Estímulos: fuerzas
   - Información externa: magnitud y dirección
4. `Auditivo`(mecánico)
   - Estímulos: vibraciones en el aire información
   - Externa: naturaleza y localización de las vibraciones
5. `Olfativo-Gustativo:`(químico)
   - Estímulos: química de vapores y objetos ingeridos.
   - Información externa: Valores bioquímicos y naturaleza de los olores.

Actualmente Internet nos provee con medios para reunirnos con diferentes personas en el mismo espacio virtual. En este sentido Internet tiende a ser un mecanismo de telepresencia. Este medio nos brinda con espacios o realidades que físicamente no existen pero que sin embargo forman parte de nuestras formas de vida.

4.2.1. Herramientas de edición

Existen diferentes herramientas de edición que ayudan ala creación de ambientes virtuales tales como: Autocad, Maya, 3D StudioMax, entre otras, de las cuales mencionamos las siguientes herramientas:

1. **Autocad** es una herramienta para la elaboración de diseños asistidos por computadora. Empleando diversos comandos para el dibujo, como pueden ser lineas, circulos, etc.[27]

   a) Ventajas
   - Dibujar de una manera ágil, rápida y sencilla, con un acabado perfecto.
   - Permite intercambiar información no solo por papel, sino mediante archivos, y esto representa una mejora en rapidez y efectividad a la hora de interpretar diseños, sobretodo en el campo tridimensional. Con herramientas para gestión de proyectos podemos compartir información de manera eficaz e inmediata. Esto es muy útil sobretodo en ensamblajes, contrastes de medidas, etc.
   - Es importante en el acabado y la presentación de un proyecto o plano, ya que tiene herramientas para que el documento en papel sea perfecto, tanto en estética, como,en información, que ha de ser muy clara. Para esto tenemos herramienta de acotación, planos en 2D a partir de 3D, cajetines, textos, colores, etc.. Sobre todo de métodos de presentación fotorrealísticos.
   - Un punto importante es que AUTOCAD se ha convertido en un estándar en el diseño por ordenador muy versátil, pudiendo ampliar el programa base mediante programación (Autolisp, DCL, Visual Basic, etc ...).
   - Por lo mismo existen más programas específicos de cada campo basados en AutoCAD como:

- Autocad Architectural desktop: centrado en arquitectura e ingeniería de edificios.
- Autocad Map, World, Mapguide: para sistemas de información geográfica y cartografía.
- Autocad Mechanical: con añadidos para optimizar producción mecánica, normalización de piezas, cálculos de ingeniería, etc.
- Mechanical Desktop: preparado para el diseño mecánico en 2D y 3D, análisis y fabricación necesarias para la producción.
- Añade el concepto de información paramétrica, un nuevo campo revolucionario en el entorno CAD.

b) Desventajas

- A pesar de que trabaja con objetos 3D, poco puede implementarse al momento de realizar la animación, pues la programación Visual Basic es el lenguaje con el que se puede trabajar.
- La poca disponibilidad que hay en Autocad para trabajar con Visual C++.
- Requiere de un mayor tiempo para su entendimiento y sacar el máximo provecho de el.
- Aunque autodesk dispone de diferentes módulos que se refirieren al diseño tridimensional. Estos módulos en general son bastante malos , sobretodo comparados con otros productos existentes en el mercado.
- El precio que tiene en estos momentos Autocad en el mercado, muchos requerimientos de hardware, ya que para ello mínimo requiere de un Pentium II con las mejores características.[28]

2. **Maya**

Maya es un software de modelado 3D, especialmente diseñado para la animación de personajes y la creación de efectos especiales.
Para tener un control muy preciso del movimiento de los personajes cuenta con un conjunto de herramientas de IK (Inverse Kinematics) entre las que destacan:

- Control de la animación facial a partir de un conjunto de Blend Shapes.
- Herramientas de modelado de la piel.
- Control de las expresiones de los personajes.
- Soporte de captura de movimiento en vivo.
- Sincronización sonora integrada.

a) Características

Entre las principales características de animación de personajes en 3D implementadas en Maya se encuentran [30]:

- Deformaciones que se pueden agrupar, reordenar y animar.
- Un conjunto de herramientas de IK (Inverse Kinematics) para tener el control preciso sobre el movimiento de los personajes.
- Control de la animación facial a partir de un conjunto de Blend Shapes.
- Herramientas de modelado de la piel.
- Control de las expresiones de los personajes.
- Soporte de captura de movimiento en vivo.
- Sincronización sonora integrada.

Maya también incorpora un avanzado conjunto de herramientas para la creación de efectos visuales, como la caída de agua sobre un molino, la simulación de la colisión de dos galaxias, entre otros.

b) Aplicaciones

- Diseño arquitectónico
- Animación en general

- Dinámica - Modelado - Renderizado

c) Requerimientos [31]

- Hardware: Procesador SGI R4000 o superior, gráficos OpenGL de 24-bit, 128 Mb de RAM
- Software: IRIX 6.2 o superior.

d) Ventajas
Maya es de arquitectura abierta Alias/Wavefront es la compañía fabricante del software Maya (forma parte del corporativo SGI, antes Silicon Graphics). Su filosofía es manufacturar un software con las mejores herramientas para el diseño y el entretenimiento, de ahí que Maya sea de arquitectura extensible y abierta.

Los precios de Maya es de 1,999 dls. (corre en Mac OSX, Windows NT/2000, Linux e Iris). Y el precio de Maya Unlimited es de 6,999 dls. (para Windows NT/2000/XP, Linux e Irix). La cuota de mantenimiento vale 1,299 dls. para Complete y 1,499 dls. para Unlimited (permite tener acceso a soporte técnico desde Canadá y a las nuevas versiones que se liberen durante el año, que comúnmente son dos ó tres). Maya 4.5 incluye ya fluidos, que es lo más avanzado. Fluidos son gases, viscosidad, vapores (con esto se puede hacer un oceáno realista o una bomba atómica) [31].

3. **3D Studio Max**

a) Definición
3D Studio Max es una aplicación basada en el entorno Windows (9x/NT) que permite crear tanto modelados como animaciones en tres dimensiones (3D) a partir de una serie de vistas o visores (planta y alzados).

La utilización de 3D Studio Max permite al usuario la fácil visualización y representación de los modelos, así como su exportación y salvado en otros formatos distintos del que utiliza el propio programa. Además de esta aplicación, existen muchas otras con los mismos fines, como pueden ser, por ejemplo, Maya, LightWave, etc.

Posee poderosas y exclusivas herramientas de trabajo. MAX como suele denominarse comúnmente exige tener una idea previa del trabajo antes de llevarlo al desarrollo para poder sacar el máximo partido de sus herramientas.

MAX es un programa de gran tamaño capaz de realizar tareas impensables hace unos años, sin embargo, los requisitos para su funcionamiento están muy por debajo de los que tienen la mayoría de los ordenadores actuales. Los requerimientos son mínimos para que el programa funcione, pero puede que las escenas muy grandes o complejas necesiten otros recursos para poder trabajar con ellas adecuadamente.

El formato de dibujo empleado en 3D Studio Max es por defecto " MAX " es decir, todos los modelos tendrán extensión " .MAX ", aunque bien es cierto que también se pueden guardar en otros formatos.

b) Entorno
El entorno representa lo que se muestra en la figura 4.1 una vez que accedemos al programa en cuestión. En este caso, se deben tener en cuenta todos los menús que aparecen : barra de menús, visores, barra de herramientas, panel de comandos menú de animación y zoom.
Dentro del entorno, el mayor espacio lo ocupan los visores (por defecto son: superior, anterior, izquierda y perspectiva). Para cambiar el modo de representación de los visores se hace clic con el botón derecho del ratón en el nombre de cada visor y se selecciona en el menú pop-up que aparece en pantalla.

c) Objetos y paramétricos

El elemento básico de MAX es el objeto. Un objeto es cualquier elemento individual que forme parte de la escena. Puede ser un objeto en 2d,en 3d, ayudante, luz, cámara o cualquier otro elemento que

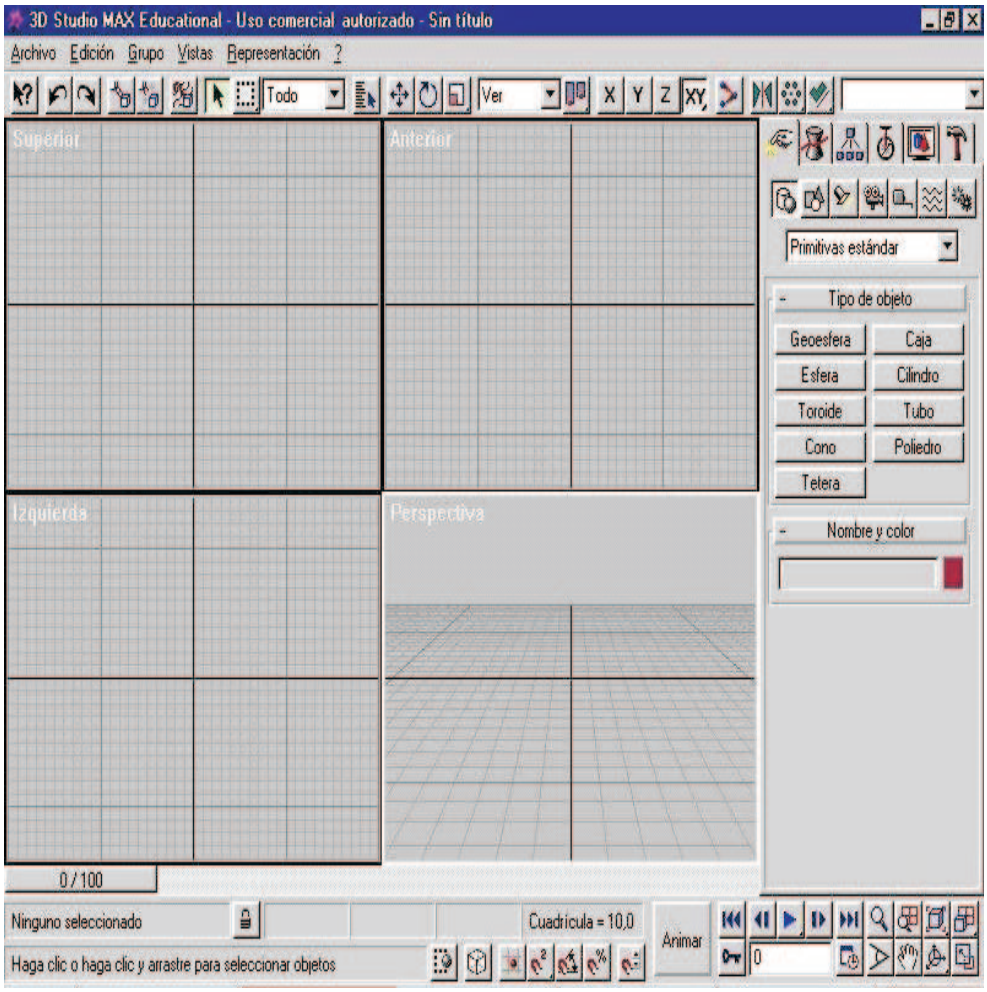

Figura 4.1: Entorno de 3DSMAX mostrando sus 4 visores.

se pueda crear en MAX. Cuando se crea un objeto, este se convierte en un miembro de la escena activa.

Por debajo del nivel del objeto, la mayoría de los objetos están compuestos por sub objetos. Los objetos 3d están compuestos por vértices, aristas, polígonos y a veces elementos. Los objetos 2d se pueden dividir en vértices, segmentos y splines. Los sub objetos se pueden editar para cambiar el aspecto y la funcionalidad de un objeto.

En el modelado informático un objeto básico y común creado por una aplicación se conoce como primitiva en MAX, las primitivas pueden ser en 3d (cuadros, conos, esferas, etc) o en 2d (rectángulos, líneas, círculos, etc.)en la mayoría de los casos, las primitivas son paramétricas.

Un objeto paramétrico es aquel cuyo parámetros (es decir, la información que lo define) se puede ajustar sin necesidad de ajustar la posición de los objetos manualmente. En el mundo real se necesitarìa su longitud, anchura y altura. Lo mismo se necesita con la primitivas cajas de MAX, que además cuenta con otros parámetros adicionales que controlan su segmentación (numero de secciones transversales en las que se divide la caja en cualquier dirección).
El material asignado a la caja también es parametrico (el color se puede dividir y se pueden ajustar

sus niveles de rojo, verde y azul [RBJ]), al igual que los modificadores que se aplican a la caja. Para aumentar la altura de un objeto con forma de caja pero no paramétrico de 2 a 5 unidades, seria necesario mover los dos vértices superiores a 3 unidades desde la base. Con un objeto caja paramétrico, solo tendría que introducir "5" en el campo del parámetro altura para poder ver el efecto reflejado en el objeto.
En todas las aplicaciones 3d, la mayoría de las aplicaciones de las paramétricas son primitivas , pues esta característica supone un importante ahorro de tiempo.

d) Requerimimientos[34]

- Procesador Intel PIII o superior 3ds max.
- Puede convivir con multiples procesadores.
- RAM y 500 Mb Swap Recomendada 1GB RAM y 2 GB Swap Tarjeta Grafica que soporte como mínimo 1024x76 8 x16-bit DirectX 8.1 o superior Hardware acelerador de OpenGL y Direct3D.
- Windows® 2000 (Service Pack 4), Windows XP Professional (Service Pack1) o Windows XP Home (Service Pack 1).
- 650 MB en espacio libre en disco duro.

e) Ventajas

- Altas resolucines gráficas que permiten una visión casi real de los objetos dando una alta calidad en animación, conteniendo 4 visores para distintas vistas para poder visionar lo que haces al momento.
- Documentación hay una disponibilidad excelente en cuanto a material, como son tutoriales.
- Facilidad de instalación e instalación rápida.
- Fácil de manejar, con mucha variedad, buena relación entre modelado, animación y puesta en escena.

4.2.2. Lenguajes

VRML(Lenguaje para Modelado de Realidad Virtual)

VRML significa Lenguaje para Modelado de Realidad Virtual (Virtual Reality Modeling Language) , básicamente lo que hace es convertir la www en un mundo de objetos tridimensionales en el cual el usuario será capaz de interactuar, es decir, el navegador tendrá la facultad de introducirse, no sólo observarlo, a este mundo por medio de ciertos dispositivos conocidos y de fácil manejo como lo son el Mouse, el Joystick, el teclado, etc. Un ejemplo de lo que podemos hacer con VRML es la representación de un robot con el cual el usuario puede interactuar y esto se muestra en la figura 4.2 [35]. El lenguaje para modelado de realidad virtual está diseñado para permitir que los autores de páginas o sitios web incrusten información sobre mundos tridimensionales y en forma tridimensional, mediante el cual se convierta a la www en una simulación interactiva, gracias a un lenguaje estándar VRML, en mundos que contengan objetos que tienen hiperenlaces a otros mundos. VRML es un lenguaje de descripción de escenas en el que cada escena se compone de un número de objetos.

Los objetos pueden ser formas sólidas situados y orientados de determinada forma u elementos intangibles que afectan a la escena como luces, sonido y distintos puntos de vista. El VRML es un lenguaje para descripción de escenas no un lenguaje de programación.
Algunos lenguajes de cómputo como "C", primero compilan el programa y después se ejecutan. El sistema VRML pasa por un análisis sintáctico antes de ser desplegado en pantalla. La descripción de escenas es un proceso estático, pues los elementos no cambian cuando el archivo VRML se carga. De hecho, es posible cambiar el punto de vista, pero no la escena misma [36].

- Aplicaciones , ventajas y desventajas.

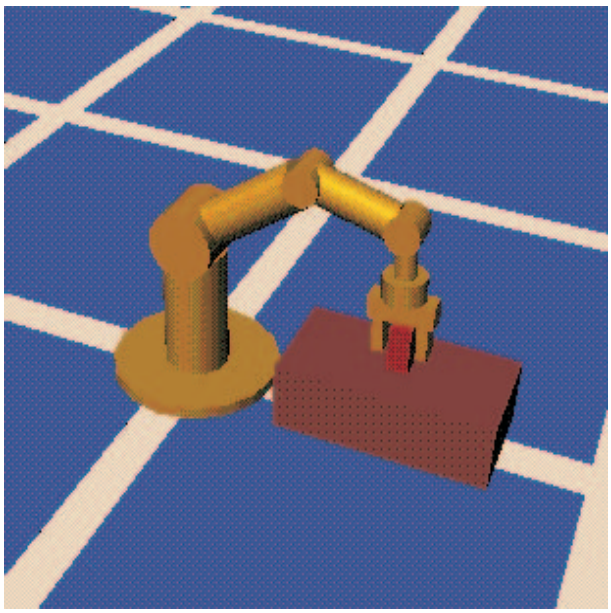

Figura 4.2: Ejemplo del escenario de un robot con el cual el usuario puede interactuar [35].

El manejo y operación de software en base a VRML no es complicado aunque requiere de conocimiento de diseño en el espacio con coordenadas tridimensionales.

Una de las ventajas de VRML es que se trata de una norma en continua evolución y su implementación se facilita por el bajo costo (gratis en muchos casos) del software que se encuentra dentro de Internet. Los editores de VRML tienen algunas ventajas: se pueden crear escenarios complejos, algunos programas de modelado gráfico en 3D puede exportar los ficheros al formato VRML.

El lenguaje para modelado de realidad virtual está emergiendo como uno de los candidatos más prometedores de entre una variedad de contendientes en representación tridimensional. Este lenguaje está diseñado para permitir que los autores web incrusten información sobre mundos tridimensionales en las páginas web.

Algunas de las ventajas que ofrece este lenguaje son [37]:

- Aumentar o disminuir el tamaño de los objetos, así como girarlos, inclinarlos o incluso voltearlos de cabeza, caminar por los espacios; es introducirse e interactuar en el mundo virtual sin necesidad de moverte de tu silla, el único requisito para ello es que la computadora cuente con un componente adicional (navegador) que permite justamente realizar todo lo mencionado.
- Independencia de plataformas: VRML define únicamente la escena como objetos en 3D y algunas interacciones de la escena con el usuario.
- Extensibilidad: VRML define la escena a través de nodos con propiedades , donde cada nodo puede ser una esfera, un cono, una transformación, etc. Para agregar nuevos objetos ala especificación basta con agregar nodos. También es posible agregar propiedades a nodos ya existentes.
- Capacidad de trabajar bien en conexiones de poco ancho de banda : VRML permite ir presentando los objetos conforme son leídos del archivo , por lo que no es necesario esperar que todo el archivo sea leído para luego presentarlo.

Pero también tienen desventajas: todos los editores de VRML son comerciales y el código generado en estos

programas puede ser mucho mas voluminoso y conseguir los mismo efectos que con el método manual.

OpenGL

1. Historia de OpenGL
   En 1982 nació en la Universidad de Standford el concepto de "graphics machine" y este fué utilizado por Silicon Graphics Corporation en su propia estación Silicon IRIS para crear un renderizador. Así nació la librería IRIS GL. A raíz de esto, por 1992 muchas empresas del hardware y software se pusieron de acuerdo para desarrollar conjuntamente una librería gráfica libre: OpenGL [37].

   Entre estas empresas destacaban Silicon Graphics Inc., Microsoft, IBM Corporation, Sun Microsystems, Digital Equipment Corporation (DEC), Hewlett-Packard Corporation, Intel e Intergraph Corporation. Así nació OpenGL (Open Graphics Library).

2. Definición
   OpenGL como se mencionó antes significa Graphic Library. No es por lo tanto ningún lenguaje de programación, sino tan sólo un conjunto de librerías que son utilizadas a través de lenguajes de programación (en este caso Visual C++) para conseguir un interfaz software entre las aplicaciones y el hardware gráfico. La librería esta formada por unas 150 instrucciones diferentes que se utilizan para especificar los objetos y las operaciones necesarias para desarrollar aplicaciones interactivas tridimensionales. Los modelos se deben construir partiendo de un pequeño conjunto de "primitivas geométricas" como puntos, líneas y polígonos.

   OpenGL es sin lugar a dudas la API que prevalece en la industria para desarrollar aplicaciones gráficas 2D y 3D. Se le considera el sucesor de IRIS GL-library de Silicon Graphics que hizo popular las estaciones de trabajo SGI como plataforma predilecta para desarrollo científico, de ingeniería y de efectos especiales.Un ejemplo de lo que podemos hacer con OpenGL es la figura 4.3 que se muestra a continuación [38].

Figura 4.3: Ejemplo de OpenGL que muestra un escenario [38].

En contraste con la antigua IRIS GL-library de SGI, OpenGL es por diseño independiente de plataformas y sistemas operativos. Es perceptiva a la red, de manera que es posible separar nuestra aplicación OpenGL en un servidor y un cliente que verdaderamente produzca los gráficos.

3. Características
   OpenGL permite al desarrollador escribir aplicaciones que se puedan desplegar en varias plataformas fácilmente. La característica de ser "Abierta"significa que un programa escrito para una plataforma puede ser fácilmente convertible a cualquier tipo de plataforma, obteniendo prácticamente los mismo resultados. Sin entrar en demasiados detalles, a continuación se describen algunas de las características que OpenGL implementa [39]:

   - Primitivas geométricas: permiten construir descripciones matemáticas de objetos. Las actuales primitivas son: puntos, líneas, polígonos, imágenes y bitmaps.
   - Codificación del color en modos RGBA (rojo-verde-azul-alfa) o de color indexado.
   - Visualización y modelado que permite disponer objetos en una escena tridimensional, mover nuestra cámara por el espacio y seleccionar la posicíon ventajosa deseada para visualizar la escena de composición [39].
   - Mapeado de texturas que ayuda a traer realismo a nuestros modelos por medio del dibujo de superficies realistas en las caras de nuestros modelos poligonales.
   - La iluminación de materiales es una parte indispensable de cualquier grágico 3D. OpenGL provee de comandos para calcular el color de cualquier punto dadas las propiedades del material y las fuentes de luz en la habitación.
   - El doble buffering ayuda a eliminar el parpadeo de las animaciones. Cada fotograma consecutivo en una animación se construye en un buffer separado de memoria y mostrado solo cuando está completo [39].
   - El anti-alizado reduce los bordes escalonados en las líneas dibujadas sobre una pantalla de ordenador. Los bordes escalonados aparecen a menudo cuando las líneas se dibujan con baja resolución. El anti-alizado es una ténica común en gráficos de ordenador que modifica el color y la intensidad de los pixels cercanos a la línea para reducir el zig-zag artificial.
   - El sombreado Gouraud es una técnica usada para aplicar sombreados suaves a un objeto 3D y producir una sutil diferencia de color por sus superficies.
   - El Z-buffering mantiene registros de la coordenada Z de un objeto 3D. El Z-buffer se usa para registrar la proximidad de un objeto al observador, y es también crucial para el eliminado de superficies ocultas [39].
   - Efectos atmosféricos como la niebla, el humo y las neblinas hacen que las imágenes producidas por ordenador sean más realistas.
     Sin efectos atmosféricos las imágenes aparecen a veces irrealmente nítidas y bien definidas. Niebla es un término que en realidad describe un algoritmo que simula neblinas, brumas, humo o polución o simplemente el efecto del aire, añadiendo profundidad a nuestras imágenes.
   - El Alpha blending usa el valor Alfa (valor de material difuso) del código RGBA, y permite combinar el color del fragmento que se procesa con el del pixel que ya está en el buffer. Imaginemos por ejemplo dibujar una ventana transparente de color azul claro enfrente de una caja roja. El Alpha blending permite simular la transparencia de la ventana, de manera que la caja vista a través del cristal aparezca con un tono magenta [39].
   - Los planos de plantilla permiten restringir el trazado a ciertas regiones de la pantalla.
   - Las listas de Display permiten almacenar comandos de dibujo en una lista para un trazado posterior, cuando las listas de display se usan apropiadamente puedan mejorar mucho el rendimiento de nuestras aplicaciones.
   - Los evaluadores polinómicos sirven para soportar B-splines racionales no uniformes, esto es para ayudar a dibujar curvas suaves a través de unos cuantos puntos de referencia, ahorrándose la necesidad de acumular grandes cantidades de puntos intermedios [38].
   - Características de feedback,selección y elección que ayudan a crear aplicaciones que permiten al usuario seleccionar una región de la pantalla o elegir un objeto dibujado en la misma. El modo de feedback permite al desarrollador obtener los resultados de los cálculos de trazado.

- Primitivas de Raster (bitmaps y rectángulos de pixels)[38].
- Operaciones con pixeles.
- Transformaciones: rotación, escalado, perspectivas en 3D.

4. Ventajas
   Para los que no lo conocen, Direct3D (o D3D) es otra API grafica 3D desarrollada por Microsoft. Para el desarrollo bajo Windows puede usarse cualquiera de las dos, pero con D3D se queda restringido solo al entorno Windows. OpenGL esta diseñada para ser utilizado en casi cualquier sistema operativo, ya sea Mac OS, Linux, Unix, Windows o Solaris. Aunque OpenGL está concebido para diseñar aplicaciones interactivas y facilita al usuario herramientas como la selección, sus capacidades resultan insuficientes para, entre otras cosas, crear interfaces gráficas con un grado mayor de interactividad.

   Estas limitaciones condujeron al desarrollo de las librerias AUX y GLUT . Las librerias AUX presentan numerosas insuficiencias, y su aplicación se limita. Por encima de todo, OpenGL es una biblioteca estilizada de trazado de gráficos de alto rendimiento, hay varias tarjetas gráficas aceleradoras y especializadas en 3D que implementan primitivas OpenGL a nivel hardware. A cambio, OpenGL, ofrece algo muy valioso : la independencia con respecto a la plataforma de hardware y el sistema operativo en que se trabaje, brindando con ello una enorme portabilidad a sus productos.

   Así, OpenGL, permite:

   - Construir formas geométricas a partir de primitivas.
   - Ubicar los objetos en el espacio tridimencional y seleccionar el punto de vista de la escena.
   - Aplicar el color a los objetos, ya sea mediante una asignación explícita de la aplicación, a partir de las condiciones de iluminación o mediante la utilización de texturas.
   - Convertir la descripción matemática de los objetos y la información sobre el color en pixels de la pantalla, proceso que se llama rasterización .

Transformaciones y coordenadas homogéneas en OpenGL

Una de las técnicas más poderosas de modelado de entidades gráficas consiste en descomponer ingeniosamente las mismas en términos de primitivas. Ya con anterioridad se ha hecho referencia a estas primitivas que permiten una descomposición jerárquica que es más rica y versátil. De esa manera, las entidades graficas son jerárquicas de estructuras, que se definen como la composición de instancias de estructuras más simples. Asi que se debe definir una única vez la estructura. Para lo cual se debe tomar en cuenta lo siguiente [41].

- Transformaciones afines: para utilizar una entidad varias veces, es decir, para que ocurran varias instancias de una entidad, es necesario hacerla “aparecer” varias veces con distintas transformaciones. Cada elemento de graficación estará definido también con transformaciones de otros elementos, las cuales deben concatenarse con la transformación correspondiente a cada casa para obtener el resultado final. Por lo tanto, las transformaciones desempeñan un papel decisivo en los modelos de la computación gráfica, en particular las transformaciones rígidas o lineales.

  La traslación, rotación (alrededor del origen) y escalamiento, conforman una base funcional para las transformaciones rígidas en un espacio lineal. Es decir, toda transformación afín o lineal puede ponerse en términos de una aplicación de estas tres operaciones.

- Coordenadas homogéneas: es necesario tener en cuenta que los espacios lineales que son conjuntos de valores vectoriales cerrados bajo suma y multiplicación escalar . Por lo tanto, los puntos en estos espacios no tienen una representación que los distinga. En un espacio vectorial de n dimensiones no afín, es posible encontrar subespacios afines de n -1 dimensiones, es decir, subespacios en los cuales la combinación afín de elementos esta bien definida.

Recíprocamente, todo espacio E de n dimensiones puede hacerse afín dentro de un espacio vectorial V de n + 1 dimensiones por medio de un procedimiento denominado homogenización. Para ello se considera que E (espacio) esta "inmerso" dentro de V (vector) como un hiperplano para un valor no trivial (distinto de cero) de una nueva variable h llamada variable de homogenización. Todo elemento del espacio homogéneo debe considerarse proyectado sobre el plano $h = 1$. De esa manera, la suma de puntos se transforma en una combinación afín.

4.2.3. Visual C++

El entorno virtual desarrollado, se encuentra bajo esta plataforma, además Windows ha sido bastante aceptado por los beneficios de su interfaz gráfica (GUI). Las características más importantes de visual C++, son:

- Procesamiento de mensajes: el sistema operativo llama a main cuando el usuario ejecuta un programa.
- Interfaz de dispositivo gráfico de Windows: se introdujo un nivel de abstracción llamado la interfaz de dispositivo gráfico (GDI) que proporcionan controladores de video e impresora conectadas al sistema; En vez de dirigirse al hardware, el programa llama a las funciones de GDI que hacen referencia a una estructura de datos llamada contexto de dispositivo, Windows relaciona el contexto de dispositivo con el dispositivo físico y emite las instrucciones de entrada/salida adecuadas.
- Pogramación basada en recursos: para realizar una programación dirigida por datos en MS-DOS, es necesario codificar los datos como constantes de inicialización o proporcionar archivos de datos aparte para que los lea el programa.
- Gestión de memoria: algunas de las funciones de gestión de memoria de Win16, como el GlobalAlloc, se trasladaron a Win32, pero se hizo para permitir a los desarrolladores modificaran el código fuente con rapidez al pasar de una versión a otra.
- Bibliotecas de enlace dinámico: permite el enlazado dinámico, lo que significa que pueden cargar y enlazar en tiempo de ejecución bibliotecas construidas de una forma especial. Múltiples aplicaciones pueden compartir bibliotecas de enlace dinámico (DLL), lo que ahorra memoria y espacio en disco.

Principales componentes de Visual C++

- El AppWizard: es un generador de código que crea un esqueleto de trabajo de una aplicación de Windows con características, nombres de clases y nombres de archivos código que se especificas a través de cuadros de diálogo. El código del AppWizard es mínimo; la funcionalidad se encuentra en las clases de base del armazón de la aplicación.
- El classWizard: es un programa implementado como DLL, simplifica el mantenimiento del código de clase de Visual C++; en caso de requerir una fusión virtual nueva o una función de gestión de mensajes, el ClassWizard escribe los prototipos, cuerpos de de la función y se es necesario , el código para enlazar el mensaje de Windows a la función.

4.3. Modelo Conceptual

4.3.1. Análisis de requerimientos

Como parte del análisis de requerimientos del sistema se realizó una entrevista al Doc. Omar Arturo Domínguez Ramírez especialista en el area de robótica a fin de establecer las necesidades que se presentan en la enseñanza y entrenamiento de operadores de robots antropomórficos para llevar a cabo la elaboración del sistema virtual desarrollado en el presente tema de tesis.

Entrevista para la validación de los propósitos del sistema desarrollado

- ¿Cuál es la importancia de la robótica en la actualidad?
  Los sistemas de control automático empleados en la industria con propósitos de realizar tareas repetitivas, con ciertos niveles de inteligencia y con el afán de efectuar toma de decisiones durante su operación dentro de un proceso de producción, están basados en mecanismos robóticos que efectúan el desplazamiento de piezas, herramientas y dispositivos especializados, describiendo trayectorias óptimas constitutivas de una tarea de movimiento.
  Los robots en la industria, incrementan la productividad sin sacrificar el índice de calidad de un producto, aminorando de manera considerable los costos de producción, sin embargo, existen innumerables aplicaciones de los robots en la actualidad, por ejemplo, en exploración local y remota, teleoperación de sistemas remotos, robots en rehabilitación, robots para entrenamiento, para entretenimiento, robots en cirugía de alta precisión, entre otras muy importantes.

- ¿Cuáles son las limitantes que se tienen en la enseñanza y el entrenamiento para operadores de robots hoy en día?
  Los principios formales para interpretar el funcionamiento de un robot manipulador, con el afán de reprogramar y planificar movimientos y tareas de alto desempeño, resultan nada triviales, y requieren de una base de conocimientos matemáticos importantes en las áreas de análisis vectorial, álgebra lineal y ecuaciones diferenciales.
  Las tareas planificadas en robots manipuladores de manera profesional, están basadas en la generación de trayectorias óptimas, libres de colisión y esfuerzos electromecánicos, sin dejar de dar atención a las formas descriptivas de posiciones y velocidades articulares, así como las definidas en el espacio operacional y el tiempo de convergencia. Existen menos de 10 libros de robótica en español, dignos de ser consultados, el resto están publicados en otros idiomas, generalmente en ingles, francés y alemán.
  La mayoría de los principiantes en robótica carecen del conocimiento de un idioma, como los mencionados previamente así como de bases matemáticas sólidas que permitan entenderlo rápidamente. Son muy escasas las herramientas didácticas dedicadas a la enseñanza de la robótica, y de las que existen, en su mayoría abordan aspectos clásicos de cinemática y controles lineales, sin abordar los fenómenos dinámicos inherentes de todo sistema robótico como un mecanismo altamente no lineal, y las estrategias de control no lineal dedicadas a estabilizar al robot en una región local o global, es decir para regulación o seguimiento de trayectorias.

- ¿Considera que la implementación de un escenario virtual, que permita el aprendizaje y el entrenamiento, sería una solución?
  Con propósitos de enseñanza, entrenamiento ó investigación, el contar con un robot manipulador industrial resulta difícil, tal situación puede ser remplazada de manera eficiente por un robot virtual con un comportamiento dinámico definido por las leyes físicas que rigen a un robot real. Si se cuenta con un ambiente virtual, constituido por un mecanismo robótico basado en su modelo dinámico y cinemático, es posible diseñar estrategias de control no lineal de movimiento y entender más los principios de operación.
  Las técnicas de diseño de controladores de movimiento para robots, consideran la simulación digital en lazo cerrado, en las que se modifican las ganancias de los controladores y se modifican a voluntad los coeficientes dinámicos del robot en fase de diseño, sin embargo, no es posible conocer el desempeño del control y del robot en su conjunto ya que las señales representativas de posición, velocidad y control, son proporcionadas en régimen del tiempo y de manera separada, sin visualizar los movimientos simultáneos en cada una de sus articulaciones, esto se resuelve con un robot virtual de forma tal que podría tenerse una sintonización eficiente, ajuste de parámetros dinámicos y planificación de trayectorias, esta última puede ser empleada para validar los movimientos antes de ser ejecutados en un robot real modelado en el ambiente virtual por el operador .

- ¿Qué importancia tiene la dinámica en el comportamiento de un robot antropomórfico?
  Todo sistema electromecánico tiene un comportamiento no lineal, los fenómenos que lo rigen están basados en leyes físicas del movimiento relativo de cuerpos en el espacio, describiendo su desempeño en función de estos. Los efectos están definidos por fuerzas inerciales, de fricción, gravitatorias, de Coriolis y centrípetas.

Los modelos clásicos que reúnen a estos fenómenos dinámicos están basados en la formulación de Euler - Lagrange, la comunidad de robótica ha utilizado esta formulación para el diseño y construcción de robots, así como para el diseño de las estrategias de control de movimiento. En consecuencia, la interpretación del comportamiento de un robot durante el desempeño de una tarea, esta basada en estos fenómenos dinámicos y de ellos depende su diseño al igual que sus leyes de control.

- De llevarse a cabo el desarrollo de un escenario virtual, ¿Qué beneficios aportaría la implementación de la dinámica del robot?
  Los movimientos del robot virtual basados en la dinámica de Euler-Lagrange permiten la asignación de comportamiento complejo similar al ocurrido en un robot real, ya que considera a los efectos dinámicos inerciales, de fricción, gravitatorios, de Coriolis y centrípetos, la aplicación del control de movimiento permitiría visualizar el desempeño del robot de una forma mas aproximada a la realidad, situación que beneficiaría el diseño de un robot, su programación de movimientos, el diseño de sus controladores, etc.

- ¿Considera que la implementación de un sistema en línea con acceso remoto a través de la red de Internet traería beneficios?, y ¿Cuáles serían dichos beneficios?
  El hecho de contar con un laboratorio virtual manipulado a través de Internet permite al usuario (estudiantes, investigadores, etc) resolver problemas de diseño cinemático, dinámico y sobre todo de control de movimiento de robots manipuladores de manera inmediata, ya que la visualización virtual, para el caso de estudio descrita por el robot manipulador permite apreciar el desempeño global del robot a partir de la solución instantánea del modelo dinámico definido por el sistema de ecuaciones diferenciales no lineales.

- ¿De cuántos grados de libertad considera sería idóneo, la representación de un robot antropomórfico?
  5 o 6 grados de libertad son los idóneos, debido a que se requieren al menos 3 para definir la posición operacional del órgano terminal y el restante para la orientación. La configuración antropomórfica resuelve el problema de posicionamiento brazo derecho e izquierdo, tal y como un ser humano ejecuta una tarea manipulación con ambos brazos y uno manipulando a la vez.

- ¿Qué beneficios tendría el desarrollo de dicho sistema para la comunidad de robótica?
  La comunidad de robótica emplea generalmente software para simulación y validación de algoritmos de robots, como Matlab, Twente Sim, SIMNON, entre otros menos clásicos, sin embargo la respuesta que proporciona un simulador de este tipo es la respuesta articular u operacional con relación al tiempo, algunos diagramas de fase y resultados que validan la estabilidad y las propiedades dinámicas y cinemáticas del robot, sin embargo, el hecho de poder visualizar el desempeño del robot en su conjunto y poder alterar las características dinámicas del mecanismo de eslabones articulados, así como sintonizar las ganancias de control para mejorar el desempeño brindan de manera objetiva un beneficio sobre el conjunto de simuladores anteriormente mencionados.

- ¿Cuáles son los parámetros que considera necesarios para referenciar a un robot antropomórfico que vaya de un punto a otro, dados por el usuario final del sistema?
  El usuario debe planificar el movimiento del robot tal que describa una tarea, y este debe ser descrito mediante las coordenadas operacionales finales, definidas de manera secuencial. En algunos casos se requiere que el robot describa una trayectoria operacional cerrada, esta debe ser planificada en la interfaz de usuario, sin embargo el caso más demandado para planificación es el de una coordenada operacional dentro del espacio de configuraciones admisibles.

- ¿Qué tipo de control aplicaría para el seguimiento de una trayectoria?
  Un control que no dependa estrictamente del modelo dinámico, que sea robusto, no lineal y de fácil implementación, por ejemplo el control por modos deslizantes de segundo orden es una buena opción.

Dada la necesidad que de incorporar el movimiento dinámico en la interfaz virtual para una mejor comprensión del movimiento por parte del usuario de lo que se esta representando en el ambiente virtual y en base al análisis realizado al cuestionario anterior se hicieron las ecuaciones descritas a continuación:

- El usuario requiere establecer una trayectoria que represente el movimiento del robot de manera inmediata.
- Necesidad de la eliminación del uso de programas externos para la obtención de la solución numérica del modelo dinámico.
- Necesidad de la comprobación de valores generados en archivo.
- Necesidad de comprobación de distintas trayectorias sin que el usuario conozca el lenguaje de programación para su modificación.

La interfaz propuesta en el presente trabajo representa la visualización del desempeño cinemático y dinámico de un dispositivo háptico PHANToM, cuya representación está basada en el diseño de opciones y funciones para su operación y visualización en el ambiente virtual; el modelo conceptual que describe los puntos antes mencionados puede observarse en "Análisis y diseño de una interfaz de visualización virtual del dispositivo háptico PHANToM 1.0 basada en la dinámica de Euler-Lagrange", la implementación hecha en el presente tema de tesis es la inclusión de la dinámica en línea, el análisis realizado para el funcionamiento del mismo se puede apreciar en el diagrama 4.4 donde se presenta a un usuario (Actor) el cuál debe proporcionar los parámetros de entrada desde la consola para la obtención de la trayectoria deseada y de esta forma representar el movimiento cinemático y dinámico.

Se pueden observar dos formas para la realización del movimiento, uno es a través de la ejecución del método de Runge Kutta con los parámetros proporcionados desde la consola el cual genera un archivo denominado Trayectoria.mat que permite la visualización del movimiento del PHANToM en el ambiente virtual; así también desde la consola se puede observar los datos generados con la ejecución del método de Runge Kutta para fines de comprobación de los valores representados del movimiento virtual. La otra forma de movimiento es la que se denomina movimiento manual cuyo estudio se a analizado en el diagramas 4.5,4.6, cuya explicación puede apreciarse en "Análisis y diseño de una interfaz de visualización virtual del dispositivo háptico PHANToM 1.0 basada en la dinámica de Euler-Lagrange".

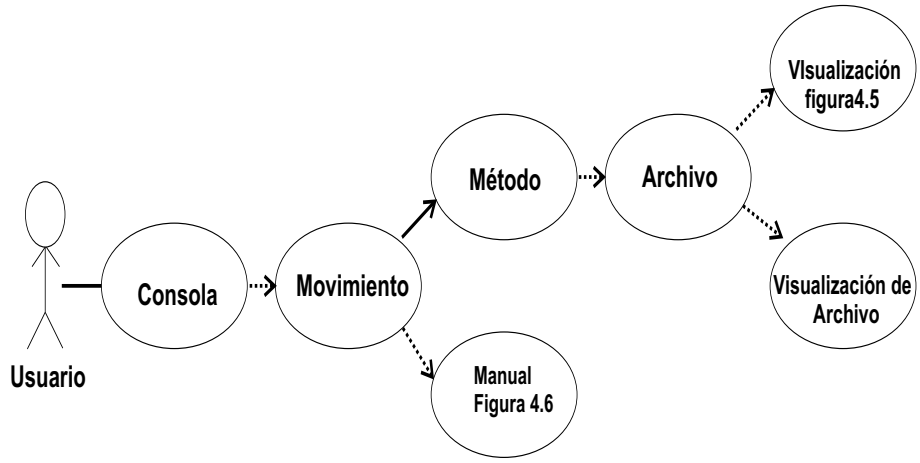

Figura 4.4: Análisis de la interfaz

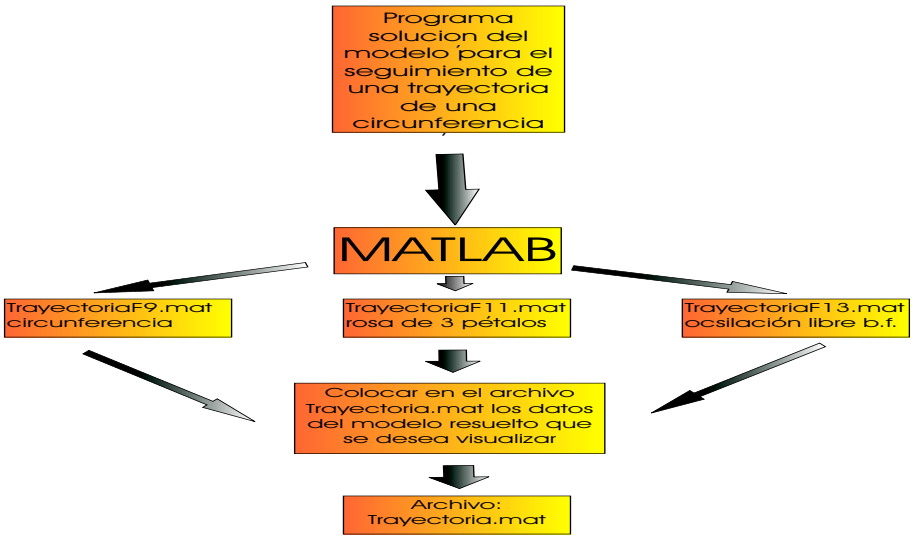

Figura 4.5: Análisis del movimiento manual del PHANToM [13].

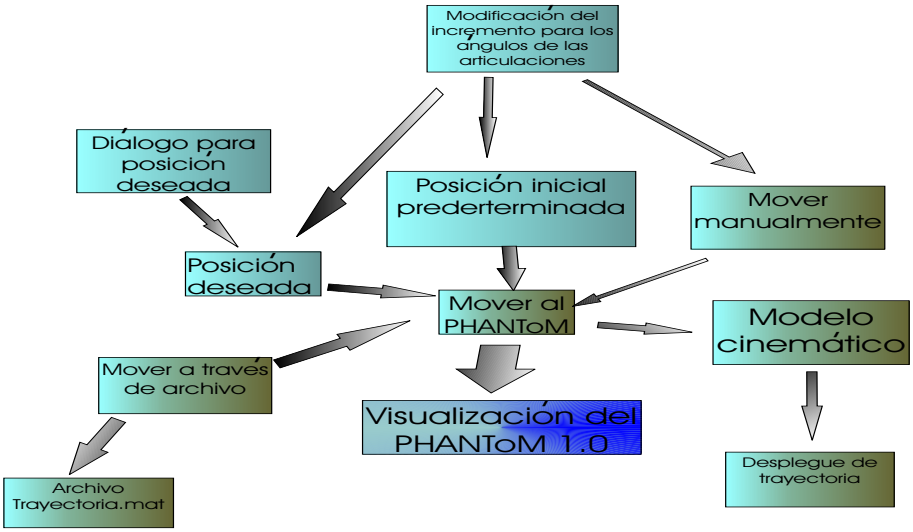

Figura 4.6: Análisis del movimiento por archivo del PHANToM [13].

4.4. Construcción del robot virtual [13]

- Primera aproximación a través de rectángulos. El robot PHANToM esta formado por tres eslabones y tres articulaciones. Los eslabones son representados por rectángulos cuyas funciones definen dibujan las 4 esquinas del rectángulo y son: $Begin(GL_QUADS)$;
  $glVertex3f(0{,}0f, 0{,}0f, 0{,}0f)$;
  glVertex3f(ESP,0.0f, 0.0f);
  glVertex3f(ESP, LongEsl1, 0.0f);
  glVertex3f(0.0f,LongEsl1, 0.0f);

gl End();
La función glRotate(..) proporciona movimiento a cada uno de los eslabones.

- Segunda aproximación a través de paralepipedos. La construcción de un paralepipedo lo conforma la creación de 6 rectángulos, y para su animación se utiliza el mismo criterio de la primera aproximación. La construcción de estos genera un movimiento entre los eslabones como se muestra en la figura 4.7.

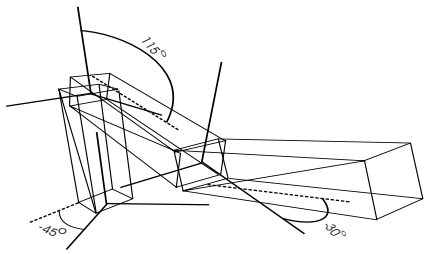

Figura 4.7: Configuración de eslabones con paralepipedos [13]

4.5. Comportamiento complejo

Se ha realizado una comparativa entre el desempeño visual cinemático y dinámico, donde puede observarse que en el desempeño cinematico realiza el movimiento sin ningun problema dado que solo se le proporciona un punto de referencia, mientras que en el dinámico realiza pequeñas oscilaciones pues ya cuenta con propiedades dinámicas con lo que sus movimientos son más parecidos a lo que en la realidad sucede con los robots antropómorficos.

4.5.1. Desempeño visual cinemático

Para integrar el modelo cinemático del robot virtual se tiene la cadena cinemática, la posición y orientación de los ejes definidos en el capítulo 5 en el apartado 5.3. Sustituyendo los ángulos en las ecuaciones del modelo cinemático se obtiene la posición inicial con la que empieza el visualizador, como lo muestra la figura 4.8, cabe aclarar que para que esto suceda, para fines de visualización deberan de agragarse 90 grados al valor del ángulo que manipula la rotación de la segunda articulación.

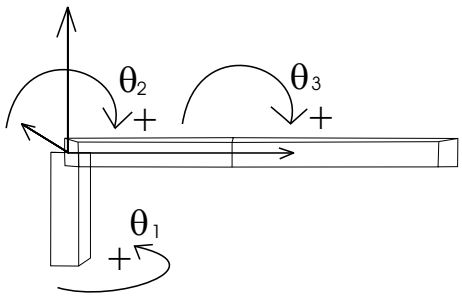

Figura 4.8: Configuración de eslabones al inicio del programa [13]

De esta forma se puede apreciar en el ambiente virtual la visualización descrita por la figura 4.9 en donde puede observarse que dicho movimiento es poco real pues no toma en cuenta factores y características que afectan el comportamiento del robot.

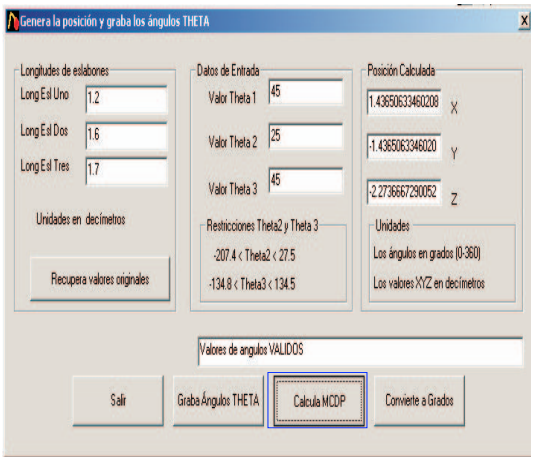

Figura 4.9: Interfaz que muestra el resultado de los cálculos del MCDP

Posteriomente se han de grabar los datos que se generaron con la ejecución del MCDP como se ilustra en la figura 4.10.

Finalmente se puede apreciar el comportamiento del robot en el ambiente virtual se presiona la tecla f4 y así se puede apreciar el comportamiento que se observa en la figura 4.11.

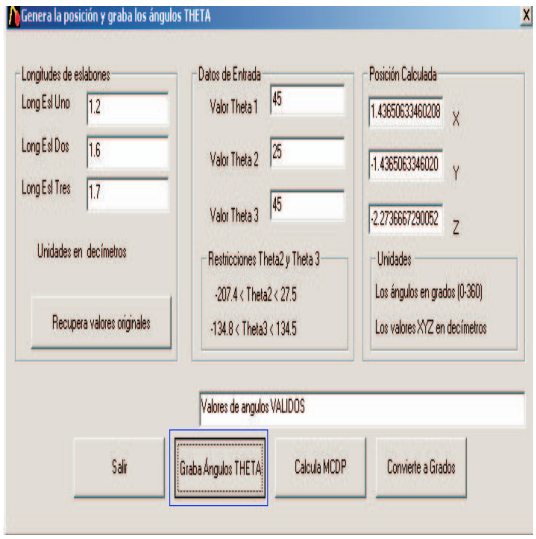

Figura 4.10: Interfaz que muestra la grabación de los datos obtenidos de los cálculos del MCDP.

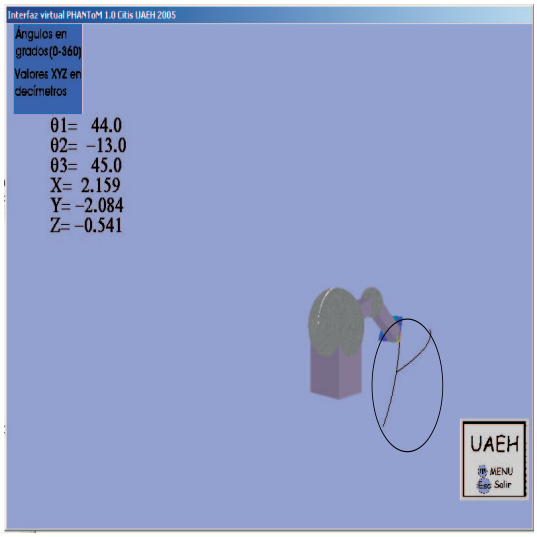

Figura 4.11: Interfaz que muestra el comportamiento MCDP.

4.5.2. Desempeño visual dinámico

La implementación del modelo dinámico en el robot se describe en el siguiente diagrama de flujo de la figura 4.12 para mayor compresión de como funciona el sistema.

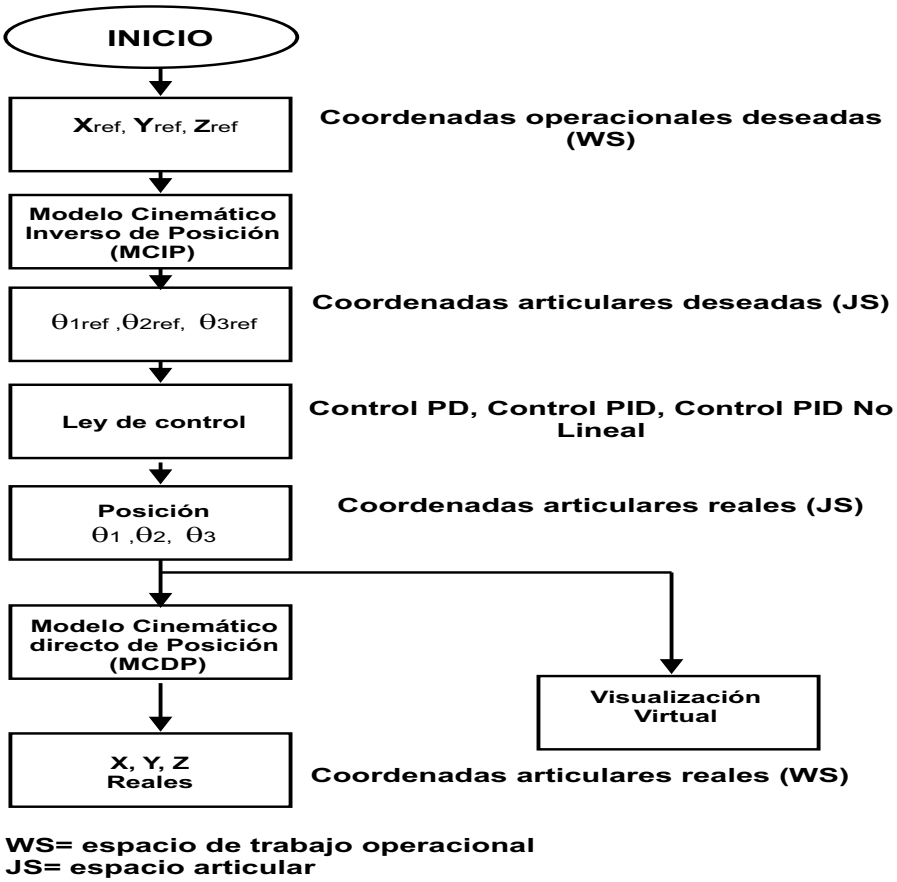

Figura 4.12: Diagrama de flujo de la implementación de la interfaz visual con el control diseñado en C++

Para la visualización de desempeño dinámico virtual se hace uso del archivo con las caraterísticas antes descritas, así también se creo una interfaz de usuario, para que pueda dar los valores iniciales a la representación de la dinámica, tal interfaz se puede observar en la figura 4.13.
Es importante señalar que para que en está representación se implemento el control pid-no lineal que hace uso del método númerico Runge-Kutta, con las formulas de control contenido en el apartado 5 (pag. 64).

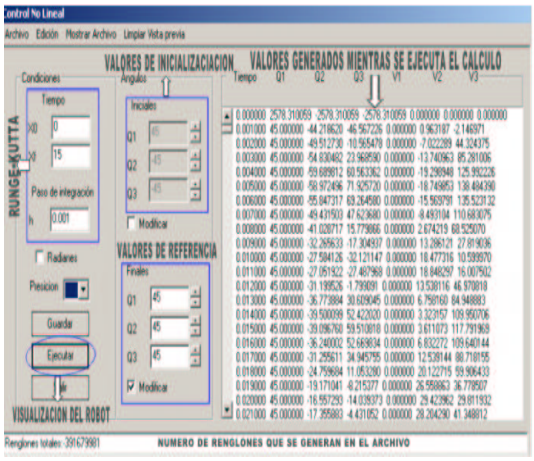

Figura 4.13: Descripción de interfaz grafica.

4.6. Robot antropomórfico virtual

La interfaz que se muestra en la figura 4.14 la cual nos proporciona el cálculo de las valoraciones para q_1,q_2,q_3, de acuerdo a los valores finales de Θ que el usuario estableció. Una vez que se haya ejecutado el control con los valores proporcionados por el usuario el método arroja los valores obtenidos del cálculo en la ventana descrita en la sección anterior.

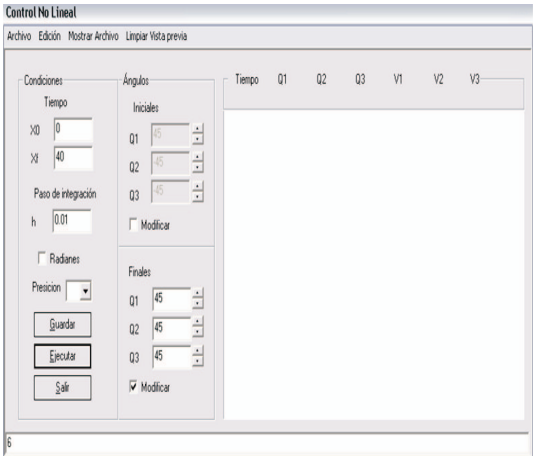

Figura 4.14: Visualización de la interfaz para el usuario.

La interfaz ejecuta el método visualizando los valores obtenidos y vinculándolos con la interfaz de visualización virtual del dispositivo háptico PANTHoM que se muestra en la figura 4.15 para tener la posición final del robot en el ambiente.

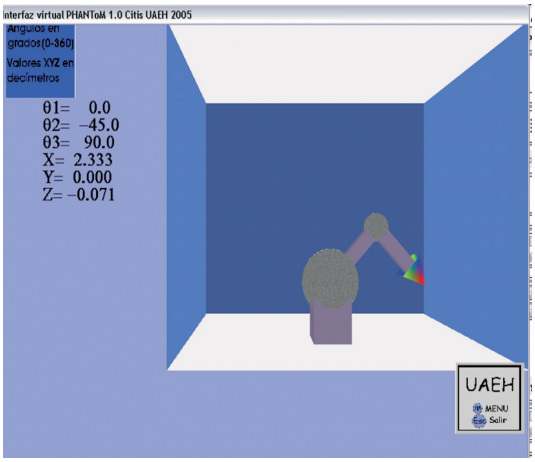

Figura 4.15: Interfaz del robot PHANToM.

En la interfaz del ambiente virtual el robot empieza a mostrar movimientos oscilatorios hasta llegar a la posición deseada por el usuario observandose la trayectoria de los movimietos del robot, vease la figura 4.16.

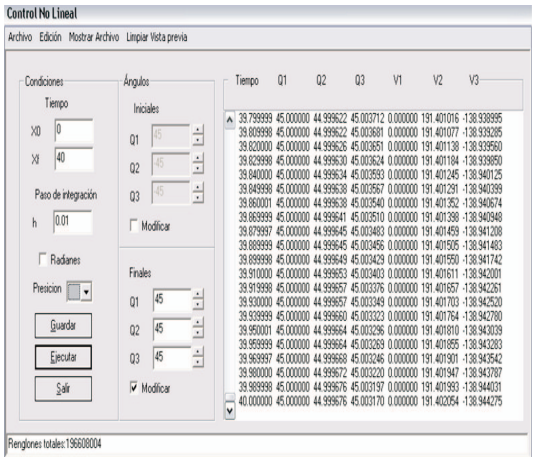

Figura 4.16: Movimientos del robot con la implementación del modelo dinámico.

Una vez llegada ala posición final el robot se estabiliza haciendo notar que ha llegado al punto de referencia dado por el usuario, los resultado se pueden apreciar en la figura 4.17

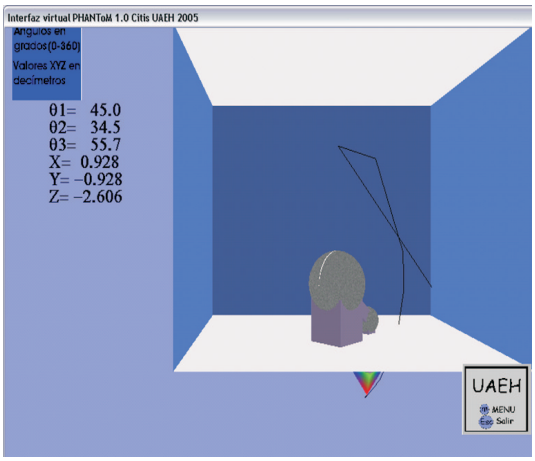

Figura 4.17: Posicion final del robot implementado el modelo dinámico.

4.7. Conclusiones

Una vez que se ha hecho un análisis de las herramientas y lenguajes y dadas las características de estos se ha elegido la utilización de OpenGl y Visual C++, para el diseño de la interfaz visual que se ha de proporcionar al usuario.

Para futuros trabajos se ha hecho una descripción de la forma en que esta compuesto el robot y la interfaz de usuario, el funcionamiento de la misma, en la cual se puede apreciar la implementación de la dinámica a través de una interfaz previa a la visualización del robot y el comportamiento del mismo que se puede apreciar en ambos modelos y que difiere bastante pues si bien en el modelo cinemático se observa que el movimiento es simple y es realizado sin ningun tipo de oscilación, mientras que la visualización del robot con la implementación de la dinámica presenta ciertas oscilaciones y un poco de inestabilidad del robot, ello se debe a que ya cuenta con propiedades (vease capítulo 2) que en la realidad si afectan el comportamiento del robot.

Capítulo 5

Modelado y control del robot antropomórfico

5.1. Introducción

En este capítulo se presentará en primer lugar la solución numérica aplicado al momento de proporcionar un control en un robot manipulador. Así también se puede apreciar el desarrollo del modelo matemático de un robot manipulador. Se observa una serie de gráficas que muestran el comportamiento del modelo matemático integrandole los tres controles descritos en el capítulo anterior.

5.2. Solución numérica a través del metodo Runge-Kutta(R-K)

5.2.1. Historia del método

Carl Runge fué un físico matemático alemán que desarrolló los métodos numéricos para solucionar las ecuaciones diferenciales que presentó en sus estudios de espectros atómicos. Él utilizó tantas matemáticas en su investigación que los físicos pensaron que él era matemático, y también utilizó tanto la física que los matemáticos pensaron que él era físico. Hoy su nombre se asocia a los métodos de Runge-Kutta para solucionar numéricamente ecuaciones diferenciales [24].

Figura 5.1: Carl Runge 30 Agosto de 1856 - 3 Junio de 1927 [24].

Martin Wilhelm Kutta especialista alemán en matemáticas aplicadas, publicó el método numérico para la aproximación de soluciones a las ecuaciones diferenciales ordinarias es el mejor conocido por el método de Runge-Kutta [24].

Figura 5.2: Martin Wilhelm Kutta 3 de Noviembre de 1867 - 25 de Diciembre de 1944 [24].

5.2.2. Solución numérica del modelo en lazo abierto y cerrado

El método de Runge-Kutta cambia la dirección en el sentido de que no sigue la misma línea de los métodos de Euler y Euler Mejorado. De hecho está basado en una aplicación de los polinomios de Taylor. Este método logra la exactitud del procedimiento de una serie de Taylor sin requerir el cálculo de derivadas superiores. Como una serie de algoritmos para calcular aproximaciones numéricas, que esta dada en función de [11]:

$$\frac{dy}{dx} = \tau^{*}(x, y); y(x0 = y0) \tag{5.1}$$

Para poder determinar que método es el que nos brinda una mayor exactitud, se seleccionaron los siguientes métodos que pueden resolver ecuaciones diferenciales [12].

- Euler
- Euler mejorado
- Runge-Kutta

Se utilizó una ecuación diferencial de primer orden para esta comparación y los resultados de dicha comparación se muestran en el cuadro 5.1 en donde se puede observar que el método de Runge-Kutta es el que nos da como resultado el valor más aproximado a la solución, está muestra es una de las razones del por que se ha hecho uso de esté método en el presente trabajo.

$$y^{\cdot} = 2xy, y(1) = 1 \tag{5.2}$$

Las condiciones iniciales son:

$$\mathrm{X_0} = 1{,}0, \mathrm{X_F} = 1{,}5, \mathrm{h} = 0{,}1 \tag{5.3}$$

Donde:

X_0: es el valor inicial del cálculo(tiempo inicial).

X_F: es el valor final del cálculo(tiempo final).

h: es el intervalo con el que se hara el cálculo(paso de integración).

Xn	Euler	Euler mejorado	Runge-Kutta	Mínimo error de convergencia
1.00	1.0000	1.0000	1.0000	1.0000
1.10	1.2000	1.2320	1.2337	1.2337
1.20	1.4640	1.5479	1.5527	1.5527
1.30	1.8154	1.9832	1.9937	1.9937
1.40	2.2874	2.5908	2.6116	2.6116
1.50	2.9278	3.4509	3.4902	3.4904

Cuadro 5.1: Tabla comparativa de la exactitud de diferentes métodos numéricos [12].

También es importante analizar por separado el error generado para cada método, teniendo la gráfica 5.3 representativa de los métodos antes mencionados incluyendo el método Heun. En donde se puede observar que el esfuerzo aplicado por el método de Runge-Kutta es menor, así como el error relativo porcentual para los métodos R-K, por tal, se establece que el método de Runge-Kutta es más efectivo que los demás [11].

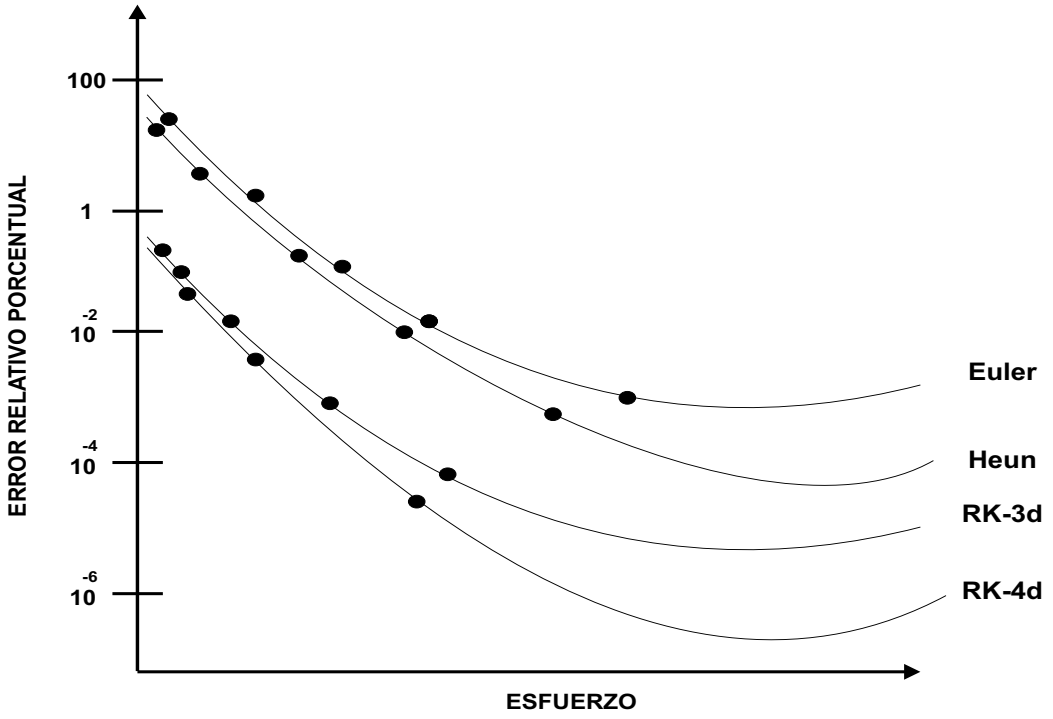

Figura 5.3: Grafica comparativa de errores generados entre diferentes métodos numéricos [11].

Al momento de valorar los resultados obtenidos para el mismo ejemplo de ecuación diferencial, en los métodos de Euler, Euler mejorado y Runge-Kutta, se pudo apreciar que el método de Runge-kutta nos ofrece una mayor exactitud, en comparación a los otros métodos.

5.2.3. Descripción del método de Runge-Kutta de 4º orden

El método se define como una serie de algoritmos para calcular aproximaciones numéricas, que esta dada en función de:

$$\frac{dy}{dx} = f^{*}(x, y); y(x0 = y0) \tag{5.4}$$

Existen diferentes versiones del método de Runge-Kutta los hay desde 1 orden hasta orden superior. El método consta de un conjunto de pendientes que van calculando una a una el intervalo generando así un conjunto de funciones incremento [11].

Variables a que se van a aproximar, las cuales tienen como valor inicial un intervalo que esta en función del tiempo. Un incremento que esta en relación al tiempo. Y el tiempo mismo como una de sus variables.

De esta forma tenemos la formulación del método es la siguiente[11]:

Calculamos las pendientes:

$$k_1 = h * f(Xn, Yn) \tag{5.5}$$

$$k_2 = h * f\left(Xn + \frac{1}{2}h, Yn + \frac{1}{2}k_1\right) \tag{5.6}$$

$$k_3 = h * f\left(Xn + \frac{1}{2}h, Yn + \frac{1}{2}k_2\right) \tag{5.7}$$

$$k_4 = h * f(Xn, +h, Yn + k_3) \tag{5.8}$$

Y así obtenemos el siguiente valor para y con la siguiente formula:

$$Y_{n+1} = Y_n + \frac{1}{6}[k_1 + 2k_2 + 2k_3 + k_4] \tag{5.9}$$

En donde:

$k_{1,2,3,4}$: son las pendientes calculadas de la curva integral.

Y_{n+1}: variable que expresa la fórmula de recurrencia en la aplicación del método.

h: es el paso de integración.

Y así sucesivamente hasta llegar al valor final dado para la solución de la ecuación, de forma gráfica esto se vería en la grafica 5.4, cabe mencionar que en un sistema de ecuaciones el método no cambia, se aplica a cada una de las ecuaciones donde habrá correspondencia entre cada una de ellas pero evaluadas de manera independiente.

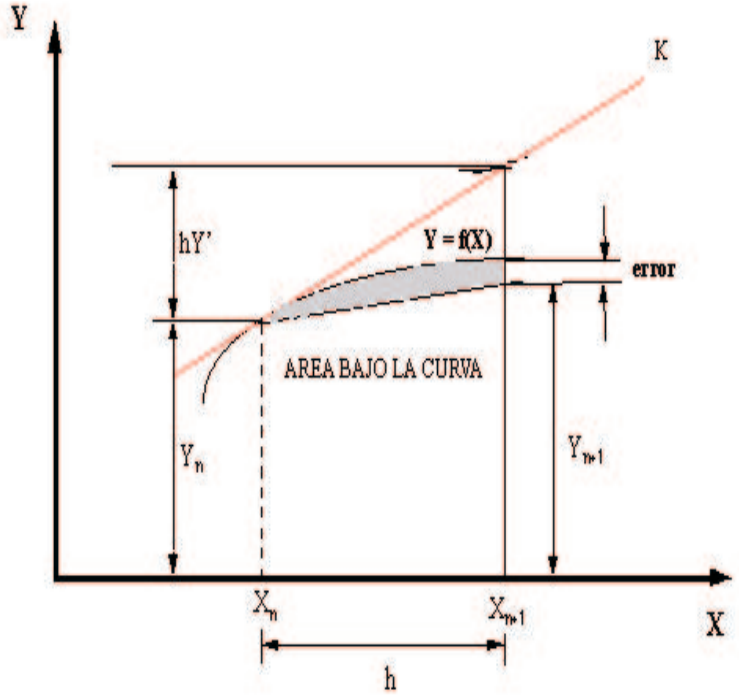

Figura 5.4: Gráfica del funcionamiento del método de Runge-Kutta [11].

Eslabón i	θ_i	d_i	a_i	α_i
1	θ_1	0	0	$-\pi/2$
2	θ_2	0	L_2	0
3	θ_3	0	L_3	0

Cuadro 5.2: Parámetros de Denavit-Hartenberg [5].

5.3. Modelo cinemático

5.3.1. Parámetros de Denavit-Hartenberg

En base a la explicación presentada en el capítulo dos se definen los parámetros de Denavit-Hartenberg para un robot antropomórfico de tres grados de libertad presentados en la tabla.

5.3.2. Matrices elementales de un robot

Las matrices elementales de la estructura de posición del robot de 3 grados de libertad son:

$$ {}^{1}_{2}T = \begin{bmatrix} c_1 & 0 & -s_1 & 0 \\ s_1 & 0 & c_1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.10} $$

$$
{}^{2}_{3}T = \begin{bmatrix} c_2 & -s_2 & 0 & L_2c_2 \\ s_1 & c_2 & 0 & L_2s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.11}
$$

$$
{}^{3}_{4}T = \begin{bmatrix} c_3 & -s_3 & 0 & L_3c_3 \\ s_3 & c_3 & 0 & L_3s_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.12}
$$

5.3.3. Matriz de transformación homogénea para un robot de 3 grados de libertad

La matriz de transformación homogénea que describe el marco ortonormal de referencia en el extremo final del último eslabón con respecto al primer marco de referencia y proporcione información de la posición en coordenadas cartesinas y orientación en ángulos Euler, se define por el producto sucesivo de las matrices elementales del robot, obteniendo la siguiente matriz de transformación homogénea 5.13 para un robot de tres grados de libertad.

$$
{}^{1}_{4}T = \begin{bmatrix} c_1\,c_{23} & -c_1\,s_{23} & -\,s_1 & L_3\,c_1\,c_{23} + L_2\,c_1\,c_2 \\ s_1\,c_{23} & -s_1\,s_{23} & c_1 & L_3\,s_1\,c_{23} + L_2\,s_1\,c_2 \\ -s_{23} & -\,c_{23} & 0 & -\,L_3\,s_{23} - L_2\,s_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.13}
$$

5.3.4. Modelo cinemático directo de posición(MCDP)

El modelo cinemático directo de posición de un robot manipulador es la relación que permite determinar las coordenadas operacionales del robot en función de las variables articulares o coordenadas generalizadas del mismo. El MCDP para R3 esta definido por las siguientes ecuaciones 5.14 , 5.15 y 5.16 [9].

$$
\mathrm{x} = \mathrm{L}_3c_1c_{23} + L_2c_1c_2 \tag{5.14}
$$

$$
\mathrm{y} = \mathrm{L}_3s_1c_{23} + L_2c_1c_2 \tag{5.15}
$$

$$
\mathrm{z} = -\mathrm{L}_3s_1c_{23} - L_2s_2 \tag{5.16}
$$

Donde:
(x, y): coordenadas operacionales.
s_1: seno de θ_1
c_1: coseno de θ_1
θ: variable articular o Coordenada generalizada.
L: longitud del eslabón.

Para determinar la matriz de transformación homógenea de R3, se tiene:

$$ {}^{1}_{4}T = {}^{1}_{2}T * {}^{2}_{3}T * {}^{3}_{4}T \tag{5.17} $$

Donde tenemos como resultado la siguiente matriz:

$$ {}^{1}_{4}T = \begin{bmatrix} E_{11} & E_{12} & E_{13} & E_{14} \\ E_{21} & E_{22} & E_{23} & E_{24} \\ E_{31} & E_{32} & E_{34} & E_{35} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.18} $$

Por lo tanto para cada posición:

$$ \begin{aligned} E_{11} &= c_1 c_{23} \\ E_{12} &= -c_1 s_{23} \\ E_{13} &= -s_1 \\ E_{14} &= L_3 c_1 c_{23} + L_2 c_1 c_2 \\ E_{21} &= s_1 c_{23} \\ E_{22} &= -s_1 s_{23} \\ E_{23} &= c_1 \\ E_{24} &= L_3 s_1 c_{23} + L_2 s_1 c_2 \\ E_{31} &= -s_{23} \\ E_{32} &= -c_{23} \\ E_{33} &= 0 \\ E_{34} &= -L_3 s_{23} - L_2 s_2 \end{aligned} \tag{5.19} $$

5.3.5. Modelo cinemático inverso de posición(MCIP)

El modelo cinemático inverso de posición está definido por la función inversa de la relación de las coordenadas operacionales [9].

$$ \theta = tg^{-1} \left\{ \frac{y}{x} \right\} \tag{5.20} $$

Esto es representado por la función 5.20 que nos proporciona el ángulo θ.

$$ \theta_1 = atan2(y, x) \tag{5.21} $$

La función atan2(x,y)

Aplicando identidades trigonométricas:

$$ \begin{aligned} sen\theta_3 &= \sqrt{1 - cos^2\theta_3} \\ \theta_3 &= atan2(sen\theta_3, cos\theta_3) \end{aligned} \tag{5.22} $$

Por lo tanto:

$$ \theta_3 = atan2(\sqrt{1 - cos^2\theta_3}, cos\theta_3) \tag{5.23} $$

y

$$cos\theta_3 = \frac{x^2+y^2+z^2-L_2^2-L_3^2}{2L_2L_3} \tag{5.24}$$

Por Ley de Senos tenemos que:

$$\theta_2 = -a\tan 2(z, \sqrt{x^2+y^2}) - a\tan 2(L_3 sen\theta_3 - L_2 + L_3) \tag{5.25}$$

5.3.6. Modelo cinemático de velocidad

Modelo cinemático directo de velocidad(MCDV)

Dada la expresión que identifica al MCDP:

$$x = f(\theta) \tag{5.26}$$

aplicando la primera derivada:

$$\dot{x} = \frac{d}{dt} f(\theta) \tag{5.27}$$

obtenemos el MCDP

$$\dot{x} = J\dot{\theta} \tag{5.28}$$

donde: $\dot{x}$: vector de velocidad del órgano terminal del robot.

$$\dot{x} = [\dot{x}, \dot{y}, \dot{z}]^T \tag{5.29}$$

$\dot{\theta}$: vector de velocidad del las articulaciones del robot.

$$\dot{\theta} = \left[\dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3\right]^T \tag{5.30}$$

$\dot{J}$: matriz Jacobiana del robot, cuyos elementos son J_{ij}.

$$J_{ij} = \frac{\partial f_i}{\partial \theta_j} \tag{5.31}$$

Dado el MCDP, derivandolo respecto al tiempo se tiene:

$$x = L_3 c_1 c_{23} + L_2 c_1 c_2 \tag{5.32}$$

$$y = L_3 s_1 c_{23} + L_2 s_1 c_2 \tag{5.33}$$

$$z = -L_3 s_{23} - L_2 s_2 \tag{5.34}$$

$$\dot{x} = -\left\{L_3\, s_1\, c_{23} + L_2\, s_3\, c_2\right\}\dot{\theta}_1 - \left\{L_3\, s_{23}\, c_1 + L_2\, s_2\, C_1\right\}\dot{\theta}_2 - \left\{L_3\, s_{23}\, c_1\right\}\dot{\theta}_3 \tag{5.35}$$

$$\dot{y} = \left\{L_3\, c_1\, c_{23} + L_2\, c_1\, c_2\right\}\dot{\theta}_1 - \left\{L_3\, s_1\, s_{23} + L_2\, s_1\, S_2\right\}\dot{\theta}_2 - \left\{L_3\, s_1\, s_{23}\right\}\dot{\theta}_3 \tag{5.36}$$

$$\dot{z} = -\left\{L_3\, c_{23} + L_2\, c_2\right\}\dot{\theta}_2 - \left\{L_3\, c_{23}\right\}\dot{\theta}_3 \tag{5.37}$$

Dado que el modelo cinemático directo de velocidad, puede ser representado en términos de una matriz Jacobiana:

$$\dot{x} = J\dot{\theta} \tag{5.38}$$

Se calcula la matriz Jacobiana quedando de la siguiente forma:

$$J = \begin{bmatrix} -L_3 s_1 c_{23} - L_2 c_1 s_2 & -L_3 s_{23} c_1 - L_2 s_2 c_1 & -L_3 s_{23} c_1 \\ L_3 c_1 c_{23} + L_2 c_1 c_2 & -L_3 s_1 s_{23} - L_2 s_1 s_2 & -L_3 s_1 s_{23} \\ 0 & -L_3 c_{23} - L_2 c_2 & -L_3 c_{23} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} \tag{5.39}$$

Modelo cinemático inverso de velocidad (MCIV)

A partir del MCDV se define la ecuación representativa del MCIV como se presenta en la siguiente ecuación:

$$\dot{q} = J^{-1}\dot{x} \tag{5.40}$$

Dado el Jacobiano en su representación reducida

$$J_* = \begin{bmatrix} -L_3 s_1 c_{23} - L_2 c_1 s_2 & -L_3 s_{23} c_1 - L_2 s_2 c_1 & -L_3 s_{23} c_1 \\ L_3 c_1 c_{23} + L_2 c_1 c_2 & L_3 s_1 s_{23} - L_2 s_1 s_2 & L_3 s_1 s_{23} \\ 0 & -L_3 c_{23} - L_2 c_2 & -L_3 c_{23} \end{bmatrix} \tag{5.41}$$

La matriz inversa del Jacobiano reducido

$$J_*^{-1} = \begin{bmatrix} -\frac{s_1}{L_2 c_2 + L_3 c_{23}} & \frac{c_1}{L_2 c_2 + L_3 c_{23}} & 0 \\ \frac{c_1 c_{23}}{L_2 s_3} & \frac{s_1 c_{23}}{L_2 s_3} & -\frac{s_{23}}{L_2 s_3} \\ -\frac{L_2 c_1 c_2 + L_3 c_1 c_{23}}{L_2 L_3 s_3} & -\frac{L_2 s_1 c_2 + L_3 - L_3 s_1 c_{23}}{L_2 L_3 s_3} & \frac{L_2 s_2 + L_3 - L_3 s_{23}}{L_2 L_3 s_3} \end{bmatrix} \tag{5.42}$$

El MCIV una vez obtenido J_*^{-1} es el siguiente:

$$\dot{\theta}_1 = \frac{1}{L_2\, c_2 + L_3\, c_{23}} \left\{ - s_1\, \dot{x} + c_1\, \dot{y} \right\} \tag{5.43}$$

5.3.7. Modelo cinemático de aceleración

Modelo cinemático directo de aceleración (MCDA)[9]

Dado el MCDV del robot, obtenido anteriormente, se procede a obtener su derivada respecto del tiempo, obteniendo así, el modelo cinemático directo de aceleración:

$$\ddot{x} = J\ddot{q} + \dot{J}\dot{q} \tag{5.44}$$

Donde:

$\ddot{x}$: corresponde al vector aceleración del órgano terminal del robot.

$\ddot{\theta}$: corresponde al vector de aceleración de las variables articulares.

$\dot{J}$: representa la primera derivada respecto al tiempo de la matriz jacobiana.

La derivada de la matriz jacobiana se puede apreciar en la ec.5.45:

$$\dot{J} = \begin{bmatrix} \dot{J}_{11} & \dot{J}_{12} & \dot{J}_{13} \\ \dot{J}_{21} & \dot{J}_{22} & \dot{J}_{23} \\ \dot{J}_{31} & \dot{J}_{32} & \dot{J}_{33} \end{bmatrix} \tag{5.45}$$

Donde:

$\dot{J}_{11}$=$L_3\, s_1\, s_{23} \left(\dot{\theta}_2 - \dot{\theta}_3 \right) - L_3\, c_1\, c_{23}\, \dot{\theta}_1 + L_2\, s_1\, s_2\, \dot{\theta}_2 - L_2\, c_1\, c_2\, \dot{\theta}_1$

$\dot{J}_{12}$=$L_3\, s_1\, s_{23}\, \dot{\theta}_1 - L_3\, c_1\, c_{23} \left(\dot{\theta}_2 - \dot{\theta}_3 \right) + L_2\, s_1\, s_2\, \dot{\theta}_1 - L_2\, c_1\, c_2\, \dot{\theta}_2$

$\dot{J}_{13}$=$L_3\, s_1\, s_{23}\, \dot{\theta}_1 - L_3\, c_1\, c_{23}\left(\dot{\theta}_2 - \dot{\theta}_3\right)$

$\dot{J}_{21}$=$- L_3\, s_{23}\, c_{23}\left(\dot{\theta}_2 - \dot{\theta}_3\right) - L_3\, s_1\, c_{23}\, \dot{\theta}_1 - L_2\, s_2\, c_1\, \dot{\theta}_2 - L_2\, s_1\, c_2\, \dot{\theta}_1$

$\dot{J}_{22}$=$- L_3\, s_1\, c_{23}\left(\dot{\theta}_2 - \dot{\theta}_3\right) - L_3\, s_{23}\, c_1\, \dot{\theta}_1 + L_2\, s_1\, c_2\, \dot{\theta}_2 - L_2\, s_2\, c_1\, \dot{\theta}_1$

$\dot{J}_{23}$=$-L_3\, s_1\, c_{23}\left(\dot{\theta}_2 - \dot{\theta}_3\right) - L_3\, s_{23}\, c_1\, \dot{\theta}_1$

$\dot{J}_{31}$=0

$\dot{J}_{32}$=$L_3\, s_{23}\left(\dot{\theta}_2 - \dot{\theta}_3\right) + L_2\, s_2\, \dot{\theta}_2$

$\dot{J}_{33}$=$- L_3\, s_{23}\left(\dot{\theta}_2 - \dot{\theta}_3\right)$

Modelo Cinemático Inverso de Aceleración(MCIA)

El modelo cinemático inverso de aceleración se deduce a partir del MCDA que se obtuvo anteriormente, cuya formula se aprecia en la ec. 5.46

$$\ddot{\theta} = J^{-1}\left[\ddot{x} - \dot{J}\dot{\theta}\right] \tag{5.46}$$

$$\ddot{y} = -L\,\theta\, sen\,\dot{\theta} + L\,\ddot{\theta}\, cos\theta \tag{5.47}$$

5.4. Modelo dinámico(MD)

El modelo dinámico se entiende como el conjunto de ecuaciones que caracterizan a los factores que intervienen en el movimiento del robot. Se considera que en los eslabones se puede localizar los centros de gravedad a la mitad de la longitud de cada eslabón como se observa en la figura 5.5,[13].

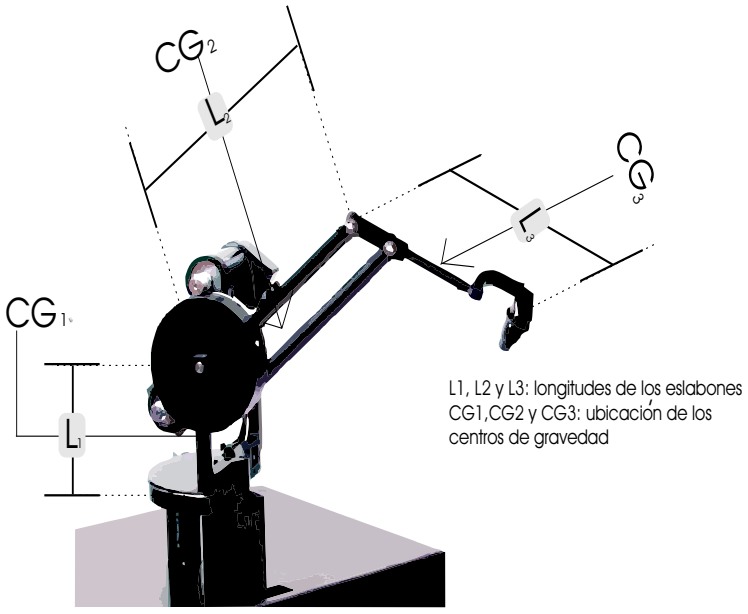

Figura 5.5: Ubicación de los centros de masa [13].

5.4.1. Obtención de velocidades en los eslabones

Las coordenadas operacionales (x_1, y_1, z_1) y sus derivadas con respecto del tiempo, para el eslabón 1 son

$$\begin{array}{ll} x_1 = 0 & \dot{x}_1 = 0 \\ y_1 = 0 & \dot{y}_1 = 0 \\ z_1 = -L_1 & \dot{z}_1 = 0 \end{array} \tag{5.48}$$

El módulo de la velocidad y altura al centro de masas para el eslabón 1 es

$$v_1^2 = \dot{x}_1^2 + \dot{y}_1^2 + \dot{z}_1^2 \tag{5.49}$$

La velocidad de acuerdo al modelo cinemático de velocidad es

$$v_1^2 = 0 \tag{5.50}$$

Y la altura para el eslabón 1 es

$$h_1 = -LCG_1 \tag{5.51}$$

donde: LCG : longitud al centro de masas.

Las coordenadas operacionales (x_2, y_2, z_2) y sus derivadas respecto del tiempo, para el eslabón 2 son

$$\begin{array}{l} x_2 = L_2 c_1 c_2 \\ \mathrm{y}_2 = L_2 s_1 c_2 \\ z_2 = -L_2 s_2 \end{array} \tag{5.52}$$

Sus derivadas con respecto al tiempo

$$\begin{array}{l} \dot{x}_2 = -L_2 s_1 c_2 \dot{\theta}_1 - L_2 s_2 c_1 \dot{\theta}_2 \\ \dot{\mathrm{y}}_2 = L_2 c_1 c_2 \dot{\theta}_1 - L_2 s_1 s_2 \dot{\theta}_2 \\ \dot{z}_2 = -L_2 c_2 \dot{\theta}_2 \end{array} \tag{5.53}$$

Continuando, el siguiente modulo es del cuadrado de la velocidad y altura al centro de masas para el eslabón 2 a partir de

$$v_2^2 = \dot{x}_2^2 + \dot{y}_2^2 + \dot{z}_2^2 \tag{5.54}$$

Substituyendo en la ecuación anterior

$$v_2^2 = \left(-L_2 s_1 c_2 \dot{\theta}_1 - L_2 s_2 c_1 \dot{\theta}_2\right)^2 + \left(L_2 c_1 c_2 \dot{\theta}_1 - L_2 s_1 s_2 \dot{\theta}_2\right)^2 + \left(-L_2 c_2 \dot{\theta}_2\right)^2 \tag{5.55}$$

desarrollando

$$v_2^2 = L_2^2 s_1^2 c_2^2 \dot{\theta}_1^2 + 2L_2^2 s_1 s_2 c_1 c_2 \dot{\theta}_1 \dot{\theta}_2 + L_2^2 s_2^2 c_1^2 \dot{\theta}_2^2 + L_2^2 c_1^2 c_2^2 \dot{\theta}_1^2 - 2L_2^2 s_1 s_2 c_1 c_2 \dot{\theta}_1 \dot{\theta}_2 + L_2^2 s_1^2 s_2^2 \dot{\theta}_2^2 + L_2^2 c_2^2 \dot{\theta}_2^2$$

$$v_2^2 = L_2^2 c_2^2 \dot{\theta}_1^2 + L_2^2 s_2^2 \dot{\theta}_2^2 + L_2^2 c_2^2 \dot{\theta}_2^2 \tag{5.56}$$

$$v_2^2 = L_2^2 c_2^2 \dot{\theta}_1^2 + L_2^2 \dot{\theta}_2^2$$

La altura del eslabón 2 se define en la siguiente ecuación

$$h_2 = -LCG_2 s_2 \tag{5.57}$$

Las coordenadas operacionales (x_3, y_3, z_3) y sus derivadas respecto del tiempo, para el eslabón 3 son las siguientes

$$\begin{aligned}
x_3 &= L_2 c_1 c_2 + L_3 c_1 c_{23} \\
y_3 &= L_2 s_1 c_2 + L_3 s_1 c_{23} \\
z_3 &= -L_2 s_2 + L_3 s_{23} \\
\dot{x}_3 &= -L_2 s_1 c_2 \dot{\theta}_1 - L_2 s_2 c_1 \dot{\theta}_2 - L_3 s_1 c_{23} \dot{\theta}_1 - L_3 s_{23} c_1 (\dot{\theta}_2 + \dot{\theta}_3) \\
\dot{\mathrm{y}}_3 &= -L_2 s_1 s_2 \dot{\theta}_2 + L_2 c_1 c_2 \dot{\theta}_1 + L_3 c_1 c_{23} \dot{\theta}_1 - L_3 s_1 c_{23} (\dot{\theta}_2 + \dot{\theta}_3) \\
\dot{z}_3 &= -L_2 c_2 \dot{\theta}_2 - L_3 c_{23} (\dot{\theta}_2 + \dot{\theta}_3)
\end{aligned} \tag{5.58}$$

Substituyendo en la ecuación de velocidad del modelo dinámico directo de velocidad $v_3^2 = \dot{x}_3^2 + \dot{y}_3^2 + \dot{z}_3^2$

$$\begin{aligned}
&v_3^2 = -(L_2 s_1 c_2 \dot{\theta}_1 + L_2 s_2 c_1 \dot{\theta}_2) - (L_3 s_1 c_{23} \dot{\theta}_1 + L_3 s_{23} c_1 (\dot{\theta}_2 + \dot{\theta}_3)) \\
&+(-L_2 s_1 s_2 \dot{\theta}_2 + L_2 c_1 c_2 \dot{\theta}_1 + L_3 c_1 c_{23} \dot{\theta}_1 - L_3 s_1 s_{23} (\dot{\theta}_2 + \dot{\theta}_3)) + \\
&(-L_2 c_2 \dot{\theta}_2 - L_3 c_{23} (\dot{\theta}_2 + \dot{\theta}_3)) \\
&simplificando \\
&v_3^2 = L_2^2 \dot{\theta}_2^2 + L_3^2 (\dot{\theta}_2 + \dot{\theta}_3)^2 + 2L_2 L_3 c_3 \dot{\theta}_2 (\dot{\theta}_2 + \dot{\theta}_3) + (L_2 c_2 \dot{\theta}_1 + \\
&L_3 c_{23} \dot{\theta}_1)^2
\end{aligned} \tag{5.59}$$

La altura del eslabón 3 se define en la siguiente ecuación

$$h_3 = -L_2 s_2 - LCG_3 s_{23} \tag{5.60}$$

5.4.2. Determinación de energías cinéticas del robot

El cálculo de las energías cinéticas para cada uno de los eslabones se presentan a continuación, a partir de la ecuación

$$K = \frac{1}{2}mv^2 \tag{5.61}$$

Para el eslabón 1

$$\begin{aligned} K_1 &= \tfrac{1}{2}m_1 v_1^2 = \tfrac{1}{2}m_1(0) = 0 \\ K_1 &= 0 \end{aligned} \tag{5.62}$$

Para el eslabón 2

$$\begin{aligned} K_2 &= \tfrac{1}{2}m_2 v_2^2 = \tfrac{1}{2}m_2\left\{L_2^2 c_2^2 \dot{\theta}_1^2 + L_2^2 \dot{\theta}_2^2\right\} \\ K_2 &= \tfrac{1}{2}m_2 L_2^2 c_2^2 \dot{\theta}_1^2 + \tfrac{1}{2}m_2 L_2^2 \dot{\theta}_2^2 \end{aligned} \tag{5.63}$$

Para el eslabón 3

$$\begin{aligned} &K_3 = \tfrac{1}{2}m_3(L_2^2\dot{\theta}_2^2 + L_3^2(\dot{\theta}_2+\dot{\theta}_3)^2 + 2L_2L_3c_3\dot{\theta}_2(\dot{\theta}_2+\dot{\theta}_3)+ \\ &(L_2c_2\dot{\theta}_1 + L_3c_{23}\dot{\theta}_1)^2) \\ &\textit{desarrollando} \\ &K_3 = \tfrac{1}{2}m_3L_2^2\dot{\theta}_2^2 + \tfrac{1}{2}m_3L_3^2(\dot{\theta}_2+\dot{\theta}_3)^2 + m_3L_2L_3c_3\dot{\theta}_2(\dot{\theta}_2+\dot{\theta}_3) + \tfrac{1}{2}m_3(L_2c_2\dot{\theta}_1 + L_3c_{23}\dot{\theta}_1)^2 \end{aligned} \tag{5.64}$$

5.4.3. Determinación de energías potenciales del robot

El cálculo de las energías potenciales para cada uno de los eslabones se presentan a continuación, a partir de la ecuación

$$P_i = m_i g h_i \tag{5.65}$$

Para el eslabón 1

$$\begin{aligned} P_1 &= m_1 g h_1 = m_1 g\{-LCG_1\} \\ P_1 &= m_1 g LCG_1 \end{aligned} \tag{5.66}$$

Para el eslabón 2

$$\begin{aligned} P_2 &= m_2 g h_2 = \{-LCG_2 s_2\} \\ P_2 &= -m_2 g LCG_2 s_2 \end{aligned} \tag{5.67}$$

Para el eslabón 3

$$\begin{aligned} P_3 &= m_3 g h_3 = m_3 g\{-L_2 s_2 - LCG_3 s_{23}\} \\ P_3 &= -m_3 g L_2 S_2 - m_3 g LCG_3 S_{23} \end{aligned} \tag{5.68}$$

5.4.4. Ecuaciones de movimiento

El lagrangiano está definido por la siguiente expresión

$$L = \sum_{i=1}^{n=3} (Ki - Pi) \tag{5.69}$$

A partir de las energías cinéticas y potenciales para el caso de estudio (3 grados de libertad) es $L = (K_1 + K_2 + K_3) - (P_1 + P_2 + P_3)$, desarrollando dicha ecuación se tiene

$$\begin{aligned} L = {} & \tfrac{1}{2} m_2 L_2^2 c_2^2 \dot{\theta}_1^2 + \tfrac{1}{2} m_2 L_2^2 c_2^2 \dot{\theta}_2^2 + \tfrac{1}{2} m_3 L_2^2 \dot{\theta}_2^2 + \tfrac{1}{2} m_3 L_3^2 (\dot{\theta}_2 + \\ & \dot{\theta}_3)^2 + m_3 L_2 L_3 c_3 \dot{\theta}_2 (\dot{\theta}_2 + \dot{\theta}_3) + \tfrac{1}{2} m_3 (L_2 c_2 \dot{\theta}_1 + L_3 c_{23} \dot{\theta}_1)^2 - \\ & m_1 g LCG_1 + m_2 g LCG_2 s_2 + m_3 g L_2 s_2 + m_3 g LCG_3 s_{23} \end{aligned} \tag{5.70}$$

simplificando y agrupando se obtine la ecuación del Lagrangiano para el caso de estudio

$$\begin{aligned} L = {} & \left\{ \tfrac{1}{2} m_2 L_2^2 c_2^2 + \tfrac{1}{2} m_3 L_2^2 c_2^2 + m_3 L_2 L_3 c_2 c_{23} + \tfrac{1}{2} m_3 L_3^2 c_{23}^2 \right\} \dot{\theta}_1^2 + \\ & \left\{ \tfrac{1}{2} m_2 L_2^2 + \tfrac{1}{2} m_3 L_2^2 + \tfrac{1}{2} m_3 L_3^2 + m_3 L_2 L_3 c_3 \right\} \dot{\theta}_2^2 + \left\{ \tfrac{1}{2} m_3 L_3^2 \right\} \dot{\theta}_3^2 + \\ & \left\{ m_3 L_3^2 + m_3 L_2 L_3 c_3 \right\} \dot{\theta}_2 \dot{\theta}_3 - m_1 g LCG_1 + m_2 g LCG_2 s_2 + \\ & m_3 g \left\{ L_2 s_2 + LCG_3 s_{23} \right\} \end{aligned} \tag{5.71}$$

5.4.5. Formulación Euler-Lagrange

Dada la formulación de Euler-Lagrange definida por la siguiente expresión

$$\tau_i = \frac{d}{dt} \frac{\partial L}{\partial \dot{\theta}_i} - \frac{\partial L}{\partial \theta_i} \tag{5.72}$$

Para el caso de estudio obtendremos una ecuación de par torsor por cada eslabón del robot, es decir tendremos

$$\begin{aligned} \tau_1 &= \tfrac{d}{dt} \tfrac{\partial L}{\partial \dot{\theta}_1} - \tfrac{\partial L}{\partial \theta_1} \\ \tau_2 &= \tfrac{d}{dt} \tfrac{\partial L}{\partial \dot{\theta}_2} - \tfrac{\partial L}{\partial \theta_2} \\ \tau_3 &= \tfrac{d}{dt} \tfrac{\partial L}{\partial \dot{\theta}_3} - \tfrac{\partial L}{\partial \theta_3} \end{aligned} \tag{5.73}$$

Obteniendo la ecuación 1

$$\tau_1 = \frac{d}{dt} \frac{\partial L}{\partial \dot{\theta}_1} - \frac{\partial L}{\partial \theta_1} \tag{5.74}$$

$$
\begin{aligned}
&\tfrac{\partial L}{\partial \dot{\theta}_1} = \left\{m_2 L_2^2 c_2^2 + m_3 L_2^2 c_2^2 + 2m_3 L_2 L_3 c_2 c_{23} + m_3 L_3^2 c_{23}^2\right\} \dot{\theta}_1 \\
&\tfrac{d}{dt}\tfrac{\partial L}{\partial \dot{\theta}_1} = \left\{(m_2 + m_3) L_2^2 c_2^2 + 2m_3 L_2 L_3 c_2 c_{23} + m_3 L_3^2 c_{23}^2\right\} \ddot{\theta}_1 \\
&\tfrac{d}{dt}\tfrac{\partial L}{\partial \dot{\theta}_1} = \left\{(m_2 + m_3) L_2^2 c_2^2 + 2m_3 L_2 L_3 c_2 c_{23} + m_3 L_3^2 c_{23}^2\right\} \ddot{\theta}_1 - \\
&\left\{2(m_2 + m_3) L_2^2 s_2 c_2 + 2m_3 L_2 L_3 (s_{23} c_2 + s_2 c_{23}) + 2m_3 L_3^2 s_{23} c_{23}\right\} \dot{\theta}_1 \dot{\theta}_2 - \\
&\left\{2m_3 L_2 L_3 s_{23} c_2 + 2m_3 L_3^2 s_{23} c_{23}\right\} \dot{\theta}_1 \dot{\theta}_3 \\
&\tfrac{\partial L}{\partial \dot{\theta}_1} = 0
\end{aligned}
\tag{5.75}
$$

La ecuación par torsor 1 es

$$
\begin{aligned}
&\tfrac{d}{dt}\tfrac{\partial L}{\partial \dot{\theta}_1} = \left\{(m_2 + m_3) L_2^2 c_2^2 + 2m_3 L_2 L_3 c_2 c_{23} + m_3 L_3^2 c_{23}^2\right\} \ddot{\theta}_1 - \\
&\left\{2(m_2 + m_3) L_2^2 s_2 c_2 + 2m_3 L_2 L_3 (s_{23} c_2 + s_2 c_{23}) + 2m_3 L_3^2 s_{23} c_{23}\right\} \dot{\theta}_1 \dot{\theta}_2 - \\
&\left\{2m_3 L_2 L_3 s_{23} c_2 + 2m_3 L_3^2 s_{23} c_{23}\right\} \dot{\theta}_1 \dot{\theta}_3
\end{aligned}
\tag{5.76}
$$

Obteniendo la ecuación 2

$$
\tau_2 = \frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_2} - \frac{\partial L}{\partial \theta_2} \tag{5.77}
$$

$$
\begin{aligned}
&\tfrac{\partial L}{\partial \dot{\theta}_2} = \left\{m_2 L_2^2 + m_3 L_2^2 + m_3 L_3^2 + 2m_3 L_2 L_3 c_3\right\} \dot{\theta}_2 + \left\{m_3 L_3^2 + m_3 L_2 L_3 c_3\right\} \dot{\theta}_3 \\
&\tfrac{d}{dt}\tfrac{\partial L}{\partial \dot{\theta}_2} = \left\{(m_2 + m_3) L_2^2 + m_3 L_3^2 + 2m_3 L_2 L_3 c_3\right\} \ddot{\theta}_2 + \left\{m_3 L_3^2 + m_3 L_2 L_3 c_3\right\} \ddot{\theta}_3 \\
&- \left\{2m_3 L_2 L_3 s_3\right\} \dot{\theta}_2 \dot{\theta}_3 - \left\{m_3 L_2 L_3 s_3\right\} \dot{\theta}_3 \\
&\tfrac{\partial L}{\partial \theta_2} = \left\{-(m_2 + m_3) L_2^2 s_2 c_2 - m_3 L_2 L_3 (c_2 s_{23} + c_{23} s_2) - m_3 L_3^2 c_{23} s_{23}\right\} \dot{\theta}_1^2 + \\
&m_2 g LCG_2 c_2 + m_3 g \left\{L_2 c_2 + LCG_3 c_{23}\right\}
\end{aligned}
\tag{5.78}
$$

la ecuación de par torsor 2 es

$$
\begin{aligned}
&\tau_2 = \left\{(m_2 + m_3) L_2^2 + m_3 L_3^2 + 2m_3 L_2 L_3 c_3\right\} \ddot{\theta}_2 + \left\{m_3 L_3^2 + m_3 L_2 L_3 c_3\right\} \ddot{\theta}_3 - \\
&\left\{2m_3 L_2 L_3 s_3\right\} \dot{\theta}_2 \dot{\theta}_3 + \left\{(m_2 + m_3) L_2^2 s_2 c_2 + m_3 L_2 L_3 (c_2 s_{23} + c_{23} s_2) + m_3 L_3^2 c_{23} s_{23}\right\} \dot{\theta}_1^2 - \\
&\left\{m_3 L_3^2 c_{23} s_{23}\right\} \dot{\theta}_3^2 + m_2 g LCG_2 c_2 + m_3 g \left\{L_2 c_2 + LCG_3 c_{23}\right\}
\end{aligned}
\tag{5.79}
$$

función	descripción
$H(q)$	Matriz de inercia
$C(q,\dot{q})$	Matriz de coriolis y centrípetas
$G(q)$	Vector de fuerzas gravitatorias
$F(\dot{q})$	Vector de fuerzas de fricción seca y viscosa

Cuadro 5.3: Descripción de las funciones del MD

Obteniendo la ecuación 3

$$T_3 = \frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_3} - \frac{\partial L}{\partial \dot{\theta}_3} \tag{5.80}$$

$$\begin{aligned}
&\tfrac{\partial L}{\partial \dot{\theta}_3} = \left\{m_3 L_3^2\right\}\dot{\theta}_3 + \left\{m_3 L_3^2 + m_3 L_2 L_3 c_3\right\}\dot{\theta}_2 \\
&\tfrac{d}{dt}\tfrac{\partial L}{\partial \dot{\theta}_3} = \left\{m_3 L_3^2\right\}\ddot{\theta}_3 + \left\{m_3 L_3^2 + m_3 L_2 L_3 c_3\right\}\ddot{\theta}_2 - \left\{m_3 L_2 L_3 s_3 \theta_3\right\}\dot{\theta}_2 \\
&\tfrac{\partial}{\partial \dot{\theta}_3} = -\left\{m_3 L_2 L_3 c_2 s_{23} + m_3 L_3^2 s_{23} c_{23}^2\right\}\dot{\theta}_1 - \left\{m_3 L_2 L_3 s_3\right\}\dot{\theta}_2 - \left\{m_3 L_2 L_3 s_3\right\}\dot{\theta}_2^2\dot{\theta}_3 + m_3 g L_{cg3} c_{23}
\end{aligned} \tag{5.81}$$

la ecuación de par torsor 3 es

$$\begin{aligned}
T_3 = &-\left\{m_3 L_3^2 + m_3 L_2 L_3 c_3\right\}\ddot{\theta}_2 - \left\{m_3 L_3^2\right\}\ddot{\theta}_3 + \left\{m_3 L_2 L_3 c_2 s_{23} + m_3 L_3^2 s_{23} c_{23}\right\}\dot{\theta}_1^2 \\
&+ \left\{m_3 L_2 L_3 s_3\right\}\dot{\theta}_2^2 + m_3 g L_{cg3} c_{23}
\end{aligned} \tag{5.82}$$

5.4.6. Representación matricial del MD

La representación matricial del modelo dinámico se basa en la siguiente expresión

$$T = H(q)\,\ddot{q} + C(q,\dot{q})\,\dot{q} + G(q) + F(\dot{q}) \tag{5.83}$$

donde:
Matriz de inercia

$$H(q) = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \tag{5.84}$$

Los elementos de la matriz de inercia son

$$
\begin{array}{l}
H_{11} = (m_2 + m_3) L_2^2 c_2^2 + 2m_3 L_2 L_3 c_2 c_{23} + m_3 L_3^2 c_{23}^2 \\
H_{12} = 0 \\
H_{13} = 0 \\
H_{21} = 0 \\
H_{22} = (m_2 + m_3) L_2^2 + m_3 L_2^2 + 2m_3 L_2 L_3 c_2 \\
H_{23} = m_3 L_3^2 + m_3 L_2 L_3 c_3 \\
H_{31} = 0 \\
H_{32} = m_3 L_3^2 + m_3 L_2 L_3 c_3 \\
H_{33} = m_3 L_3^2
\end{array}
\tag{5.85}
$$

La matriz de fuerzas de coriolis y centrípetas, así como sus elementos se definen a continuación

$$
C(q, \ddot{q}) = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix}
\tag{5.86}
$$

en donde:

$$
\begin{array}{l}
C_{11} = 0 \\
C_{12} = -\left\{ \begin{array}{l} 2(m_2 + m_3) L_2^2 s_2 c_2 + 2m_3 L_2 L_3 (s_{23} c_2 + s_2 c_{23}) + \\ 2m_3 L_3^2 s_{23} c_{23} \end{array} \right\} \dot{\theta}_1 \\
C_{13} = \left\{ 2m_3 L_2 L_3 s_{23} c_2 + 2m_3 L_3^2 s_{23} c_{23} \right\} \\
C_{22} = \left\{ \begin{array}{l} (m_2 + m_3) L_2^2 s_2 c_2 + m_3 L_2 L_3 (s_{23} c_2 + s_2 c_{23}) + \\ m_3 L_3^2 s_{23} c_{23} \end{array} \right\} \\
C_{21} = -\left\{ 2m_3 L_2 L_3 s_3 \right\} \dot{\theta}_3 \\
C_{21} = -\left\{ m_3 L_2 L_3 s_3 \right\} \dot{\theta} \\
C_{31} = \left\{ m_3 L_2 L_3 s_{23} c_2 + m_3 L_3^2 s_{23} c_{23} \right\} \\
C_{32} = \left\{ m_3 L_2 L_3 s_3 \right\} \dot{\theta} \\
C_{33} = 0
\end{array}
\tag{5.87}
$$

El vector de fuerzas gravitatorias se define a continuación

$$
G(q) = \begin{bmatrix} G_{11} \\ G_{21} \\ G_{31} \end{bmatrix}
\tag{5.88}
$$

y sus elementos son:

$$
\begin{array}{l}
G_{11} = 0 \\
G_{21} = -m_2 L_{CG2} c_2 - m_3 \left\{ L_2 c_2 - L_{CG3} c_{23} \right\} \\
G_{31} = -m_3 L_{CG3} c_{23}
\end{array}
\tag{5.89}
$$

Vector	Descripción
$Fv\left(\dot{q}\right)$	Fricción viscosa
$Fs\left(\dot{q}\right)$	Fricción seca

Cuadro 5.4: Descripción de vectores para la fricción

El MD que descrito en la ecuación 5.83 ha tomado en cuenta la fricción viscosa y seca, este vector fue definido como $F\left(\dot{q}\right)$ el cual es la suma del vector de fricción seca y el vector de fricción viscosa que a continuación se describe

$$F\left(\dot{q}\right) = Fv\left(\dot{q}\right) + Fs\left(\dot{q}\right) \tag{5.90}$$

En donde:
$Fv\left(\dot{q}\right)$ se describe a continuación

$$Fv\left(\dot{q}\right) = \begin{bmatrix} Fv_1\left(\dot{q}\right) \\ Fv_2\left(\dot{q}\right) \\ Fv_3\left(\dot{q}\right) \end{bmatrix} = \begin{bmatrix} b_1\dot{\theta}_1 \\ b_2\dot{\theta}_2 \\ b_2\dot{\theta}_2 \end{bmatrix} \tag{5.91}$$

en donde b_i es el coeficiente de fricción viscosa o rozamiento dinámico y $\dot{\theta}_1$ es la velocidad angular
$Fs\left(\dot{q}\right)$ se describe a continuación

$$Fs\left(\dot{q}\right) = \begin{bmatrix} Fs_1\left(\dot{q}\right) \\ Fs_2\left(\dot{q}\right) \\ Fs_3\left(\dot{q}\right) \end{bmatrix} = \begin{bmatrix} K_1\text{sgn}\left(\dot{\theta}_1\right) \\ K_2\text{sgn}\left(\dot{\theta}_2\right) \\ K_2\text{sgn}\left(\dot{\theta}_3\right) \end{bmatrix} = \begin{bmatrix} K_1\tanh\left(\beta_1\dot{\theta}_1\right) \\ K_2\tanh\left(\beta_2\dot{\theta}_2\right) \\ K_2\tanh\left(\beta_3\dot{\theta}_3\right) \end{bmatrix} \tag{5.92}$$

en donde K_i es el coeficiente de fricción seca que depende del grado de lubricación de la articulación, sgn es la función signo y $\dot{\theta}_i$ es la velocidad angular.

5.5. Control

Las formulas evaluadas por el método de Runge-Kutta son las ecuaciones obtenidas a partir del modelo dinámico, para ello hay que despejar las aceleraciones articulares resultantes de dicho modelo, como se muestra en las siguientes formulas.

En primer lugar se despeja la aceleración 1 de la formula de control para la variable articular numero 1 5.93.

$$\tau_1 = H_{11}\,\ddot{\theta}_1 + C_{12}\,\dot{\theta}_2 + C_{13}\,\dot{\theta}_3 + F_1\,\dot{\theta}_1 \tag{5.93}$$

Despejando:

$$\ddot{\theta}_1 = \frac{1}{H_{11}}\left\{\tau_1 - C_{12}\,\dot{\theta}_2 - C_{13}\,\dot{\theta}_3 - F_1\,\dot{\theta}_1\right\} \tag{5.94}$$

La aceleración 2 parte de la formula de τ_2 de lo cual se obtiene:

$$\tau_2 = H_{22}\,\ddot{\theta}_2 + H_{23}\,\ddot{\theta}_3 + C_{21}\,\dot{\theta}_1 + C_{22}\,\dot{\theta}_2 + C_{23}\,\dot{\theta}_3 + G_2 + F_2\,\dot{\theta}_2 \tag{5.95}$$

Despejando:

$$\ddot{\theta}_2 = \frac{1}{H_{22}}\left\{\tau_2 - H_{23}\,\ddot{\theta}_3 + C_{21}\,\dot{\theta}_1 + C_{22}\,\dot{\theta}_2 + C_{23}\,\dot{\theta}_3 + G_2 + F_2\,\dot{\theta}_2\right\} \tag{5.96}$$

Por último tenemos para τ_3 la ecuación:

$$\tau_3 = H_{32}\,\ddot{\theta}_2 + H_{33}\,\ddot{\theta}_3 + C_{31}\,\dot{\theta}_1 + C_{32}\,\dot{\theta}_2 + G_3 + F_3\,\dot{\theta}_3 \tag{5.97}$$

Despejando:

$$\ddot{\theta}_3 = \frac{H_{32}\left[\tau_2 - H_{23}\,\ddot{\theta}_3 + C_{21}\,\dot{\theta}_1 + C_{22}\,\dot{\theta}_2 + C_{23}\,\dot{\theta}_3 + G_2 + F_2\,\dot{\theta}_2\right] + \left[C_{31}\,\dot{\theta}_1 + C_{32}\,\dot{\theta}_2 + G_3 + F_3\,\dot{\theta}_3 + \tau_3\right]}{H_{23}\,H_{23} - H_{22}\,H_{33}} \tag{5.98}$$

En las siguientes secciones se presentan los diversos comportamientos con respecto a posición, velocidad, error y fuerza de par o torque en los diferentes controles que se analizarón en el presente trabajo.
Para la ejecución del programa se establecierón condiciones que corresponden a un modelo de una interfaz háptica real con lo cual se observa una modelación de la realidad y no del mundo virtual pues con ello en un futuro puede integrarse la realidad y el ámbiente virtual para observar el movimiento real de un dispositivo háptico Phantom, de tal manera que se tienen los siguientes parámetros para el cálculo del modelo matemático:

- Masas de los eslabones:
  1. Longitud de los eslabones
  2. Primer eslabón m1=0.45;
  3. Segundo eslabón m2=0.25;
  4. Tercer eslabón m3=0.15;
- Longitudes de los eslabones:
  1. Primer eslabón l1=0.35;
  2. Segundo eslabón l2=0.35;
  3. Tercer eslabón l3=0.2;

- Longitud al centro de masas:
  1. Primer eslabón LCG1=0.5*l1;
  2. Segundo eslabón LCG2=0.5*l2;
  3. Tercer eslabón LCG3=0.5*l3;
- Fricción seca:
  1. Primer eslabón f1=0.1;
  2. Segundo eslabón f2=0.5;
  3. Tercer eslabón f3=0.01;
- Fricción Viscosa:
  1. Primer eslabón b1=0.3;
  2. Segundo eslabón b2=0.15;
  3. Tercer eslabón b3=0.4;
- Gravedad gr=9.81;

Para las variables articulares de referencia y de inicio se considerarón en cada uno de los controles análizados en el presente trabajo de estudio los siguientes valores:

θ_1 = 0.8 radianes

θ_2 = - 0.8 radianes

θ_3 =-0.8 radianes

Así como los valores siguientes para las variables de referencia:

θ_{ref1} = 0.8 radianes

θ_{ref2} = 0.8 radianes

θ_{ref3} = 0.8 radianes

El paso de integración dentro del método para la ejecución del control establecido para las comprobaciones es siempre de h = 0.001.

5.5.1. Control PD

Una vez ejecutado el programa, se hizo la comprobación gráfica de resultados con la ayuda de MATLAB, cuyos resultados aparecen en las gráficas siguientes.

El punto de referencia para cada una de las variables articulares es de 0.8 radianes, en primera instancia vemos en la siguiente gráfica que θ_1 intenta alcanzar este punto, primero hace una oscilación y después llega al equilibrio, sin embargo tiene un error de .0095 radianes, pero el comportamiento es óptimo. Con estos valores se tiene el comportamiento de la primera variable articular en la gráfica 5.6. La variable articular dos tiene un comportamiento parecido a θ_1, la diferencia radica en que este tiene una oscilación mayor como lo muestra la gráfica 5.7.

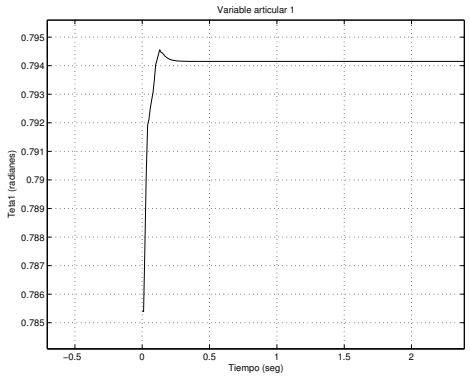

Figura 5.6: Gráfica de la variable articular 1 del control PD.

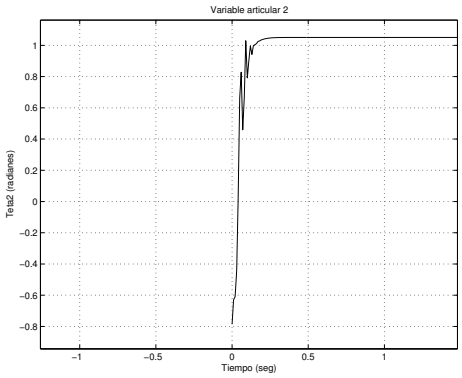

Figura 5.7: Gráfica de la variable articular 2 del control PD.

La variable articular tres sufre las mismas condiciones que los ángulos anteriores, pero se diferencia su comportamiento ya que tiene una oscilación mas acentuada y el error es aun mayor, tal fenómeno se muestra en la gráfica 5.8. Las velocidades que adquiere el punto de referencia para al ángulo uno que oscila en un principio pero se estabilizan, como se muestra en gráfica 5.9.

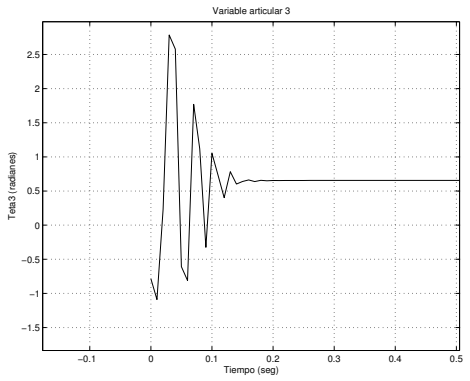

Figura 5.8: Gráfica de la variable articular 3 del control PD.

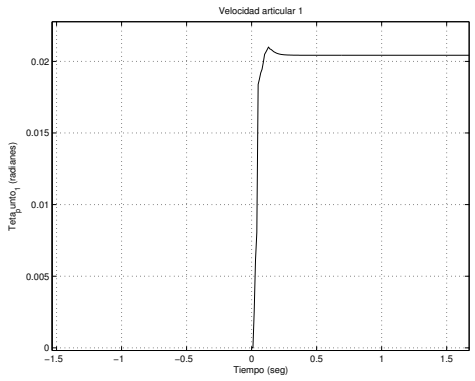

Figura 5.9: Gráfica de la velocidad articular 1 del control PD.

La velocidad articular dos tiene una mayor oscilación, sin embargo llega a un estado estable como lo muestra la gráfica 5.10.

La velocidad articular 3, tiene una visible variación al inicio, esto debido a que las otras articulaciones intervienen en la dinámica del tercero, pero finalmente se estabiliza 5.11.

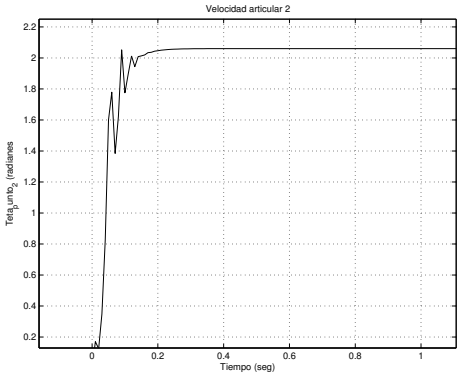

Figura 5.10: Gráfica de la velocidad articular 2 del control PD.

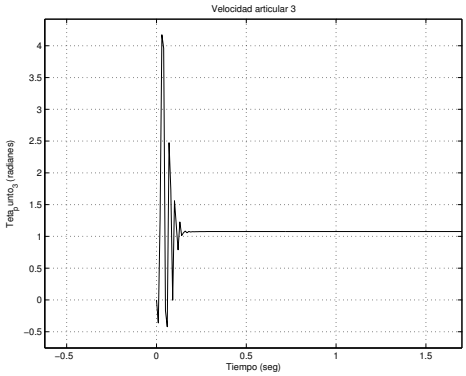

Figura 5.11: Gráfica de la velocidad articular 3 del control PD.

De igual forma se ha hecho una comparación gráfica del error obtenido al momento de realizar la integración para posicionar a las variables articulares en los puntos de referencia antes mencionados y los resultados se muestran más adelante.

Error dado para θ_1.

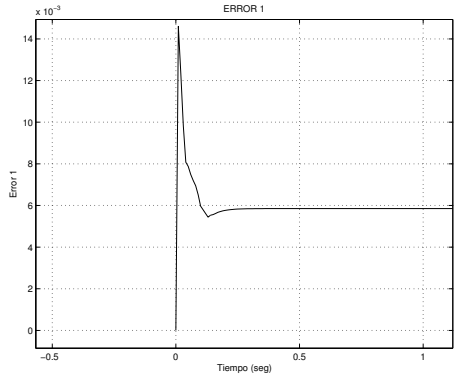

Figura 5.12: Gráfica del error 1 del control PD.

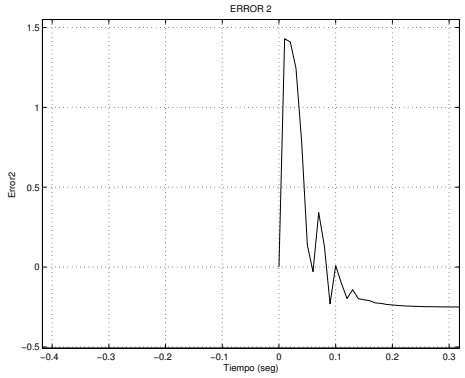

Figura 5.13: Gráfica del error 2 del control PD.

El error que se presenta en θ_2, durante el tiempo en que se ejecuta tiende a converger a cero, como se muestra en la gráfica 5.13.

Asi mismo, el error que se presenta en θ_3, se comporta con oscilación, en un inicio , pero finalmente converge a cero 5.14.

También se muestra el control dado para que θ alcance los puntos de referencia, este torque se ha definido en base al control PD, obteniendo de manera gráfica los siguientes resultados:

Se puede interpretar, al torque como la fuerza que es aplicada, a cada eslabon del robot, haciendo que se posicione en la referencia. En cuanto al τ_1 inicia en cero, pero al ser aplicada la fuerza se mueve el eslabón, aunque se disminuye porque el eslabón se mueve cada vez más cerca del punto deseado vease gráfica 5.15.

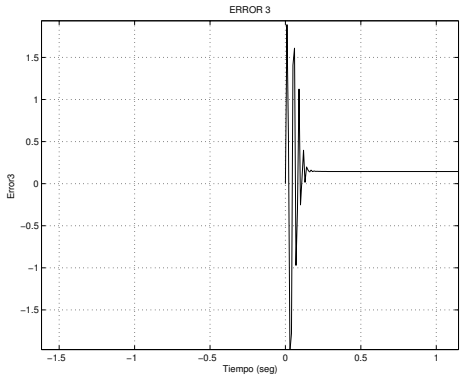

Figura 5.14: Gráfica del error 3 del control PD.

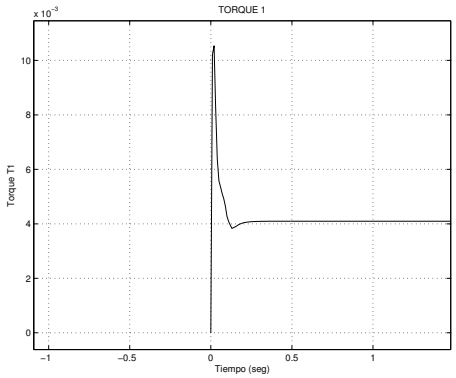

Figura 5.15: Gráfica de τ_1 para el control PD.

En el segundo eslabón se observa que en τ_2 el comportamiento es un poco diferente, esto debido a que el movimiento del primer eslabón impide que el segundo y tercero lleguen al punto de referencia, por ello se aplica una fuerza mayor a la aplicada en el primer eslabón, como se muestra en la gráfica 5.16.

Hay que recordar que si el brazo del robot se soltara, simplemente se caería a una posición en que llegaría a un estado de reposo, pero la finalidad del control, es que se lleve el brazo a una posición, y una vez alcanzada dicha posición se mantenga, por lo tanto la aplicación del control permite sostener el brazo para que esté no caiga, de esta forma τ_3, mantiene un valor por encima de cero y se mantiene ahí, vease la gráfica 5.17.

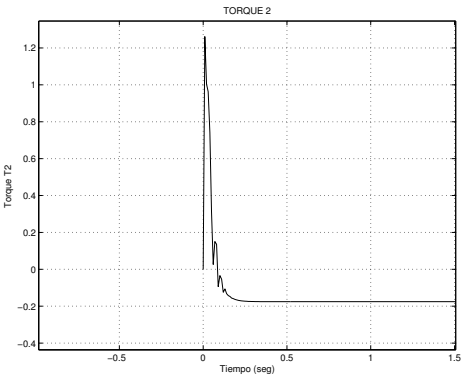

Figura 5.16: Gráfica de τ_2 para el control PD.

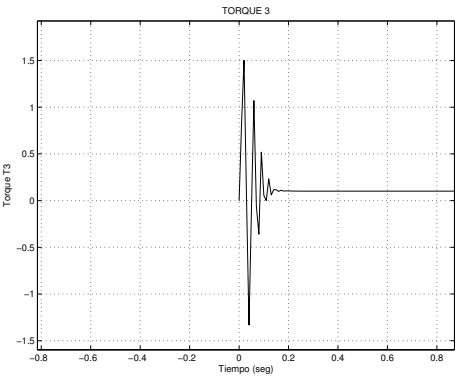

Figura 5.17: Gráfica de τ_3 para el control PD.

5.5.2. Control PID

Una vez que se ha ejecutado el programa y hecha la interacción con MATLAB para fines de graficación de resultados obtenemos las gráficas que se muestran en la siguiente sección.

Recordando que el punto de referencia para cada una de las θ's es de 0.8 radianes, en primera instancia vemos en la gráfica 5.18 que θ_1 intenta alcanzar este punto, primero hace una oscilación y después llega al equilibrio, sin embargo tiene un error de .0095 radianes, pero el comportamiento es óptimo.

La variable articular dos tiene un comportamiento parecido a θ_1, la diferencia radica en que este tiene una oscilación muy ligera que es casi imperceptible como lo muestra la gráfica 5.19. La variable θ_3 sufre las mismas

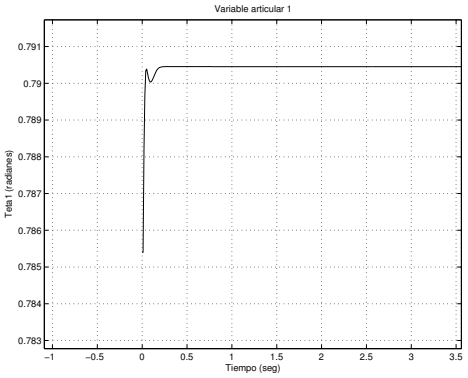

Figura 5.18: Variable articular 1 del control PID.

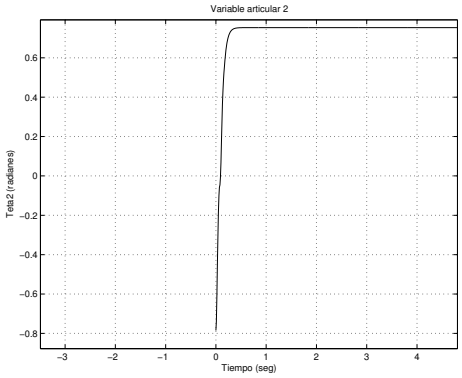

Figura 5.19: Variable articular 2 del control PID.

condiciones que las θ's anteriores, pero se diferencia su comportamiento ya que tiene una oscilación mas acentuada y el error es aun mayor, tal fenómeno se muestra en la siguiente gráfica 5.20.
La velocidad alcanzada por cada una de las variables articulares se muestra en las gráficas a las que más adelante se hace referencia.
En la gráfica 5.21 se muestra la velocidad articular 1 en donde la oscilación es menor a comparación de los resultados anteriores acercandose asi al valor deseado.
Para el valor de la velocidad para θ se muesta en la gráfica 5.23 en donde se aprecia una oscilación mayor, siendo éste más cercano al valor de referencia.

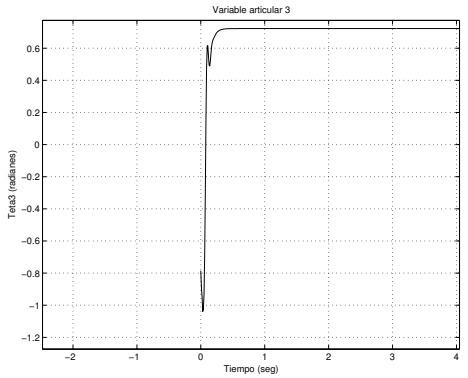

Figura 5.20: Variable articular 3 del control PID.

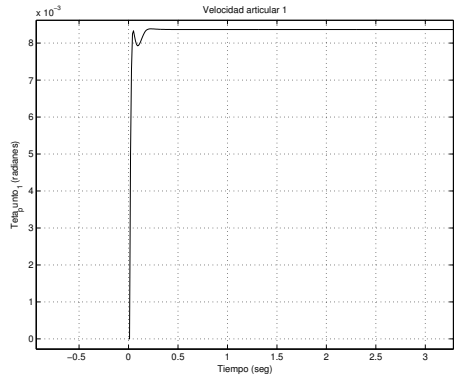

Figura 5.21: Velocidad Articular 1 del control PID.

La velocidad en θ_2 se aprecia en la gráfica 5.22 donde la velocidad demostrada es casi perfecta ya que la oscilación es casi nula pero solo se acerca más al valor deseado.
Para el valor de la velocidad de θ_3 se muestra una oscilación mayor a la del eslabón anterior, sin embargo llega a la posición deseada, como se ilustra en la gráfica 5.23.

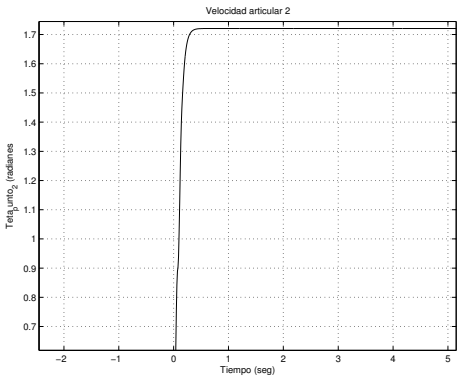

Figura 5.22: Velocidad articular 2 del control PID.

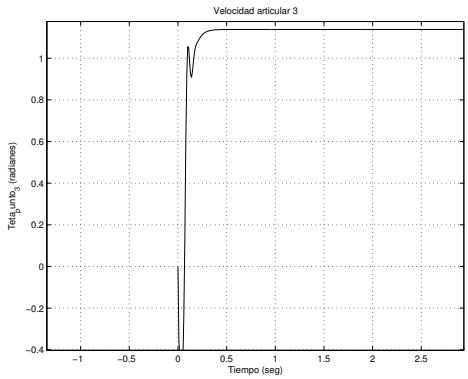

Figura 5.23: Velocidad articular 3 del control PID.

También se ha hecho una comparación gráfica del error obtenido al momento de realizar la integración para posicionar a las variables articulares en los puntos de referencia antes mencionados y los resultados se muestran en las siguientes gráficas.
En la gráfica 5.24 se puede observar que a diferencia del control PD este se aproxima más a cero, este error para θ_1 muestra una oscilación que tiende a acercarse más a cero pero inmediatamente se establece alejándose de cero.
En la gráfica 5.25 que muestra el error para θ_2 la aproximación a cero esta más cerca oscilando hasta un valor estable que casi es similar al valor deseado.

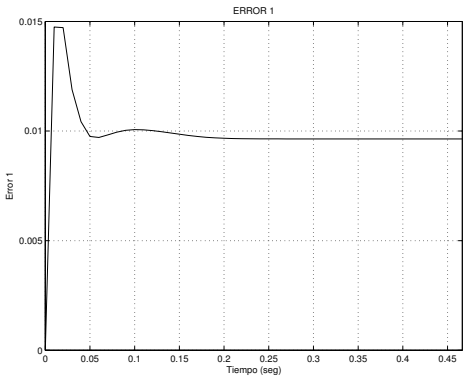

Figura 5.24: Error para la articulación 1 aplicando el control PID.

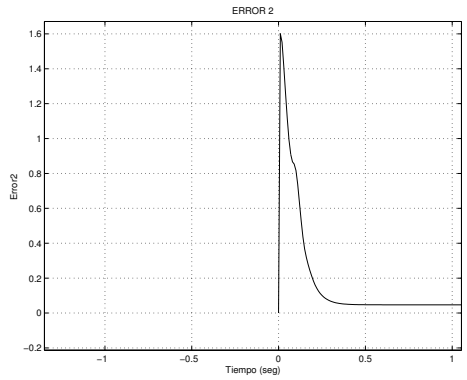

Figura 5.25: Error para la articulación 2 aplicando el control PID.

La gráfica 5.26 nos muestra una vez mas la aproximación que tiene a cero, la oscilación es un poco mas que las anteriores , pero en el momento en que se llega a establecer esta llega a acercarse al punto deseado.

El control dado para que θ alcance los puntos de referencia, esta definido en base al control PID, obteniendo de manera gráfica los siguientes resultados.
La gráfica 5.27 muestra los resultados al aplicar una fuerza sobre el torque 1, se muestra una pequeña oscilación en donde el brazo del robot tendría un pequeño movimiento para después establecer su valor, la oscilación es mínima pero no la óptima. En la gráfica 5.28 se muestra la fuerza aplicada al τ_2 y su comportamiento es más

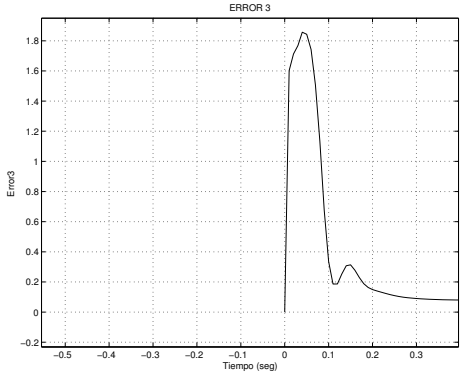

Figura 5.26: Error para la articulación 3 aplicando el control PID.

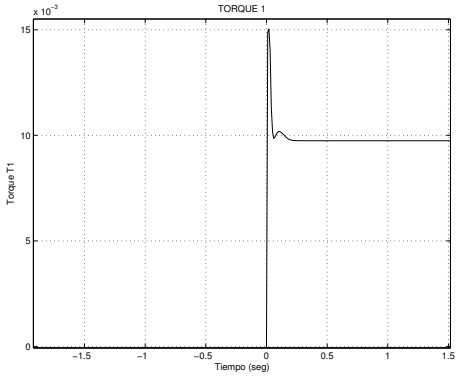

Figura 5.27: τ_1 aplicado en el control PID.

estable que el anterior ya que el brazo del robot estaría oscilando muy poco.

La gráfica 5.29muestra la fuerza aplicada en τ_3 la cual demuestra más oscilación que las anteriores y el tiempo en que se establece es mayor, una vez que se estabiliza casi llega al punto deseado.

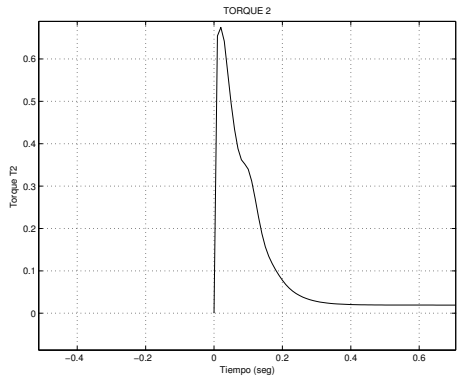

Figura 5.28: τ_2 aplicado en el control PID.

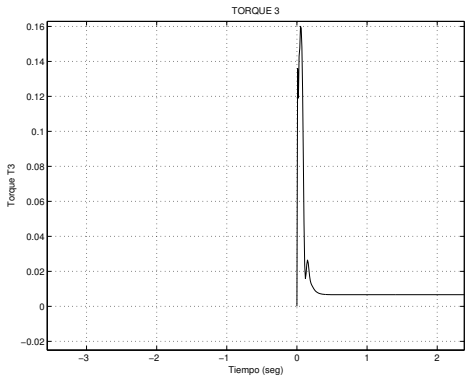

Figura 5.29: τ_3 aplicado en el control PID.

5.5.3. Control PID No Lineal

Una vez ejecutado el programa se obtienen las siguientes gráficas, en primer lugar se tiene θ_1 cuyo comportamiento se ilustra en la gráfica 5.30 en donde se puede observar que dicha variable alcanza un estado estable rápidamente y llega a la posición de referencia dado en el programa que para este caso es de 0.8 radianes.

En el caso de θ_2 tienen un comportamiento similar ya que llega al estado estable en un mayor tiempo pero con la misma exactitud que el anterior, como se puede apreciar en la gráfica 5.31.

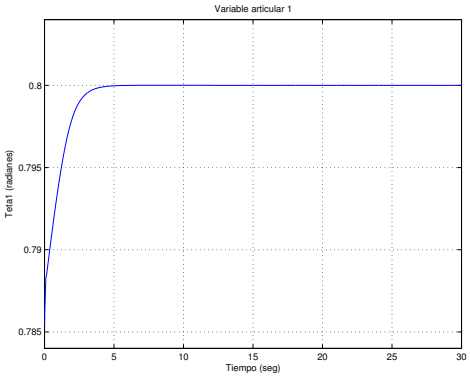

Figura 5.30: Variable 1 en el control PID NO LINEAL.

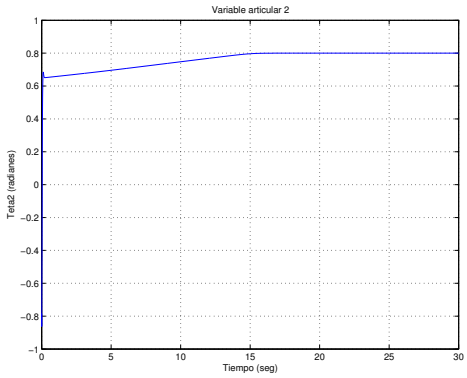

Figura 5.31: Variable 2 en el control PID NO LINEAL.

Para θ_3 la exactitud es la misma pero la rapidez en alcanzar dicha posición es mayor como lo muestra la gráfica 5.32.

En el caso de la velocidad articular 1 el comportamiento es bastante estable y preciso como se puede observar en la gráfica 5.33

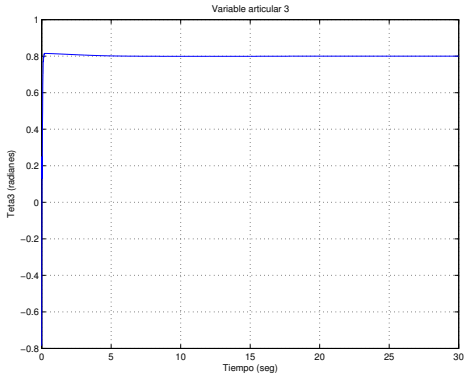

Figura 5.32: Variable 3 en el control PID NO LINEAL.

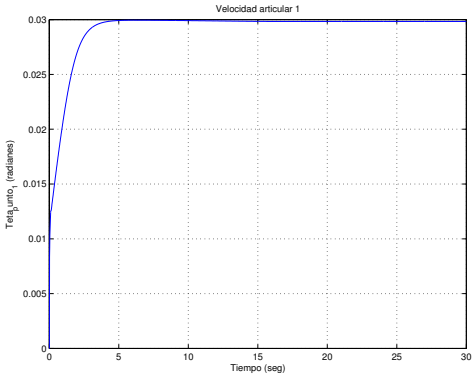

Figura 5.33: Velocidad articular 1 en el control PID NO LINEAL.

Para la velocidad de θ_2 se puede observar en la gráfica 5.34 que el comportamiento de está es similar a la de la articulación 1.

La velocidad para θ_3 como en cada uno de los casos de las velocidades esta tiende a cero y con un comportamiento bastante estable vease gráfica 5.35. En el caso de los errores podemos observar que el uso de este control

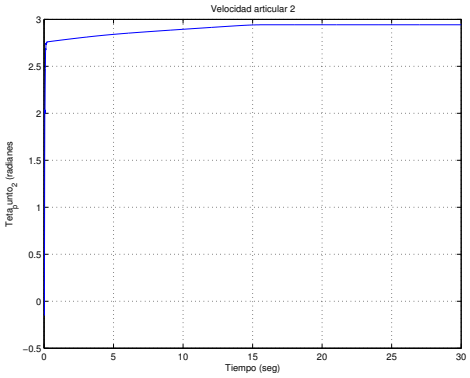

Figura 5.34: Velocidad articular 2 en el control PID NO LINEAL.

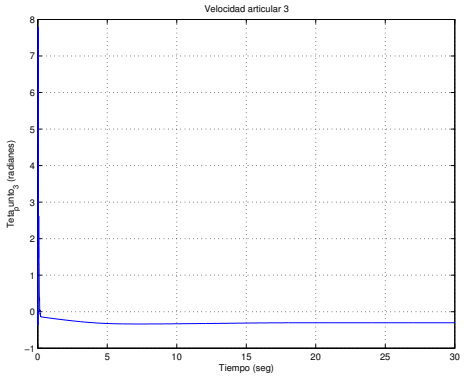

Figura 5.35: Velocidad articular 3 en el control PID NO LINEAL.

en este caso en particular no tiene errores como lo muestran las siguientes figuras una por cada articulación. El Error para θ_1 se muestra en la gráfica 5.36.
El Error para θ_2 se muestra en la gráfica 5.37.

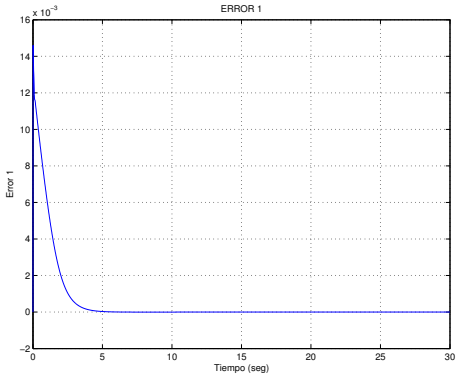

Figura 5.36: Error 1 en el control PID NO LINEAL.

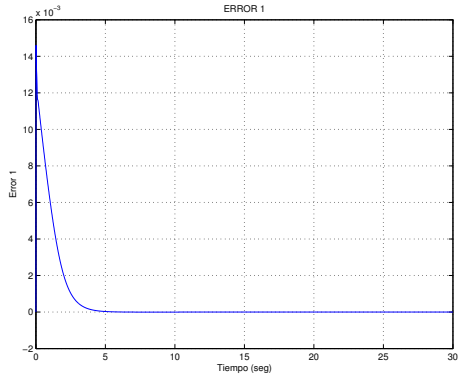

Figura 5.37: Error 2 en el control PID NO LINEAL.

El Error para la variable articular número tres se muestra en la gráfica 5.38.

En el caso de la fuerza de par el comportamiento para cada una de las articulaciones es que esta fuerza es ejercida y proporciona el control que se esta buscando para que una vez que ya no se efectua tal fuerza caiga en cero dicha fuerza como se puede observar en las siguientes gráficas.
El control aplicado en la variable articular número uno se muestra en la gráfica 5.39. El control aplicado en θ_2 se

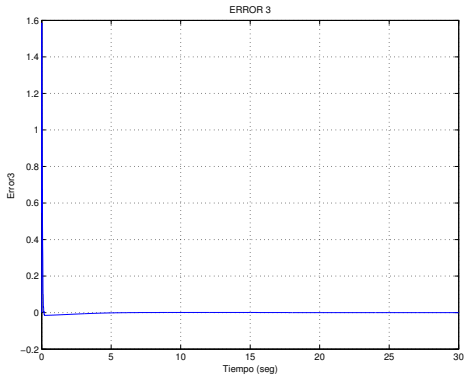

Figura 5.38: Error 3 en el control PID NO LINEAL.

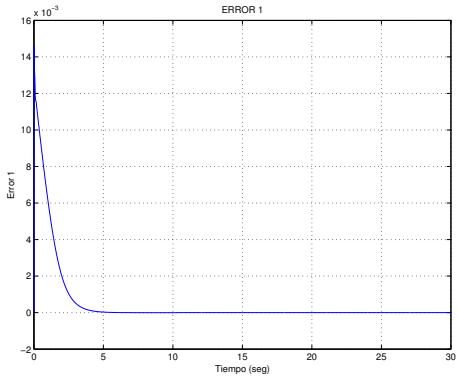

Figura 5.39: τ_1 en el control PID NO LINEAL.

muestra en la gráfica 5.40.

El control aplicado en θ_3 se muestra en la gráfica 5.41.

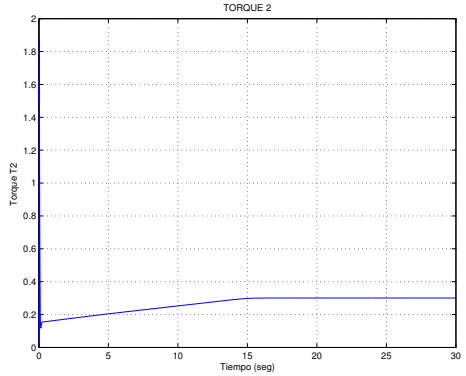

Figura 5.40: τ_2 en el control PID NO LINEAL.

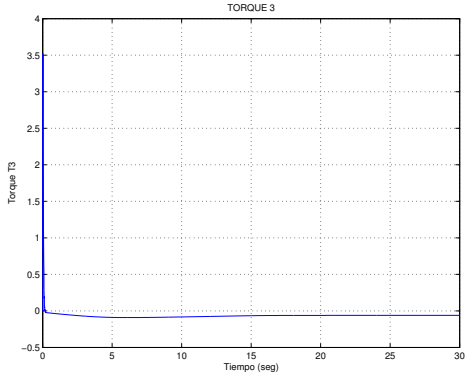

Figura 5.41: τ_3 en el control PID NO LINEAL.

Control PD						
Tiempo	X	Y	Z	$\dot{X}$	$\dot{Y}$	$\dot{Z}$
0.000000	0.785398	-0.785398	-0.785398	0.000000	0.000000	0.000000
30.00000	0.794724	0.995144	0.722283	0.019499	2.347016	-0.10516
Control PID						
Tiempo	X	Y	Z	$\dot{X}$	$\dot{Y}$	$\dot{Z}$
0.000000	0.785398	-0.785398	-0.785398	0.000000	0.000000	0.000000
30.00000	0.790453	0.753363	0.722372	0.008365	1.720657	1.138081
Control PID No Lineal						
Tiempo	X	Y	Z	$\dot{X}$	$\dot{Y}$	$\dot{Z}$
0.000000	0.785398	-0.785398	-0.785398	0.000000	0.000000	0.000000
30.00000	0.800000	0.800000	0.800000	0.029835	2.942263	-0.30211

Cuadro 5.5: Tabla comparativa del resumen de los resultados de la evaluación de los controles utilizados de posiciones y velocidades

Control PD						
Tiempo	$Error_1$	$Error_2$	$Error_3$	T_1	T_2	T_3
0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
30.00000	0.005276	-0.19514	0.077717	0.003694	-0.136601	0.054402
Control PID						
Tiempo	$Error_1$	$Error_2$	$Error_3$	T_1	T_2	T_3
0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
30.00000	0.009642	0.047101	0.078400	0.009740	0.019251	0.006654
Control PID No Lineal						
Tiempo	$Error_1$	$Error_2$	$Error_3$	T_1	T_2	T_3
0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
30.00000	0.000000	0.000000	0.000000	0.032545	0.300528	-0.06129

Cuadro 5.6: Tabla comparativa del resumen de los resultados de la evaluación de los controles utilizados de errores y Torques

5.6. Conclusiones

La relevancia del método numérico creado por Runge y Kutta en el presente trabajo está dado por la exactitud que proporciona ya que el estudio comparativo de dicho método asi lo demuestra,más aún que de esta forma ya no se depende de otros programas en el momento de hacer la visualización analizada en el capítulo 4.

El modelo dinámico para este trabajo permitío la evaluación del comportamiento de los diferentes controles, es por ello que es importante saber el como se obtiene dicho modelo, que como se puede ver en este capítulo parte del modelo cinemático.

De está forma una vez hecho el análisis comparativo y viendo el comportamiento que cada uno de los controles como se muestra con anterioridad se ha decidido hacer uso del **control no lineal**, para la visualización, ya que de los tres es el que tiene un mejor comportamiento.

Capítulo 6

Conclusiones y perspectivas

6.1. Conclusiones generales

Los resultados relevantes de este trabajo de tesis corresponden a resolver las limitaciones del primer visualizador virtual de robots manipuladores antropomórficos desarrollado en el CITIS, y cuyos beneficios con respecto de los existentes en el estado de arte es el de considerar la dinámica del robot. El propósito inicial es la de evaluar estrategias de control clásico y moderno para planificación de tareas y visualización del dispositivo háptico PHANToM 1.0. Resultando muy generoso para verificar el desempeño del robot en el espacio de trabajo bajo la influencia de un determinado controlador. Los resultados obtenidos que permiten concluir este trabajo de tesis son:

- Solución numérica de la dinámica del robot en lazo abierto y en lazo cerrado empleando el método de integración Runge-Kutta de 4º orden y su comprobación con el integrador ODE45 de Matlab.

- Implementación del método de integración Runge-Kutta de 4º orden al visualizador virtual en Visual C++.

- Evaluación de dos controladores clásicos (lineales) y un moderno (no lineal) y el correspondiente estudio comparativo.

- Desarrollo e implementación de una interfaz de usuario que permite la interacción del operador con el mundo virtual, tal que es posible definir la referencia de movimiento (posición y velocidad) y conocer las variables que definen el desempeño del robot en lazo cerrado para un controlador determinado.

- Dejar en arquitectura abierta al sistema con el afán de poder implementar diversos controladores en este robot, con el afán de verificar el desempeño.

Por lo anterior se concluye que los objetivos propuestos al inicio de este trabajo de tesis fueron cumplidos y que el simulador y visualizador virtual de robots es más eficiente con los aportes anteriormente mencionados.

6.2. Perspectivas y trabajo futuro

Los trabajos a futuro que se pretenden desarrollar en el simulador y visualizador virtual de robots están definidos a continuación:

- Aplicación de otras técnicas de control de movimiento.

- Planificación de trayectorias cerradas y su visualización.
- Planificación de tareas a partir de trayectorias abiertas.
- Identificación paramétrica de la dinámica de robots.
- Evasión de colisiones con objetos virtuales estáticos.
- Evasión de colisiones con objetos virtuales dinámicos.
- Robots cooperativos virtuales.
- Convergencia en tiempo finito para robots cooperativos.
- Establecimiento de una página WEB que permita el entrenamiento remoto.

Apéndice A

Abreviaturas, Glosario y Apéndices

A.1. Lista de abreviaturas

E-L Euler - Lagrange.

ISO International Standard Organization(Organización Internacional de Estandarización).

L Lagrangiano.

LCG Longitud al Centro de Masas.

LVRM Laboratorio Virtual de Robótica y Manufactura.

MCDA Modelo Cinemático Directo de Aceleración.

MCIA Modelo Cinemático Inverso de Aceleración.

MCDP Modelo Cinemático Directo de Posición.

MCIP Modelo Cinemático Inverso de Posición.

MCDV Modelo Cinemático Directo de Velocidad.

MCIV Modelo Cinemático Inverso de Velocidad.

MD Modelo Dinámico.

N-E Newton-Euler.

PD Proporcional Derivativo.

PID Proporcional Integral Derivativo.

PDH Parámetros de Denavit- Hartenberg.

R-K Runge-Kutta.

VRML Virtual Reality Modeling Languaje(Lenguaje de Modelado de Realidad Virtual).

A.2. Glosario

Articulación de rotación: suministra un grado de libertad consistente en una rotación alrededor del eje de la articulación. Está articulación es, con diferencia, la más empleada [17].

Articulación prismática: el grado de libertad consiste en una traslación a lo largo del eje de la articulación [9]

Cadena cinemática: Es un ensamble de eslabones y juntas interconectados de modo que proporcionen un movimiento de salida controlado en respuesta a un movimiento de entrada proporcionado[9].

Centro de masa: Es la posición geométrica de un cuerpo rígido en la cual se puede considerar concentrada toda su masa; corresponde a la posición promedio de todas las partículas de masa que forman el cuerpo rígido. El centro de masa de cualquier objeto simétrico homogéneo, se ubica sobre un eje de simetría[9].

Cinemática: es la parte de la física que estudia el movimiento de los cuerpos, aunque sin interesarse por las causas que originan dicho movimiento. Un estudio de las causas que lo originan es lo que se conoce como dinámica. Las magnitudes que define la cinemática son principalmente tres, la posición, la velocidad y la aceleración [7]

Control: Selección de las entradas del sistema de manera que los estados o salidas cambien de acuerdo a una manera deseada[7].

Dinámica: estudia precisamente por qué se mueven los cuerpos, es decir, cuáles son las causas que crean la variación de su estado de movimiento. La dinámica es un campo de las ciencias dedicado al estudio de las fuerzas requeridas para producir el movimiento [10]

Energía Cinética: la energía cinética de un cuerpo en movimiento se define como la capacidad de efectuar trabajo en virtud de su movimiento [10].

Energía Potencial: es una medida de la cantidad de trabajo que realizara una fuerza conservativa cuando se desplaza desde una posición fija con respecto del punto de referencia [10]. Cambio de una función de la posición del cuerpo realizado por algunas fuerzas [10]

Eslabón: Cuerpo rígido que posee al menos dos nodos, que son los puntos de unión con otros eslabones. El número de nodos le da su nombre al eslabón: Binario = dos nodos, Terciario = tres nodos,etc [14].

Junta o par cinemático: Conexión entre dos o más eslabones que permite algún movimiento o movimiento potencial entre los eslabones conectados[9].

Pueden clasificarse en varios modos:
Por el número de grados de libertad.
- Rotacional 1 GDL
- Prismática o Deslizante 1 GDL

Espacio de Trabajo: Región del Espacio compuesta por todos los puntos que pueden ser alcanzados por el final del brazo del robot (brida o TCP 0), sin considerar el elemento terminal. El robot tiende a tener una geometría fija, y limitada. El espacio de trabajo es el límite de posiciones en espacio que el robot puede alcanzar. Para un robot cartesiano (como una grúa arriba) los espacios de trabajo podrían ser un cuadrado, para los robots más sofisticados los espacios podrían ser de una forma esférica [9].
El torque: es la fuerza aplicada en una palanca que hace rotar alguna cosa. Al aplicar fuerza en el extremo de una llave se aplica un torque que hace girar las tuercas. En términos científicos el torque es la fuerza aplicada multiplicada por el largo de la palanca (Torque = F x D) y se mide comúnmente en Newtons metro [9].

Fuerza Centrífuga: Esta, es definida como la fuerza que tiende a que todos los cuerpos en rotación traten de alejarse de su eje. Es una de las fuerzas dominantes en el estudio de las alas rotativas [7].

Fuerza centrípeta: cuando una partícula se limita a recorrer una trayectoria circular con una rapidez constante, existe una fuerza normal ejercida sobre la partícula debida ala restricción puesto que esta fuerza siempre se dirige hacia el centro de la trayectoria [10]

Fricción: Fuerza de reacción tangencial entre dos superficies en contacto. Estas fuerzas de reacción físicamente son el resultado de diferentes mecanismos, los cuales dependen de la geometría y topología de contacto, propiedades del cuerpo y de los materiales de la superficie del mismo, desplazamiento y velocidad relativa de los cuerpos y de la presencia de lubricación [9].

Hamiltoniano: Es una reformulación de la mecánica clásica. Los mecánica hamiltoniana puede ser formulada por si misma, usando los espacios simplécticos, sin referir a cualesquiera conceptos anteriores de fuerza o de la mecánica lagrangiana. Vea la sección en su formulación matemática para esto. Para la primera parte de este artículo, mostraremos cómo surge históricamente del estudio de la mecánica lagrangiana [9].

Inercia: Propiedad de un cuerpo que tiende a oponerse a toda variación en su estado de reposo o de movimiento [9].

Lagrangiano: es la diferencia entre la energía cinética y la energía potencial de un cuerpo en movimiento en un punto dado de su trayectoria [15].

Matriz de rotación : Una matriz de rotación 3 X 3 se puede definir como una matriz de transformación que opera sobre un vector de posición en un espacio euclídeo tridimensional y transforma sus coordenadas ligado al cuerpo a un sistema de coordenadas de referencia OXYZ. En la figura siguiente se dan dos sistemas de coordenadas rectangulares, uno el sistema de coordenada OXYZ; con OX, OY y OZ como sus ejes de coordenadas, y el sistema de coordenadas OUVW, con OU, OV, OW como sus ejes de coordenadas. En ambos sistemas de coordenadas tiene sus orígenes coincidentes en el punto O [9].

Matriz de transformación: la matriz de transformación homogénea es una matriz 4 X 4 que transforma un vector de posición expresado en coordenadas homogéneas desde un sistema de coordenadas hasta otra sistema de coordenadas. Una matriz de transformación homogénea se puede considerar que consiste en cuatro submatrices [14].

Mecanismo: Es una cadena cinemática en la cual por lo menos un eslabón ha sido fijado o sujetado al marco de referencia (el cual puede estar en movimiento) [9].

Modelo: Representación de una manera más sencilla de ciertos aspectos de un proceso o elemento, y el cual es utilizado con objetivos de análisis, control y predicción. Todo modelo se basa en una teoría, pero dicha teoría puede no estar indicada en una forma concisa. Tipos de modelos: a escala, descriptivos, gráficos, analógicos y matemáticos [7].

Momento: el momento de una fuerza alrededor de un punto dado es el producto de dicha fuerza y la distancia en ángulo recto desde ese punto a la línea de acción de la fuerza. M = r x F [9].

Momento de inercia: es una medida de la inercia rotacional o la tendencia de un cuerpo a resistirse al cambio en su movimiento rotacional. Aunque se dice que I debe ser constante para un cuerpo rígido, y que es el análogo rotacional de la inercia, corresponde a un eje determinado y puede tener valores diferentes para ejes diferentes. El momento de inercia depende también de la distribución de la masa referente al eje de rotación.Momento de inercia: expresiones del movimiento de cuerpos regidos en las que interviene el producto de la masa de un pequeño elemento del cuerpo por el cuadrado de su distancia a una recta de interés. Este producto recibe el nombre de momento de inercia, el cual en una masa será siempre positivo [16]. Es una forma de medir la resistencia opuesta por un cuerpo ala aceleración angular, igual que la masa es una medición de la resistencia opuesta por el

campo ala aceleración [10]

Organo terminal: El manipulador, ejecutor o efector final: (dispositivos de manipulación: brazos, muñecas, manos, herramientas, dispositivos de succión y magnéticos. El manipulador o brazo son los elementos mecánicos que propician el movimiento del elemento terminal. Los elementos rígidos del brazo están relacionados entre sí mediante articulaciones, las cuales pueden ser giratorias o prismáticas. El número de elementos del brazo y el de las articulaciones que los relacionan determinan los grados de libertad del manipulador, que en los robots industriales suele ser seis [10].

Realidad virtual: es una representación de las cosas a través de medios electrónicos, que nos da la sensación de estar en una situación real en la que podemos interactuar con lo que nos rodea [10].

Robot: La definición técnica adoptada por el Instituto Norteamericano de Robótica y aceptada internacionalmente es la siguiente: Ün robot es un manipulador multifuncional reprogramable, diseñado para mover materiales, piezas, herramientas o dispositivos especiales mediante movimientos programados y variables que permiten realizar diversas tareas". Suelen tener forma de brazo articulado, en cuyo extremo incorporan elementos de sujeción o herramientas. Realizan tareas repetitivas en industrias de automoción [7].

Robot manipulador: Constituye la estructura mecánica del robot. Conjunto de barras conectadas por pares cinemáticos de modo que constituyan una cadena cinemática abierta. Mecanismo compuesto generalmente de elementos en serie, articulados o deslizantes entre si, cuyo objetivo es el agarre y el desplazamiento de objetos siguiendo diversos grados de libertad. Es multifuncional y puede ser mandado directamente por un operador humano o por cualquier sistema lógico (levas, lógica pneumática, lógica eléctrica cableada o bien programado)[1].

Sensor : Dispositivo que convierte un parámetro físico (como temperatura, presión, flujo, velocidad, posición) en una señal eléctrica. En algunos casos se le considera un sinónimo de transductor, pro un verdadero sensor contiene un sistema de acondicionamiento de la señal, de manera que es mucho más sencillo realizar una medición. Un sensor mide una característica del ambiente o espacio en el que está y proporciona señales eléctricas. Estos dispositivos tratan de emular los sentidos humanos, es decir el olfato, la visión, el tacto, etc. Pero estas máquinas tienen la ventaja de poder detectar información acerca de los campos magnéticos u onda ultrasónicas [10].

A.3. Apendice: Código de Visual C++ del Runge Kutta de 4 orden

/*Runge-Kutta para un sistema de ecuaciones rk42() esté metodo usa un intervalo de paso igual a h, las condiciones iniciales son x0,y0,z0. Las funciones a evaluar son: f(x,y,z) , g(x,y,z) y e(x,y,z)

Programadores: Camargo Montaño Alejandro Garcia Meneses Maria del Carmen Ortiz Robles Ibeth */
/*Librerias necesarias para la ejecución del programa, en las cuales de incluye la de matemáticas para funciones de orden superior como lo son las tangentes hiperbólicas*/
#include < *iostream.h* >
#include < *iomanip.h* >
#include < *conio.h* >
#include < *math.h* >
#include < *stdlib.h* >
#include < *dos.h* >

//Funciones a evaluar para el cálculo del método de integración
//bajo la forma f(x,y1,y2,y3), para lo cual se evalua una formula para cada una de las ecuaciones de forma independiente
double f(float x,double y1,double y2,double y3)
return y2;
}

double g(float x,double y1,double y2,double y3)
return -2*y2-y1+1;
}

double e(float x,double y1,double y2,double y3)
return -2*y2-y1+y3+1;
}
/* El método de integración recibe como parámetros el tiempo inicial (x0), ángulos iniciales (y10,y20,y30), el tiempo final (xf) y el paso de integración para el cálculo de las pendientes que conforman al método (h).*/ void rk42(float x0, float y10,float y20,float y30,float xf,float h)

double x=x0, y1=y10,y2=y20,y3=y30;
// Declaración de las variables que sirven para el calculo de las pendientes. double k1,k2,k3,k4;
double m1,m2,m3,m4;
double p1,p2,p3,p4;

gotoxy(20,2);
cout<<RUNGE-KUTTA-4 PARA UN SISTEMA DE ECUACIONES "<<endl;
gotoxy(7,4);
cout<<"x y1 y2 y3"<<endl;
gotoxy(5,5);
cout<<————————————————————";
cout<<endl;

while(x¡=xf)
cout<<setw(10)<<setprecision(5)<<setiosflags(ios::fixed)<<x;
cout<<setw(10)<<setprecision(5)<<setiosflags(ios::fixed)<<y1;
cout<<setw(10)<<setprecision(5)<<setiosflags(ios::fixed)<<y2;
cout<<setw(10)<<setprecision(5)<<setiosflags(ios::fixed)<<y3<<endl;

```
    k1=h*f(x,y1,y2,y3);
m1=h*g(x,y1,y2,y3);
p1=h*e(x,y1,y2,y3);

    k2=h*f(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
m2=h*g(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
p2=h*e(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);

    k3=h*f(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
m3=h*g(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
p3=h*e(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);

    k4=h*f(x+h,y1+k3,y2+m3,y3+p3);
m4=h*g(x+h,y1+k3,y2+m3,y3+p3);
p4=h*e(x+h,y1+k3,y2+m3,y3+p3);

    y1+=(k1+2*k2+2*k3+k4)/6.0;
y2+=(m1+2*m2+2*m3+m4)/6.0;
y3+=(p1+2*p2+2*p3+p4)/6.0;
x+=h;
}
cout<<setw(10)<<setprecision(5)<<setiosflags(ios::fixed)<<x;
cout<<setw(10)<<setprecision(5)<<setiosflags(ios::fixed)<<y1;
cout<<setw(10)<<setprecision(5)<<setiosflags(ios::fixed)<<y2;
cout<<setw(10)<<setprecision(5)<<setiosflags(ios::fixed)<<y3<<endl;
}

    int Menu(char *titulo,char *opciones[])

int op,i;
clrscr();
i=0;
gotoxy(34,4);
cout<<titulo<<endl; cout<<endl;
for(;*opciones[i];i++)
cout<<" t t t t "<< (i+1) << ".- "<< opciones[i] << endl;
cout<<" t t t t "<< (i+1) << ".- Salir "<< endl;
cout<<endl;
cout<<" t t t t Opcion-¿";
cin>>op;
return op;
}

    void Lee(float ,float ,float ,float ,float);

    int main(void)


    int op;
enumRK42=1,SALIR;
char *Opciones[]=
Runge-Kutta Para un Sistema de Ecuaciones",
```

```
};
do
op=Menu("ECUACIONES DIFERENCIALES",Opciones);
switch(op)

   case RK42:
clrscr();

   rk42(0, 0, 0, 0, 1, 0.1);
getch();
break;

   case SALIR: gotoxy(34,20);

   gotoxy(34,21);
cout<<"Espere un momento...";
exit(0);
break;
default:gotoxy(34,15);
cout<<.°pcion no permitida";
getch();
break;
}
}while(op!=SALIR);
return 0;
}
```

A.4. . Código de la dinámica del robot antropomórfico en lazo abierto con Runge Kutta de 4o orden

```
#include < stdio.h >
#include < iostream.h >
#include < iomanip.h >
#include < conio.h >
#include < math.h >
#include < stdlib.h >
#include < dos.h >
#define p 4
#define pi 3.1415926536
//declaración de la función para la conversión de grados a radianes. #define rad(a) (pi/180)*(a)

FILE *FL; //puntero de la estructura FILE

double x0=0.0, xf=1.0, h=0.01;
// Condiciones iniciales
double q1=0, q2=-(rad(90)), q3=0, qp1=0, qp2=0, qp3=0;
//masas de los eslabones
double m1=0.45;
double m2=0.25;
double m3=0.15;
// longitud de los eslabones
double l1=0.35;
double l2=0.35;
double l3=0.2;
//longitud al centro de masas
double lcg1=0.5*l1;
double lcg2=0.5*l2;
double lcg3=0.5*l3;
//Coeficiente de gravitación double gr=9.81;
//Fricción seca double f1=0.0001;
double f2=0.0005;
double f3=0.00002;
double beta=10;

//Fricción viscosa
double b1=0.0003;
double b2=0.001199;
double b3=0.00472;

// Matriz de inercia
double y[]=0,q1,qp1,q2,qp2,q3,qp3,

H11=(m2+m3)*pow(l2,2)*pow((cos(y[3])),2)+2*m3*l2*l3*cos(y[3])*cos(y[3]+y[5])+
m3*pow(l3,2)*pow((cos(y[3]+y[5])),2),
H12=0,
H13=0,
H21=0,
H22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(y[5]),
H23=m3*pow(l3,2)+m3*l2*l3*cos(y[5]),
```

```
H31=0,
H32=m3*pow(l3,2)+m3*l2*l3*cos(y[5]), H33=m3*pow(l3,2);

   //Vector de fuerzas de coriolis y de fuerzas centripetas
double et_1=sin(y[3])*cos(y[3]),
et_2=sin(y[3]+y[5])*cos(y[3])+sin(y[3])*cos(y[3]+y[5]),
et_3=sin(y[3]+y[5])*cos(y[3]+y[5]), et_4=sin(y[3]+y[5])*cos(y[3]),

   C11=0,
C12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*y[2],
C13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*y[2],

   C21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*y[2],
C22=-(2*m3*l2*l3*sin(y[5]))*y[6],
C23=-(m3*l2*l3*sin(y[5]))*y[6],
C31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*y[2],
C32=(m3*l2*l3*sin(y[5]))*y[4],
C33=0,

   //Matriz de gravedad
G1=0,
G2=-(m2*gr*lcg2+m3*gr*l2)*cos(y[3])-m3*gr*lcg3*cos(y[3]+y[5]),
G3=-m3*gr*lcg3*cos(y[3]+y[5]),

   //Matriz de fricción viscosa y seca
Fr1=f1*tanh(beta*y[2]),
Fr2=f2*tanh(beta*y[4]),
Fr3=f3*tanh(beta*y[6]),

   // Variables articulares y derivadas
q[]=0,y[1],y[3],y[5],
qp[]=0,y[2],y[4],y[6],

   // Matrices del robot
M[4][4]=0,0,0,0,0,H11,H12,H13,0,H21,H22,H23,0,H31,H32,H33,
C[4][4]=0,0,0,0,0,C11,C12,C13,0,C21,C22,C23,0,C31,C32,C33,
G[4]=0,G1,G2,G3,

   // Ecuación de control
T[4]=0,0,0,0;

   void actualiza(double,double,double,double,double,double);
double f(double,double,double,double);
double e(double,double,double,double);
double g(double,double,double,double);
void rk42(double,double,double,double,double,double);

   int main(void)
//Funciones a evaluar para el cálculo de la dinámica
//f(x,y1,y2,y3)
double f(double x,double y01,double y02,double y03)
double
```

```
ty[]=0,y01,qp[1],y02,qp[2],y03,qp[3]},

   tH11=(m2+m3)*pow(l2,2)*pow((cos(ty[3])),2)+2*m3*l2*l3*cos(ty[3])*cos(ty[3]+ty[5])+
m3*pow(l3,2)*pow((cos(ty[3]+ty[5])),2),
tH12=0,
tH13=0,
tH21=0,
tH22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(ty[5]),
tH23=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]),
tH31=0,
tH32=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]),
tH33=m3*pow(l3,2);

   //Vector de Fuerzas de Coriolis ty de Fuerzas Centripetas
double tet_1=sin(ty[3])*cos(ty[3]),
tet_2=sin(ty[3]+ty[5])*cos(ty[3])+sin(ty[3])*cos(ty[3]+ty[5]),
tet_3=sin(ty[3]+ty[5])*cos(ty[3]+ty[5]),
tet_4=sin(ty[3]+ty[5])*cos(ty[3]),

   tC11=0,
tC12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*ty[2],
tC13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*ty[2],

   tC21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*ty[2],
tC22=-(2*m3*l2*l3*sin(ty[5]))*ty[6],
tC23=-(m3*l2*l3*sin(ty[5]))*ty[6],

   tC31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*ty[2],
tC32=(m3*l2*l3*sin(ty[5]))*ty[4], tC33=0,

   //Matriz de Gravedad
tG1=0,
tG2=-(m2*gr*lcg2+m3*gr*l2)*cos(ty[3])-m3*gr*lcg3*cos(ty[3]+ty[5]),
tG3=-m3*gr*lcg3*cos(ty[3]+ty[5]),

   //Matriz de Fricción Viscosa ty Seca
tFr1=f1*tanh(beta*ty[2]),
tFr2=f2*tanh(beta*ty[4]),
tFr3=f3*tanh(beta*ty[6]),

   // Variables Articulares ty Derivadas
tq[]=0,ty[1],ty[3],ty[5]},
tqp[]=0,ty[2],ty[4],ty[6]},

   // Matrices del robot
tM[4][4]=0,0,0,0,0,tH11,tH12,tH13,0,tH21,tH22,tH23,0,tH31,tH32,tH33},
tC[4][4]=0,0,0,0,0,tC11,tC12,tC13,0,tC21,tC22,tC23,0,tC31,tC32,tC33},
tG[4]=0,tG1,tG2,tG3};
// Ecuación de control
return 1/tM[1][1]*(T[1]-tC[1][2]*qp[2]-tC[1][3]*qp[3]-b1*qp[1]-tFr1);
}
```

```
   //g//f(x,y1,y2,y3)
double g(double x,double y01,double y02,double y03)
double
ty[]=0,y01,qp[1],y02,qp[2],y03,qp[3]},

   tH11=(m2+m3)*pow(l2,2)*pow((cos(ty[3])),2)+2*m3*l2*l3*cos(ty[3])*cos(ty[3]+ty[5])+
m3*pow(l3,2)*pow((cos(ty[3]+ty[5])),2),
tH12=0,
tH13=0,
tH21=0,
tH22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(ty[5]),
tH23=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]),
tH31=0,
tH32=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]),
tH33=m3*pow(l3,2);

   //Vector de Fuerzas de Coriolis ty de Fuerzas Centripetas
double tet_1=sin(ty[3])*cos(ty[3]),
tet_2=sin(ty[3]+ty[5])*cos(ty[3])+sin(ty[3])*cos(ty[3]+ty[5]),
tet_3=sin(ty[3]+ty[5])*cos(ty[3]+ty[5]),
tet_4=sin(ty[3]+ty[5])*cos(ty[3]),

   tC11=0,
tC12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*ty[2],
tC13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*ty[2],

   tC21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*ty[2],
tC22=-(2*m3*l2*l3*sin(ty[5]))*ty[6],
tC23=-(m3*l2*l3*sin(ty[5]))*ty[6],

   tC31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*ty[2],
tC32=(m3*l2*l3*sin(ty[5]))*ty[4],
tC33=0,

   //Matriz de Gravedad
tG1=0,
tG2=-(m2*gr*lcg2+m3*gr*l2)*cos(ty[3])-m3*gr*lcg3*cos(ty[3]+ty[5]),
tG3=-m3*gr*lcg3*cos(ty[3]+ty[5]),

   //Matriz de Fricción Viscosa ty Seca
tFr1=f1*tanh(beta*ty[2]),
tFr2=f2*tanh(beta*ty[4]),
tFr3=f3*tanh(beta*ty[6]),

   // Variables Articulares ty Derivadas
tq[]=0,ty[1],ty[3],ty[5]},
tqp[]=0,ty[2],ty[4],ty[6]},

   // Matrices del robot
tM[4][4]=0,0,0,0,0,tH11,tH12,tH13,0,tH21,tH22,tH23,0,tH31,tH32,tH33},
tC[4][4]=0,0,0,0,0,tC11,tC12,tC13,0,tC21,tC22,tC23,0,tC31,tC32,tC33},
tG[4]=0,tG1,tG2,tG3};
```

```
   return 1/tM[2][2]*(T[2]-tM[2][3]*e(x,y01,y02,y03)-tC[2][1]*qp[1]-tC[2][2]*qp[2]-
tC[2][3]*qp[3]-tG[2]-b2*qp[2]-tFr2);
}
//e//f(x,y1,y2,y3)
double e(double x,double y01,double y02,double y03)
double
ty[]=0,y01,qp[1],y02,qp[2],y03,qp[3]},

   tH11=(m2+m3)*pow(l2,2)*pow((cos(ty[3])),2)+2*m3*l2*l3*cos(ty[3])*cos(ty[3]+ty[5])+
m3*pow(l3,2)*pow((cos(ty[3]+ty[5])),2),
tH12=0,
tH13=0,
tH21=0,
tH22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(ty[5]),
tH23=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]),
tH31=0,
tH32=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]),
tH33=m3*pow(l3,2);

   //Vector de Fuerzas de Coriolis ty de Fuerzas Centripetas
double tet_1=sin(ty[3])*cos(ty[3]),
tet_2=sin(ty[3]+ty[5])*cos(ty[3])+sin(ty[3])*cos(ty[3]+ty[5]),
tet_3=sin(ty[3]+ty[5])*cos(ty[3]+ty[5]),
tet_4=sin(ty[3]+ty[5])*cos(ty[3]),

   tC11=0,
tC12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*ty[2],
tC13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*ty[2],

   tC21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*ty[2],
tC22=-(2*m3*l2*l3*sin(ty[5]))*ty[6],
tC23=-(m3*l2*l3*sin(ty[5]))*ty[6],

   tC31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*ty[2],
tC32=(m3*l2*l3*sin(ty[5]))*ty[4],
tC33=0,

   //Matriz de Gravedad
tG1=0,
tG2=-(m2*gr*lcg2+m3*gr*l2)*cos(ty[3])-m3*gr*lcg3*cos(ty[3]+ty[5]),
tG3=-m3*gr*lcg3*cos(ty[3]+ty[5]),

   //Matriz de Fricción Viscosa ty Seca
tFr1=f1*tanh(beta*ty[2]),
tFr2=f2*tanh(beta*ty[4]),
tFr3=f3*tanh(beta*ty[6]),

   // Variables Articulares ty Derivadas
tq[]=0,ty[1],ty[3],ty[5]},
tqp[]=0,ty[2],ty[4],ty[6]},
```

```
   // Matrices del robot
tM[4][4]=0,0,0,0,0,tH11,tH12,tH13,0,tH21,tH22,tH23,0,tH31,tH32,tH33},
tC[4][4]=0,0,0,0,0,tC11,tC12,tC13,0,tC21,tC22,tC23,0,tC31,tC32,tC33},
tG[4]=0,tG1,tG2,tG3};

   return (tM[3][2]*(T[2]-tC[2][1]*qp[1]-tC[2][2]*qp[2]-tC[2][3]*qp[3]-tG[2]-b2*qp[2]-tFr2)+
tM[2][2]*(tC[3][1]*qp[1]+tC[3][2]*qp[2]+tG[3]+b3*qp[3]+tFr3-T[3]))/(tM[3][2]*tM[2][3]-
tM[2][2]*tM[3][3]);
}
//Utilización del método de integración para la obtención de los valores con el uso de la dinámica. void rk42(double
x0, double y10,double y20,double y30,double xf,double h)


   double x=x0, y1=y10,y2=y20,y3=y30;
double k1,k2,k3,k4;
double m1,m2,m3,m4;
double p1,p2,p3,p4;
/*Preparación del archivo para grabar los datos que posteriormente se han de utilizar en la representación del
modelo dinámico en el ambiente virtual.*/ if ((FL= fopen (”Datos.mat”, ”w”))==NULL)}
fprintf(FL,”
   double
y01=y1,
y02=y2,
y03=y3,
qp01=qp[1],
qp02=qp[2],
qp03=qp[3];

   for (int cont=1; x¡=xf; cont++)
y01=y1,
y02=y2,
y03=y3,
qp01=qp[1],
qp02=qp[2],
qp03=qp[3];

   actualiza(y1,qp01, y2, qp02, y3, qp03);

   //se calcula teta punto porque en las funciones f,g,e, teta_pp=1/M[...]

   k1=h*f(x,y1,y2,y3);
m1=h*g(x,y1,y2,y3);
p1=h*e(x,y1,y2,y3);

   k2=h*f(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
m2=h*g(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
p2=h*e(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);

   k3=h*f(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
m3=h*g(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
p3=h*e(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
```

```
    k4=h*f(x+h,y1+k3,y2+m3,y3+p3);
m4=h*g(x+h,y1+k3,y2+m3,y3+p3);
p4=h*e(x+h,y1+k3,y2+m3,y3+p3);

    qp01+=(k1+2*k2+2*k3+k4)/6.0;
qp02+=(m1+2*m2+2*m3+m4)/6.0;
qp03+=(p1+2*p2+2*p3+p4)/6.0;
//en caso de q se desen graficar los valores de la velocidas angular(teta punto)
actualiza(y1,qp01, y2, qp02, y3, qp03);
//se calcula teta porque en las funciones f,g,e, teta_punto
k1=h*f(x,y1,y2,y3);
m1=h*g(x,y1,y2,y3);
p1=h*e(x,y1,y2,y3);

    k2=h*f(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
m2=h*g(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
p2=h*e(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);

    k3=h*f(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
m3=h*g(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
p3=h*e(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);

    k4=h*f(x+h,y1+k3,y2+m3,y3+p3);
m4=h*g(x+h,y1+k3,y2+m3,y3+p3);
p4=h*e(x+h,y1+k3,y2+m3,y3+p3);

    y1+=(k1+2*k2+2*k3+k4)/6.0;
y2+=(m1+2*m2+2*m3+m4)/6.0;
y3+=(p1+2*p2+2*p3+p4)/6.0;

    //resultados de teta enviados al archivo para ser graficados en matlab
x+=h;
actualiza(y1,qp01, y2, qp02, y3, qp03);
}
fclose(FL);
cout<<setw(12)<<setprecision(p)<<setiosflags(ios::fixed)<<x;
cout<<setw(12)<<setprecision(p)<<setiosflags(ios::fixed)<<y1;
cout<<setw(12)<<setprecision(p)<<setiosflags(ios::fixed)<<y2;
cout<<setw(12)<<setprecision(p)<<setiosflags(ios::fixed)<<y3<<endl;
}
}

    void actualiza(double q01,double qp01,double q02,double qp02,double q03,double qp03)
y[0]=0; y[1]=q01; y[2]=qp01; y[3]=q02; y[4]=qp02; y[5]=q03;
y[6]=qp03;
// Variables Articulares y Derivadas

    q[0]=0;
q[1]=y[1];
q[2]=y[3];
q[3]=y[5];
```

```
    qp[0]=0;
qp[1]=y[2];
qp[2]=y[4];
qp[3]=y[6];

    H11=(m2+m3)*pow(l2,2)*pow((cos(y[3])),2)+2*m3*l2*l3*cos(y[3])*cos(y[3]+y[5])+
m3*pow(l3,2)*pow((cos(y[3]+y[5])),2);
H22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(y[5]);
H23=m3*pow(l3,2)+m3*l2*l3*cos(y[5]);
H32=m3*pow(l3,2)+m3*l2*l3*cos(y[5]);
H33=m3*pow(l3,2);

    //Vector de Fuerzas de Coriolis y de Fuerzas Centripetas

    et_1=sin(y[3])*cos(y[3]);
et_2=sin(y[3]+y[5])*cos(y[3])+sin(y[3])*cos(y[3]+y[5]);
et_3=sin(y[3]+y[5])*cos(y[3]+y[5]); et_4=sin(y[3]+y[5])*cos(y[3]);

    C12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*y[2];
C13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*y[2];

    C21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*y[2];
C22=-(2*m3*l2*l3*sin(y[5]))*y[6];
C23=-(m3*l2*l3*sin(y[5]))*y[6];

    C31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*y[2];
C32=(m3*l2*l3*sin(y[5]))*y[4];

    //Matriz de Gravedad

    G2=-(m2*gr*lcg2+m3*gr*l2)*cos(y[3])-m3*gr*lcg3*cos(y[3]+y[5]);
G3=-m3*gr*lcg3*cos(y[3]+y[5]);

    // Matrices del robot
M[0][0]=0;
M[0][1]=0;
M[0][2]=0;
M[1][0]=0;
M[2][0]=0;
M[3][0]=0;

    M[1][1]=H11;
M[1][2]=H12;
M[1][3]=H13;
M[2][1]=H21;
M[2][2]=H22;
M[2][3]=H23;
M[3][1]=H31;
M[3][2]=H32;
M[3][3]=H33;

    C[0][0]=0;
```

```
C[0][1]=0;
C[0][2]=0;
C[1][0]=0;
C[2][0]=0;
C[3][0]=0;

    C[1][1]=C11;
C[1][2]=C12;
C[1][3]=C13;
C[2][1]=C21;
C[2][2]=C22;
C[2][3]=C23;
C[3][1]=C31;
C[3][2]=C32;
C[3][3]=C33;

    G[0]=0;
G[1]=G1;
G[2]=G2;
G[3]=G3;
}
```

A.5. . Código de la dinámica del robot antropomórfico en lazo cerrado con Runge Kutta de 4o orden (PD)

```
   #include < stdio.h >
#include < iostream.h >
#include < iomanip.h >//archivo
#include < conio.h >
#include < math.h >
#include < stdlib.h >
#include < dos.h >
#define p 6
#define pi 3.1415926536
#define rad(a) (pi/180)*(a)

FILE *FL; //puntero de la estructura FILE
// Condiciones iniciales
double x0=0.0, xf=10.0, h=0.01; double
//tetas de referencia
//qr[4]=0,rad(45),rad(30),rad(20),
qr[4]=0,0.8,0.8,0.8,
//errores
er[4]=0,0,0,0, de[4]=0,0,0,0,

//constantes derivativa, porcentual
//kd=2, //kp=1; kp=.7, kd=(2* sqrt(kp));

//variables articulares
double q1=rad(45), q2=-rad(45), q3=-rad(45), qp1=rad(0),
qp2=rad(0), qp3=rad(0);

//masas de los eslabones
double m1=0.45; double m2=0.25; double m3=0.15;

// longitud de los eslabones
double l1=0.35; double l2=0.35; double l3=0.2;

//longitud al centro de masas
double lcg1=0.5*l1; double lcg2=0.5*l2; double lcg3=0.5*l3;

//Coeficiente de gravitación
double gr=9.81;

// Friccion Seca o Coulomb
double f1=0.001; double f2=0.0005; double f3=0.0001; double
beta=10;

//Fricción Viscosa
double b1=0.3; double b2=0.15; double b3=0.05;

// Matriz de Inercia
double y[]=0,q1,qp1,q2,qp2,q3,qp3,
```

```
H11=(m2+m3)*pow(l2,2)*pow((cos(y[3])),2)+2*m3*l2*l3*cos(y[3])*cos(y[3]+y[5])+
m3*pow(l3,2)*pow((cos(y[3]+y[5])),2),
H12=0, H13=0, H21=0,
H22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(y[5]),
H23=m3*pow(l3,2)+m3*l2*l3*cos(y[5]), H31=0,
H32=m3*pow(l3,2)+m3*l2*l3*cos(y[5]), H33=m3*pow(l3,2);

//Vector de Fuerzas de Coriolis y de Fuerzas Centripetas
double et_1=sin(y[3])*cos(y[3]),
et_2=sin(y[3]+y[5])*cos(y[3])+sin(y[3])*cos(y[3]+y[5]),
et_3=sin(y[3]+y[5])*cos(y[3]+y[5]),
et_4=sin(y[3]+y[5])*cos(y[3]),

C11=0,
C12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*y[2],
C13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*y[2],

C21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*y[2],
C22=-(2*m3*l2*l3*sin(y[5]))*y[6], C23=-(m3*l2*l3*sin(y[5]))*y[6],

C31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*y[2],
C32=(m3*l2*l3*sin(y[5]))*y[4], C33=0,

//Matriz de Gravedad
G1=0,
G2=-(m2*gr*lcg2+m3*gr*l2)*cos(y[3])-m3*gr*lcg3*cos(y[3]+y[5]),
G3=-m3*gr*lcg3*cos(y[3]+y[5]),

//Matriz de Fricción Viscosa y Seca
Fr1=f1*tanh(beta*y[2]), Fr2=f2*tanh(beta*y[4]),
Fr3=f3*tanh(beta*y[6]),

// Variables Articulares y Derivadas
q[]=0,y[1],y[3],y[5], qp[]=0,y[2],y[4],y[6],

// Matrices del robot
M[4][4]=0,0,0,0,0,H11,H12,H13,0,H21,H22,H23,0,H31,H32,H33,
C[4][4]=0,0,0,0,0,C11,C12,C13,0,C21,C22,C23,0,C31,C32,C33,
G[4]=0,G1,G2,G3,

// Ecuación de control
T[4]=0,(kp*er[1]),(kp*er[2]),(kp*er[3]);

// Ecuaciones de Estado
void actualiza(double,double,double,double,double,double);
double f(double,double,double,double);
double e(double,double,double,double);
double g(double,double,double,double);
void rk42(double,double,double,double,double,double);

int main(void)
```

```
void rk42(double x, double y1,double y2,double y3,double xf,double h)
double ea[4]=0,0,0,0,en[4]=0,0,0,0;
double k1,k2,k3,k4;
double m1,m2,m3,m4;
double p1,p2,p3,p4;

if ((FL= fopen ("Datos.mat", "w"))==NULL)
fprintf(FL,"qp[1],qp[2],qp[3],en[1],en[2],en[3], T[1],T[2],T[3]);//graba datos en archivo

//fprintf(FL,//calculando el error inicial para proporcionar al robot el control en su movimiento
er[1]=(qr[1]-q[1]);
er[2]=(qr[2]-q[2]);
er[3]=(qr[3]-q[3]);
//calculando los errores anteriores
ea[1]=er[1];
ea[2]=er[2];
ea[3]=er[3];

for (int cont=1; x¡=xf; cont++)

y1=q[1];
y2=q[2];
y3=q[3];

//se calcula teta punto porque en las funciones f,g,e, teta_pp=1/M[...]
k1=h*f(x,y1,y2,y3);
m1=h*g(x,y1,y2,y3);
p1=h*e(x,y1,y2,y3);

k2=h*f(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
m2=h*g(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
p2=h*e(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);

k3=h*f(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
m3=h*g(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
p3=h*e(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);

k4=h*f(x+h,y1+k3,y2+m3,y3+p3);
m4=h*g(x+h,y1+k3,y2+m3,y3+p3);
p4=h*e(x+h,y1+k3,y2+m3,y3+p3);

qp[1]+=(k1+2*k2+2*k3+k4)/6.0;
qp[2]+=(m1+2*m2+2*m3+m4)/6.0;
qp[3]+=(p1+2*p2+2*p3+p4)/6.0;

//resultados de teta punto
//fprintf(FL,"
actualiza(y1,qp[1], y2, qp[2], y3, qp[3]);
//se calcula teta porque en las funciones f,g,e, teta_p=...
k1=h*f(x,y1,y2,y3);
m1=h*g(x,y1,y2,y3);
p1=h*e(x,y1,y2,y3);
```

```
k2=h*f(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
m2=h*g(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
p2=h*e(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);

k3=h*f(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
m3=h*g(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
p3=h*e(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);

k4=h*f(x+h,y1+k3,y2+m3,y3+p3);
m4=h*g(x+h,y1+k3,y2+m3,y3+p3);
p4=h*e(x+h,y1+k3,y2+m3,y3+p3);

y1+=(k1+2*k2+2*k3+k4)/6.0;
y2+=(m1+2*m2+2*m3+m4)/6.0;
y3+=(p1+2*p2+2*p3+p4)/6.0;
x+=h;

actualiza(y1,qp[1], y2, qp[2], y3, qp[3]);
//——————————-verificar error——————————-//
//calculando el error nuevo en cada iteracion
en[1]=(qr[1]-q[1]);
en[2]=(qr[2]-q[2]);
en[3]=(qr[3]-q[3]);

de[1]=(ea[1]-en[1])/cont;
de[2]=(ea[2]-en[2])/cont;
de[3]=(ea[3]-en[3])/cont;
//printf(//aplicando el control a T
T[1]=(kp*en[1])+(kd*de[1]);
T[2]=(kp*en[2])+(kd*de[2]);v T[3]=(kp*en[3])+(kd*de[3]);

//printf(//actualizando valores para siguiente iteracion
ea[1]=en[1];
ea[2]=en[2];
ea[3]=en[3];
//——————————-verificar error——————————-//
//resultados de teta
fprintf(FL,"qp[1],qp[2],qp[3],en[1],en[2],en[3], T[1],T[2],T[3]);//graba datos en archivo
printf("
} fclose(FL);
}
//Funciones a evaluar
//f(x,y1,y2,y3)
double f(double x,double y01,double y02,double y03)
// Matriz de Inercia
double
ty[]=0,y01,qp[1],y02,qp[2],y03,qp[3],

tH11=(m2+m3)*pow(l2,2)*pow((cos(ty[3])),2)+2*m3*l2*l3*cos(ty[3])*cos(ty[3]+ty[5])+
m3*pow(l3,2)*pow((cos(ty[3]+ty[5])),2),
tH12=0, tH13=0, tH21=0,
```

```
tH22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(ty[5]),
tH23=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]), tH31=0,
tH32=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]), tH33=m3*pow(l3,2);

//Vector de Fuerzas de Coriolis ty de Fuerzas Centripetas
double tet_1=sin(ty[3])*cos(ty[3]),
tet_2=sin(ty[3]+ty[5])*cos(ty[3])+sin(ty[3])*cos(ty[3]+ty[5]),
tet_3=sin(ty[3]+ty[5])*cos(ty[3]+ty[5]),
tet_4=sin(ty[3]+ty[5])*cos(ty[3]),

tC11=0,
tC12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*ty[2],
tC13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*ty[2],

tC21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*ty[2],
tC22=-(2*m3*l2*l3*sin(ty[5]))*ty[6],
tC23=-(m3*l2*l3*sin(ty[5]))*ty[6],

tC31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*ty[2],
tC32=(m3*l2*l3*sin(ty[5]))*ty[4], tC33=0,

//Matriz de Gravedad
tG1=0,
tG2=-(m2*gr*lcg2+m3*gr*l2)*cos(ty[3])-m3*gr*lcg3*cos(ty[3]+ty[5]),
tG3=-m3*gr*lcg3*cos(ty[3]+ty[5]),

//Matriz de Fricción Viscosa ty Seca
tFr1=f1*tanh(beta*ty[2]),
tFr2=f2*tanh(beta*ty[4]),
tFr3=f3*tanh(beta*ty[6]),

// Variables Articulares ty Derivadas
tq[]=0,ty[1],ty[3],ty[5], tqp[]=0,ty[2],ty[4],ty[6],

// Matrices del robot
tM[4][4]=0,0,0,0,0,tH11,tH12,tH13,0,tH21,tH22,tH23,0,tH31,tH32,tH33,
tC[4][4]=0,0,0,0,0,tC11,tC12,tC13,0,tC21,tC22,tC23,0,tC31,tC32,tC33,
tG[4]=0,tG1,tG2,tG3;
// Ecuación de control
return 1/tM[1][1]*(T[1]-tC[1][2]*qp[2]-tC[1][3]*qp[3]-b1*qp[1]-tFr1);
}
double g(double x,double y01,double y02,double y03)
double
ty[]=0,y01,qp[1],y02,qp[2],y03,qp[3],

tH11=(m2+m3)*pow(l2,2)*pow((cos(ty[3])),2)+2*m3*l2*l3*cos(ty[3])*cos(ty[3]+ty[5])+
m3*pow(l3,2)*pow((cos(ty[3]+ty[5])),2),
tH12=0, tH13=0, tH21=0,
tH22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(ty[5]),
tH23=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]), tH31=0,
tH32=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]), tH33=m3*pow(l3,2);
```

```
//Vector de Fuerzas de Coriolis ty de Fuerzas Centripetas
double tet_1=sin(ty[3])*cos(ty[3]),
tet_2=sin(ty[3]+ty[5])*cos(ty[3])+sin(ty[3])*cos(ty[3]+ty[5]),
tet_3=sin(ty[3]+ty[5])*cos(ty[3]+ty[5]),

tet_4=sin(ty[3]+ty[5])*cos(ty[3]), tC11=0,
tC12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*ty[2],
tC13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*ty[2],

tC21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*ty[2],
tC22=-(2*m3*l2*l3*sin(ty[5]))*ty[6],
tC23=-(m3*l2*l3*sin(ty[5]))*ty[6],

tC31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*ty[2],
tC32=(m3*l2*l3*sin(ty[5]))*ty[4], tC33=0,

//Matriz de Gravedad
tG1=0,
tG2=-(m2*gr*lcg2+m3*gr*l2)*cos(ty[3])-m3*gr*lcg3*cos(ty[3]+ty[5]),
tG3=-m3*gr*lcg3*cos(ty[3]+ty[5]),

//Matriz de Fricción Viscosa ty Seca
tFr1=f1*tanh(beta*ty[2]), tFr2=f2*tanh(beta*ty[4]),
tFr3=f3*tanh(beta*ty[6]),

// Variables Articulares ty Derivadas
tq[]=0,ty[1],ty[3],ty[5], tqp[]=0,ty[2],ty[4],ty[6],

// Matrices del robot
tM[4][4]=0,0,0,0,0,tH11,tH12,tH13,0,tH21,tH22,tH23,0,tH31,tH32,tH33,
tC[4][4]=0,0,0,0,0,tC11,tC12,tC13,0,tC21,tC22,tC23,0,tC31,tC32,tC33,
tG[4]=0,tG1,tG2,tG3;

return 1/tM[2][2]*(T[2]-tM[2][3]*e(x,y01,y02,y03)-tC[2][1]*qp[1]-tC[2][2]*qp[2]-
tC[2][3]*qp[3]-tG[2]-b2*qp[2]-tFr2);
}
//e//f(x,y1,y2,y3)
double e(double x,double y01,double y02,double y03)
double
ty[]=0,y01,qp[1],y02,qp[2],y03,qp[3],

tH11=(m2+m3)*pow(l2,2)*pow((cos(ty[3])),2)+2*m3*l2*l3*cos(ty[3])*cos(ty[3]+
ty[5])+m3*pow(l3,2)*pow((cos(ty[3]+ty[5])),2),
tH12=0, tH13=0, tH21=0,
tH22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(ty[5]),
tH23=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]), tH31=0,
tH32=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]), tH33=m3*pow(l3,2);

//Vector de Fuerzas de Coriolis ty de Fuerzas Centripetas
double tet_1=sin(ty[3])*cos(ty[3]),
tet_2=sin(ty[3]+ty[5])*cos(ty[3])+sin(ty[3])*cos(ty[3]+ty[5]),
tet_3=sin(ty[3]+ty[5])*cos(ty[3]+ty[5]),
```

```
tet_4=sin(ty[3]+ty[5])*cos(ty[3]),

tC11=0,
tC12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*ty[2],
tC13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*ty[2],

tC21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*ty[2],
tC22=-(2*m3*l2*l3*sin(ty[5]))*ty[6],
tC23=-(m3*l2*l3*sin(ty[5]))*ty[6],

tC31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*ty[2],
tC32=(m3*l2*l3*sin(ty[5]))*ty[4], tC33=0,

//Matriz de Gravedad
tG1=0,
tG2=-(m2*gr*lcg2+m3*gr*l2)*cos(ty[3])-m3*gr*lcg3*cos(ty[3]+ty[5]),
tG3=-m3*gr*lcg3*cos(ty[3]+ty[5]),

//Matriz de Fricción Viscosa ty Seca
tFr1=f1*tanh(beta*ty[2]), tFr2=f2*tanh(beta*ty[4]),
tFr3=f3*tanh(beta*ty[6]),

// Variables Articulares ty Derivadas
tq[]=0,ty[1],ty[3],ty[5], tqp[]=0,ty[2],ty[4],ty[6],
// Matrices del robot
tM[4][4]=0,0,0,0,0,tH11,tH12,tH13,0,tH21,tH22,tH23,0,tH31,tH32,tH33,
tC[4][4]=0,0,0,0,0,tC11,tC12,tC13,0,tC21,tC22,tC23,0,tC31,tC32,tC33,
tG[4]=0,tG1,tG2,tG3;

return (tM[3][2]*(T[2]-tC[2][1]*qp[1]-tC[2][2]*qp[2]-tC[2][3]*qp[3]-tG[2]-b2*qp[2]-tFr2)+
tM[2][2]*(tC[3][1]*qp[1]+tC[3][2]*qp[2]+tG[3]+b3*qp[3]+tFr3-T[3]))/(tM[3][2]*tM[2][3]-
tM[2][2]*tM[3][3]);
}
int Menu(char *titulo,char *opciones[])
int op,i;
i=0;
cout<<titulo<<endl; cout<<endl;
for(;*opciones[i];i++)
cout<<" t t "<< (i+1) << ".- "<< opciones[i] << endl;
cout<<" t t "<< (i+1) << ".- Salir "<< endl;
cout<<endl;
cout<<" t t Opcion-¿";
cin>>op;
return op;
}
void actualiza(double q01,double qp01,double q02,double
qp02,double q03,double qp03)

y[0]=0; y[1]=q01; y[2]=qp01; y[3]=q02; y[4]=qp02; y[5]=q03;
y[6]=qp03;

// Variables Articulares y Derivadas
```

```
q[0]=0; q[1]=y[1]; q[2]=y[3]; q[3]=y[5];
qp[0]=0; qp[1]=y[2]; qp[2]=y[4]; qp[3]=y[6];

H11=(m2+m3)*pow(l2,2)*pow((cos(y[3])),2)+2*m3*l2*l3*cos(y[3])*cos(y[3]+y[5])+
m3*pow(l3,2)*pow((cos(y[3]+y[5])),2);
H22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(y[5]);
H23=m3*pow(l3,2)+m3*l2*l3*cos(y[5]);
H32=m3*pow(l3,2)+m3*l2*l3*cos(y[5]);
H33=m3*pow(l3,2);

//Vector de Fuerzas de Coriolis y de Fuerzas Centripetas

et_1=sin(y[3])*cos(y[3]);
et_2=sin(y[3]+y[5])*cos(y[3])+sin(y[3])*cos(y[3]+y[5]);
et_3=sin(y[3]+y[5])*cos(y[3]+y[5]);
et_4=sin(y[3]+y[5])*cos(y[3]);

C12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*y[2];
C13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*y[2];

C21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*y[2];
C22=-(2*m3*l2*l3*sin(y[5]))*y[6]; C23=-(m3*l2*l3*sin(y[5]))*y[6];

C31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*y[2];
C32=(m3*l2*l3*sin(y[5]))*y[4];

//Matriz de Gravedad

G2=-(m2*gr*lcg2+m3*gr*l2)*cos(y[3])-m3*gr*lcg3*cos(y[3]+y[5]);
G3=-m3*gr*lcg3*cos(y[3]+y[5]);

// Matrices del robot
M[0][0]=0; M[0][1]=0; M[0][2]=0; M[1][0]=0; M[2][0]=0; M[3][0]=0;

M[1][1]=H11; M[1][2]=H12; M[1][3]=H13; M[2][1]=H21; M[2][2]=H22;
M[2][3]=H23; M[3][1]=H31; M[3][2]=H32; M[3][3]=H33;

C[0][0]=0; C[0][1]=0; C[0][2]=0; C[1][0]=0; C[2][0]=0; C[3][0]=0;

C[1][1]=C11; C[1][2]=C12; C[1][3]=C13; C[2][1]=C21; C[2][2]=C22;
C[2][3]=C23; C[3][1]=C31; C[3][2]=C32; C[3][3]=C33; G[0]=0;
G[1]=G1; G[2]=G2; G[3]=G3;

}
```

A.6. . Código de la dinámica del robot antropomórfico en lazo cerrado con Runge Kutta de 4o orden (PID)

```
   #include < stdio.h >
#include < iostream.h >
#include < iomanip.h >//archivo
#include < conio.h >
#include < math.h >
#include < stdlib.h >
#include < dos.h >
#define p 6
#define pi 3.1415926536
#define rad(a) (pi/180)*(a)
FILE *FL;
//puntero de la estructura FILE
// Condiciones iniciales
double x0=0.0, xf=30.0, h=0.01;
double
//tetas de referencia
qr[4]=0,0.8,0.8,0.8,
//errores
er[4]=0,0,0,0, de[4]=0,0,0,0,
//constantes derivativa, porcentual

   kp1=1, kd1=(2* sqrt(kp1)), ki1=(2*pow(kp1,1.5))/(kp1*100),

   kp2=(kp1/2.5),//.25, kd2=(2* sqrt(kp2)),
ki2=(2*pow(kp2,1.5))/(kp2*100),

   kp3=(kp2/5),//.125, kd3=(2* sqrt(kp3)),
ki3=(2*pow(kp3,1.5))/(kp3*100);

   //variables articulares
double q1=rad(45), q2=-rad(45), q3=-rad(45), qp1=rad(0),
qp2=rad(0), qp3=rad(0);

   //masas de los eslabones
double m1=0.45; double m2=0.25; double m3=0.15;

   // longitud de los eslabones
double l1=0.35; double l2=0.35; double l3=0.2;

   //longitud al centro de masas
double lcg1=0.5*l1; double lcg2=0.5*l2; double lcg3=0.5*l3;

   //Coeficiente de gravitación
double gr=9.81;

   double f1=0.1; double f2=0.5; double f3=0.00001; double beta=10;

   //Fricción Viscosa
double b1=0.3; double b2=0.15; double b3=0.00005;
```

```
   // Matriz de Inercia
double y[]=0,q1,qp1,q2,qp2,q3,qp3,

   H11=(m2+m3)*pow(l2,2)*pow((cos(y[3])),2)+2*m3*l2*l3*cos(y[3])*cos(y[3]+y[5])+
m3*pow(l3,2)*pow((cos(y[3]+y[5])),2),
H12=0, H13=0, H21=0,
H22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(y[5]),
H23=m3*pow(l3,2)+m3*l2*l3*cos(y[5]), H31=0,
H32=m3*pow(l3,2)+m3*l2*l3*cos(y[5]), H33=m3*pow(l3,2);

   //Vector de Fuerzas de Coriolis y de Fuerzas Centripetas
double et_1=sin(y[3])*cos(y[3]),
et_2=sin(y[3]+y[5])*cos(y[3])+sin(y[3])*cos(y[3]+y[5]),
et_3=sin(y[3]+y[5])*cos(y[3]+y[5]), et_4=sin(y[3]+y[5])*cos(y[3]),

   C11=0,
C12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*y[2],
C13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*y[2],

   C21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*y[2],
C22=-(2*m3*l2*l3*sin(y[5]))*y[6], C23=-(m3*l2*l3*sin(y[5]))*y[6],

   C31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*y[2],
C32=(m3*l2*l3*sin(y[5]))*y[4], C33=0,

   //Matriz de Gravedad
G1=0,
G2=-(m2*gr*lcg2+m3*gr*l2)*cos(y[3])-m3*gr*lcg3*cos(y[3]+y[5]),
G3=-m3*gr*lcg3*cos(y[3]+y[5]),

   //Matriz de Fricción Viscosa y Seca
Fr1=f1*tanh(beta*y[2]), Fr2=f2*tanh(beta*y[4]),
Fr3=f3*tanh(beta*y[6]),

   // Variables Articulares y Derivadas
q[]=0,y[1],y[3],y[5], qp[]=0,y[2],y[4],y[6],

   // Matrices del robot
M[4][4]=0,0,0,0,0,H11,H12,H13,0,H21,H22,H23,0,H31,H32,H33,
C[4][4]=0,0,0,0,0,C11,C12,C13,0,C21,C22,C23,0,C31,C32,C33,
G[4]=0,G1,G2,G3,

   // Ecuación de control
T[4]=0,(kp1*er[1])+(kd1*de[1]),(kp2*er[2])+(kd2*de[2]),(kp3*er[3])+(kd3*de[3]);
// Ecuaciones de Estado

   void actualiza(double,double,double,double,double,double);
double f(double,double,double,double);
double e(double,double,double,double);
double g(double,double,double,double);
double ce1(double,double,double,double);
```

```
double ce2(double,double,double,double);
double ce3(double,double,double,double);

   void rk42(double,double,double,double,double,double);

   int main(void)


   void rk42(double x, double y1,double y2,double y3,double xf,double h)


   double ea[4]=0,0,0,0,en[4]=0,0,0,0;
double k1,k2,k3,k4;
double m1,m2,m3,m4;
double p1,p2,p3,p4,pd1=0,pd2=0,pd3=0;

   if ((FL= fopen ("Datos.mat", "w"))==NULL)
fprintf(FL,"x,y1,y2,y3,qp[1],qp[2],qp[3],en[1],en[2],en[3], T[1],T[2],T[3]);
//graba datos en archivo

   //fprintf(FL,//calculando el error inicial
er[1]=(qr[1]-q[1]);
er[2]=(qr[2]-q[2]);
er[3]=(qr[3]-q[3]);
//calculando los errores anteriores
ea[1]=er[1];
ea[2]=er[2];
ea[3]=er[3];

   for (int cont=1; x¡=xf; cont++)


   y1=q[1];
y2=q[2];
y3=q[3];

   //se calcula teta punto porque en las funciones f,g,e, teta_pp=1/M[...]

   k1=h*f(x,y1,y2,y3);
m1=h*g(x,y1,y2,y3);
p1=h*e(x,y1,y2,y3);

k2=h*f(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
m2=h*g(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
p2=h*e(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
k3=h*f(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
m3=h*g(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
p3=h*e(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
k4=h*f(x+h,y1+k3,y2+m3,y3+p3);
m4=h*g(x+h,y1+k3,y2+m3,y3+p3);
p4=h*e(x+h,y1+k3,y2+m3,y3+p3);
```

```
qp[1]+=(k1+2*k2+2*k3+k4)/6.0;
qp[2]+=(m1+2*m2+2*m3+m4)/6.0;
qp[3]+=(p1+2*p2+2*p3+p4)/6.0;

//resultados de teta punto
//fprintf(FL,"
actualiza(y1,qp[1], y2, qp[2], y3, qp[3]);
//se calcula teta porque en las funciones f,g,e, teta_p=...
k1=h*f(x,y1,y2,y3);
m1=h*g(x,y1,y2,y3);
p1=h*e(x,y1,y2,y3);

k2=h*f(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
m2=h*g(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
p2=h*e(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);

k3=h*f(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
m3=h*g(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
p3=h*e(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);

k4=h*f(x+h,y1+k3,y2+m3,y3+p3);
m4=h*g(x+h,y1+k3,y2+m3,y3+p3);
p4=h*e(x+h,y1+k3,y2+m3,y3+p3);

y1+=(k1+2*k2+2*k3+k4)/6.0;
y2+=(m1+2*m2+2*m3+m4)/6.0;
y3+=(p1+2*p2+2*p3+p4)/6.0;

//————————————-verificar error————————————-//
//calculando el error nuevo en cada iteracion
en[1]=(qr[1]-q[1]);
en[2]=(qr[2]-q[2]);
en[3]=(qr[3]-q[3]);

de[1]=(ea[1]-en[1])/cont;
de[2]=(ea[2]-en[2])/cont;
de[3]=(ea[3]-en[3])/cont;

pd1=kp1*en[1];
pd2=kp2*en[2];
pd3=kp3*en[3];

actualiza(y1,qp[1], y2, qp[2], y3, qp[3]);

//se calula el error para el segundo control Ki

k1=h*ce1(x,y1,y2,y3);
m1=h*ce2(x,y1,y2,y3);
p1=h*ce3(x,y1,y2,y3);

k2=h*ce1(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
m2=h*ce2(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
```

```
p2=h*ce3(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);

k3=h*ce1(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
m3=h*ce2(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
p3=h*ce3(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);

k4=h*ce1(x+h,y1+k3,y2+m3,y3+p3);
m4=h*ce2(x+h,y1+k3,y2+m3,y3+p3);
p4=h*ce3(x+h,y1+k3,y2+m3,y3+p3);

en[1]+=(k1+2*k2+2*k3+k4)/6.0;
en[2]+=(m1+2*m2+2*m3+m4)/6.0;
en[3]+=(p1+2*p2+2*p3+p4)/6.0;

x+=h;

//aplicando el control a T

T[1]=(pd1)+ki1*en[1]+(kd1*de[1]);
T[2]=(pd2)+ki2*en[2]+(kd2*de[2]);
T[3]=(pd3)+ki3*en[3]+(kd3*de[3]);

//actualizando valores para siguiente iteracion
ea[1]=en[1];
ea[2]=en[2];
ea[3]=en[3];
//——————————-verificar error——————————-//
//resultados de teta
fprintf(FL,"x,y1,y2,y3,qp[1],qp[2],qp[3],en[1],en[2],en[3], T[1],T[2],T[3]);//graba datos en archivo
printf(//graba datos en archivo

}//fin de for
fclose(FL);
}

//ei: error a integrar
//ce: calculo del error a integrar
double ce1(double x,double ei1, double ei2, double ei3)

return (qr[1]-ei1);
}

double ce2(double x,double ei1, double ei2, double ei3)

return (qr[2]-ei2);
}

double ce3(double x,double ei1, double ei2, double ei3)

return (qr[3]-ei3);
}
//Funciones a evaluar
```

```
//f(x,y1,y2,y3)
double f(double x,double y01,double y02,double y03)
// actualiza(y1,qp[1], y2, qp[2], y3, qp[3]);
// Matriz de Inercia
double
ty[]=0,y01,qp[1],y02,qp[2],y03,qp[3],

tH11=(m2+m3)*pow(l2,2)*pow((cos(ty[3])),2)+2*m3*l2*l3*cos(ty[3])*cos(ty[3]+ty[5])+
m3*pow(l3,2)*pow((cos(ty[3]+ty[5])),2),
tH12=0, tH13=0, tH21=0,
tH22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(ty[5]),
tH23=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]), tH31=0,
tH32=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]), tH33=m3*pow(l3,2);

//Vector de Fuerzas de Coriolis ty de Fuerzas Centripetas
double tet_1=sin(ty[3])*cos(ty[3]),
tet_2=sin(ty[3]+ty[5])*cos(ty[3])+sin(ty[3])*cos(ty[3]+ty[5]),
tet_3=sin(ty[3]+ty[5])*cos(ty[3]+ty[5]),
tet_4=sin(ty[3]+ty[5])*cos(ty[3]),

tC11=0,
tC12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*ty[2],
tC13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*ty[2],

tC21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*ty[2],
tC22=-(2*m3*l2*l3*sin(ty[5]))*ty[6],
tC23=-(m3*l2*l3*sin(ty[5]))*ty[6],

tC31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*ty[2],
tC32=(m3*l2*l3*sin(ty[5]))*ty[4], tC33=0,

//Matriz de Gravedad
tG1=0,
tG2=-(m2*gr*lcg2+m3*gr*l2)*cos(ty[3])-m3*gr*lcg3*cos(ty[3]+ty[5]),
tG3=-m3*gr*lcg3*cos(ty[3]+ty[5])

   //Matriz de Fricción Viscosa ty Seca
tFr1=f1*tanh(beta*ty[2]), tFr2=f2*tanh(beta*ty[4]),
tFr3=f3*tanh(beta*ty[6]),

   // Variables Articulares ty Derivadas
tq[]=0,ty[1],ty[3],ty[5], tqp[]=0,ty[2],ty[4],ty[6],

   // Matrices del robot
tM[4][4]=0,0,0,0,0,tH11,tH12,tH13,0,tH21,tH22,tH23,0,tH31,tH32,tH33,
tC[4][4]=0,0,0,0,0,tC11,tC12,tC13,0,tC21,tC22,tC23,0,tC31,tC32,tC33,
tG[4]=0,tG1,tG2,tG3;
// Ecuación de control
return 1/tM[1][1]*(T[1]-tC[1][2]*qp[2]-tC[1][3]*qp[3]-b1*qp[1]-tFr1);
}

//g//f(x,y1,y2,y3)
```

```
double g(double x,double y01,double y02,double y03)
double
ty[]=0,y01,qp[1],y02,qp[2],y03,qp[3],

tH11=(m2+m3)*pow(l2,2)*pow((cos(ty[3])),2)+2*m3*l2*l3*cos(ty[3])*cos(ty[3]+ty[5])+
m3*pow(l3,2)*pow((cos(ty[3]+ty[5])),2),
tH12=0, tH13=0, tH21=0,
tH22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(ty[5]),
tH23=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]), tH31=0,
tH32=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]), tH33=m3*pow(l3,2);

   //Vector de Fuerzas de Coriolis ty de Fuerzas Centripetas
double tet_1=sin(ty[3])*cos(ty[3]),
tet_2=sin(ty[3]+ty[5])*cos(ty[3])+sin(ty[3])*cos(ty[3]+ty[5]),
tet_3=sin(ty[3]+ty[5])*cos(ty[3]+ty[5]),

tet_4=sin(ty[3]+ty[5])*cos(ty[3]), tC11=0,
tC12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*ty[2],
tC13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*ty[2],

tC21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*ty[2],
tC22=-(2*m3*l2*l3*sin(ty[5]))*ty[6],
tC23=-(m3*l2*l3*sin(ty[5]))*ty[6],

tC31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*ty[2],
tC32=(m3*l2*l3*sin(ty[5]))*ty[4], tC33=0,

//Matriz de Gravedad
tG1=0,
tG2=-(m2*gr*lcg2+m3*gr*l2)*cos(ty[3])-m3*gr*lcg3*cos(ty[3]+ty[5]),
tG3=-m3*gr*lcg3*cos(ty[3]+ty[5]),

//Matriz de Fricción Viscosa ty Seca
tFr1=f1*tanh(beta*ty[2]), tFr2=f2*tanh(beta*ty[4]),
tFr3=f3*tanh(beta*ty[6]),

// Variables Articulares ty Derivadas
tq[]=0,ty[1],ty[3],ty[5], tqp[]=0,ty[2],ty[4],ty[6],

// Matrices del robot
tM[4][4]=0,0,0,0,0,tH11,tH12,tH13,0,tH21,tH22,tH23,0,tH31,tH32,tH33,
tC[4][4]=0,0,0,0,0,tC11,tC12,tC13,0,tC21,tC22,tC23,0,tC31,tC32,tC33,
tG[4]=0,tG1,tG2,tG3;

return 1/tM[2][2]*(T[2]-tM[2][3]*e(x,y01,y02,y03)-tC[2][1]*qp[1]-tC[2][2]*qp[2]-
tC[2][3]*qp[3]-tG[2]-b2*qp[2]-tFr2);

}

//e//f(x,y1,y2,y3)
double e(double x,double y01,double y02,double y03)
double
```

```
ty[]=0,y01,qp[1],y02,qp[2],y03,qp[3],

tH11=(m2+m3)*pow(l2,2)*pow((cos(ty[3])),2)+2*m3*l2*l3*cos(ty[3])*cos(ty[3]+ty[5])+
m3*pow(l3,2)*pow((cos(ty[3]+ty[5])),2),
tH12=0, tH13=0, tH21=0,
tH22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(ty[5]),
tH23=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]), tH31=0,
tH32=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]), tH33=m3*pow(l3,2);

//Vector de Fuerzas de Coriolis ty de Fuerzas Centripetas
double tet_1=sin(ty[3])*cos(ty[3]),
tet_2=sin(ty[3]+ty[5])*cos(ty[3])+sin(ty[3])*cos(ty[3]+ty[5]),
tet_3=sin(ty[3]+ty[5])*cos(ty[3]+ty[5]),
tet_4=sin(ty[3]+ty[5])*cos(ty[3]),

tC11=0,
tC12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*ty[2],
tC13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*ty[2],

tC21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*ty[2],
tC22=-(2*m3*l2*l3*sin(ty[5]))*ty[6],
tC23=-(m3*l2*l3*sin(ty[5]))*ty[6],

tC31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*ty[2],
tC32=(m3*l2*l3*sin(ty[5]))*ty[4], tC33=0,

//Matriz de Gravedad
tG1=0,
tG2=-(m2*gr*lcg2+m3*gr*l2)*cos(ty[3])-m3*gr*lcg3*cos(ty[3]+ty[5]),
tG3=-m3*gr*lcg3*cos(ty[3]+ty[5]),

//Matriz de Fricción Viscosa ty Seca
tFr1=f1*tanh(beta*ty[2]), tFr2=f2*tanh(beta*ty[4]),
tFr3=f3*tanh(beta*ty[6]),

// Variables Articulares ty Derivadas
tq[]=0,ty[1],ty[3],ty[5], tqp[]=0,ty[2],ty[4],ty[6],

// Matrices del robot
tM[4][4]=0,0,0,0,0,tH11,tH12,tH13,0,tH21,tH22,tH23,0,tH31,tH32,tH33,
tC[4][4]=0,0,0,0,0,tC11,tC12,tC13,0,tC21,tC22,tC23,0,tC31,tC32,tC33,
tG[4]=0,tG1,tG2,tG3;

return (tM[3][2]*(T[2]-tC[2][1]*qp[1]-tC[2][2]*qp[2]-tC[2][3]*qp[3]-tG[2]-b2*qp[2]-tFr2)+
tM[2][2]*(tC[3][1]*qp[1]+tC[3][2]*qp[2]+tG[3]+b3*qp[3]+tFr3-T[3]))/(tM[3][2]*tM[2][3]-
tM[2][2]*tM[3][3]);
}


//Función para actualizar todas las variables de cada una de las matrices del robot. void actualiza(double
q01,double qp01,double q02,double
qp02,double q03,double qp03)
```

```
y[0]=0; y[1]=q01; y[2]=qp01; y[3]=q02; y[4]=qp02; y[5]=q03;
y[6]=qp03;

// Variables Articulares y Derivadas

q[0]=0; q[1]=y[1]; q[2]=y[3]; q[3]=y[5];

qp[0]=0; qp[1]=y[2]; qp[2]=y[4]; qp[3]=y[6];

H11=(m2+m3)*pow(l2,2)*pow((cos(y[3])),2)+2*m3*l2*l3*cos(y[3])*cos(y[3]+y[5])+
m3*pow(l3,2)*pow((cos(y[3]+y[5])),2);
H22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(y[5]);
H23=m3*pow(l3,2)+m3*l2*l3*cos(y[5]);
H32=m3*pow(l3,2)+m3*l2*l3*cos(y[5]);
H33=m3*pow(l3,2);

//Vector de Fuerzas de Coriolis y de Fuerzas Centripetas

et_1=sin(y[3])*cos(y[3]);
et_2=sin(y[3]+y[5])*cos(y[3])+sin(y[3])*cos(y[3]+y[5]);
et_3=sin(y[3]+y[5])*cos(y[3]+y[5]);
et_4=sin(y[3]+y[5])*cos(y[3]);

C12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*y[2];
C13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*y[2];

C21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*y[2];
C22=-(2*m3*l2*l3*sin(y[5]))*y[6]; C23=-(m3*l2*l3*sin(y[5]))*y[6];

C31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*y[2];
C32=(m3*l2*l3*sin(y[5]))*y[4];

//Matriz de Gravedad

G2=-(m2*gr*lcg2+m3*gr*l2)*cos(y[3])-m3*gr*lcg3*cos(y[3]+y[5]);
G3=-m3*gr*lcg3*cos(y[3]+y[5]);

// Matrices del robot
M[0][0]=0; M[0][1]=0; M[0][2]=0; M[1][0]=0; M[2][0]=0; M[3][0]=0;

M[1][1]=H11; M[1][2]=H12; M[1][3]=H13; M[2][1]=H21; M[2][2]=H22;
M[2][3]=H23; M[3][1]=H31; M[3][2]=H32; M[3][3]=H33;

   C[0][0]=0; C[0][1]=0; C[0][2]=0; C[1][0]=0; C[2][0]=0; C[3][0]=0;
C[1][1]=C11; C[1][2]=C12; C[1][3]=C13; C[2][1]=C21; C[2][2]=C22;
C[2][3]=C23; C[3][1]=C31; C[3][2]=C32; C[3][3]=C33; G[0]=0;
G[1]=G1; G[2]=G2; G[3]=G3;
}
```

A.7. . Código de la dinámica del robot antropomórfico en lazo cerrado con Runge Kutta de 4o orden (PID NO LINEAL)

```
   #include < stdio.h >
#include < iostream.h >
#include < iomanip.h >//archivo
#include < conio.h >
#include < math.h >
#include < stdlib.h >
#include < dos.h >
#define p 6
#define pi 3.1415926536
#define rad(a) (pi/180)*(a)
FILE *FL;//puntero de la estructura FILE

   //Condiciones iniciales
double x0=0.0, xf=30.0, h=0.01; double conta=0,
//tetas de referencia
qr[4]=0,0.8,0.8,0.8,
//errores
er[4]=0,0,0,0, de[4]=0,0,0,0,
//constantes derivativa, porcentual
wn[4]=0,1,1,1.3,

   kd1=(pow(wn[1],2)), ki1=(2*pow(kd1,1.5))/(kd1*100),

   kd2=(pow(wn[2],2)), ki2=(2*pow(kd2,1.5))/(kd2*100),

   kd3=(pow(wn[3],2)), ki3=(2*pow(kd3,1.5))/(kd3*100);

   double alfa1=2/wn[1], alfa2=2/wn[2], alfa3=2/wn[2];

   //declaracion de variables para evaluar S
double s1=0, s2=0, s3=0,

   //declaracion de variables para la integral de la tanh
itanh[4]=0,0,0,0;

   //variables articulares
double q1=rad(45), q2=-rad(45), q3=-rad(45), qp1=rad(0),qp2=rad(0), qp3=rad(0);

   //masas de los eslabones
double m1=0.45; double m2=0.25; double m3=0.15;

   // longitud de los eslabones
double l1=0.35; double l2=0.35; double l3=0.2;

   //longitud al centro de masas
double lcg1=0.5*l1; double lcg2=0.5*l2; double lcg3=0.5*l3;

   //Coeficiente de gravitación
double gr=9.81;
```

```
   double f1=0.1; double f2=0.5; double f3=0.01; double beta=10;

   //Fricción Viscosa
double b1=0.3; double b2=0.15; double b3=0.4;

   // Matriz de Inercia
double y[]=0,q1,qp1,q2,qp2,q3,qp3,

   H11=(m2+m3)*pow(l2,2)*pow((cos(y[3])),2)+2*m3*l2*l3*cos(y[3])*cos(y[3]+y[5])+
m3*pow(l3,2)*pow((cos(y[3]+y[5])),2),
H12=0,
H13=0,
H21=0,
H22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(y[5]),
H23=m3*pow(l3,2)+m3*l2*l3*cos(y[5]),
H31=0,
H32=m3*pow(l3,2)+m3*l2*l3*cos(y[5]),
H33=m3*pow(l3,2);

   //Vector de Fuerzas de Coriolis y de Fuerzas Centripetas
double et_1=sin(y[3])*cos(y[3]),
et_2=sin(y[3]+y[5])*cos(y[3])+sin(y[3])*cos(y[3]+y[5]),
et_3=sin(y[3]+y[5])*cos(y[3]+y[5]), et_4=sin(y[3]+y[5])*cos(y[3]),

   C11=0,
C12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*y[2],
C13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*y[2],

   C21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*y[2],
C22=-(2*m3*l2*l3*sin(y[5]))*y[6],
C23=-(m3*l2*l3*sin(y[5]))*y[6],

   C31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*y[2],
C32=(m3*l2*l3*sin(y[5]))*y[4],
C33=0,

   //Matriz de Gravedad
G1=0,
G2=-(m2*gr*lcg2+m3*gr*l2)*cos(y[3])-m3*gr*lcg3*cos(y[3]+y[5]),
G3=-m3*gr*lcg3*cos(y[3]+y[5]),

   //Matriz de Fricción Viscosa y Seca
Fr1=f1*tanh(beta*y[2]),
Fr2=f2*tanh(beta*y[4]),
Fr3=f3*tanh(beta*y[6]),

   // Variables Articulares y Derivadas
q[]=0,y[1],y[3],y[5],
qp[]=0,y[2],y[4],y[6],

   // Matrices del robot
```

```
M[4][4]=0,0,0,0,0,H11,H12,H13,0,H21,H22,H23,0,H31,H32,H33,
C[4][4]=0,0,0,0,0,C11,C12,C13,0,C21,C22,C23,0,C31,C32,C33,
G[4]=0,G1,G2,G3,

   // Ecuación de control
T[4]=0,0,0,0;
// Ecuaciones de Estado
void actualiza(double,double,double,double,double,double);
double f(double,double,double,double);
double e(double,double,double,double);
double g(double,double,double,double);
double ce1(double,double,double,double);
double ce2(double,double,double,double);
double ce3(double,double,double,double);

   void rk42(double,double,double,double,double,double);

   int Menu(char *titulo, char *opciones[]);
int main(void)


   void rk42(double x, double y1,double y2,double y3,double xf,double h)

double ea[4]=0,0,0,0,en[4]=0,0,0,0;
double k1,k2,k3,k4;
double m1,m2,m3,m4;
double p1,p2,p3,p4,pd1=0,pd2=0,pd3=0;
if ((FL= fopen ("Datos.mat", "w"))==NULL)
fprintf(FL,"qp[1],qp[2],qp[3],en[1],en[2],en[3], T[1],T[2],T[3]);//graba datos en archivo
//calculando el error inicial
er[1]=(qr[1]-q[1]);
er[2]=(qr[2]-q[2]);
er[3]=(qr[3]-q[3]);
//calculando los errores anteriores
ea[1]=er[1];
ea[2]=er[2];
ea[3]=er[3];
for (int cont=1; x¡=xf; cont++)

y1=q[1];
y2=q[2];
y3=q[3];
//se calcula teta punto porque en las funciones f,g,e, teta_pp=1/M[...]

   k1=h*f(x,y1,y2,y3);
m1=h*g(x,y1,y2,y3);
p1=h*e(x,y1,y2,y3);

   k2=h*f(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
m2=h*g(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
p2=h*e(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
```

```
    k3=h*f(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
m3=h*g(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
p3=h*e(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);

    k4=h*f(x+h,y1+k3,y2+m3,y3+p3);
m4=h*g(x+h,y1+k3,y2+m3,y3+p3);
p4=h*e(x+h,y1+k3,y2+m3,y3+p3);

    qp[1]+=(k1+2*k2+2*k3+k4)/6.0;
qp[2]+=(m1+2*m2+2*m3+m4)/6.0;
qp[3]+=(p1+2*p2+2*p3+p4)/6.0;

    actualiza(y1,qp[1], y2, qp[2], y3, qp[3]);
//se calcula teta porque en las funciones f,g,e, teta_p=...
k1=h*f(x,y1,y2,y3);
m1=h*g(x,y1,y2,y3);
p1=h*e(x,y1,y2,y3);

    k2=h*f(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
m2=h*g(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);
p2=h*e(x+h/2.0,y1+k1/2.0, y2+m1/2.0, y3+p1/2.0);

    k3=h*f(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
m3=h*g(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);
p3=h*e(x+h/2.0,y1+k2/2.0, y2+m2/2.0, y3+p2/2.0);

    k4=h*f(x+h,y1+k3,y2+m3,y3+p3);
m4=h*g(x+h,y1+k3,y2+m3,y3+p3);
p4=h*e(x+h,y1+k3,y2+m3,y3+p3);

    y1+=(k1+2*k2+2*k3+k4)/6.0;
y2+=(m1+2*m2+2*m3+m4)/6.0;
y3+=(p1+2*p2+2*p3+p4)/6.0;
//————————————-verificar error————————————-//
//calculando el error nuevo en cada iteracion
en[1]=(qr[1]-q[1]);
en[2]=(qr[2]-q[2]);
en[3]=(qr[3]-q[3]);
de[1]=(ea[1]-en[1])/cont;
de[2]=(ea[2]-en[2])/cont;
de[3]=(ea[3]-en[3])/cont;

    actualiza(y1,qp[1], y2, qp[2], y3, qp[3]);
//integrando la tangente hiperbolica como parte del control
s1=ea[1]+alfa1*de[1];
s2=ea[2]+alfa2*de[2];
s3=ea[3]+alfa3*de[3];

    k1=h*ce1(x,s1,s2,s3);
m1=h*ce2(x,s1,s2,s3);
p1=h*ce3(x,s1,s2,s3);
```

```
    k2=h*ce1(x+h/2.0,s1+k1/2.0, s2+m1/2.0, s3+p1/2.0);
m2=h*ce2(x+h/2.0,s1+k1/2.0, s2+m1/2.0, s3+p1/2.0);
p2=h*ce3(x+h/2.0,s1+k1/2.0, s2+m1/2.0, s3+p1/2.0);

    k3=h*ce1(x+h/2.0,s1+k2/2.0, s2+m2/2.0, s3+p2/2.0);
m3=h*ce2(x+h/2.0,s1+k2/2.0, s2+m2/2.0, s3+p2/2.0);
p3=h*ce3(x+h/2.0,s1+k2/2.0, s2+m2/2.0, s3+p2/2.0);

    k4=h*ce1(x+h,s1+k3,s2+m3,s3+p3);
m4=h*ce2(x+h,s1+k3,s2+m3,s3+p3);
p4=h*ce3(x+h,s1+k3,s2+m3,s3+p3);

    itanh[1]+=(k1+2*k2+2*k3+k4)/6.0;
itanh[2]+=(m1+2*m2+2*m3+m4)/6.0;
itanh[3]+=(p1+2*p2+2*p3+p4)/6.0;

    x+=h;
//aplicando el control a T

    T[1]=kd1*s1+ki1*itanh[1];
T[2]=kd2*s2+ki2*itanh[2];
T[3]=kd3*s3+ki3*itanh[3];
//actualizando valores para siguiente iteracion
ea[1]=en[1];
ea[2]=en[2];
ea[3]=en[3];
//————————————-verificar error————————————-//
//resultados de teta
fprintf(FL,"qp[1],qp[2],qp[3],en[1],en[2],en[3], T[1],T[2],T[3]);//graba datos en archivo
//igualando el contador global
conta=cont;
}//fin de for

    fclose(FL);
//ei: error a integrar
//ce: calculo del error a integrar
double ce1(double x,double ei1, double ei2, double ei3)

return (tanh(10*beta*ei1));
}

    double ce2(double x,double ei1, double ei2, double ei3)

return (tanh(10*beta*ei2));
}

    double ce3(double x,double ei1, double ei2, double ei3)

return (tanh(10*beta*ei3));
}
//Funciones a evaluar
//f(x,y1,y2,y3)
```

```
double f(double x,double y01,double y02,double y03)
// Matriz de Inercia
double
ty[]=0,y01,qp[1],y02,qp[2],y03,qp[3],

   tH11=(m2+m3)*pow(l2,2)*pow((cos(ty[3])),2)+2*m3*l2*l3*cos(ty[3])*cos(ty[3]+ty[5])+
m3*pow(l3,2)*pow((cos(ty[3]+ty[5])),2),
tH12=0,
tH13=0,
tH21=0,
tH22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(ty[5]),
tH23=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]),
tH31=0,
tH32=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]),
tH33=m3*pow(l3,2);
//Vector de Fuerzas de Coriolis ty de Fuerzas Centripetas
double tet_1=sin(ty[3])*cos(ty[3]),
tet_2=sin(ty[3]+ty[5])*cos(ty[3])+sin(ty[3])*cos(ty[3]+ty[5]),
tet_3=sin(ty[3]+ty[5])*cos(ty[3]+ty[5]),
tet_4=sin(ty[3]+ty[5])*cos(ty[3]),

   tC11=0,
tC12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*ty[2],
tC13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*ty[2],

   tC21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*ty[2],
tC22=-(2*m3*l2*l3*sin(ty[5]))*ty[6],
tC23=-(m3*l2*l3*sin(ty[5]))*ty[6],

   tC31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*ty[2],
tC32=(m3*l2*l3*sin(ty[5]))*ty[4],
tC33=0,

   //Matriz de Gravedad
tG1=0,
tG2=-(m2*gr*lcg2+m3*gr*l2)*cos(ty[3])-m3*gr*lcg3*cos(ty[3]+ty[5]),
tG3=-m3*gr*lcg3*cos(ty[3]+ty[5]),

   //Matriz de Fricción Viscosa ty Seca
tFr1=f1*tanh(beta*ty[2]), tFr2=f2*tanh(beta*ty[4]),
tFr3=f3*tanh(beta*ty[6]),

   // Variables Articulares ty Derivadas
tq[]=0,ty[1],ty[3],ty[5], tqp[]=0,ty[2],ty[4],ty[6],

   // Matrices del robot
tM[4][4]=0,0,0,0,0,tH11,tH12,tH13,0,tH21,tH22,tH23,0,tH31,tH32,tH33,
tC[4][4]=0,0,0,0,0,tC11,tC12,tC13,0,tC21,tC22,tC23,0,tC31,tC32,tC33,
tG[4]=0,tG1,tG2,tG3;
return 1/tM[1][1]*(T[1]-tC[1][2]*qp[2]-tC[1][3]*qp[3]-b1*qp[1]-tFr1);

   }
```

```
   //g//f(x,y1,y2,y3)
double g(double x,double y01,double y02,double y03)
double ty[]=0,y01,qp[1],y02,qp[2],y03,qp[3],

   tH11=(m2+m3)*pow(l2,2)*pow((cos(ty[3])),2)+2*m3*l2*l3*cos(ty[3])*cos(ty[3]+ty[5])+
m3*pow(l3,2)*pow((cos(ty[3]+ty[5])),2),
tH12=0,
tH13=0,
tH21=0,
tH22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(ty[5]),
tH23=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]),
tH31=0,
tH32=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]),
tH33=m3*pow(l3,2);

   //Vector de Fuerzas de Coriolis ty de Fuerzas Centripetas
double tet_1=sin(ty[3])*cos(ty[3]),
tet_2=sin(ty[3]+ty[5])*cos(ty[3])+sin(ty[3])*cos(ty[3]+ty[5]),
tet_3=sin(ty[3]+ty[5])*cos(ty[3]+ty[5]),

   tet_4=sin(ty[3]+ty[5])*cos(ty[3]), tC11=0,
tC12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*ty[2],
tC13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*ty[2],

   tC21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*ty[2],
tC22=-(2*m3*l2*l3*sin(ty[5]))*ty[6],
tC23=-(m3*l2*l3*sin(ty[5]))*ty[6],

   tC31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*ty[2],
tC32=(m3*l2*l3*sin(ty[5]))*ty[4],
tC33=0,

   //Matriz de Gravedad
tG1=0,
tG2=-(m2*gr*lcg2+m3*gr*l2)*cos(ty[3])-m3*gr*lcg3*cos(ty[3]+ty[5]),
tG3=-m3*gr*lcg3*cos(ty[3]+ty[5]),

   //Matriz de Fricción Viscosa ty Seca
tFr1=f1*tanh(beta*ty[2]), tFr2=f2*tanh(beta*ty[4]),
tFr3=f3*tanh(beta*ty[6]),

   // Variables Articulares ty Derivadas
tq[]=0,ty[1],ty[3],ty[5], tqp[]=0,ty[2],ty[4],ty[6],

   // Matrices del robot
tM[4][4]=0,0,0,0,0,tH11,tH12,tH13,0,tH21,tH22,tH23,0,tH31,tH32,tH33,
tC[4][4]=0,0,0,0,0,tC11,tC12,tC13,0,tC21,tC22,tC23,0,tC31,tC32,tC33,
tG[4]=0,tG1,tG2,tG3;

   return 1/tM[2][2]*(T[2]-tM[2][3]*e(x,y01,y02,y03)-tC[2][1]*qp[1]-tC[2][2]*qp[2]-
tC[2][3]*qp[3]-tG[2]-b2*qp[2]-tFr2);
```

```
    }

    //e//f(x,y1,y2,y3)
double e(double x,double y01,double y02,double y03)
double
ty[]=0,y01,qp[1],y02,qp[2],y03,qp[3],

    tH11=(m2+m3)*pow(l2,2)*pow((cos(ty[3])),2)+2*m3*l2*l3*cos(ty[3])*cos(ty[3]+ty[5])+
m3*pow(l3,2)*pow((cos(ty[3]+ty[5])),2),
tH12=0, tH13=0, tH21=0,
tH22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(ty[5]),
tH23=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]),
tH31=0,
tH32=m3*pow(l3,2)+m3*l2*l3*cos(ty[5]),
tH33=m3*pow(l3,2);

    //Vector de Fuerzas de Coriolis ty de Fuerzas Centripetas
double tet_1=sin(ty[3])*cos(ty[3]),
tet_2=sin(ty[3]+ty[5])*cos(ty[3])+sin(ty[3])*cos(ty[3]+ty[5]),
tet_3=sin(ty[3]+ty[5])*cos(ty[3]+ty[5]),
tet_4=sin(ty[3]+ty[5])*cos(ty[3]),

    tC11=0,
tC12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*ty[2],
tC13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*ty[2],

    tC21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*ty[2],
tC22=-(2*m3*l2*l3*sin(ty[5]))*ty[6],
tC23=-(m3*l2*l3*sin(ty[5]))*ty[6],

    tC31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*ty[2],
tC32=(m3*l2*l3*sin(ty[5]))*ty[4],
tC33=0,

    //Matriz de Gravedad
tG1=0,
tG2=-(m2*gr*lcg2+m3*gr*l2)*cos(ty[3])-m3*gr*lcg3*cos(ty[3]+ty[5]),
tG3=-m3*gr*lcg3*cos(ty[3]+ty[5]),

    //Matriz de Fricción Viscosa ty Seca
tFr1=f1*tanh(beta*ty[2]), tFr2=f2*tanh(beta*ty[4]),
tFr3=f3*tanh(beta*ty[6]),

    // Variables Articulares ty Derivadas
tq[]=0,ty[1],ty[3],ty[5], tqp[]=0,ty[2],ty[4],ty[6],

    // Matrices del robot
tM[4][4]=0,0,0,0,0,tH11,tH12,tH13,0,tH21,tH22,tH23,0,tH31,tH32,tH33,
tC[4][4]=0,0,0,0,0,tC11,tC12,tC13,0,tC21,tC22,tC23,0,tC31,tC32,tC33,
tG[4]=0,tG1,tG2,tG3;
```

```
    return (tM[3][2]*(T[2]-tC[2][1]*qp[1]-tC[2][2]*qp[2]-tC[2][3]*qp[3]-tG[2]-b2*qp[2]-tFr2)+
tM[2][2]*(tC[3][1]*qp[1]+tC[3][2]*qp[2]+tG[3]+b3*qp[3]+tFr3-T[3]))/(tM[3][2]*tM[2][3]-
tM[2][2]*tM[3][3]);
}

    void actualiza(double q01,double qp01,double q02,double
qp02,double q03,double qp03)

    y[0]=0;
y[1]=q01;
y[2]=qp01;
y[3]=q02;
y[4]=qp02;
y[5]=q03;
y[6]=qp03;

    // Variables Articulares y Derivadas

    q[0]=0;
q[1]=y[1];
q[2]=y[3];
q[3]=y[5];

    qp[0]=0;
qp[1]=y[2];
qp[2]=y[4];
qp[3]=y[6];

    H11=(m2+m3)*pow(l2,2)*pow((cos(y[3])),2)+2*m3*l2*l3*cos(y[3])*cos(y[3]+y[5])+
m3*pow(l3,2)*pow((cos(y[3]+y[5])),2);
H22=(m2+m3)*pow(l2,2)+m3*pow(l3,2)+2*m3*l2*l3*cos(y[5]);
H23=m3*pow(l3,2)+m3*l2*l3*cos(y[5]);
H32=m3*pow(l3,2)+m3*l2*l3*cos(y[5]);
H33=m3*pow(l3,2);

    //Vector de Fuerzas de Coriolis y de Fuerzas Centripetas

    et_1=sin(y[3])*cos(y[3]);
et_2=sin(y[3]+y[5])*cos(y[3])+sin(y[3])*cos(y[3]+y[5]);
et_3=sin(y[3]+y[5])*cos(y[3]+y[5]); et_4=sin(y[3]+y[5])*cos(y[3]);

    C12=-(2*(m2+m3)*pow(l2,2)*et_1+2*m3*l2*l3*et_2+2*m3*pow(l3,2)*et_3)*y[2];
C13=-(2*m3*l2*l3*et_4+2*m3*pow(l3,2)*et_3)*y[2];

    C21=((m2+m3)*pow(l2,2)*et_1+m3*l2*l3*et_2+m3*pow(l3,2)*et_3)*y[2];
C22=-(2*m3*l2*l3*sin(y[5]))*y[6]; C23=-(m3*l2*l3*sin(y[5]))*y[6];

    C31=(m3*l2*l3*et_4+m3*pow(l3,2)*et_3)*y[2];
C32=(m3*l2*l3*sin(y[5]))*y[4];

    //Matriz de Gravedad
```

```
    G2=-(m2*gr*lcg2+m3*gr*l2)*cos(y[3])-m3*gr*lcg3*cos(y[3]+y[5]);
G3=-m3*gr*lcg3*cos(y[3]+y[5]);

    // Matrices del robot
M[0][0]=0;
M[0][1]=0;
M[0][2]=0;
M[1][0]=0;
M[2][0]=0;
M[3][0]=0;

    M[1][1]=H11;
M[1][2]=H12;
M[1][3]=H13;
M[2][1]=H21;
M[2][2]=H22;
M[2][3]=H23;
M[3][1]=H31;
M[3][2]=H32;
M[3][3]=H33;

    C[0][0]=0;
C[0][1]=0;
C[0][2]=0;
C[1][0]=0;
C[2][0]=0;
C[3][0]=0;

    C[1][1]=C11;
C[1][2]=C12;
C[1][3]=C13;
C[2][1]=C21;
C[2][2]=C22;
C[2][3]=C23;
C[3][1]=C31;
C[3][2]=C32;
C[3][3]=C33;
G[0]=0;
G[1]=G1;
}
```

A.8. . Construcción de un Escenario Virtual

La realización del escenario virtual realizada para poder demostrar los resultados del método y los pasos para poder crearlo son los que se mencionan a continuación. Para el modelo del robot se tomaron las medidas reales a escala del robot PUMA, tomando como base la figura A.1 y mediante la unión y modificación de objetos se logra la forma deseada.

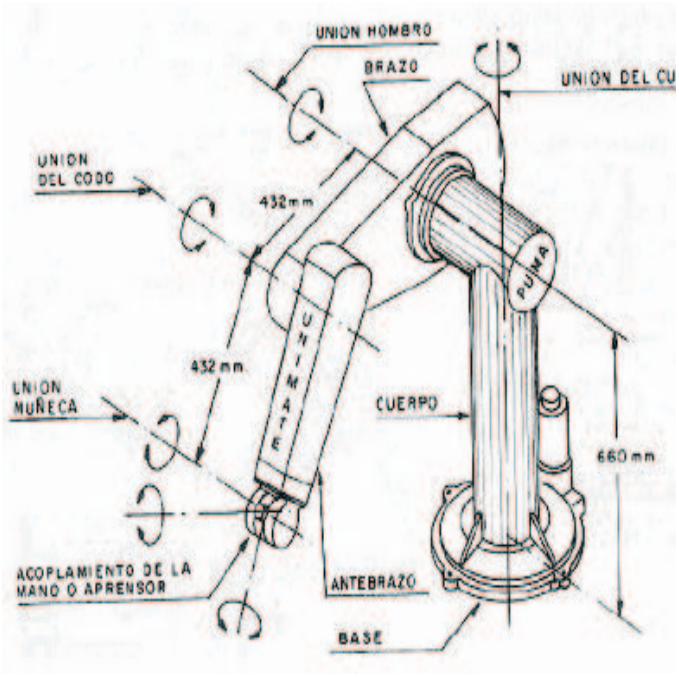

Figura A.1: Medidas reales de un robot PUMA.

En primer lugar se realiza la base fija del robot que consta de un cilindro, cuyas medidas se han modificado a una escala mas pequeña y esto se logra dando dichas medidas en las propiedades del objeto cilindro o base del robot y la base quedara tal y como se muestra en la figura A.2 que se presenta a continuación:

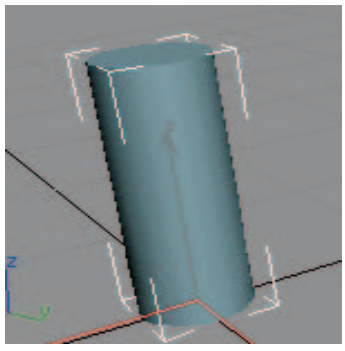

Figura A.2: Base del robot PUMA mediante la elaboración de un cillindro.

Una vez establecida la base se pusieron los moldes que sustentan dicha base, esto se realiza con la creación de aros también llamados dentro del entorno de 3D Studio como Toroides, para ello se selecciona en la persiana Tipo

de Objeto la categoría aro (dentro de Primitivas estándar). En cualquier visor se hace clic y se arrastra el ratón para definir los eslabones rectangulares de la caja (manteniendo pulsado CTRL se consigue una base cuadrada).

Cuando se suelta el ratón se establece la longitud y anchura. Para definir la altura se arrastra el ratón hacia arriba o abajo y llegado el punto deseado, se hace clic. Y posteriormente se puede modificar su tamaño de acuerdo alas medidas del robot.
Después se elabora un segundo cilindro que se coloca de forma horizontal sobre la base fija el cual sostendrá las articulaciones del robot constituyendo así el primer eslabón del robot como se muestra en la figura A.3.

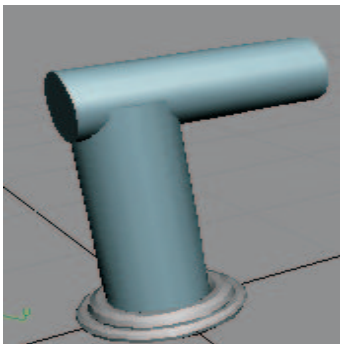

Figura A.3: Primer eslabón del robot.

Para crear las siguientes partes de la extremidad se tomaran dos rectángulos y se modificara su estructura, mediante la técnica de mayado se movieron sus vértices hasta dar la forma deseada, un cilindro más para el segundo eslabón entre la primer parte de la extremidad y la segunda; de igual forma para la segunda parte de la extremidad se tomó un rectángulo y se deformó con la técnica de mayado y se deforma para tomar la forma final que se muestra e la figura A.4.

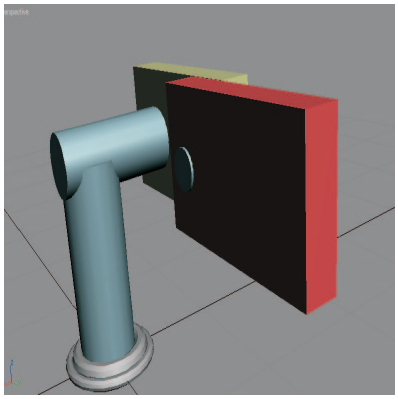

Figura A.4: Extremidades del robot PUMA.

Para modificar la forma de los rectángulos como se muestra en la figura A.5 se seleccionan y dando clic con el botón derecho del Mouse y convertirlo en malla editable, enseguida seleccionar los vértices y moverlos hasta crear la forma deseada.

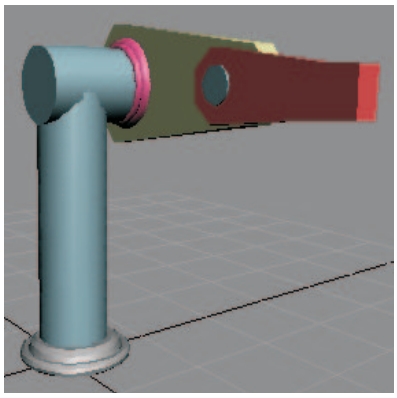

Figura A.5: Modificación de las extremidades.

Posteriormente con la ayuda de un rectángulo, 3 cilindros del mismo tamaño y mas pequeños que en rectángulo, un cilindro más largo y menos ancho, se forma la base para el efector final como se muestra en la figura A.6.

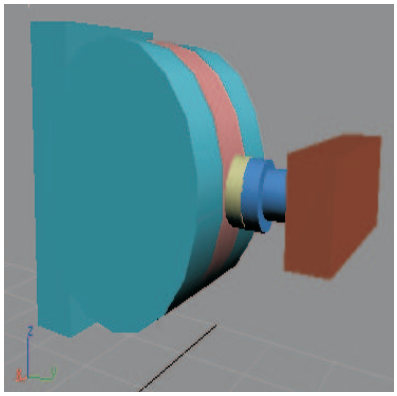

Figura A.6: Efector final de robot.

La realización de la pinza se hace con la ayuda de dos rectangulos teniendo como efector final una pinza como se muestra en la figura A.7.

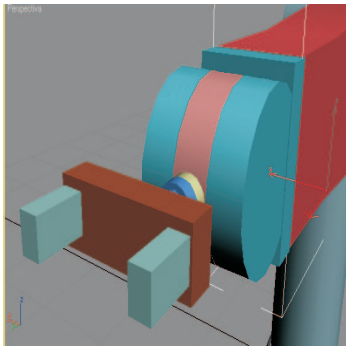

Figura A.7: Pinza del robot.

Despues de haber realizado todos estos pasos se obtendra como resultado un robot como el que se muestra en la figura A.8.

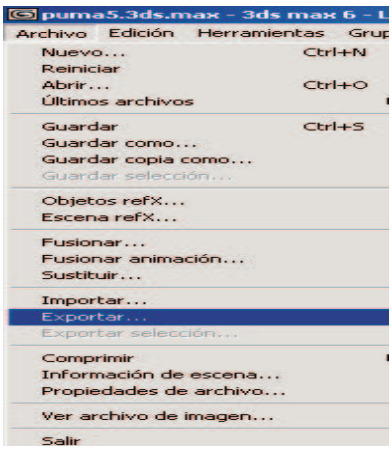

Figura A.8: Robot.

En este punto el modelo es exportado a el formato “3ds”, con el fin de dejarlo listo para la siguiente transformación, que es convertirlo a un proyecto de Visual C++ y se puedan utilizar funciones de OpenGL (llamadas gluts),la exportación se muestra en la figura A.9 :
Lo guardamos en la opcion “Guardar como” con el formato 3ds como se muestra en la figura A.10

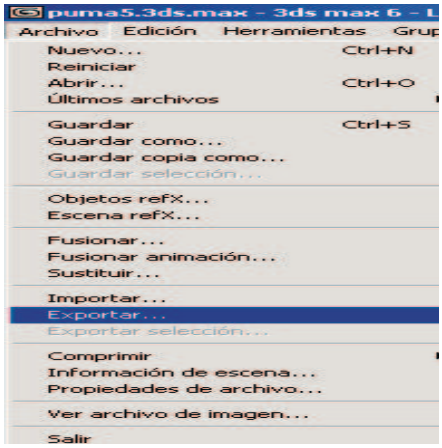

Figura A.9: Exportación del robot.

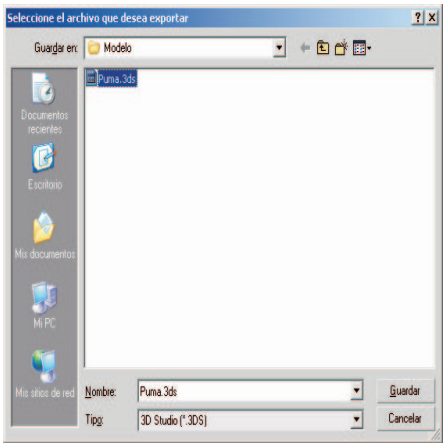

Figura A.10: Formato en 3ds.

Ya en el entorno de trabajo de 3D Exploration como se muestra en la figura A.11 observaremos la exportacion del robot ya realizada.

A continuación se hace la exportación a un formato comprensible para las librerías de Opengl en este caso C, como se ilustra en la figura A.12 . Cabe mencionar que para generar el proyecto como tal hay que tomar en cuenta las directrices que se muestran en la figura A.13.

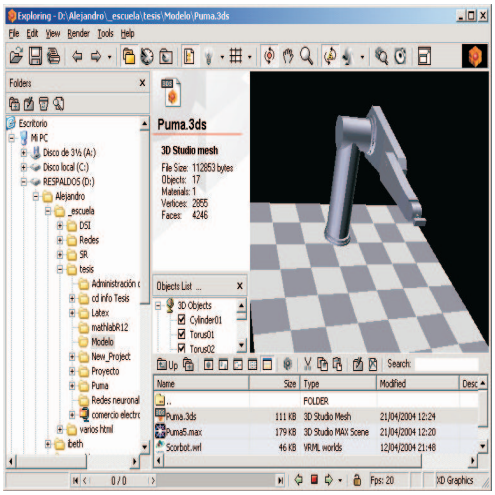

Figura A.11: En 3D Exploration.

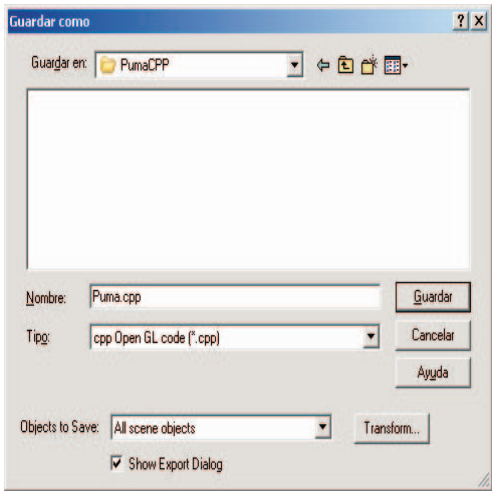

Figura A.12: Exportando al lenguaje C.

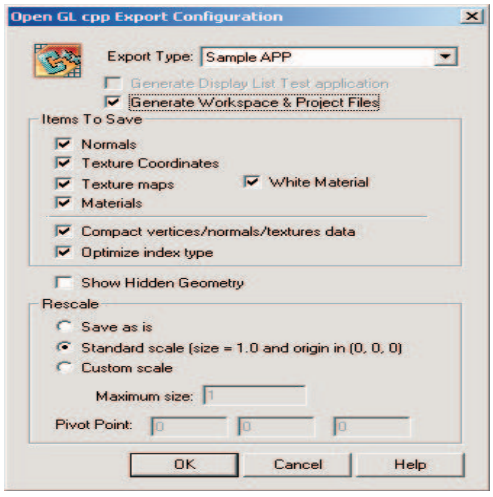

Figura A.13: Directrices para la exportación.

Bibliografía

[1] Arimoto, S., *"Control Theory of Non-Linear Mechanical Systems"*, Oxford Science Publications, 1996.

[2] 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.

[3] Burdea G. C. and Coiffet, P., *"Virtual Reality Technology"*, John Wiley and Sons, Inc., New York, 1994.

[4] 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.

[5] Domínguez-Ramírez, O. A. and V. Parra-Vega, *"Interacción Háptica de Alto Desempeño con Objetos Virtuales Dinámicos Deformables"*, Avances en la Ciencia de la Computación en México, ISBN 970-36-0026-3, Centro de Investigación en Computación del Instituto Politécnico Nacional y ACM México. Research on Computing Science, vol 2, pp 294-310, 2003.

[6] D. Feygin, M. Keehner, and F. Tendick, "Haptic Guidance: Experimental Evaluation of a Haptic Training Method for a Perceptual Motor Skill." Proceedings 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, part of IEEE Virtual Reality 2002, Orlando, FL, pp. 40–47, Mar. 2002

[7] 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.

[8] Slotine, J.J. and W. Li, *"On the Adaptive Control of Manipulators"*, International Journal of Robotics Research, Vol. 6, No. 3, 1987.

[9] Spong, M. and Vidyasagar, M., *"Robot Dynamics and Control"*, , John Wiley and Sons, 1989

[10] Mendoza Vazquez Jose Rafael, *"Diseño del control de un robot de dos grados de libertad para aplicaciones de seguimientos de objetos"*, Instituto Nacional de Astrofísica , Óptica y Electrónica, Puebla,México.

[11] Steven C. Chapra, Raymond P. Canale, *"Métodos Numéricos para Ingenieros"*, ed.Mc GrawHill, tercera edición, México.

[12] Dennis G.Zill, *"Ecuaciones diferenciales con aplicaciones"*, ed.Iberoamericana S.A. de C.V, tercera edición, México, 1993.

[13] Herbert Lara Ordaz, *"Análisis y diseño de una interfaz de visualización virtual del dispositivo háptico PHANToM 1.0 basada en la dinámica de Euler-Lagrange "*, Universidad Autónoma del Estado de Hidalgo, Pachuca, Hidalgo,2005.

[14] Omar A. Domínguez-Ramírez, V. López-Morales, R. Samperio-Llano *"Resultados Preliminares sobre Interacción Háptica en Laberintos Virtuales, con Propósitos de Diagnóstico en Pacientes con Discapacidades Neuropsicológicas"*,Universidad Autónoma del Estado de Hidalgo, Centro de Investigación en Tecnologías de Información y Sistemas Grupo Científico de Electrónica y Control.

[15] Domínguez-Ramírez , O. A. , Herbert Lara Ordaz Ramón Soto de la Cruz, Virgilio López-Morales, *"Visualization and Control of Virtual Robot Manipulators"*, Citis, UAEH, CICINDI 2004, ISBN:970-36-0189-8

[16] Benjamin C. Kuo *"Sistemas de Control Digital"*, ed. Cecsa primera edicion, México, 1997.

[17] Katsuhiko Ogata *"Ingeniería de control moderna"*, ed. Prentice Hall, primera edición, México, 1980.

[18] $http://www.control-systems.net/resources/glosario/r.htm$

[19] $http://proton.ucting.udg.mx/materias/robotica/r166/r66/r66.htm$

[20] $http://bellota.ele.uva.es/\ imartin/libro/node9.html$

[21] $http://www.ciencia.net/VerArticulo/Momento-de-Inercia?idArticulo=5137$

[22] $http://www.roboticajoven.mendoza.edu.ar/rob_sis5.htm$

[23] $http://www4.geometry.net/authors/asimov_isaac_page_no_2.html$

[24] $http://www-groups.dcs.st-andrews.ac.uk/\ history/Indexes/Full_Alph.html$

[25] $http://www.activamente.com.mx/vrml$

[26] $http://www.cuadernoscervantes.com/multi_47_adquisicion.html$

[27] $http://www.cp.com.uy/82/82autodesk.htm$

[28] $www.publispain.com/supertutoriales/grafico/autocad/$

[29] $www.guiaautocad.com$

[30] $http://rnasa.tic.udc.es/gc/$

[31] $http://www.omegasystems.cl/index/maya$

[32] $http://www.boxxtech.com/applications/alias_maya.asp$

[33] $http://www.puntocad.com/descripcion/3dsmax.htm$

[34] $http://www,3dyanimacion.com/Analisis/analisis.cfm?link=deepexp$

[35] $http://agamenon.uniandes.edu.co/\ fde/isis3802003-2/ejemplos/$

[36] $http://www.activamente.com.mx/vrml/$

[37] $http://www.med.ub.es/\ aprats/curvrml/html/cvrml03.htm$

[38] $http://dac.escet.urjc.es/docencia/GV3D/OpenGl1.pdf$

[39] $http://trevinca.ei.uvigo.es/\ formella/doc/ig03/node31.html$

[40] $http://www.linuxfocus.org/Castellano/January1998/article15.html$

[41] $http://dac.escet.urjc.es/docencia/IG/PresentacionOpenGL.pdf$

[42] $http://www.info-ab.uclm.es/labelec/solar/electronica/index.htm$

[43] $http://www.mathworks.com/$

[44] $http://www.cia.mty.itesm.mx/\ gordillo/GV/personal.html$

[45] $http://www.disclab.ua.es/robolab/principal.htm$

[46] $http://www.cosimir.com/VR/English/Technologie/Introduction.htm$

[47] $http://www.cp.com.uy/82/82autodesk.htm$