UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO

CENTRO DE INVESTIGACIÓN EN TECNOLOGÍAS DE INFORMACIÓN Y SISTEMAS

INSTITUTO DE CIENCIAS BÁSICAS E INGENIERÍA

"Scorbot ER-VII virtual: modelado matemático y control de movimiento"

Proyecto de tesis para obtener el título de:

Lic. en Sistemas Computacionales

Presentan:

López Pacheco Liana
Martínez Olguín Verónica

Director de tesis:

Dr. Omar Arturo Domínguez Ramírez

Co-Asesor de tesis:

M. en C. Luis Ramón Macías Pulido

Pachuca de Soto, Hgo. 2006

Agradecimientos

"Dios no te hubiera dado la capacidad de soñar sin darte también la posibilidad de convertir tus sueños en realidad". Héctor Tassinari

A Dios

Por darme la oportunidad de conocer a gente maravillosa y estar conmigo siempre que lo necesito.

A mis padres

Por la confianza, apoyo incondicional y esfuerzos que hicieron para que continuara con mi educación. Al brindarme seguridad en mis decisiones tomadas y compartir todas mis alegrías. Ser apoyo fiel en mis tristezas dándome una palabra de aliento cuando la he necesitado. A mi padre por que fue la chispa que dio inicio a mi inquietud por conocer el comportamiento de los de los robots.

A mis grandes amigos

No hay palabras para poder agradecer todo el apoyo incondicional que me han brindado desde el inicio hasta el final de esta carrera. Por brindarme su amistad y compartir conmigo los buenos, malos y maravillosos momentos de su vida. Y solo puedo decir gracias. Seguiré adelante.

A papi Omar

Mi director de tesis y amigo. A ti dedico estas líneas, por brindarme esta gran amistad y compartir conmigo geniales momentos. Por el gran apoyo incondicional, comentarios, consejos profesionales y sobre todo por el tiempo dedicado e invaluable colaboración.

Liana López Pacheco

Agradecimientos

Jamás dejes que las dudas paralicen tus acciones. Toma siempre todas las decisiones que necesites tomar, aún cuando no tengas la seguridad de estar decidiendo correctamente. Paulo Coelho, Brida.

A Dios

Gracias por estar conmigo cuando te he necesitado y darme la oportunidad de vivir y compartir la vida misma.

A mis padres

Gracias por darme la vida, por su amor, por el dolor, por las sonrisas, por el sufrimiento, por los regaños, gracias por enseñarme a crecer, a través del sufrimiento, curándome las heridas y consolándome en mis lamentos, gracias por el ejemplo tan lindos que me han dado, por los regaños que ahora me doy cuenta que eran solo muestras de su gran amor, perdónenme por no ser como ustedes querían.

Gracias con todo mi corazón, gracias por ser como son y por ser la pareja que ustedes son.

A mis dos grandes amigos Phoenix Seyer y Lianita

Gracias por todos los momentos que hemos compartido momentos llenos de sentimientos y pensamientos compartidos, sueños y anhelos, secretos, risas y lágrimas, gracias por dedicarme tiempo para demostrar su preocupación por mi, tiempo para escuchar mis problemas y ayudarme a buscar solución, gracias por ser lo que son unas persona maravillosa, gracias a ustedes comencé a conocerme e incluso a apreciar lo que soy y por todo lo que han hecho por mi, los quiero mucho amigo(a).

Gracias a papi Dr. Omar por la amistad, confianza y consejos que me ha brindado y los conocimientos que he adquirido y al M. en C. David Hdz por su gran confianza, amistad y apoyo en los momentos que lo he necesitado.

Verónica Martínez Olguín.

Resumen

Este trabajo de investigación y desarrollo tecnológico presenta como resultado el diseño e integración de un ambiente virtual dinámico de un robot Scorbot ER-VII empleado para evaluar estrategias de control de movimiento a partir de la respuesta transitoria de sus variables articulares de posición y velocidad, así como una trayectoria para la consigna de movimiento descrita a partir de sus ecuaciones paramétricas, en este caso el control de modos deslizantes hace el seguimiento de trayectorias programadas haciendo que el error sea compensado con la tarea a seguir. El sistema fue elaborado en Visual C++ con librerías de OpenGL y herramientas de autoría como 3D Studio MAX para el modelado de la estructura del robot con detalles cinemáticos finos, para la exportación a código OpenGL se utilizó Deep Exploration edición CAD. La respuesta de movimientos controlados fué adquirida con Matlab y su integrador.

Existe una serie de paradigmas, estos paradigmas persiguen el objetivo de la construcción de un producto de buena calidad. Adicionalmente que el desempeño del sistema resultante sea el fiel reflejo de los requerimientos del usuario. A demas que el software debe cumplir con los requerimientos mínimos de robustez y estabilidad, es por ello que se utilizo como herramienta para el análisis, diseño, desarrollo, implementación y evaluación del sistema para nuestro software la metodología desarrollo de sistemas (SMD).

Índice general

Índice de figuras

Capítulo 1

Introducción

1.1. Introducción

Los robots manipuladores a través del tiempo han resuelto problemas prácticos de posicionamiento de objetos en el espacio y otro tipo de tareas que permiten incrementar la productividad sin sacrificar la calidad de un producto terminado. Un robot manipulador, es diseñado y construido de acuerdo a la tarea que va a desempeñar, y en general se debe cumplir una regla de diseño, en ella es necesario considerar el diseño electromecánico a partir del modelo matamático (cinemática y dinámica), mismos que permiten determinar la ley de control de movimiento para el desempeño eficiente de la tarea. Este aspecto relevante en el diseño y puesta en marcha de un manipulador requiere de la evaluación a nivel simulación digital y corroborar con ello la funcionalidad de todo el sistema bajo ciertas condiciones. Estas simulaciones son a menudo digitales, es decir respuestas en régimen del tiempo, situación que no permite el conocimiento exhaustivo del desempeño del robot y de la ley de control evaluada, así como la trayectoria planificada. Este trabajo de investigación tiene el propósito de desarrollar un sistema de visualización científica con ambientes de realidad virtual cuyo comportamiento complejo esta basado en la solución numérica del modelo dinámico y cinemático en laso cerrado con la ley de control de movimiento evaluada para el seguimiento de la tarea. En esta tésis se reporta los modelos cinemáticos de posición y diferencial del robot Scorbot ER-VII (caso de estudio), el modelo dinámico basado en formulación Euler-Lagrange de su estructura de posición y se diseñan y evalúa el control por modos deslizantes de segundo orden. Se diseña el ambiente virtual del robot experimental en Visual C++ 6.0 con OpenGL como resultados finales.

1.2. Objetivo general

Diseñar e integrar al robot Scorbot ER-VII en un mundo virtual en el que sea posible interactuar con sus modelos matemáticos, con el propósito de que éste efectúe la simulación de movimiento controlado utilizando herramientas computacionales de última generación para la asignación de comportamientos complejos en mundos virtuales dinámicos.

1.3. Objetivos específicos

- Obtener el modelo cinemático directo e inverso de posición, así como el modelo diferencial empleando el algoritmo de Denavit-Hartenberg.
- Obtener el modelo dinámico basado en la formulación de Euler-Lagrange y análisis de propiedades dinámicas.
- Evaluación de algoritmos para el control de movimiento del tipo de modos deslizantes de segundo orden por modelo de referencia.

- Diseñar e integrar un ambiente virtual con Visual C++ empleando las librerías de OpenGL y la cinemática del robot.
- Asignar el comportamiento complejo del robot utilizando la dinámica en lazo cerrado con cada uno de los controladores a partir de la simulación digital en Matlab.
- Desarrollar una interfaz de usuario de entrada y salida numérica que permita la planificación de movimientos y la lectura de las variables del robot, utilizando como herramienta la metodología desarrollo de sistemas (SMD).

1.4. Justificación

La robótica, y más aún la robótica de manipuladores industriales tiene un impácto relevante en diferentes procesos industriales tales como el posicionamiento de objetos en el espacio, el desarrollo de tareas de barrenado, pintura, montaje, entre muchas otras; sin embargo la ejecución inicial de la tarea una vez planificada, implica fallos de operación por no considerar o considerar parcialmente la dinámica del robot y de su entorno. Las reglas de diseño y planificación de una tarea industrial implican la evaluación de la técnica de control a nivel simulación digital y para ellos son empleados distintos tipos de simuladores digitales comerciales o bien los que son de propósito específico del manipulador en cuestión, sin embargo los resultados proporcionados son exclusivamente en régimen transitorio, es decir solo podemos apreciar la respuesta de las variables articulares y operacionales en el dominio del tiempo, y en el mejor de los casos la respuesta operacional en el espacio de trabajo, esto a partir de la estela que deja el órgano terminal al ejecutar la tarea. En la industria se aprecia conocer el desempeño del robot en su conjunto al evaluar la tarea, es decir la ubicación instantánea del mecanismo de eslabones articulados e inclusive su respuesta ante la presencia de objetos existentes en su mundo, esto por supuesto de manera previa a la ejecución real. La realidad virtual es una alternativa para poder visualizar los eventos definidos por un robot de manera previa a su ejecución, son mucho los trabajos que se han propuesto al momento, sin embargo estos evaluan exclusivamente la cinemática de posición del robot, sin considerar los efectos que la dinámica pueda afectar durante el desarrollo de la tarea. En este trabajo de tesis se propone un nuevo simulador virtual basado en la dinámica no lineal y la aplicación de estrategias de control de movimiento no lineal basado y no en la estrucutra del modelo, el caso de estudio es el robot Scorbot ER-VII. El propósito de la virtualización del robot Scorbot ER-VII es de brindar una metodología nueva utilizando herramientas computacionales y de control moderno que permiten eficientar el proceso de diseño y construcción de un sistema de automatización de procesos industriales. Adicionalmente, este sistema puede ser utilizado como herramienta didáctica en el proceso de enseñanza-aprendizaje de materias de licenciatura y posgrado como: robótica, mecatrónica, automatización, control, realidad virtual y programación orientada a objetos.

1.5. Planteamiento del problema

En la industria, la automatización ocupa un lugar relevante y cuyo propósito es el incrementar la productividad sin sacrificar la calidad final del producto, los robots ocupan un espacio primordial en esta tarea, sin embargo y en la mayoría de los casos se tiene la siguiente problemática:

- Las estrategias de control son lineales, cuando un robot es de naturaleza no lineal.
- Las técnicas de simulación son de régimen transitorio y no permiten la validación de los movimientos en su conjunto.
- Los simuladores virtuales existentes son basados en la cinemática, es decir no consideran los afectos dinámicos.

1.6. Estado del arte

Para el año de 1965, el pionero de la computación gráfica, Ivan Sutherland propone una interfaz de despliegue visual para mundos virtuales que incluye retroalimentación háptica. En 1967, Frederick Brooks Jr., y coloaboradores de la universidad de Carolina del Norte, motivados por la visión de Ivan Sutherland sobre gráficas computacionales interactivas con retroalimentación de fuerzas, desarrollaron el proyecto GROPE, cuyo ambicioso objetivo fué, desarrollar en tiempo real, cortes con retroalimentación de fuerzas de moleculas tridimencionales. Proyecto que permitió desarrollar investigación de retroalimentación de fuerzas, durante un largo periodo. En 1971, el GROPE-I permitió resolver el problema de simulación de campo de fuerza en dos dimenciones. En 1976 éste equipo de investigación, utilizó accesorios y dispositivos sobrantes del proyecto de Goertz, para simular fuerzas de colisión en tres dimenciones. El equipo de investigación de Brooks perseveró en lograr su objetivo orinal y 20 años más tarde, con equipo computacional de alto desempeño, logró culminar su trabajo.

A finales de los 80's, investigadores de NASA (National Aeronautics and Space Administration) y de JPL (Jet Propulsion Laboratories) desarrollaron un brazo maestro de seis grados de libertad con retroalimentción de fuerzas con propósitos de teleoperación, y cuyas aplicaciones correspondían a las misiones de reparación espacial. Para efectos de prueba experimental, utilizaron el Salisbury/JPL Arm. El esquema en este momento requería que los modelos cinemáticos de los dispositivos maestro y esclavo fueran similares, situación que propiciaba la construcción de dispositivos maestros de excedidas dimenciones. Los trabajos experimentales en el Salisbury/JPL Arm permitieron introducir un a estrategia de control cartesiano, situación que permitió disminuir las dimenciones del dispositivo maestro, y con la capacidad de teleoperar robots esclavos con diferentes configuraciones cinemática. Posteriormente, se desarrollaron aplicaciones con excelente desempeño en el que los dispositivos esclavos fueron definidos por ambientes virtuales flexibles con propósito de entrenamiento.

De 1993 a 1995 surgieron de manera comercial, dispositivos con retroalimentación háptica, visual y auditiva. La tendencia del costo fué disminuyendo y con ello logrando que la comunidad de interfaces hápticas diversificara las aplicaciones, propusiera algoritmos nuevos y más rápidos, logrará la integración con otros dispositivos especializados, etc. Los dispositivos que lograron este propósito son el Touch Master y el SAFIRE Master en 1995, el PHANToM Arm en 1994 utilizando experimentalmente en este trabajo de investigación, y el Impulse Engine en 1955.

En 1995, Colgate, Stanley, y Brown proponen una estrategia de acoplamiento virtual entre el dispositivo háptico y el embiente virtual para eliminar el problema de baja estabilidad. J. Michael Broun exploró las condiciones bajo las cuales los parámetros del acoplamiento virtual garantiza una interfaz pasiva para el operador humano. El desarrolló un criterio de diseño para un arbitrario ambiente pasivo discreto en el tiempo y para una simulación virtual no pasiva.

Algunos trabajos recientes están enfocados con la teoría de pasividad. Adams y colaboradores derivaron un método de diseño basado en acoplamiento virtual con teoría de redes de dos puertos aplicando combinaciones causales. Determinaron parámetros óptimos de acoplamiento virtual utilizando el modelo dínámico del dispositivo háptico y satisfaciendo el criterio de estabilidad absoluta de Lewellyn, logrando un acoplamiento óptimo del ambiente virtual y del dispositivo háptico. Este procedimento garantiza estabilidad de contacto y acoplamiento virtual de alto desempeño tanto como el ambiente virtual sea pasivo. Miller, Colgate y Freeman derivaron otros procedimentos, en el cual estienden el análisis en embientes no lineales y definen un parémetro amortiguador para garantizar una operación estable. Resientemente se propuso un nuevo metodo basado en la energía, el observador de pasividad y el controlador de pasividad, este nuevo método puede ser aplicado en interfaces hápticas con requerimeintos de estabilidad durante el contacto y en teleoperadores bilaterales [1].

La robótica virtual es un sistema de simulación en el que permite la telemanipulación via remota. La telepresencia es un sistema que hoy en día es usado por diversas organizaciones ya que por medio de ella es posible comunicarse y realizar un trabajo en un lugar remoto y a bajo costo. Un ejemplo de ello es la robótica en cirugía, es un sistema asistido por computadora en donde el cirujano manipula a un robot y determina las maniobras que este efectuara sobre un paciente.

Otro es el caso de los robots moviles, el ITESM campus Morelos realizó un laboratorio de robótica movíl el cual tiene el proposito de probar algoritmos de navegación para robots móviles. El ambiente virtual contempla el modelo de los sensores del robot real, sonares, sensores de contacto, cámaras, para reproducir y visualizar los desplazamientos y giros, posteriormente ejecutar los programas en el robot real.

El 25 de junio del 2006 Microsoft presentó una serie de herramientas de software que facilitará a los creadores el diseño de robots y la creación de programas de prueba que funcionen en una amplia variedad de máquinas, desde juguetes hasta aspiradoras usadas en líneas de fabricación industriales. El primer producto del grupo, llamado Microsoft Robotics Studio, está diseñado por aficionados, estudiantes o inventores comerciales, que han tenido que reinventar la rueda cada vez que usaban un hardware diferente para construir el robot.

Microsoft ofrece una muestra técnica que se puede descargar gratuitamente en la página oficial de la compañía, la figura 1.1. muestra el software de Microsoft Robotics Studio[2].

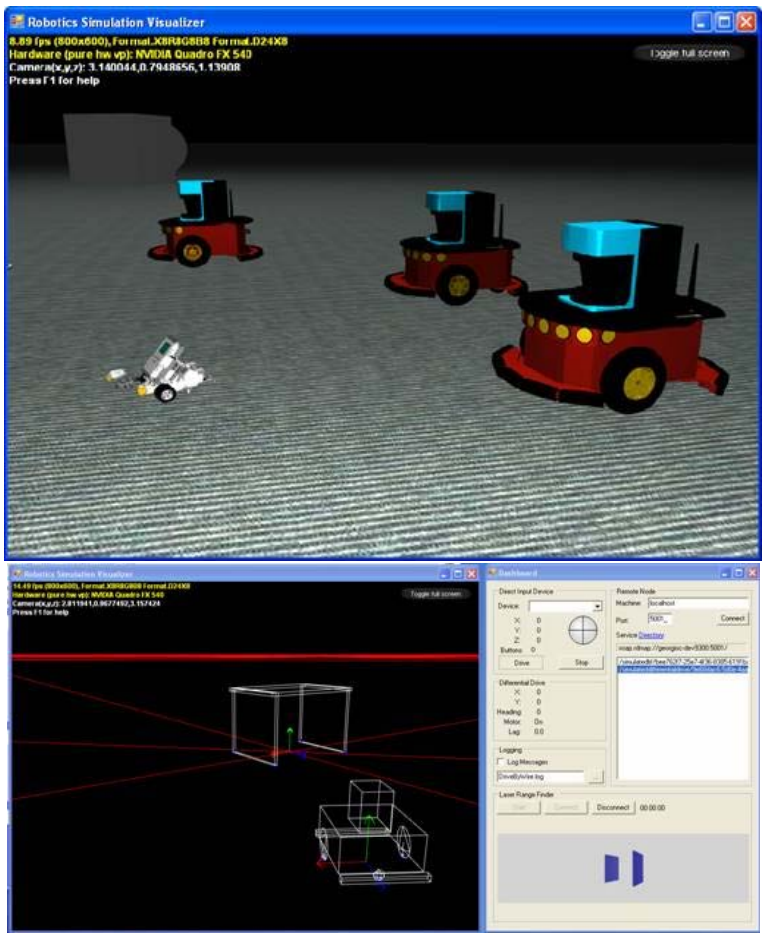


Figura 1.1: Microsoft Robotics Studio [3].

1.7. Organización de la tesis

Esta tesis esta organizada en 5 capítulos y apéndices, los cuales se describen a continuación:

- Capítulo 1: Se presentan brevemente los objetivos generales, los objetivos específicos, la justificación, el planteamineto del problema de esta tesis así como también el estado de arte, haciendo una breve descripción de los mundos virtuales.
- Capítulo 2: Se hace una breve descripción de la realidad virtual, los diferentes dispositivos con los cuales uno puede interactuar dentro del ambiente, así como tambien describimos algunos mundos virtuales aplicados a la enseñanza, la aplicación que tiene la realidad virtual, la descripción de algunos sw en donde se modela y programa los ambientes, el impacto que ha tenido la robótica en la realidad virtual, asi como también la simulación y aplicación de algunos laboratorios virtuales de robótica.
- Capítulo 3: Se hace una pequeña introducción de los robots manipuladores, la obtención de la cadena cinemática, los PDH, las matrices elementales y de tranformación homogenea asi como la cinemática directa e inversa de posición, velocidad y aceleración, la dinámica de Euler Lagrange y su aplicación del control de modos deslizantes obteniendo su comprobación y simulación el MATLAB.
- Capítulo 4: Se describen brevemente las herramientas de edición que se utilizarón para la construcción del escenario virtual así como también la aplicación de los modelos matemáticos y de control para su utilización y evaluación haciendo la simulación en el mundo virtual, para su comprobación.
- Capítulo 5: Se plantean las conclusiones y perspectivas de la tesis.
- Apéndice A: Glosario de terminos: Se describen algunos conceptos generales empleados en esta tesis.
- Apéndice B: Acrónimos: Se describe el lenguaje técnico utilizado.
- Apéndice C: Manual de usuario: En este apartado se proporciona una guía para la interacción con el ambiente virtual.
- Apéndice D: Programación del robot Scorbot ER-VII: Se escribe el codigo del software.
- Apéndice E: Programas en MatLab: Se escribe el codigo de la dinámica y el control de modos deslizantes.
- Apéndice F: Publicaciones: Se muestra el artículo y poster publicados a nivel nacional e internaciomal.
- Apéndice G: Metodología: Se describe la metodologia utilizada como herramienta para el analisis, diseño, desarrollo, implementación y evaluación del software.

Capítulo 2

Robótica y mundos virtuales.

2.1. Introducción

La realidad virtual ha ido evolucionando inconsiderablemente a tal grado que es uno de los temas que ha tenido mayor auge en este siglo, ya que a través de los mundos virtuales podemos interactuar con ellos simulando experiencias, experimentos, conocimientos o simplemente navegaciones, es por ello que en este capítulo se describen algunos conceptos preliminares de realidad virtual, sus diferentes sistemas como son inmersivos, semi-inmersivo y no inmersivo interactuando a través de los guantes, cascos, etc., o simplemente por medio del teclado y/o mouse haciéndonos sentir, ver y escuchar experiencias, experimentos, aplicaciones, etc.

La realidad virtual tiene diferentes aplicaciones como: en las ciencias, medicina, entretenimiento como los videojuegos y en cualquiera de estas el impacto que ha tenido es bastante favorable ya que nos permite no simplemente generar un metodo de enseñanza-aprendizaje, sino de simulaciones, aplicaciones, relagamentos mentales, pruebas, etc. Hay que recordar que la realidad virtual es un tema demasiado amplio y solo se mencionan algunos conceptos que hemos considerado. Para poder crear nuestro ambiente virtual es necesario de la utilización de la programación y modelado de objetos en 3D y para ello podemos utilizar software que cumple con estos requisitos como es 3D MAX, Autocad, maya, etc., y para su manipulación visual c++, Delhi, Visual Basic, etc.
La robótica que en este caso es nuestro tema fuerte también ha tenido grandes avances en mundos virtuales ya que se aplica sobre temas como: regulaciones, controles, tareas programadas como montajes, pintados, etc., para que posteriormente esto sea programado en robots reales o simplemente vía remota.

2.2. Realidad virtual.

La Realidad Virtual(RV) es un medio electrónico gracias al cual podemos explorar y examinar, en tiempo real, un mundo virtual desde cualquier perspectiva además de interactuar a la vez con los distintos elementos que lo integran.

El usuario, en estos escenarios mantiene un control continuo de sus movimientos y sus sentidos, lo cual hace que el mundo virtual 3D parezca virtualmente real.

A diferencia de otros escenarios tradicionales, como la animación, el software de Realidad Virtual calcula, genera y muestra en pantalla una perspectiva actualizada del mundo virtual de múltiples formas y veces por segundo, podemos interactuar y manipular cualquier objeto del mundo virtual, lo que no hacen los software de animación tradicional que trabajan a partir de un cálculo y una reacción previa del modelo [4].

Generalmente, se ve a la realidad virtual como una tecnología nueva, sin embargo, sus bases, es decir sus orígenes se remontan a los años 60's cuando emergen los primeros simuladores de vuelo construidos para la fuerza aérea

estadounidense, donde los estudiantes de piloto aprendían a manejar aviones entrenando en cabinas aéreas montadas en plataformas movibles las cuales subían y bajaban, o movían hacia los lados dependiendo de las acciones que el piloto realizara sobre los controles [5].

Otro precursor dentro del área fue Ivan Sutherland, quién en 1965, publicó un artículo denominado "The Ultimate Display", donde sentó las bases del concepto de realidad virtual, Sutherland estipulaba "La pantalla es una ventana a través de la cual uno ve un mundo virtual. El desafío es hacer que ese mundo se vea real, actúe real, suene real, se sienta real". Posteriormente en 1966, el mismo Sutherland creó un casco visor de realidad virtual al montar tubos de rayos catódicos en un armazón de alambre, este instrumento fue llamado "Espada de Damocles"debido a que el aparato requería de un sistema de apoyo que pendía del techo. En 1972 se desarrolló el primer simulador de vuelo computarizado, el cual fue importante para el despegue de la realidad virtual [5].

La era de la realidad virtual inicia en los 80's, cuando la NASA inició con el sistema de imágenes generadas por computadora, y así a principios de los 80's Jaron Lanier acuñó la expresión Realidad Artificial", también tomó parte en el desarrollo de guantes y visores. En 1985 fue construido el primer sistema práctico de visores estereoscópicos para la NASA [5].

En 1989 el Departamento de defensa de los E.U crea a Simnet (Simulador de Red), una red experimental de estaciones de trabajo basadas en microprocesadores que habilitaban al personal a prácticas de operaciones de combate interactivamente en sistema de entrenamiento de tiempo real. De hecho este sistema se uso para entrenar al ejército en la Guerra del Golfo Pérsico en 1991.

En los últimos años la realidad virtual se ha enfocado a los sistemas no inmersivos donde el usuario tiene la posibilidad de desarrollar aplicaciones de RV de bajo costo y de manera accesible [5].

En 1991 VPL Research Inc., una compañía fundada por el pionero de la realidad virtual Jaron Lanier, creó un modelo computarizado tridimensional del nervio óptico que puede ser estudiado desde cualquier ángulo. En 1994, los radiólogos Ron Kinkinis y Ferenc A. Jolesz y el neurocirujano Peter M. Black, todos del Brighamn and Women's Hospital en Boston, empezaron a realizar neurocirugías en las que una representación en tercera dimensión del cerebro, formada por el ingreso continuo de imágenes de tomas de resonancia magnética, se sobrepuso a la imagen en vivo del video. Con esta tecnología, los cirujanos pueden señalar la ubicación de un tumor cerebral dentro de los 0.5 milímetros precisión sin precedentes. En 1997, los cirujanos Jacques Himpens y Guy Bernard Cadiére del Saint Blasius Hospital, cerca de Bruselas, realizaron la primera operación de vesícula por medio de telecirugía, o cirugía realizada desde una localidad remota. Algunos componentes de la cirugía en la era de la informática lo que muchos llaman cibercirugía ocupan ya un lugar, como es el caso de la inteligencia artificial y la computación de alta resolución [6].

Los sistemas de realidad virtual se clasifican en diversos tipos como se describen acontinuación:

Sistemas inmersivos

Los sistemas inmersivos son aquellos sistemas donde el usuario se siente dentro del mundo virtual que está explorando. Este tipo de sistemas utiliza diferentes dispositivos denominados accesorios, como pueden ser guantes, trajes especiales, visores o cascos, estos últimos le permiten al usuario visualizar los mundos a través de ellos, y precisamente estos son el principal elemento que lo hacen sentirse inmerso dentro de estos mundos. Este tipo de sistemas son ideales para aplicaciones de entrenamiento o capacitación [5].

Sistemas semi-inmersivos

Los sistemas semi-inmersivos o inmersivos de proyección se caracterizan por ser 4 pantallas en forma de cubo (tres pantallas forman las paredes y una el piso), las cuales rodean al observador, el usuario usa lentes y un dispositivo de seguimiento de movimientos de la cabeza, de esta manera al moverse el usuario las proyecciones perspectivas son calculadas por el motor de RV para cada pared y se despliegan en proyectores que están conectados a la computadora. Este tipo de sistemas son usados principalmente para visualizaciones dondese requiere

que el usuario se mantenga en contacto con elementos del mundo real [5].

Sistemas no inmersivos
Los sistemas no inmersivos o de escritorio, son aquellos donde el monitor es la ventana hacia el mundo virtual y la interacción es por medio del teclado, micrófono, mouse o joystick, este tipo de sistemas son idóneas para visualizaciones científicas, también son usadas como medio de entretenimiento (como son los casos de los juegos de arcada) y aunque no ofrecen una total inmersión son una buena alternativa de bajo costo [5].
Aunque no existen reglas fijas y específicas sobre lo que deben o no incorporar los sistemas de realidad virtual, los mejores sistemas utilizan tres elementos básicos: inmersión, navegación y manipulación [5].

Inmersión Se necesita inmersión, propiedad mediante la cual el usuario tiene la sensación de encontrarse dentro de ese mundo tridimensional [4].

Sumergirse en un sistema de realidad virtual es sentir que se está experimentando desde dentro relacionadose con los otros personajes, que ahora tienen tamaño natural y que pueden aparecer por detras, a la izquierda o a la derecha, e incluso por encima de su cabeza [4].

Un sistema típico de realidad virtual consiste en uno o más dispositivos de entrada (sea un joystick, un volante o un arnés corporal), varias formas de salida (sea luz, sonido y presión), y un ordenador que maipule todos estos datos [7].

Sensorama

La máquina Sensorama de Morton Heilig, dispositivo multisensorial diseñado en 1962 para estimular los sentidos de la vista, tacto, oído y olfato del usuario, es un ejemplo de sistema de realidad virtual no computarizado que utiliza las técnicas de inmersión. El usuario se sentaba en un taburete frente a una pantalla de retroproyección y hacia un par de palancas.

El Sensorama conducía al usuario a través de varias escenas, incluyendo un paseo en bicicleta por Brooklyn y sobre las dunas de California. Estimulaba el sentido del tacto a través de las vibraciones de las palancas y de un asiento que se movía bruscamente a medida que la bicicleta se desplazaba. Estimulaba los sentidos de la vista y el oído mostrándole al usuario una película con sonidos sincronizados, y el sentido del olfato lanzando aromas a través de pequeños surtidores que apuntaban hacia la cara del usuario. El paseo terminaba con una visita a un espectáculo sobre la danza del vientre acompañadolo del olor de un perfume barato.

Aunque la experiencia del Sensorama se basaba en una tecnología ánticuada, supo sumergir al usuario en un entorno virtual durante corto tiempo. El aparato de Heilig no se fabricó nunca en una cantidad [4].

Navegación: propiedad que a su vez permite al usuario cambiar su posición de observación [4].

La inmersión lo lleva a un engaño y le hace pensar que se encuentra en una realidad virtual, mientras que la navegación le da oportunidad de explorarla. La navegación es la habilidad de desplazarse dentro del ambiente virtual generado por el ordenador, explorarlo e interactuar con él a voluntad. Naturalmente, esto no significa que realmente vaya a alguna parte; es la sensación de que puede moverse dentro, lo que hace que un entorno sea virtual [7].

Manipulación: La manipulación es simplemente la posibilidad de alcanzar y llamar a una puerta virtual, o disparar sobre un adversario virtual, y hacer que el mundo virtual responda de la manera apropiada. Por ejemplo, podrá ponerse un guante de datos, asir y rotar un objeto dentro del campo de visión virtual. El objeto responderá como si fuese un objeto real que gira en un mundo real. Lo bueno de esto es que sentira como hace

rotar el objeto, por medio de pequeños estimuladores táctiles (llamados tactors) incorporados al guante [7].

2.3. Dispositivos de entrada y/o salida

Un sistema de realidad virtual necesita una correspondencia entre la entrada (los datos suministrados al ordenador) y la salida (los datos que salen de él). En la mayoría de los casos, el tipo de datos que alimentan un sistema de realidad virtual consite en señales generadas por algún tipo de dispositivos de entrada, raton, teclado un guante electrónico Figura 2.1, para la manipulación en el medio ambiente artificial además de proporcionar la sensación de tacto, un casco estereoscópico, para proyectar secuencias estereoscópicas, (sonido y visión) figura 2.2 para la determinación de la posición y del movimiento de la cabeza del usuario, banda transportadora y timón, para dar la sensación de estar caminando y navegar y la computadora como medio principal no unido al cuerpo, con capacidad para simular los procesos asociados con un sistema de realidad virtual [7].

Figura 2.1: Guante electrónico [8].

Figura 2.2: Cascos estereoscópicos [9], [8].

2.4. Mundos virtuales

Los mundos virtuales son lugares donde experimentamos nuevas realidades. Contienen todos los objetos que podemos ver y manejar. Nos permiten, además experimentar cosas que no son posibles en el mundo real.

Para poder experimentar la realidad virtual es necesario el punto de vista de un usuario en primera persona que debe realizar movimientos completos a voluntad en tiempo real, la capacidad de visualizar y modificar el entorno virtual en tiempo real [10].

2.4.1. Realidad Virtual Distribuida para soportar la Educación a Distancia en Colombia

En este software se desarrolla en tres etapas. En la primera, llamada AVALÓN 1, se construyó una interfaz para comunicación usando texto. En la segunda, AVALÓN 2, estan construyendo una aplicación de Realidad Virtual que permite que dos usuarios se encuentren en un mundo virtual para realizar una tutoría o para recorrer conjuntamente un mundo virtual. En la tercera, AVALÓN 3, se podrán encontrar más de dos personas.

En la figura 2.3 se aprecia el aspecto actual del mundo virtual, incluyendo dos personas. Las personas se encuen-

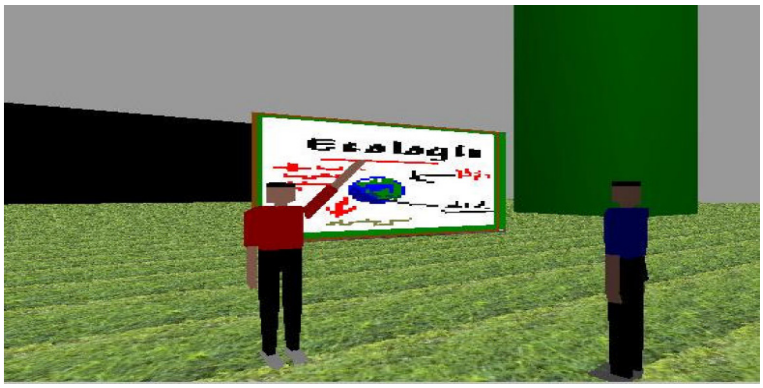


Figura 2.3: Realidad Virtual Distribuida para soportar la Educación a Distancia [11].

tran en un mundo virtual que simula un aula. Disponen de un tablero para hacer dibujos que apoyen sus ideas y un proyector de acetatos para compartir imágenes disponibles. La componente de visualización está construida utilizando las librerías gráficas de OpenGL para Win32, lo cual nos permite crear un código más independiente de un fabricante determinado, y aprovechar la rapidez de despliegue de las tarjetas aceleradoras 3D. El navegador actúa sobre un conjunto jerárquico de objetos, definido en un formato propio del proyecto, con el cual se puede especificar las características del ambiente que se supone será el Aula Virtual. Ya que no es un formato muy complicado, el usuario del navegador podría perfectamente cambiar el entorno y personalizar así su ambiente virtual [11].

2.4.2. La realidad virtual como herramienta en la enseñanza-aprendizaje de la anatomía humana para el nivel EGB II

En este software se desarrolló un entorno que expone la importancia y pertinencia que tiene desarrollar y explorar el campo de la RV como área de investigación y estudio de la anatomía humana en la educación. El software El cuerpo humano: una máquina perfecta tiene la capacidad de representar los siguientes sistemas funcionales del ser humano: sistema digestivo, sistema respiratorio, sistema urinario, sistema circulatorio.

Las escenas gráficas se realizaron en el programa de diseño gráfico 3D Studio Max. Se procedió a efectuar cada sistema de manera independiente y a su vez, cada órgano de manera diferente. Cabe aclarar que los modelos de los sistemas del cuerpo humano fueron integrados a un entorno de aprendizaje web con el propósito de incorporar también en él los contenidos teóricos.

El modelo virtual del Sistema digestivo muestra los siguientes órganos: la boca, faringe, esófago, vesícula biliar, hígado, duodeno, intestino grueso, intestino delgado, apéndice, ano, recto, páncreas y estómago figura 2.4. Se agregaron las cámaras que permiten obtener diferentes vistas del sistema[12].

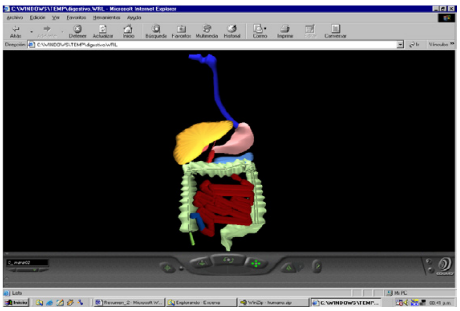


Figura 2.4: Sistema digestivo [12].

2.4.3. Desarrollo de una ambiente virtual para interactuar con el guante P5.

Este software presenta la interacción entre un esqueleto virtual que representa la mano derecha de un humano y el guante de datos P5, puesto que cuenta con una interfaz donde se aplica un grado de inmersión mayor al ambiente virtual en VRML, haciendo uso de la tecnología de un dispositivo de entrada de datos como lo es el guante de datos P5, proporcionando una forma más intuitiva de navegar en los escenarios virtuales, proporcionando una mayor sensación de inmersión hacia el usuario, haciendo que involucre más sentidos al interactuar con el ambiente, permitiéndole retener ideas y el aprendizaje de cualquier actividad es retenido por un tiempo prolongado. El guante P5 es un dispositivo que cuenta con una serie de características interesantes y útiles para navegar en un ambiente virtual, sin embargo, no permite retroalimentación del ambiente al usuario, por lo que este no puede tener una percepción de los objetos que se encuentran en el escenario ver figura 2.5 [13].

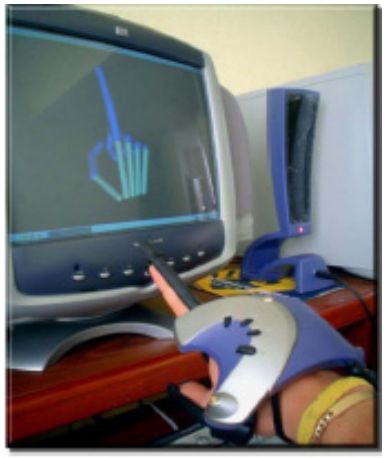


Figura 2.5: Interacción entre el guante P5 y el modelo virtual [13].

2.4.4. Desarrollo de Software Educativo Caso de estudio: Matemáticas de Sexto de Primaria Área de Conocimiento: Computación Educativa

Es un software educativo multimedia que integra diversas herramientas para edición de texto, imágenes, sonido, objetos tridimensionales y video, con el fin de producir un material pedagógico autodidacta para facilitar el aprendizaje de matemáticas de sexto año de primaria y al mismo tiempo que sirva de apoyo a profesores que impartan este mismo curso. Este sistema se ha desarrollado en base a la Metodología para el desarrollo de aplicaciones educativas en ambientes multimedios, figura 2.6 [14].

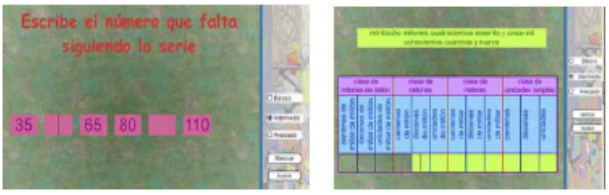


Figura 2.6: Software educativo [14].

2.5. Aplicación de la realidad virtual

La Realidad Virtual es una tecnología que puede ser aplicada en cualquier campo, como la educación, gestión, telecomunicaciones, juegos, entrenamiento militar, procesos industriales, medicina, trabajo a distancia, consulta de información, marketing, turismo, etc.

La Realidad Virtual es una tecnología especialmente adecuada para la enseñanza, debido a su facilidad para captar la atención de los estudiantes mediante su inmersión en mundos virtuales relacionados con las diferentes ramas del saber, lo cual puede ayudar en el aprendizaje de los contenidos de cualquier materia, como se muestra a continuación:

2.5.1. Física

Visualización de fluídos de partículas.

Una aplicación en el área de visualización es el fluído de partículas figura 2.7. Existen proyectos que modelan este tipo de fenómenos, donde el propósito principal es el fácil análisis de una gran cantidad de datos que facilitan el estudio de los modelos. Se cuenta con una herramienta auxiliar que permite visualizar modelos complicados de interpretar si solo se analizan tal cual. Este proyecto corresponde a un tipo de realidad no inmersiva [8].

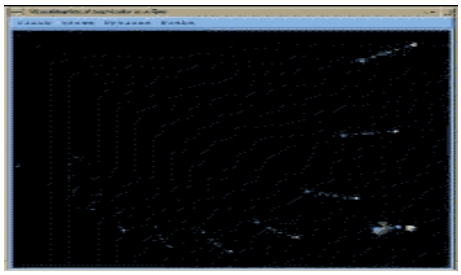


Figura 2.7: Visualización de fluídos de partículas [8].

2.5.2. Psicología

Las aplicaciones desarrolladas por PREVI consisten, genéricamente, en un software (formato PC CD-ROM) de Realidad Virtual, cuyos escenarios virtuales se han diseñado en función del trastorno mental concreto que se pretende abordar, y que se han demostrado eficaces en su evaluación y tratamiento. Por tanto, los programas se diversifican en varias aplicaciones en función de los trastornos o problemas a los que se vayan dirigidos. Todos reúnen las siguientes características [9]:

Clínicamente validados. Alta calidad en los acabados de las imágenes, buscando un máximo realismo. Un desarrollo totalmente exclusivo y pionero. Previ cuenta actualmente con las siguientes aplicaciones:

- Virtual Open Out (claustrofobia)
- Virtual Flight (miedo a volar)
- Virtual & Body (trastornos alimentarios)
- Virtual Going Out (agorafobia)
- Virtual Pathological Gambling (juego patológico)
- Virtual Acrophobia

El software se acompaña de una GUÍA CLÍNICA (manual del terapeuta). En este manual se expone la base lógica que se explica al paciente respecto al tratamiento con RV y el modo en que el terapeuta tiene que ir actuando paso a paso a lo largo de las sesiones. Un área de aplicación de la RV en el campo de la salud es su uso como herramienta en el tratamiento de diversos problemas psicológicos. La idea de usar la RV para el tratamiento de estos problemas, se concibió por primera vez en noviembre de 1992 en el Human-Computer Interaction Group de la Clark Atlanta University y, desde entonces, las aplicaciones se han ido desarrollado rápidamente [9].

2.5.3. Ingeniería

Dentro de las áreas de ingeniería hay proyectos de manipulación remota como lo son la manipulación de robots, o procesos de ensamblado, también existen áreas dedicadas al desarrollo de prototipos virtuales. Todas estas aplicaciones facilitan la automatización dentro de diferentes áreas, como las presentadas acontinuación [8].

2.5.4. Proceso de ensamblado

Cuando se tiene un proceso de ensamblado de algún producto se presentan distintos acontecimientos como puede ser las deformaciones de plástico, fricción externa, fenómeno termal, absorción, y factores como el desgaste de herramientas, ocasionando errores de dimensión y forma. Si se tiene información adicional sobre el efecto de los parámetros antes mencionados sobre la variación en los valores de tolerancia y dimensión se puede desarrollar mecanismos para el ensamblado automático. Usando un modelo de elementos finitos se puede visualizar las fuerzas que actuan en el proceso de manufactura y la deformación del equipo bajo la acción de estos fuerzas. Si se tiene un ingeniero en diseño y manufactura que pueda observar el ensamblado de una de las partes por medio de la computadora y dispositivos especiales, puede sugerir cambios en la tolerancia de los valores basandose en las condiciones de las máquinas, herramientas, fisuras y requerimientos de diseño. Un tipo de aplicación como ésta puede permitir obtener un configuración de ensamblado óptimo para satisfacer los requerimientos funcionales, por lo que, es un tipo de herramienta efectiva para el proceso de toma de decisiones. Este tipo de proyectos son totalmente inmersivos [8].

2.5.5. Ciencias de la tierra

Visualización de fenómenos volcánicos sin duda, el riesgo de potenciales erupciones volcánicas es un problema que se tiene en todo el mundo. Las simulaciones de fenómenos volcánicos permiten analizar la pérdida de vida y la destrucción de la infraestructura. Los modelos de flujos permiten estimar los movimientos de materiales volcánicos dentro y sobre la superficie. Este tipo de aplicaciones permite el entendimiento de los peligros de estos fenómenos antes de que sucedan, además, del desarrollo de mapas de riesgo, asistencia en crisis y reconstrucción post-crisis. La Universidad de Buffalo, desarrolló un sistema de visualización de estos fenómenos, el cual

es utilizado para el análisis de varios tipos de flujos que van desde lava de movimiento lento y flujos saturados, que le permitirá a oficiales públicos, científicos y la población en general entender el efecto de varios fenómenos volcánicos y sus áreas locales y diseñar planes apropiados de migración. Cabe destacar que se han escogido tres volcanes en México para desarrollar y probar los modelos científicos de este estudio, estos son el Popocatepetl, Colima y Pico de Orizaba interactuando con vulcanólogos de la UNAM. Este tipo de aplicación corresponde a la categoría de realidad virtual no inmersiva [8].

2.5.6. Medicina

En la Medicina, el empleo de esta técnicas es de gran valor y utilidad en la cirugía general y específica, donde se requiere un alto grado de destreza y capacidad de reconocimiento de los órganos apropiados [4].

2.5.7. Simulación de cirugías virtuales

Una aplicación más de la realidad virtual en la medicina son los proyectos de cirugías virtuales. La idea general es proveer al cirujano con una herramienta que le permita experimentar diferentes procedimientos quirúrgicos en un ambiente artificial. Las aplicaciones de éste tipo se puede utilizar tambien para el entrenamiento de estudiantes de medicina, donde ellos pueden realizar operaciones en modelos virtuales permitiendoles observar los resultados. Este tipo de simulaciones tridimensionales todavía se pueden perfeccionar, sin embargo, existen modelos que ya se están implementando actualmente [8].

2.5.8. Arquitectura

En arquitectura, para el diseño y recorrido del modelo arquitectónico, que permite visualizar la proporción entre los elementos de construcción, la estética y plástica de los colores y el recorrido interno y externo de la obra, aún antes de su construcción [4].

La manera en que los arquitectos comunican sus ideas la mayor parte de tiempo es en forma visual, el utilizar alguna forma de visualización facilita la comprensión de información compleja y facilita la comunicación. Hoy en día, cada vez son más los arquitectos que utilizan a la realidad virtual como una herramienta más para participar a los demás de sus ideas y trabajos. Algunos de los enfoques más comunes que los arquitectos dan al uso de realidad virtual es en el modelado virtual de sus diseños de casas y edificios, donde además de hacer los diseños tradicionales como planos y maquetas elaboran un modelo tridimensional interactivo, donde sus clientes pueden contemplar de una manera más real"los diseños o inclusive adentrarse en estos edificios o casas y recorrerlos libremente, teniendo así una visión mas clara de las ideas que se tratan de expresar.

Además, existe un vínculo entre la arquitectura como tal y diseñadores urbanos, donde no solo se realizan los diseños de una casa o edificio, sino de un planeamiento más amplio como es el diseño de una ciudad o una parte de ella. En este tipo de proyectos, la visualización va un poco mas lejos, se trata de planear con anticipación el crecimiento de una ciudad o una parte de ella, creando no solo edificios o avenidas con una belleza por si solas sino en armonía con la infraestructura ya existente [8].

2.5.9. Educación

La educación y capacitación de personas, representa mayores ventajas para el desarrollo con la realidad virtual, la realización de actividades que requieren coordinación motora pueden beneficiarse especialmente, ya que es posible evaluar los movimientos dentro de las trayectorias preescritas y si se ejercen una presión o fuerza apropiada [4].

Todo esto puede aplicarse en diferentes formas: para aprender a tocar instrumentos musicales, manejar y experimentar el uso de fuentes informativas y acontecimientos trascendentales, conducir automóviles, soldar componentes electrónicos, escribir a máquina o jugar tenis, entre otras cosas [4].

2.6. Realidad virtual en Aplicaciones de problemas psicologicos

Por lo que se refiere la realidad virtual en sus aplicaciones en Psicología, cabe destacar su utilidad en la evaluación. Cuando queremos saber qué es lo que le ocurre a una persona con un problema determinado, el camino más directo es preguntarle a la persona qué le pasa. La RV no sustituye en absoluto este modo de evaluar problemas, pero amplía el uso de esta estrategia, ya que además de que en nuestra consulta le preguntemos a la persona por lo que le pasa, podemos recrear un ambiente en el que lo observemos directamente.

Por lo que respecta a los tratamientos, en la RV confluyen muchos aspectos que se han considerados centrales para la buena marcha del proceso terapéutico. Entre las principales ventajas que ofrece la utilización de la RV en los tratamientos psicológicos se encuentran las siguientes:

1. El ambiente virtual es seguro, ofrece un ambiente terapéutico especial y protegido que permite al usuario explorar, experimentar y, en definitiva, actuar y conocer situaciones que siempre ha considerado amenazadoras. Poco a poco, a partir del conocimiento y dominio de las interacciones con las diversas partes del mundo virtual, el usuario podrá enfrentarse al mundo real.
2. Permite al usuario adoptar distintos papeles y así explorar, analizar, probar y con ello encontrar un modo satisfactorio de funcionar.
3. La RV asegura la confidencialidad y privacidad. Cualquier problema o situación embarazosa que el paciente teme que se pudiera producir cuando se enfrenta a su problema, ocurre en la privacidad de la consulta.

2.6.1. Acrofobia

La acrofobia es un miedo intenso a los lugares elevados. se clasifica dentro de los Trastornos de ansiedad como una Fobia específica, es decir, un miedo intenso y específico a situaciones y objetos concretos. Dado que la acrofobia es un miedo a los espacios elevados, las situaciones que se evitan son terrazas, escaleras, ascensores, edificios altos, puentes, aviones, etc. En general, las personas con acrofobia no suelen tener miedo sólo a una de estas situaciones, sino que suelen sentir temor en cualquier situación que implique altura. La persona no teme la situación en sí misma, sino las posibles consecuencias negativas de estar en ese sitio. El miedo más frecuente es a caerse. En los últimos años, se esté observando y reconociendo que esta fobia puede llegar a ser muy incapacitante.

Se emula diferentes escenarios significativos para personas con acrofobia. La aplicación consiste en diversos entornos virtuales que suelen suscitar ansiedad a una persona con miedo a las alturas (asomarse desde diferentes alturas en un edificio; montar en una noria, etc.). Cada una de estas situaciones cuenta con distintos estímulos que facilitan la exposición del paciente a aquello que teme, y hacerlo, además, reiteradamente y a su propio ritmo. Los entornos virtuales están especialmente diseñados para facilitar y potenciar el procesamiento emocional y el cambio de significado de situaciones que el paciente siempre ha considerado como peligrosas, ver figura 2.8 [9].

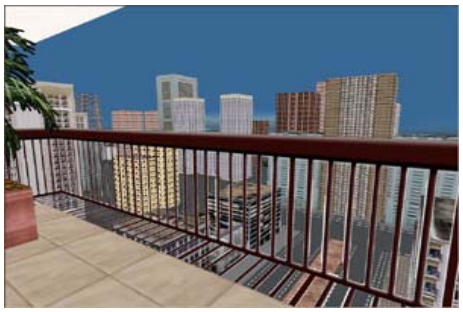


Figura 2.8: Acrofobia [9].

2.6.2. Agorafobia

El trastorno de pánico consiste en crisis de ansiedad de gran intensidad, en las que la persona experimenta una serie de sensaciones (taquicardia, dificultades para respirar, opresión en el pecho, etc.). Estas crisis se producen de forma inesperada y, en la mayoría de los casos, la persona siente un miedo repentino sin saber porqué, lo que contribuye a que se asuste más. Se emula diferentes escenarios significativos para personas con Agorafobia. Se trata de un entorno virtual que contiene situaciones significativas para este problema, como el hecho de que la persona se plantee la posibilidad de salir de casa; de estar en recintos pequeños con o sin gente; utilizar transportes públicos; alejarse de casa, y realizar diversas compras en un gran almacén. Cada una de estas situaciones se puede graduar estableciéndose así, distintos niveles de dificultad para cada persona, y todo ello sin salir de la consulta del profesional, ver figura 2.9 [9].

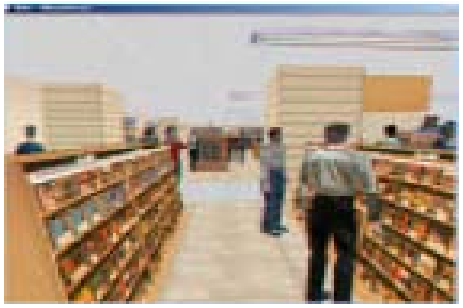


Figura 2.9: Agorafobia [9].

2.6.3. Alteraciones de la imagen corporal en los transtornos alimentarios

Emula diferentes escenarios significativos para personas con trastornos alimentarios que presentan alteraciones en su imagen corporal. Se trata de un entorno virtual que posibilita a los pacientes confrontar su imagen corporal de forma visual e interactiva con la realidad. Se trata de una herramienta versátil que completa el tratamiento del profesional con la posibilidad de manejar ingestas virtuales de varios alimentos, evaluar el impacto que ello tiene en la imagen corporal de la persona, y la modificación interactiva de sus diversos componentes, ver figura 2.10 [9].

Figura 2.10: Transtornos alimenticios [9].

2.6.4. Fobia a volar

Emula diferentes escenarios significativos para personas con miedo a volar. Se trata de un entorno virtual diseñado para ayudar a su tratamiento que permite secuencializar distintos grados de amenaza e incidentes de vuelo. Permite realizar exposiciones individualizadas totalmente ajustadas a las situaciones de malestar, incluyendo la preparación del viaje, paneles de información dinámicos, conversaciones en el aeropuerto, tormentas, despegues en diversas condiciones climáticas, vuelos nocturnos, turbulencias, etc., que el clínico puede controlar y lograr un alto grado de realismo, ver figura 2.11 [9].

Figura 2.11: Fobia a volar [9].

2.6.5. Claustrofobia

Se trata de un entorno virtual que permite que la persona se sienta inmersa en situaciones que constituyen una fuente de ansiedad, como habitaciones con puertas y ventanas bloqueables que pueden quedar totalmente a oscuras; ascensores con vibraciones, fallos mecánicos, cortes de iluminación y cabina de tamaño ajustable, y espacios con paredes móviles que permiten ajustar el espacio de forma personalizada a la ansiedad del paciente, ver figura 2.12 [9].

Figura 2.12: Claustrofobia [9].

2.7. Realidad virtual en el entretenimiento

Existe mayor aplicación de la realidad virtual principalmente en los juegos electrónicos de diversión ya que son un medio de distracción, en donde se tiene la posibilidad de experimentar e interactuar con distintos ambientes ofreciendo una enorme fascinación para la mayoría de las personas gustosas de esta aplicación [8].

2.7.1. Juegos patológicos

Incluye distintos entornos virtuales significativos para este trastorno de falta de control del impulso de jugar. La aplicación consiste en diversos entornos de juego (tragaperras, casino, etc.) con diferentes estímulos discriminativos que hacen posible y facilitan la exposición y prevención de respuestas del paciente. De esta manera, el paciente puede ir exponiéndose a su propio ritmo a las situaciones que le producen el impulso de jugar [9].

2.7.2. Videojuegos

En la actualidad, los videojuegos pueden contener tablas de gráficos capaces de representar más de 180.000 tipos de formas gráficas por segundo. Gracias a la tecnología de Realidad Virtual, el juego responde en tiempo real a las reacciones del usuario.

Las primeras versiones de juegos de computadora en dos dimensiones se han ido mejorando hacia versiones de cabina en tres dimensiones con tecnología de VR y cada vez más interactivas.

La mayoría de los aficionados prefieren utilizar equipos de cabeza y visualizaciones tridimensionales, por medios de pequeños CRTSs, que producen un óptica especial de la trama del juego y un efecto estéreo de sonido.

En el año 1985, los desarrolladores de videojuegos empiezan a programar en tres dimensiones, aunque con diseños muy sencillos y cajas sin texturas. Después, este motor evoluciona y nace el primer programa de creación de entornos para plataforma PC. Pero no será hasta 1992 cuando se popularice la Realidad Virtual, con un motor render más avanzado, que funcionaba en los procesadores 386. El usuario ya podía manejar una cámara en primera persona, que daba la sensación de estar en dicho escenario.

Con la evolución del Pentium, se va consiguiendo más realismo y se logra una consola destinada a ejecutar potentes motores para los videojuegos. Con los sistemas en tres dimensiones, se posibilita la opción multijugador, en el que varias personas pueden encontrarse y hablar dentro del mismo mundo virtual.

Los juegos virtuales sumergen al participante en mundos imaginarios por medio de varios dispositivos para conseguir diferentes tipos de interacción, como un casco de VR, guante virtual, micrófonos, mouse 3D, volante y pedales o plataformas móviles. Después, los desarrolladores de juegos se vuelven más ambiciosos. Persiguen conectar a los usuarios tridimensionales a través de la Red y surge una nueva dimensión: la Realidad Virtual en Internet [15].

Virtuality, BattleTech y el Cybertron son las atracciones más conocidas de Realidad Virtual en lugares de EE.UU. Virtuality de W-Industries está instalado en alrededor de 20 lugares de EE.UU., y hay entre 100 y 150 lugares más donde está proyectado su instalación.

Se trata de un producto británico, que representa un escenario de resolución de un puzzle, en el que el jugador vuela a través de un territorio de fantasía, esquivando y disparando a los oponentes que percibe.

Se están planeando parques de atracciones que incorporan las tecnologías de Realidad Virtual para representar mundos virtuales interactivos que los usuarios puedan experimentar, convirtiéndose en los personajes que los usuarios deseen ver.

Las sillas de juego son cómodas sillas equipadas con aparatos montados sobre la cabeza y serán las principales atracciones en muchos centros de entretenimiento por vídeo. Estas pueden ser reclinables, en las que la persona se sienta con los pies ligeramente elevados y gira un BOOM enfrente de su cabeza para visualizar.

Los sistemas de sillas de juego para dos personas cuestan alrededor de 100.000 dólares. Un sistema para ocho personas cuesta alrededor de 300.000 dólares. Muchos parques de atracciones y centros comerciales de EE.UU. ya cuentan con simuladores, que combinan las atracciones con efectos visuales y sonoros [15].

2.8. Programación y modelados de los mundos virtuales

En la actualidad para poder desarrollar un ambiente virtual se tiene una gran variedad de lenguajes de programación que son en su mayoria multi-plataformas y graficadores de 3D los cuales son usados de acuerdo a las características de la aplicación a desarrollar en las cuales muestra como resultado que el navegador en realidad sienta, escuche, vea e interactue con el ambiente, entre ellos tenemos lo siguientes:

2.8.1. VRML

En Internet, nace VRML (Virtual Reality Modeling Language), que es un estándar para la creación de objetos en 3D y que permite combinarlos en escenas y mundos virtuales figura 2.13. Se utiliza para representar simulaciones interactivas, que incorporan animaciones, contenidos multimedia y la participación de múltiples usuarios en tiempo real. A estos mundos se puede acceder a través del Web, mediante un navegador de tipo Netscape Navigator o Microsoft Explorer dotado de un plug-in o mediante un navegador que sólo pueda visualizar VRML [15].

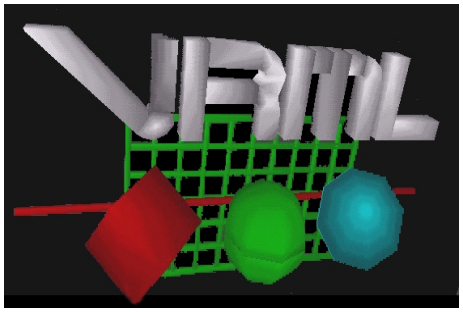


Figura 2.13: VRML[16].

2.8.2. OpenGL

OpenGL provee de un interfaz grafica de hardware. Es un software de librerías de renderizado y modelado, disponible sobre todas las plataformas principales, con el amplio apoyo de hardware. Es diseñado para el empleo en cualquier uso de gráficos, como modelado de juegos CAD.

OpenGL puede ser utilizado a bajo, OpenGL es solamente una biblioteca de gráficos; a diferencia de DirectX, esto no incluye el apoyo al sonido, introduce, la interconexión, o algo más no directamente relacionado con la gráfica.

Las GLUT son las liblerías de soporte para cualquier plataforma, ya que OpenGL no tiene eventos sobre la ventana, menus o salidas. Esto es donde las GLUT entran. Esto proporciona la funcionalidad básica en todas aquellas áreas, dejando la plataforma independiente, de modo que usted fácilmente pueda mover la aplicación de las GLUT, por ejemplo Windows, UNIX. Las Glut son faciles de usarse y aprender [17].

2.8.3. Delphi

Delphi, combina la idea del desarrollo visual de aplicaciones con pascal, un lenguaje de programación extremadamente versatil y potente. El resultado de esta relación se puede clasificar perfectamente de extraordinario. Delphi ofrece al usuario poco iniciado elementos de programa orientado a objetos y una programación visual por componentes, de tal forma que pasado un cierto tiempo de aclimatación cualquier usuario podrá crear comodamente aplicaciones windows de calidad. A los programadores más experimentados la combinación de programación visual y tradicional les permite trabajar de forma eficiente en el desarrollo de aplicaciones complejas y profesionales en todos los campos.

Junto al desarrollo de aplicaciones windows convencionales, Delphi ofrece un amplio apoyo en la programación de Base de Datos y programas de internet, así como la creación de programas de consola no gráficos para MS-Dos, figura 2.14 [18].

Figura 2.14: Delphi 5.0 [19].

2.8.4. Java 3D

El sistema de Java 3D es una colección bien desarrollada de clases y puede ser descargado y ser instalado de la misma manera que el JDK estándar. Java 3D fue diseñada para permitir que un observador humano se mueva a través de un universo virtual y ver el escenario desde varias posiciones. Java 3D no solo se utiliza para la exhibición estándar, estática que aparece en un monitor o una impresora, pero también para producir los gráficos que se pueden utilizar con los cascos virtuales de la realidad y los juegos tridimensionales. El universo entero de objetos y del observador gráficos se reúne en una estructura de datos conocida como gráfico de la escena, la figura 2.15 muestra el entono de objetos en 3D [20].

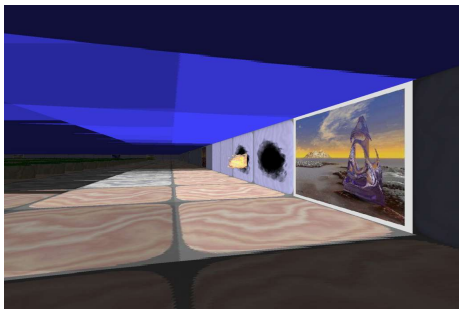


Figura 2.15: Java3D[21].

2.8.5. Maya

Maya es un programa, usado para el modelado, animación y renderizado de escenas 3D. Las escenas 3D creadas con Maya pueden ser aplicadas en películas, televisión, publicidad, videojuegos, visualización de productos y en la web. Con maya pueden crearse y animarse escenas 3D y mostrarlas como imagenes fijas o como secuencias de animación [22].

2.8.6. 3D Studio Max

3ds max 6© crea animaciones de alta calidad, ambientes, y efectos visuales, incluyendo como usar plug-ins, todo en una escena animada dada, también como hacen películas cortas animadas, y películas animadas de cuerpo entero. 3ds max 6© crea efectos visuales y de iluminación [23].

2.8.7. Autocad

AutoCAD nació hace más de dos décadas, a la vez cuando la mayoría de la gente pensó que los ordenadores personales no eran capaces de tareas industrialstrength como el CAD. AutoCAD fue nacido hace más de una década, como mucha alternativa de necesidad, pero no todo, es utilizado para dibujar AutoCAD es una evolución de AutoCAD LT, que fue mas complejo, Aotocad es una herramienta la puedes dibujar lineas, circulos, cuadros, etc., o algo mas complejo dibujos, o cualquier cosa, basandose en sus herramientas para tener un mejor efecto, en la figura 2.16 se visualiza el software [24].

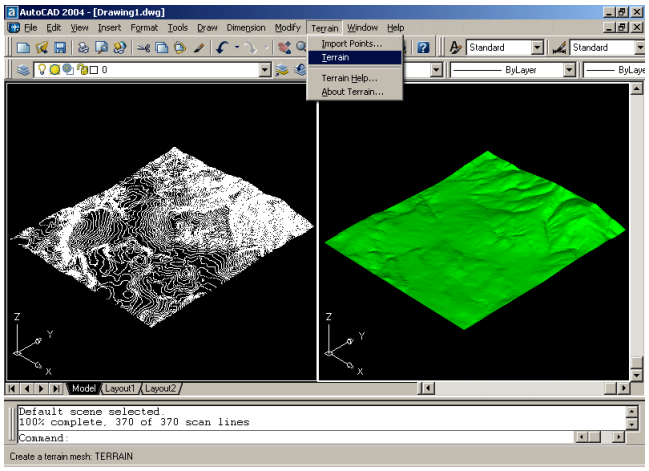


Figura 2.16: Autocad[25].

2.9. Robótica y realidad virtual

En la actualidad, podemos decir que los robots han llegado a ser una parte integral y necesaria de un importante número de sistemas automatizados en la industria. Estos son usados en una variedad de aplicaciones, tales como estaciones de ensamble, estaciones de pintado, estaciones de soldaje, estaciones de atención de máquinas CNC, entre otras. El reemplazo del hombre en estas actividades evita las tareas fatigosas, rutinarias, peligrosas o excesivas para las capacidades musculares del ser humano.

El uso de la Robótica en la industria trae como consecuencias más evidentes el aumento de la productividad y la mejora de la calidad de los productos fabricados. Esto se debe a las características fundamentales de estos dispositivos que son la repetibilidad y la precisión. Es por ello que con todas estas caracteristicas antes mencionadas la simulación de robots en ambientes virtuales son de gran importancia ya que para tareas programadas se requieren de gran precisión, ademas de que se puede interactuar de manera remota, algunas de estas características en aplicaciones de la robótica y la realidad virtual se muestran a continuación:

2.9.1. Robótica en cirugía

La cirugía de telepresencia también llamada cirugía robótica o cirugía asistida por computadoras es un sistema interactivo computarizado figura 2.17, tan veloz e intuitivo, que la computadora desaparece de la mente del cirujano, dejando como real el entorno generado por el sistema.

A través de la realidad virtual, el cirujano determina las maniobras que el robot ejecutará en el paciente. La consola de mando donde trabaja el cirujano puede situarse en el mismo quirófano, y eventualmente en otro lugar de la misma ciudad o incluso en otro país.

La cirugía robótica o cirugía de telepresencia está basada en dos conceptos fundamentales que son: Realidad virtual y Cibernética.

Se habla de realidad virtual porque se logran los efectos de inmersión en tercera dimensión, navegación, interacción y simulación, sólo que ésta es sustituida por tiempo real, es decir lo que se ve en tercera dimensión en el monitor, es real y lo que se toca a través del robot, también es real.

En cuanto a cibernética, es la rama de la informática que digitaliza el movimiento y se divide en tres áreas importantes que son: autómata, biónica y robótica. Esta última estudia el desarrollo de robots que son mecanismos articulados programados, con partes mecánicas, motores, grados de libertad, cámaras, sensores, transductores, almacenamiento de información, programas especializados para procesamiento de datos, optimización de funciones e interfaces conectados a elementos ejecutores de tareas específicas.

En la cirugía de telepresencia se utiliza un robot esclavo que no puede hacer ningún tipo de movimiento sin las órdenes del cirujano; es decir que es absolutamente dependiente del juicio, de los conocimientos y de la habilidad del médico. Consta de una estructura que semeja la anatomía de los brazos humanos, capaz de imitar los movimientos de diversas articulaciones como las del hombro, codo, muñeca y manos [26].

Figura 2.17: Robótica en cirugía [27].

2.9.2. Manipulación remota de robots

Es claro que los robots dan una gran aportación a los procesos de ensamblado de la industria. El agregar la característica de manipulación desde un lugar remoto abre las posibilidades para el mejoramiento de este tipo de procesos, puesto que se puede tener un robot que realice proceso definidos y donde su manipulación sea dada desde un lugar distinto de donde se encuentra físicamente. Las aplicaciones forman parte un nuevo enfoque del manejo

de procesos y refleja las nuevas tendencias actuales, donde los lugares se vuelven más cercanos y la distancia deja de ser un factor a considerar. Éste proyecto es un tipo de realidad inmersiva [8].

2.10. Laboratorios virtuales de robótica.

Existe hoy en dia una gran infinidad de laboratorios virtuales de robótica, estos en su mayoria para aplicaciones de enseñanza-aprendizaje ya que por medio de estos se puede tener una mejor visión de la simulación. Algunos de ellos se muestran a continuación:

2.10.1. Laboratorio virtual de robótica móvil

Este trabajo fue realizado por:

- Investigadores: L. Enrique Sucar Succar, Víctor Hugo Zárate Silva, Eduardo Morales Manzanares, Rogerio Enríquez Caldera, Roberto Valdivia Bautelspacher.
- Asistentes de Investigación: Marco A. López Trinidad, Miguel Salas Zuñiga.
- Estudiantes: Alma Rocío Ligonio Velasco, Pedro Salguero Centeno, Rogelio Ferreira Escutia, Giovani Gómez Estrada, Leonardo Romero, José Rodolfo Martínez y Cárdenas.

En el Instituto Tecnológico y de Estudios Superiores de Monterrey Campus Morelos. En este trabajo se presenta el diseño de un ambiente virtual para la programación de robots móviles, en parti-cular, el ambiente es una representación tridimensional del Laboratorio de Sistemas Inteligentes y del robot Nomad 200 del ITESM Campus Morelos. El modelo del laboratorio y del robot fue desarrollado a partir del "Modelador **3D"** construido por el grupo del Dr. José Luis Gordillo del ITESM Campus Monterrey.

El Modelador es un sistema basado en Java 3D que permite construir rápidamente mundos virtuales a partir de formas básicas como son: cubos, esferas, cilindros y toroides. Para la animación de los elementos en el ambiente, se utiliza un conjunto de instrucciones de alto nivel o "Scripts".

El ambiente virtual desarrollado, tiene como propósito probar algoritmos de navegación para robots móviles. La representación virtual contempla el modelo de los sensores de un robot real; sonares, sensores de contacto, cámara, etc. De esta forma es posible reproducir y visualizar los movimientos, desplazamientos, giros, etc.; realizados por el robot móvil virtual, para posteriormente ejecutar los programas en un robot real, figura 2.18 [28].

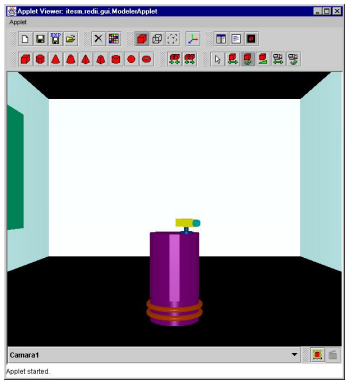


Figura 2.18: Universo Virtual del Laboratorio y del robot simulado [28].

2.10.2. RV-M1

Basado en el Robot Industrial Mitsubishi RV-M1, incorpora todas las funciones de la máquina real, Display doble: muestra simultáneamente la Teach Box y la vista del Robot, Programación de la teach box a través del ratón figura 2.19, La programación del robot incluye operaciones anidadas, jogging y de control de la mano, Simulación en tiempo real, Posibilidad de comunicación y comando de un RV-M1 real,RV-M1 levanta y dejar objetos, utilizando los comandos básicos de movimiento, tiene: manipulación y posicionamiento, cuidadoso de objetos, incluyendo la detección de colisiones técnicas de inteligencia artificial robótica para tomar y posicionar objetos, simulación segura de un peligroso entorno industrial [29].

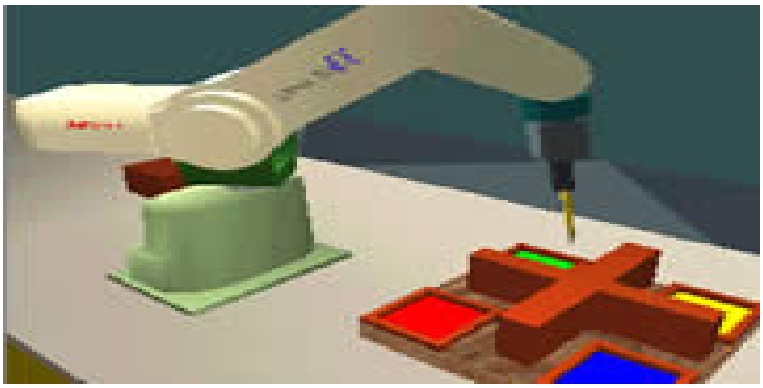


Figura 2.19: RV-M1 [29].

2.10.3. RoboCell

RoboCell permite que los alumnos creen, programen, simulen y controlen toda la operativa de las células de robótica y sistemas de fabricación flexible (FMS).

RoboCell integra el software de control de robótica SCORBASE son software interactivo de simulación de sólidos en 3D. Los dispositivos y robots virtuales que aparecen en RoboCell copian de manera exacta las dimensiones y funciones de los equipos SCORBOT. Los alumnos pueden enseñar posiciones, escribir programas y depurar aplicaciones de robótica de forma offline antes de ejecutarlas en las células de trabajo real. RoboCell permite a los alumnos experimenten con una gran variedad de células de trabajo simuladas que no existen fisicamente en el laboratorio. Los alumnos de nivel mas avanzado pueden diseñar objetos en 3D e importarlos a RoboCell para usarlos en los trabajos virtuales, ver figura 2.20[30].

2.10.4. Robolab

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.

Este laboratorio virtual remoto para robótica y evaluación de su impacto en la docencia, ya que es una herramienta adecuada para introducir a los alumnos en cuestiones básicas de robótica y control cinemática, gracias a su simulación gráfica con una interfaz de usuario compacta de fácil utilización. Además ofrece la capacidad ejecución remota que permite al alumno probar los resultados de los ejercicios prácticos en el robot real. Todo ello, sin

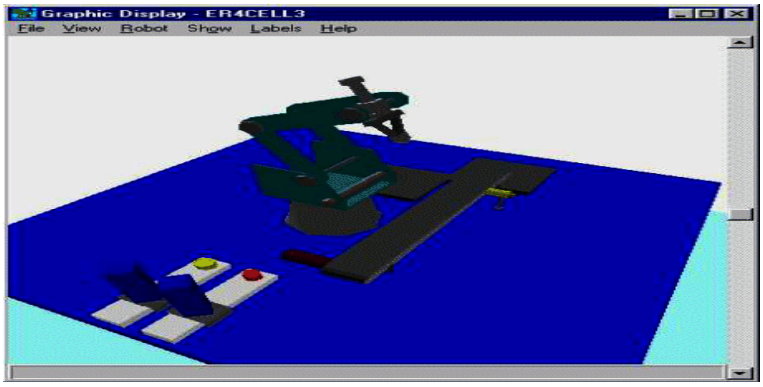


Figura 2.20: RoboCell [30].

que el alumno necesite desplazarse al laboratorio, haciendo este sistema en un proceso de enseñanza-aprendizaje.

La principal diferencia de la nueva versión del sistema con respecto a la anterior, es que la representación virtual del robot se realiza mediante Java-3D. Con esto se consigue que la interfaz de usuario (cliente) esté contenida toda en un applet de forma compacta, solucionando algunos problemas de compatibilidad e instalación con determinados sistemas operativos o navegadores que presenta la versión basada en VRML, la figura 2.21 muestra el entorno [31].

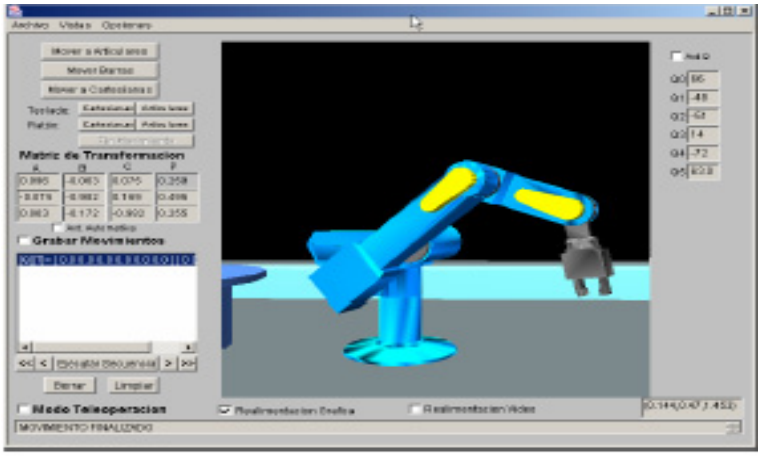


Figura 2.21: Robolab [31].

2.11. Conclusiones

En este capitulo se menciona la gran importancia que tiene hoy en día la realidad virtual y su interacción con aplicaciones educativas en la ciencia como la física, ingeniería, ciencias de la tierra, medicina, arquitectura, etc., el entretenimiento como: videojuegos, películas, problemas psicológicos entre ellos acrofobia, agorafobia alteraciones de imagenes corporales en los transtornos alimenticios, fovia a volar, claustrofobia, etc., además de que la realidad virtual nos ayuda no simplemente a navegar por el mundo virtual que no existe si no también a simular experiencias o experimentos en los cuales uno puede aprender o a resolver problemas que pueden ser fatales en la vida diaria.

Capítulo 3

Características y modelo matemático de un Robot antropomórfico de 3 grados de libertad

3.1. Introducción

En este capítulo se hace una breve descripción de los robots manipuladores describiendonos las partes que lo componen, sus grados de libertad, así como también describiendo su clasificación, además se evaluo al robot Scorbot ER-VII considerando 3 grados de libertad: base, hombro y codo, este último engloba al pich y roll, se desarrollo el modelo matemático, ya que este nos permite obtener la posición, velocidada y aceleración, sin dejar de considerar la metodología para obtenerlos, resolviendo la posición y orientación del elemento terminal, o sea la situación de las articulaciones y determinando la magnitud de los parámetros característicos de los grados de libertad conociendo su posición y orientación, la dinámica de Euler-Lagrange, aplicando el control de modo deslizante, haciendo la simulación mostrando los resultados optenidos en MATLAB.

El control de robots nos permite tener una gran presición en sus resultados que los robots industriales no pueden hacer, ya que estos controles son muy socorridos en aplicaciones industriales en donde se debe de tener tareas programables como: pintura, soldadura, esmaltado, corte, arenado, pulido, desbardado, etc.

3.2. Introducción a los robots manipuladores

La palabra robot proviene del checo y la usó por primera vez el escritor Karel Capek en 1917 para referirse, en sus obras, a máquinas con forma humanoide.

Con el objetivo de diseñar una máquina flexible, adaptable al entorno y de fácil manejo, George Devol, pionero de la Robótica Industrial, patentó, en 1956, un manipulador programable que fue el germen del robot industrial.

Una peculiaridad de los robots es su estructura en forma de brazo mecánico y otra su adaptabilidad a diferentes aprehensores o herramientas. Otra característica específica del robot, es la posibilidad de llevar acabo trabajos completamente diferentes e incluso tomar decisiones según la información procedente del mundo exterior[32].

Clasificación de los robots
La maquinaria para la automatización rígida dio paso al robot con el desarrollo de controladores rápidos, basados en el microprocesador, así como con el empleo de servos en bucle cerrado, que permiten establecer con exactitud la posición deseada. Esta evolución ha dado origen a una serie de tipos de robots, que se citan a continuación:

1. **Manipuladores**

Son sistemas mecánicos multifuncionales con un sencillo sistema de control que permite gobernar el movimiento de sus elementos de los siguientes modos:

a) Manual: Cuando el operador controla directamente la tarea del manipulador.
b) De secuencia fija: Cuando se repite de forma invariable el proceso de trabajo preparado previamente.
c) De secuencia variable: Se puede alterar algunas características de los ciclos de trabajo.

Recibe el nombre de manipulador o brazo de un robot el conjunto de elementos mecánicos que propician el movimiento del elemento terminal (aprehensor o herramienta). Dentro de la estructura interna del manipulador se alojan en muchas ocasiones los elementos motrices engranajes y transmisiones que soportan el movimiento de las cuatro partes que generalmente suelen conformar el brazo como lo muestra la figura 3.1:

l) Base o pedestal de fijación.
ll) Cuerpo.
lll) Brazo.
lV) Antebrazo.

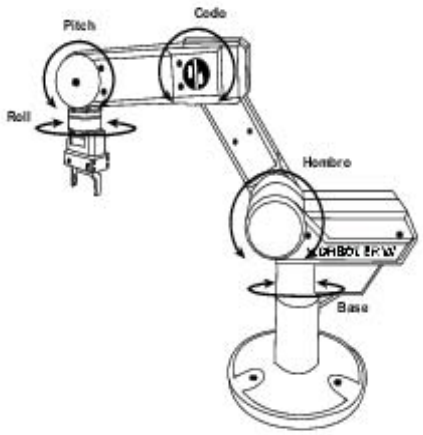


Figura 3.1: Esquema del manipulador

Los cuatro elementos rígidos del brazo están relacionados entre sí mediante articulaciones las cuales pueden ser giratorias cuando el movimiento permitido es el de rotación o prismáticas en las que existe un movimiento de translación entre los elementos que relacionan figura 3.2.

A semejanza con el brazo humano a las uniones o articulaciones del manipulador se les denomina:

-Unión del cuerpo(base-cuerpo).
-Unión hombro(cuerpo-brazo).
-Unión codo(brazo-antebrazo).
-Unión muñeca(antebrazo-aprehensor).

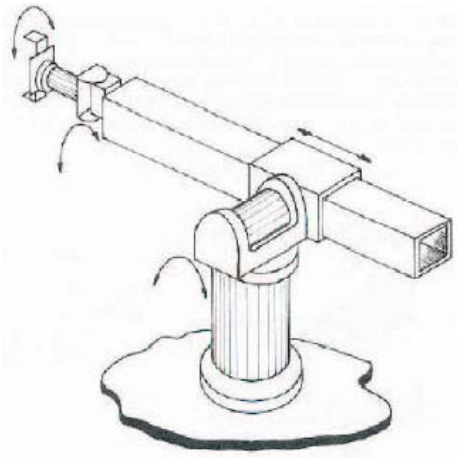


Figura 3.2: Manipulador con una articulación prismática [32].

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 suelen ser 6, que coinciden con los movimientos independientes que posicionan las partes del brazo en el espacio. Tres de ellos definen la posición en el espacio y los otros tres la orientación del elemento terminal.

El controlador

Recibe este nombre el dispositivo que se encarga de regular el movimiento de los elementos del manipulador y todo tipo de acciones, cálculos y procesado de información que se realiza. La complejidad del control varía según los parámetros que se gobiernan:

- Controladores de posición: sólo intervienen en el control de la posición del elemento terminal.
- Control cinemático: cuando además de la posición se regula la velocidad.
- Control dinámico: se tiene encuenta también las propiedades dinámicas del manipulador, motores y elementos asociados.
- Control adaptativo: además de lo indicado en los anteriores controles, también se considera la variación de las características del manipulador al variar la posición.

El control puede llevarse acabo en lazo abierto o cerrado. En el caso del control en lazo abierto, se produce una señal de consigna que determina el movimiento, pero no se analiza si se ha realizado con exactitud o se ha producido un error.

La mayoría de los sistemas de robots industriales poseen un control en lazo cerrado figura 3.3, con retroalimentación, este control hace uso de un transductor o sensor de la posición real de la articulación o del elemento terminal, cuya información se compara con el valor de la señal de mando o consigna que indica la posición deseada. El error entre estas dos magnitudes se trata de diversas formas para obtener una señal final, que aplicada a los elementos motrices varíe la posición real hasta hacerla coincidir con la deseada

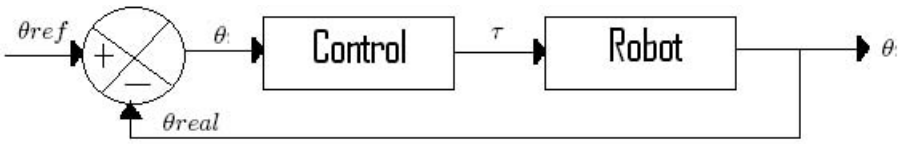


Figura 3.3: Control en lazo cerrado [5].

Los elementos motrices o actuadores

Son los encargados de producir el movimiento de las articulaciones directamente o a través de poleas, cables, cadenas, etc. Se clasifican en 3 grandes grupos atendiendo a la energía que utilizan:

a) Neumáticos.
b) Hidráulicos.
c) Eléctricos.

El elemento terminal

A la muñeca del manipulador se acopla una garra o una herramienta que será la encargada de materializar el trabajo previsto, el cual adopta formas muy diversas[32].

2. **Robots de repetición o aprendizaje**

Son manipuladores que se limitan a repetir una secuencia de movimientos, previamente ejecutada por un operador humano, haciendo uso de un controlador manual o un dispositivo auxiliar. En este tipo de robots el operador en la fase de enseñanza se vale de una pistola de programación con diversos pulsadores, teclas, joystics, o bien utiliza un maniquí, y algunas veces desplaza directamente la mano del robot.
Los robots de aprendizaje son más conocidos en la actualidad, en los ambientes industriales y el tipo de programación que incorporan recibe el nombre de "gestual"[32].

3. **Robots con control por computador**

Son manipuladores o sistemas mecánicos multifuncionales controlados por un computador, que habitualmente suele ser un microordenador.

En este tipo de robots el programador no necesita mover realmente el elemento de la máquina cuando la prepara para realizar un trabajo. El control por computador dispone de un lenguaje específico compuesto por varias instrucciones adaptadas al robot con las que puede confeccionar un programa de aplicación utilizando sólo el terminal, no el brazo. A esta programación se la denomina textual y se creá .ºf-line", es decir, sin la intervención del manipulador[32].

4. **Robots inteligentes**

Son similares a los grupos anteriores pero además son capaces de relacionarse con el mundo que les rodea a través de sensores y tomar decisiones en tiempo real(autoprogramables).
De momento, son muy poco conocidos en el mercado y se hallan en fase experimental, en la que se esfuerzan los grupos investigradores por potenciarles y hacerles más efectivos, al mismo tiempo que más accesibles[32].

5. **Micro-robots**

Con fines educacionales de entrenamiento o investigación existen numerosos robots de formación o microrobots, cuya estructura y funcionamiento son similares a los de aplicación industrial[32].

3.3. Modelo cinemático

La cinemática de robots se dedica al análisis y solución de los problemas derivados del posicionamiento de los elementos del manipulador.

La cinemática de robot hace uso de tres conceptos fundamentales:

1. Estructura mecánica del manipulador.
2. Grados de libertad para el posicionamiento del elemento terminal.
3. Solución de los modelos directo e inverso.

Con los grados de libertad propios de los elementos se consigue posicionar en el punto de la zona operativa el extremo libre o muñeca y con los tres grados de libertad de esta última se logra orientar en cualquier dirección al elemento terminal.

La cinemática sólo se ocupa de definir la posición del manipulador respecto a un sistema de coordenadas, a lo largo del tiempo. Mediante la cinemática se resuelven los dos problemas característicos en el posicionamiento del manipulador:

- Problema cinemático "directo". Resuelve la posición y orientación del elemento terminal, conociendo los parámetros que definen los grados de libertad, o sea, la situación de las articulaciones.
- Problema cinemático "inverso". Determina la magnitud de los parámetros característicos de los grados de libertad, conociendo la posición y orientación del elemento terminal [32].

3.3.1. Cadena cinemática

La cadena cinemática está constituida por los eslabones, las articulaciones y en su extremo final por un órgano ternimal o efector final. Los eslabones y articulaciones se enumeran comenzando por 1 hasta n (número de grados de libertad). Existen reglas para construir cadenas cinemáticas del manipulador. La figura 3.4 muestra la representación de la cadena cinemática.

Descipción:
θ_i: Ángulo de articulaciones del eje X_i al eje X_{i+1} medido respecto a Z_i.
d_i : Distancia entre X_i y X_{i+1} medido sobre Z_i.
L_i : Longitud de la perpendicular común a los ejes de las articulaciones.
α_i : Ángulo de separación del eje Z_i al eje Z_{i+1} medido respecto a Z_{i+1}.

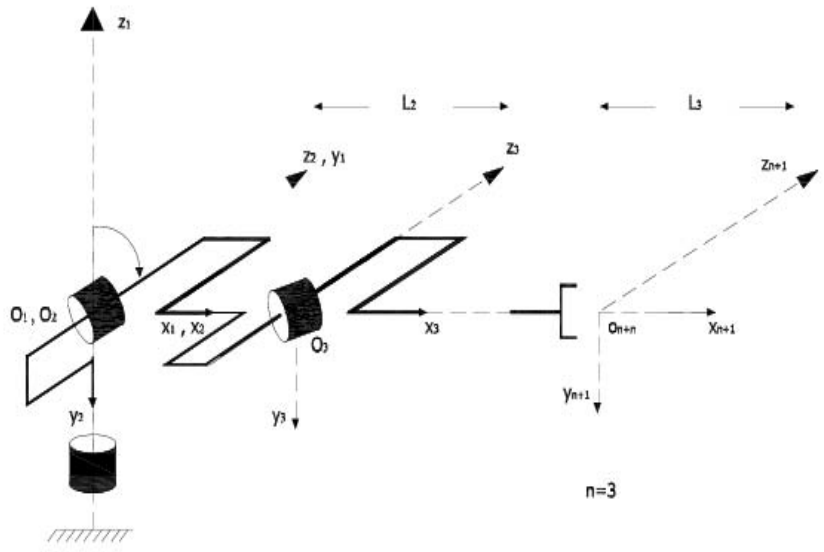


Figura 3.4: Cadena cinemática.

3.3.2. Parámetros Denavit-Hartenberg (PDH)

Denavit y Hartenberg propusieron un método sistemático para describir y representar la geometría espacial de los elementos de una cadena cinemática y en particular la de un robot con respecto a un sistema de referencia fijo. Este método utiliza una matriz de transformación homogénea para describir la relación espacial entre dos elementos rígidos adyacentes, reduciéndose el problema cinemático directo en una matriz de transformación homogénea 4 X 4 que relacione la localización (orientación y posición) 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. Estas transformaciones básicas consisten en una sucesión de rotaciones y traslaciones que permitan relacionar el sistema de referencia del elemento i con el sistema del elemento i-1. Las transformaciones en cuestión son las siguientes[33][34]:

1. Rotación al rededor del eje Z_{i-1} un ángulo θ_i
2. Traslación a lo largo de Z_{i-1} una distancia d_i
3. Traslación a lo largo de X_i una distancia a_i
4. Rotación al rededor del eje X_i, un ángulo α_i

Una vez que los marcos de cada eslabón han sido establecidos, la posicón del eslabón i con respecto al eslabón i-1 son especificados por los siguientes parámetros:

θ_i : Es el ángulo el cual hay que girar en torno a Z_{i-1} para que X_{i-1} y X_i queden paralelos.

d_i : Es la distancia medida entre X_i y X_{i-1} a lo largo de Z_{i-1}.

a_i : Distancia entre O_i y O_{i+1}.

α_i : Ángulo entre los ejes Z_{i-1} y Z_i a través del eje X_i para ser positivo cuando la rotación va en el sentido de las manecillas del reloj.

La figura 3.5 muestra los resultados obtenidos de la cadena cinemática, así como sus parámetros se muestran en el cuadro 3.1.

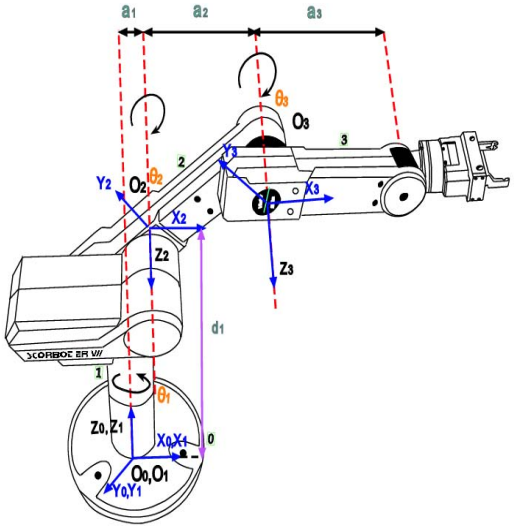


Figura 3.5: Scorbot ER-VII y representación de PDH

Parámetros Denavit-Hartenberg

i	α_{i+1}	a_{i+1}	d_i	θ_i
1	-90	a_1	d_1	θ_1
2	0	a_2	0	θ_2
3	0	a_3	0	θ_3

Cuadro 3.1: Parámetros Denavit-Hartenberg

3.3.3. Matriz elementales y de transformacion homogénea

Definen la posición y orientación de un marco de referencia respecto a otro y se obtienen sustituyendo los parámetros Denavit Hartenberg en la siguiente matriz:

$$T_{i+1}^{i} = \begin{bmatrix} C\theta_i & -S\theta_i C\alpha_i & S\theta_i S\alpha_i & a_i C\theta_i \\ S\theta_i & C\theta_i C\alpha_i & -C\theta_i S\alpha_i & a_i S\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

Donde:

$$C\theta_i = \cos\theta_i$$

$$S\theta_i = \sin\theta_i$$

$$C\alpha_1 = \cos\alpha_i$$

$$S\alpha_i = \sin\alpha_i$$

Sustituyendo los parámetros PDH se obtienen las matrices elementales: 3.2, 3.3, 3.4

$$T_2^1 = \begin{bmatrix} C_1 & 0 & -S_1 & 0 \\ S_1 & -1 & C_1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.2}$$

$$T_3^2 = \begin{bmatrix} C_2 & -S_2 & 0 & L_2C_2 \\ S_2 & C_2 & 0 & L_2S_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.3}$$

$$T_4^3 = \begin{bmatrix} C_3 & -S_3 & 0 & L_3C_3 \\ S_3 & C_3 & 0 & L_3S_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

La matriz de transformación homogénea generalizada se obtiene multiplicando las 3 matrices elementales:

$$T_4^1 = \begin{bmatrix} T_2^1 & T_3^2 & T_4^3 \end{bmatrix} \tag{3.5}$$

Empleando identidades trigonométricas para la suma de ángulos y realizando simplificaciones en la matriz 3.5 se obtiene la siguiente matriz:

$$T_4^1 = \begin{bmatrix} C_1C_{23} & -C_1S_{23} & -S_1 & L_3C_1C_{23} + L_2C_1C_2 \\ S_1C_{23} & -S_1S_{23} & C_1 & L_3S_1C_{23} + L_2S_1C_2 \\ -S_{23} & -C_{23} & 0 & -L_3S_{23} - L_2S_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.6}$$

3.3.4. Modelado cinemático directo de posición (MCDP)

Permite determinar las coordenadas operacionales X, Y, Z en función de las variables articulares θ_1, θ_2,θ_3 además de conocer la posición instantánea del órgano terminal.

$$X = f(\theta) \tag{3.7}$$

Las coordenadas de posición cartesiana son:

$$X_p = [X, Y, Z] = P[e_{14}, e_{24}, e_{34}] \tag{3.8}$$

Las coordenadas de orientación son:

$$X_0 = [\lambda, \mu, \nu] \tag{3.9}$$

$$\lambda = atan2(-e_{23}, e_{33}) \tag{3.10}$$

$$\mu = atan2(e_{13}, \frac{e_{33}}{\cos\lambda}) \tag{3.11}$$

$$\nu = atan2(-e_{12}, e_{11}) = \theta_2 + \theta_3 \tag{3.12}$$

El MCDP se obtiene de $X_p \bigcup X_0$

$$X = \begin{pmatrix} X \\ Y \\ Z \\ \lambda \\ \mu \\ \nu \end{pmatrix} = \begin{pmatrix} L_3C_1C_{23} + L_2C_1C_2 \\ L_3S_1C_{23} + L_2S_1C_2 \\ -L_3S_{23} - L_2S_2 \\ 0 \\ 0 \\ \theta_2 + \theta_3 \end{pmatrix} \tag{3.13}$$

3.3.5. Modelado cinemático directo de velocidad (MCDV)

Nos proporciona el conocimiento de la velocidad instántanea que el órgano terminal o efector final adquiere durante la ejecución de una tarea referente al marco operacional o espacio de trabajo y donde además, dicho modelo puede ser representado en términos de una matriz jacobiana.

Dada la expresión que identifica al MCDP (ver ecuación 3.7) y aplicando la primera derivada tenemos que:

$$\dot{X} = \frac{d}{dt} f(\theta)$$

Obtenemos el MCDV

$$\dot{X} = J_c\dot{\theta} \tag{3.14}$$

Donde:

$\dot{X}$ = Es el vector de velocidad del Órgano terminal del robot $[\dot{X}, \dot{Y}, \dot{Z}, \dot{\lambda}, \dot{\mu}, \dot{\nu}]$
$\dot{\theta}$= Es el vector de velocidad de las articulaciones del robot $\dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3$
J_c= Es la matriz jacobiana del robot cuyos elementos son J_y

$$\dot{J}_y = \frac{\partial f_i}{\partial \theta_j} \tag{3.15}$$

Derivando al MCDP con respecto al tiempo, conseguimos las ecuaciones que sirven para controlar la posición instantanea del robot:

$$X = P_x = L_3C_1C_{23} + L_2C_1C_2$$

$$Y = P_y = L_3S_1C_{23} + L_2S_1C_2$$

$$Z = P_z = -L_3S_{23} - L_2S_2$$

$$\lambda = 270°$$

$$\mu = 270°$$

$$\nu = \theta_2 + \theta_3$$

Derivando X

$\frac{dX}{dt} = \dot{X} = \{L_3C_1C_{23} + L_2C_1C_2\}$

$\dot{X} = L_3[C_1\{-S_{23}(\dot{\theta}_2 + \dot{\theta}_3)\} + C_{23}\{-S_1(\dot{\theta}_1)\}] + L_2[C_1\{-S_2(\dot{\theta}_2)\} + C_2\{-S_1(\dot{\theta}_1)\}]$

Simplificando términos

$= -L_3S_{23}C_1(\dot{\theta}_2 + \dot{\theta}_3) - L_3S_1C_{23}\dot{\theta}_1 - L_2S_2C_1\dot{\theta}_2 - L_2S_1C_2\dot{\theta}_1$

$$\dot{X} = [-L_3S_1C_{23} - L_2S_1C_2]\dot{\theta}_1 + [-L_3S_{23}C_1 - L_2S_2C_1]\dot{\theta}_2 + [-L_3S_{23}C_1]\dot{\theta}_3$$

Derivando Y

$\frac{dY}{dt} = \dot{Y} = \{L_3S_1C_{23} + L_2S_1C_2\}$

$\dot{Y} = L_3[S_1\{-S_{23}(\dot{\theta}_2 + \dot{\theta}_3)\} + C_{23}\{C_1\dot{\theta}_1\}] + L_2[S_1\{-S_2\dot{\theta}_2\} + C_2\{C_1\dot{\theta}_1\}]$

Simplificando términos

$= -L_3S_1S_{23}(\dot{\theta}_2 + \dot{\theta}_3) + L_3C_1C_{23}\dot{\theta}_1 - L_2S_1S_2\dot{\theta}_2 + L_2C_1C_2\dot{\theta}_1$

$$\dot{Y} = [L_3C_1C_{23} + L_2C_1C_2]\dot{\theta}_1 + [-L_3S_1S_{23} - L_2S_1S_2]\dot{\theta}_2 + [-L_3S_1S_{23}]\dot{\theta}_3$$

Derivando Z

$\frac{dZ}{dt} = \dot{Z} = \{-L_3S_{23} - L_2S_2\}$

$= -L_3[C_{23}(\dot{\theta}_2 + \dot{\theta}_3)] - L_3[C_2(\dot{\theta}_3)]$

$$\dot{Z} = [-L_3C_{23} - L_2C_2]\dot{\theta}_2 + [-L_3C_{23}]\dot{\theta}_3$$

Finalmente tenemos que:

$$\frac{d\lambda}{dt} = \dot{\lambda} = \frac{d}{dt}270 = 0$$

$$\frac{d\mu}{dt} = \dot{\mu} = \frac{d}{dt}270 = 0$$

$$\frac{d\nu}{dt} = \dot{\nu} = \frac{d}{dt}(\theta_2 + \theta_3) = \dot{\theta}_2 + \dot{\theta}_3$$

La velocidad del órgano terminal queda:

$$\text{Vel} = \begin{pmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \\ \dot{\lambda} \\ \dot{\mu} \\ \dot{\nu} \end{pmatrix} = \begin{pmatrix} (-L_3S_1C_{23} - L_2S_1C_2)\dot{\theta}_1 + & (-L_3S_{23}C_1 - L_2S_2C_1)\dot{\theta}_2 + & (-L_3S_{23}C_1)\dot{\theta}_3 \\ (L_3C_1C_{23} + L_2C_1C_2)\dot{\theta}_1 + & (-L_3S_1S_{23} - L_2S_1S_2)\dot{\theta}_2 + & (-L_3S_1S_{23})\dot{\theta}_3 \\ 0 + & (-L_3C_23 - L_2C_2)\dot{\theta}_2 + & (-L_3C_{23})\dot{\theta}_3 \\ 0 + & 0 + & 0 \\ 0 + & 0 + & 0 \\ 0 + & \dot{\theta}_2 + & \dot{\theta}_3 \end{pmatrix} \quad (3.16)$$

Dado que el MCDV puede ser representado en una matriz jacobiana queda:

$$J_c = \begin{pmatrix} \frac{\partial x}{\partial \theta_1} & \frac{\partial x}{\partial \theta_2} & \frac{\partial x}{\partial \theta_3} \\ \frac{\partial y}{\partial \theta_1} & \frac{\partial y}{\partial \theta_2} & \frac{\partial y}{\partial \theta_3} \\ \frac{\partial z}{\partial \theta_1} & \frac{\partial z}{\partial \theta_2} & \frac{\partial z}{\partial \theta_3} \\ \frac{\partial \lambda}{\partial \theta_1} & \frac{\partial \lambda}{\partial \theta_2} & \frac{\partial \lambda}{\partial \theta_3} \\ \frac{\partial \mu}{\partial \theta_1} & \frac{\partial \mu}{\partial \theta_2} & \frac{\partial \mu}{\partial \theta_3} \\ \frac{\partial \nu}{\partial \theta_1} & \frac{\partial \nu}{\partial \theta_2} & \frac{\partial \nu}{\partial \theta_3} \end{pmatrix} \quad (3.17)$$

Por lo tanto la matriz jacobiana queda:

$$J_c = \begin{pmatrix} -L_3S_1C_{23} - L_2S_1C_2 + & -L_3S_{23}C_1 - L_2S_2C_1 + & -L_3S_{23}C_1 \\ L_3C_1C_{23} + L_2C_1C_2 + & -L_3S_1S_{23} - L_2S_1S_2 + & -L_3S_1S_{23} \\ 0 + & -L_3C_23 - L_2C_2 + & -L_3C_{23} \\ 0 + & 0 + & 0 \\ 0 + & 0 + & 0 \\ 0 + & \dot{\theta}_2 + & \dot{\theta}_3 \end{pmatrix} \quad (3.18)$$

3.3.6. Modelado cinemático directo de aceleración (MCDA)

Sustituyendo la ecuación 3.16 tenemos que :

$$\ddot{X} = \begin{pmatrix} (-L_3S_1C_{23} - L_2S_1C_2)\ddot{\theta}_1 + (L_3S_{23}C_1 - L_2S_2C_1)\ddot{\theta}_2 - (L_3S_{23}C_1)\ddot{\theta}_3 \\ +[L_3S_1S_{23}(\dot{\theta}_2 + \dot{\theta}_3) - L_3C_1C_{23}\dot{\theta}_1 + L_2S_1S_2\dot{\theta}_2 - L_2C_1C_2\dot{\theta}_1] \\ +[L_3S_1S_{23}\dot{\theta}_1 - L_3C_1C_{23}(\dot{\theta}_2 + \dot{\theta}_3) + L_2S_1S_2\dot{\theta}_1 - L_2C_1C_2\dot{\theta}_2]\dot{\theta}_2 \\ +[L_3S_1S_{23}\dot{\theta}_1 - L_3C_1C_{23}(\dot{\theta}_2 + \dot{\theta}_3)]\dot{\theta}_3 \end{pmatrix} \quad (3.19)$$

$$\ddot{Y} = \begin{pmatrix} (\text{-L}_3C_1C_{23} + L_2C_1C_2)\ddot{\theta}_1 + (-L_3S_1S_{23} - L_2S_1S_2)\ddot{\theta}_2 - (L_3S_1S_{23})\ddot{\theta}_3 \\ +[\text{-L}_3S_{23}C_1(\dot{\theta}_2 + \dot{\theta}_3) - L_3S_1C_{23}\dot{\theta}_1 - L_2S_2C_1\dot{\theta}_2 - L_2S_1C_2\dot{\theta}_1]\dot{\theta}_1 \\ +[\text{-L}_3S_1C_{23}(\dot{\theta}_2 + \dot{\theta}_3) - L_3S_{23}C_1\dot{\theta}_1 - L_2S_1C_2\dot{\theta}_2 - L_2S_2C_1\dot{\theta}_1]\dot{\theta}_2 \\ +[\text{-L}_3S_1C_{23}(\dot{\theta}_2 + \dot{\theta}_3) - L_3S_{23}C_1\dot{\theta}_1]\dot{\theta}_3 \end{pmatrix} \tag{3.20}$$

$$\ddot{Z} = \begin{pmatrix} (\text{-L}_3C_{23} - L_2C_2)\ddot{\theta}_2 - (L_2C_{23})\ddot{\theta}_3 \\ +[\text{L}_3S_{23}(\dot{\theta}_2 + \dot{\theta}_3) + L_2S_2\dot{\theta}_2]\dot{\theta}_2 \\ +[\text{L}_3S_{23}(\dot{\theta}_2 + \dot{\theta}_3)]\dot{\theta}_3 \end{pmatrix} \tag{3.21}$$

Finalmente, la segunda derivada de $\lambda, \mu y \nu$:

$$\ddot{\lambda} = 0$$

$$\ddot{\mu} = 0$$

$$\ddot{\nu} = \ddot{\theta}_2 + \ddot{\theta}_3$$

3.3.7. Modelado cinemático inverso de posición (MCIP)

Es la función inversa del MCDP, nos permite conocer las variables articulares que el robot debera adquirir para lograr una acción deseada del órgano terminal.

$$\theta = f^{-1}(X) \tag{3.22}$$

Obtención θ_1: este ángulo se obtiene por medio de la figura 3.6 que se construye entorno a la primera articulación,

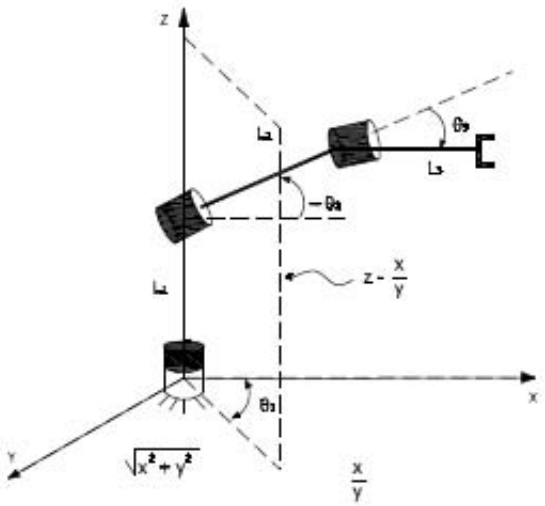


Figura 3.6: Cadena cinemática sintética.

obteniendo la ecuación 3.24:

$$tan\theta_1 = \frac{Y}{X} \tag{3.23}$$

Teniendo así la expresión que resuelve la primera variable articular:

$$\theta_1 = atan2(Y, X) \tag{3.24}$$

Obtención θ_3: por conveniencia se determina primero el valor de θ_3 antes que el valor de θ_2. Para el análisis, con-

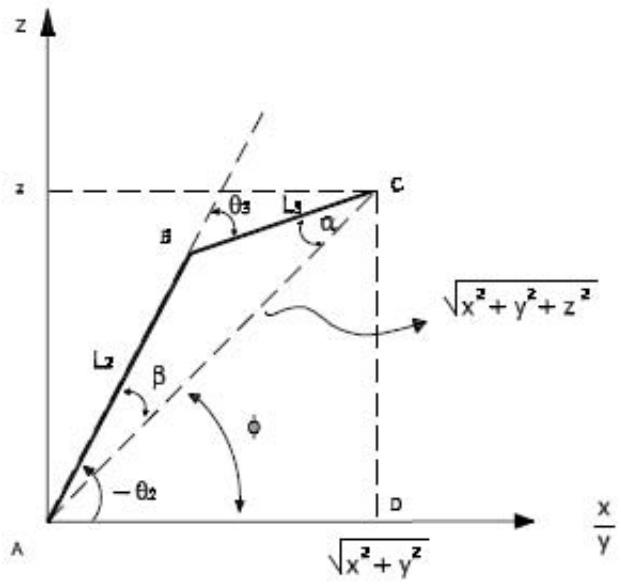


Figura 3.7: Triángulo formado por los eslabones L2, L3 y sus proyecciones sobre el plano XY.

sideramos el esquema derivado de la cadena cinemática, observado de la figura 3.7 conformada por los eslabones L2 y L3. Estableciendo relaciones en la figura y aplicando la ley de cosenos se obtiene la ecuación 3.25:

$$\cos\theta_3 = \frac{X^2 + Y^2 + Z^2 - L_2^2 - L_3^2}{2L_2L_3} \tag{3.25}$$

Para relacionar la función atan2, se obtiene la ecuación 3.26:

$$\sin\theta_3 = \sqrt{1 - \cos^2\theta_3} \tag{3.26}$$

Para que finalmente obtener la ecuación 3.27:

$$\theta_3 = atan2(\sin\theta_3, \cos\theta_3) \tag{3.27}$$

Obtención θ_2: tomando encuenta que el ángulo θ_2 se mide en sentido de las manecillas del reloj y observando nuevamente la figura 3.7 se obtiene la siguiente 3.28:

$$\theta_2 = -\varphi - \beta \tag{3.28}$$

En donde:

$$\varphi = atan2(Z\sqrt{X^2 + Y^2}) \tag{3.29}$$

Sabiendo que la suma de los ángulos internos de un triangulo es de 180°, podemos relacionar β con θ_2 y φ para obtener la ecuación 3.30:

$$\beta = atan2(L_3 \sin\theta_3, L_2 + L_3 \cos\theta_3) \tag{3.30}$$

Finalmente, θ_2 es :

$\theta_2 = -\varphi - \beta$

$$\theta_2 = -atan2(Z, \sqrt{X^2 + Y^2}) - atan2(L_3 \sin\theta_3, L_2 + L_3 \cos\theta_3) \tag{3.31}$$

3.3.8. Modelado cinemático inverso de velocidad (MCIV)

El modelo cinemático inverso de velocidad (MCIV) se obtuvo partiendo del modelo anterior, y este permite controlar al robot manipulador en cuanto a la velocidad. Así mismo, es posible identificar las singularidades, es decir, las confuguraciones inadmisibles (arreglos θ_1, θ_2, θ_3)deseados que no puede el RM.

Apartir de la formula del MCDV se define la ecuación representativa del MCIV:

$$\dot{X} = \frac{d}{dt} f(\theta) = J\dot{\theta} \tag{3.32}$$

Donde despejando a $\dot{\theta}$, tenemos que el jacobiano no se puede obtener su inversa, así que se utiliza la representación reducida del jacobiano (J_R):

$$J_R = \begin{pmatrix} -L_3S_1C_{23} - L_2S_1C_2 & + & -L_3S_{23}C_1 - L_2S_2C_1 & + & -L_3S_{23}C_1 \\ L_3C_1C_{23} + L_2C_1C_2 & + & -L_3S_1S_{23} - L_2S_1S_2 & + & -L_3S_1S_{23} \\ 0 & + & -L_3C_23 - L_2C_2 & + & -L_3C_{23} \end{pmatrix} \tag{3.33}$$

$\dot{X} = J_R\dot{\theta}$
$J_R^{-1}\dot{X} = J_R^{-1}J\dot{\theta}$

$$\dot{X} = J_R^{-1}\dot{X} \tag{3.34}$$

La derivada de cada termino de la matriz jacobiana es:

$J_{11} = L_3S_1S_{23}(\dot{\theta}_2 + \dot{\theta}_3) - L_3C_1C_{23}\dot{\theta}_1 + L_2S_1S_2\dot{\theta}_2 - L_2C_1C_2\dot{\theta}_1$
$J_{12} = L_3S_1S_{23}\dot{\theta}_1 - L_3C_1C_{23}(\dot{\theta}_2 + \dot{\theta}_3) + L_2S_1S_2\dot{\theta}_1 - L_2C_1C_2\dot{\theta}_2$
$J_{13} = L_3S_1S_{23}\dot{\theta}_1 - L_3C_1C_{23}(\dot{\theta}_2 + \dot{\theta}_3)$

$J_{21} = -L_3S_{23}C_1(\dot{\theta}_2 + \dot{\theta}_3) - L_3S_1C_{23}\dot{\theta}_1 - L_2S_2C_1\dot{\theta}_2 - L_2S_1C_2\dot{\theta}_1$
$J_{22} = -L_3S_1C_{23}(\dot{\theta}_2 + \dot{\theta}_3) - L_3S_{23}C_1\dot{\theta}_1 - L_2S_1C_2\dot{\theta}_2 - L_2S_2C_1\dot{\theta}_1$
$J_{23} = -L_3S_1C_{23}(\dot{\theta}_2 + \dot{\theta}_3) - L_3S_{23}C_1\dot{\theta}_1$

$J_{31} = 0$
$J_{32} = L_3S_{23}(\dot{\theta}_2 + \dot{\theta}_3) + L_2S_2\dot{\theta}_2$
$J_{33} = L_3S_{23}(\dot{\theta}_2 + \dot{\theta}_3)$

$J_{41} = 0$ $\quad$ $J_{51} = 0$ $\quad$ $J_{61} = 0$
$J_{42} = 0$ $\quad$ $J_{52} = 0$ $\quad$ $J_{62} = 0$
$J_{43} = 0$ $\quad$ $J_{53} = 0$ $\quad$ $J_{63} = 0$

3.3.9. Modelado cinemático inverso de aceleración (MCIA)

El MCIA se obtiene despejando a $\ddot{\theta}$ del MCDA de la siguente manera:

$$\ddot{X} = J_C\ddot{\theta} + J_c\dot{\theta} \tag{3.35}$$

$$J_c\ddot{\theta} = \ddot{X} - J_c\dot{\theta} \tag{3.36}$$

Sin embargo, se trabaja con el jacobiano reducido para obtener la inversa.

$$\ddot{\theta} = J_R^{-1}[\ddot{X} - \dot{J}_R\dot{\theta}] \tag{3.37}$$

Representando los elementos de las matrices:

$$\begin{pmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\theta}_3 \end{pmatrix} = J_R^{-1} \begin{pmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{pmatrix} - \begin{pmatrix} \dot{J}_{11} & \dot{J}_{12} & \dot{J}_{13} \\ \dot{J}_{21} & \dot{J}_{22} & \dot{J}_{23} \\ \dot{J}_{31} & \dot{J}_{32} & \dot{J}_{33} \end{pmatrix} \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{pmatrix}$$

Realizando el producto matricial indicado y realizando sustituciones se obtiene:

$$\ddot{\theta}_1 = -\tfrac{S_1}{L_2C_2+L_3C_{23}}[\ddot{X} - \{\{L_3S_1S_{23}(\dot{\theta}_2+\dot{\theta}_3) - L_3C_1C_{23}\dot{\theta}_1 + L_2S_1S_2\dot{\theta}_2 - L_2C_1C_2\dot{\theta}_1\}\dot{\theta}_1$$
$$+\{L_3S_1S_{23}\dot{\theta}_1 - L_3C_1C_{23}(\dot{\theta}_2+\dot{\theta}_3) + L_2S_1S_2\dot{\theta}_1 - L_2C_1C_2\dot{\theta}_2\}\dot{\theta}_2$$
$$+\{L_3S_1S_{23}\dot{\theta}_1 - L_3C_1C_{23}(\dot{\theta}_2+\dot{\theta}_3)\}\dot{\theta}_3\}]$$
$$+\tfrac{C_1}{L_2C_2+L_3C_{23}}[\dot{Y} - \{\{-L_3S_{23}C_1(\dot{\theta}_2+\dot{\theta}_3) - L_3S_1C_{23}\dot{\theta}_1 - L_2S_2C_1\dot{\theta}_2 - L_2S_1C_2\dot{\theta}_1\}\dot{\theta}_1$$
$$+\{-L_3S_1C_{23}(\dot{\theta}_2+\dot{\theta}_3) - L_3S_{23}C_1\dot{\theta}_1 - L_2S_1C_2\dot{\theta}_2 - L_2S_2C_1\dot{\theta}_1\}\dot{\theta}_2$$
$$\{-L_3S_1C_{23}(\dot{\theta}_2+\dot{\theta}_3) - L_3S_{23}C_1\dot{\theta}_1\}\dot{\theta}_3\}]$$

$$\ddot{\theta}_2 = \tfrac{C_1C_{23}}{L_2S_3}[\ddot{X} - \{\{L_3S_1S_{23}(\dot{\theta}_2+\dot{\theta}_3) - L_3C_1C_{23}\dot{\theta}_1 + L_2S_1S_2\dot{\theta}_2 - L_2C_1C_2\dot{\theta}_1\}\dot{\theta}_1$$
$$+\{L_3S_1S_{23}\dot{\theta}_1 - L_3C_1C_{23}(\dot{\theta}_2+\dot{\theta}_3) + L_2S_1S_2\dot{\theta}_1 - L_2C_1C_2\dot{\theta}_2\}\dot{\theta}_2$$
$$+\{L_3S_1S_{23}\dot{\theta}_1 - L_3C_1C_{23}(\dot{\theta}_2+\dot{\theta}_3)\}\dot{\theta}_3\}]$$
$$+\tfrac{S_1C_{23}}{L_2S_3}[\dot{Y} - \{\{-L_3S_{23}C_1(\dot{\theta}_2+\dot{\theta}_3) - L_3S_1C_{23}\dot{\theta}_1 - L_2S_2C_1\dot{\theta}_2 - L_2S_1C_2\dot{\theta}_1\}\dot{\theta}_1$$
$$+\{-L_3S_1C_{23}(\dot{\theta}_2+\dot{\theta}_3) - L_3S_{23}C_1\dot{\theta}_1 - L_2S_1C_2\dot{\theta}_2 - L_2S_2C_1\dot{\theta}_1\}\dot{\theta}_2$$
$$\{-L_3S_1C_{23}(\dot{\theta}_2+\dot{\theta}_3) - L_3S_{23}C_1\dot{\theta}_1\}\dot{\theta}_3\}]$$
$$-\tfrac{S_{23}}{L_2S_3}[\ddot{Z} - \{\{L_3S_{23}(\dot{\theta}_2+\dot{\theta}_3) + L_2S_2\dot{\theta}_2\}\dot{\theta}_2 + \{L_3S_{23}(\dot{\theta}_2+\dot{\theta}_3)\}\dot{\theta}_3\}]$$

$$\ddot{\theta}_3 = \tfrac{-L_2C_1C_2+L_3C_1C_{23}}{L_2L_3S_3}[\ddot{X} - \{\{L_3S_1S_{23}(\dot{\theta}_2+\dot{\theta}_3) - L_3C_1C_{23}\dot{\theta}_1 + L_2S_1S_2\dot{\theta}_2 - L_2C_1C_2\dot{\theta}_1\}\dot{\theta}_1$$
$$+\{L_3S_1S_{23}\dot{\theta}_1 - L_3C_1C_{23}(\dot{\theta}_2+\dot{\theta}_3) + L_2S_1S_2\dot{\theta}_1 - L_2C_1C_2\dot{\theta}_2\}\dot{\theta}_2$$
$$+\{L_3S_1S_{23}\dot{\theta}_1 - L_3C_1C_{23}(\dot{\theta}_2+\dot{\theta}_3)\}\dot{\theta}_3\}]$$

$$+\frac{-L_2S_1C_2+L_3S_1C_{23}}{L_2L_3S_3}[\ddot{Y}-\{\{-L_3S_{23}C_1(\dot{\theta}_2+\dot{\theta}_3)-L_3S_1C_{23}\dot{\theta}_1-L_2S_2C_1\dot{\theta}_2-L_2S_1C_2\dot{\theta}_1\}\dot{\theta}_1$$

$$+\{-L_3S_1C_{23}(\dot{\theta}_2+\dot{\theta}_3)-L_3S_{23}C_1\dot{\theta}_1-L_2S_1C_2\dot{\theta}_2-L_2S_2C_1\dot{\theta}_1\}\dot{\theta}_2$$

$$\{-L_3S_1C_{23}(\dot{\theta}_2+\dot{\theta}_3)-L_3S_{23}C_1\dot{\theta}_1\}\dot{\theta}_3\}]$$

$$+\frac{L_2S_2+L_3S_{23}}{L_2L_3S_3}[\ddot{Z}-\{\{L_3S_{23}(\dot{\theta}_2+\dot{\theta}_3)+L_2S_2\dot{\theta}_2\}\dot{\theta}_2+\{L_3S_{23}(\dot{\theta}_2+\dot{\theta}_3)\}\dot{\theta}_3\}]$$

3.4. Modelo dinámico

La dinámica del robot trata con la formulación matemática de las ecuaciones del movimiento del brazo. Las ecuaciones dinámicas de movimiento de un manipulador son un conjunto de ecuaciones matemáticas que describen la conducta dinámica del manipulador. El modelo dinámico real de un brazo se puede obtener de leyes físicas conocidas tales como las leyes de newton y la mecánica lagrangiana. Esto conduce el desarrollo de las ecuaciones dinámicas del movimiento para las distintas articilaciones del manipulador en términos de los parámetros geométricos e inerciales especificados para los distintos elementos. Se pueden aplicar sistemáticamente enfoques convencionales como las formulaciones de Lagrange-Euler y de Newton-Euler para desarrollar las ecuaciones de movimiento del robot.

La formulación de Euler-Lagrange.

Las ecuaciones de movimiento general de un manipulador se pueden expresar convenientemente mediante la aplicación directa de la formulación de Euler-Lagrange. Muchos investigadores utilizan la representación de Denavit-Hartenberg para describir el desplazamiento espacial entre los sistemas de coordenadas de elementos vecinos para obtener la información cinemática del elemento, y emplean la técnica dinámica lagrangiana para deducir las ecuaciones dinámicas de un manipulador. La aplicación directa se la formulación dinámica lagrangiana, junto con la representación de coordenadas de elementos de Denavit-Hartenberg, resulta en una descripcción algorítmica conveniente y compacta de las ecuaciones de movimiento del manipulador.

La derivación de las ecuaciones dinámicas de un manipulador con n grados de libertad se basa en la comprensión de: 1. La matriz de transformación de coordenadas homogéneas 4 x 4, ${}^{i-1}A_i$, que describe la relación espacial entre $(i-1)$-ésimo. Relaciona un punto fijado en el elemento i expresado en coordenadas homogéneas con respecto al sistema de coordenadas i-ésimo en el sistema de coordenadas $(i-1)$-ésimo. 2. La ecuación Euler-Lagrange

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i}-\frac{\partial L}{\partial q_i}=\tau_i \tag{3.38}$$

$i = 1, 2, ..., n$

donde
L = función lagrangiana.
K = energía cinética total del brazo.
P = energía potencial total de brazo.
q_i = coordenada generalizada del brazo.
$\dot{q}_i$ = primera derivada respecto al tiempo de la coordenada generalizada q_i.
τ_i = par de fuerzas o torque generalizado aplicado al sistema en la articulación i para mover al elemento i.

El modelo dinámico del antropomórfico en su estructura se define acontinuación, dada la representación gráfica de la cadena cinemática, considerando los parametros dinámicos nos permitirá en forma auxiliar derivar el MD.

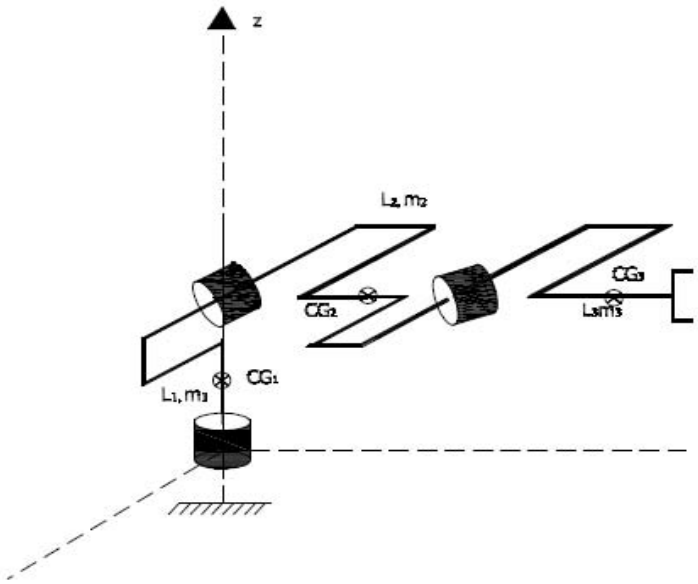


Figura 3.8: Representación del modelo dinámico en su representación.

Donde:

L_1, L_2, L_3 : longitud del los eslabones.
m_1, m_2, m_3 : masa de los eslabones.
CG_1, CG_2, CG_3 : centros de masas o de gravedad.
Localizados en la longitud media, considerando que estan construidos de material homogeneo.

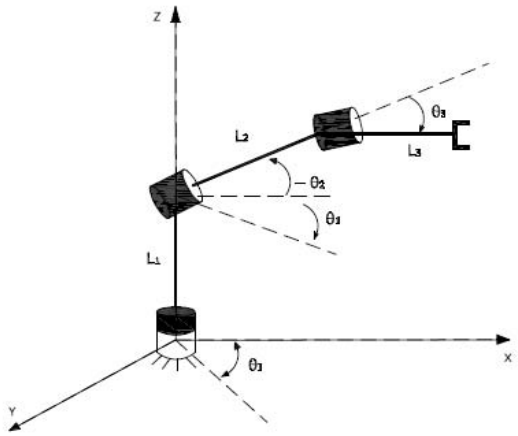


Figura 3.9: Representación de las articulaciones del Scorbot ER-VII.

Cálculo de las velocidades

Para obtener la velocidad y la altura al centro de masas para cada uno de los eslabones se hace referencia al MCDP y al MCDV.

Eslabón 1

Las coordenadas operacionales (X_1, Y_1 y Z_1) y sus derivadas con respecto al tiempo son:

$X_1 = 0 \qquad \dot{X}_1 = 0$
$Y_1 = 0 \qquad \dot{Y}_1 = 0$
$Z_1 = 0 \qquad \dot{Z}_1 = 0$

$v_1^2 = \dot{X}_1^2 + \dot{Y}_1^2 + \dot{Z}_1^2 \;\; v_1^2 = 0$

$h_1 = LCG_1$

Eslabón 2

Las coordenadas operacionales ($X_2, Y_2 y Z_2$) y sus derivadas con respecto del tiempo son:

$X_2 = L_2C_1C_2 \qquad \dot{X}_2 = -L_2S_1C_2\dot{\theta}_1 - L_2S_2C_1\dot{\theta}_2$
$Y_2 = L_2S_1C_2 \qquad \dot{Y}_2 = L_2C_1C_2\dot{\theta}_1 - L_2S_1S_2\dot{\theta}_2$
$Z_2 = -L_2S_2 \qquad \dot{Z}_2 = -L_2C_2\dot{\theta}_2$

$v_2^2 = \dot{X}_2^2 + \dot{Y}_2^2 + \dot{Z}_2^2$
$v_1^2 = L_2^2C_2^2\dot{\theta}_1^2 + L_2S_2^2\dot{\theta}_2^2 + L_2^2C_2^2\dot{\theta}_2^2$
$v_1^2 = L_2^2C_2^2\dot{\theta}_1^2 + L_2\dot{\theta}_2^2$

$h_2 = -LCGS_2$

Eslabón 3

$X_3 = L_2C_1C_2 + L_3C_1C_{23} \qquad \dot{X}_3 = -L_2S_1C_2\dot{\theta}_1 - L_2S_2C_1\dot{\theta}_2 - L_3S_1C_{23}\dot{\theta}_1 - L_3S_{23}C_1(\dot{\theta}_2 + \dot{\theta}_3)$
$Y_3 = L_2S_1C_2 + L_3S_1C_{23} \qquad \dot{Y}_3 = -L_2S_1S_2\dot{\theta}_2 + L_2C_1C_2\dot{\theta}_1 + L_3C_1C_{23}\dot{\theta}_1 - L_3S_1S_{23}(\dot{\theta}_2 + \dot{\theta}_3)$
$Z_3 = -L_2S_2 - L_3S_{23} \qquad \dot{Z}_3 = -L_2C_2\dot{\theta}_2 - L_3C_{23}(\dot{\theta}_2 + \dot{\theta}_3)$

$v_3^2 = \dot{X}_3^2 + \dot{Y}_3^2 + \dot{Z}_3^2$

De manera que simplificamos se obtiene:

$v_3^2 = L_2^2\dot{\theta}_2^2 + L_3^2(\dot{\theta}_2 + \dot{\theta}_3)^2 + 2L_2L_3C_3\dot{\theta}_2^2(\dot{\theta}_2 + \dot{\theta}_3) + (L_2C_2\dot{\theta}_1 + L_3C_{23}\dot{\theta}_1)^2$

$h_3 = -L_2S_2 - LCG_3S_{23}$

Cálculo de la energía cinética
El cálculo para cada uno de los eslabones, se presenta a continuación:

Eslabón 1

$k_1 = \frac{1}{2}m_1v_1^2$

$k_1 = \frac{1}{2} m_1 (0)$

$$k_1 = 0 \tag{3.39}$$

Eslabón 2

$k_2 = \frac{1}{2} m_2 v_2^2$

$k_2 = \frac{1}{2} m_2 \{ L_2^2 C_2^2 \dot{\theta}_1^2 + L_2^2 \dot{\theta}_2^2 \}$

$$k_2 = \frac{1}{2} m_2 L_2^2 C_2^2 \dot{\theta}_1 + \frac{1}{2} m_2 L_2^2 \dot{\theta}_2^2 \tag{3.40}$$

Eslabón 3

$k_3 = \frac{1}{2} m_3 v_3^2$

$k_3 = \frac{1}{2} m_3 \{ L_2^2 \dot{\theta}_2^2 + L_3^2 (\dot{\theta}_2 + \dot{\theta}_3)^2 + 2 L_2 L_3 C_3 \dot{\theta}_2 (\dot{\theta}_2 + \dot{\theta}_3)^2 + (L_2 C_2 \dot{\theta}_1 + L_3 C_{23} \dot{\theta}_1) \}$

$$k_3 = \frac{1}{2} m_3 L_2^2 \dot{\theta}_2^2 + \frac{1}{2} m_3 L_3^2 (\dot{\theta}_2 + \dot{\theta}_3)^2 + m_3 L_2 L_3 C_3 \dot{\theta}_2 (\dot{\theta}_2 + \dot{\theta}_3)^2 + \frac{1}{2} m_3 (L_2 C_2 \dot{\theta}_1 + L_3 C_{23} \dot{\theta}_1)^2 \tag{3.41}$$

Cálculo de la energía potencial

El cálculo de la energía potencial para cada uno de los eslabones, se presenta a continuación:

Eslabón 1

$P_1 = m_1 g h_1$
$P_1 = m_1 g \{ -LCG_1 \}$

$$P_1 = m_1 g LCG_1 \tag{3.42}$$

Eslabón 2

$P_2 = m_2 g h_2$
$P_2 = m_2 g \{ -LCG_2 S_2 \}$

$$P_2 = -m_2 g LCG_2 S_2 \tag{3.43}$$

Eslabón 3

$P_3 = m_3 g h_3$
$P_3 = m_3 g \{ -L_2 S_2 - LCG_3 S_{23} \}$

$$P_3 = -m_3 g L_2 S_2 - m_3 g LCG_3 S_{23} \tag{3.44}$$

Formulación de Lagrange-Euler

Es el balance de todas las fuerzas que interactuan de forma general en los eslabones.

$$\tau = \frac{d}{dt}\frac{\partial}{\partial \dot{\theta}}L - \frac{\partial}{\partial \theta}L \tag{3.45}$$

Donde:

t : Tiempo
θ : Variable articular o coordenada generalizada
$\dot{\theta}$: Velocidad de la variable articular

$$\dot{\theta} = \tfrac{d}{dt}\theta; \theta = \theta(t)$$

Lagrangiano:

$$L = \Sigma_{i=1}^{n} k_i - p_i \tag{3.46}$$

$$k_i = \frac{1}{2} m v_i^2 \tag{3.47}$$

$$p_i = m_i g h_i \tag{3.48}$$

Donde:

k_i : Energía cinética del i-ésimo eslabón
p_i : Energía potencial del i-ésimo eslabón
m_i : Masa constante del i-ésimo eslabón
v_i : velocidad del i-ésimo eslabón
h_i : altura del i-ésimo eslabón
g : graviedad $9{,}81 m/s^2$

Cálculo del Lagrangiano

El lagrangiano (L), es la diferencia entre la energía potencial y la energía cinética de un sistema.

Para nuestro caso tenemos que:

$$L = \sum_{i=1}^{n=3} ki - pi$$

Donde:

El lagrangiano para 3 grados de libertad a partir de las energías cinéticas y potenciales es:

$L = k_1 + k_2 + k_3 - (p_1 + p_2 + p_3)$

Con base en las energías cinéticas y potenciales calculadas, se tiene:

$L = \frac{1}{2}m_2L_2^2C_2^2\dot{\theta}_1^2 + \frac{1}{2}m_2L_2\dot{\theta}_2^2 + \frac{1}{2}m_3L_2^2\dot{\theta}_2^2 + \frac{1}{2}m_3L_3^2(\dot{\theta}_2+\dot{\theta}_3)^2 + m_3L_2L_3C_3\dot{\theta}_2(\dot{\theta}_2+\dot{\theta}_3)$
$+\frac{1}{2}m_3(L_2C_2\dot{\theta}_1 + L_3C_{23}\dot{\theta}_1)^2 - m_1gLCG_1 + m_2gLCG_2S_2 + m_3gL_2S_2 + m_3gLCG_3S_{23}$

de manera que, simplificando y agrupando:

$$L = \{\tfrac{1}{2}m_2L_2^2C_2^2 + \tfrac{1}{2}m_3L_2^2C_2^2 + m_3L_2L_3C_2C_{23} + \tfrac{1}{2}m_3L_3^2C_{23}\}\dot{\theta}_1^2$$
$$+\{\tfrac{1}{2}m_2L_2^2 + \tfrac{1}{2}m_3L_2^2 + \tfrac{1}{2}m_3L_3^2 + m_3L_2L_3C_3\}\dot{\theta}_2^2 + \{\tfrac{1}{2}m_3L_3^2\}\dot{\theta}_3^2$$
$$+\{m_3L_3^2 + m_3L_2L_3c_3\}\dot{\theta}_2\dot{\theta}_3 - m_1gLCG_1 + m_2gLCG_2S_2 + m_3g\{L_2S_2 + LCG_3S_{23}\} \tag{3.49}$$

Las ecuaciones de movimiento de un sistema mecánico con coordenadas generalizadas $\theta\varepsilon R^m$ y lagrangiano L. Y al aplicarlas a un robot, se deben calcular su energía cinética y potencial de los eslabones en función de los ángulos y velocidades de las articulaciones.

Las ecuaciones de Lagrange-Euler, son escritas para nuestro caso de estudio, como:

$$\tau_1 = \frac{d}{dt}\frac{\partial}{\partial\dot{\theta}_1}L - \frac{\partial}{\partial\theta_1}L \tag{3.50}$$
$$\tau_2 = \frac{d}{dt}\frac{\partial}{\partial\dot{\theta}_2}L - \frac{\partial}{\partial\theta_2}L \tag{3.51}$$
$$\tau_3 = \frac{d}{dt}\frac{\partial}{\partial\dot{\theta}_3}L - \frac{\partial}{\partial\theta_3}L \tag{3.52}$$

donde:

τ :es la fuerza externa que actúa sobre la coordenada generalizada i.

$$i = 1, 2, 3, ..., m$$

Las ecuaciones de Lagrange-Euler reducen el número de ecuaciones que se necesitan para describir el movimiento de un sistema.

Por lo tanto, para (3.37) tenemos que:

$\frac{\partial L}{\partial\dot{\theta}_1} = \{m_2L_2^2C_2^2 + m_3L_2^2C_2^2 + 2m_3L_2L_3C_2C_{23} + m_3L_3^2C_{23}^2\}\dot{\theta}_1$

$\frac{d}{dt}\frac{\partial L}{\partial\dot{\theta}_1} = \{(m_2+m_3)L_2^2C_2^2 + 2m_3L_2L_3C_2C_{23} + m_3L_3^2C_{23}^2\}\ddot{\theta}_1$
$-\{2(m_2+m_3)L_2^2S_2C_2 + 2m_3L_2L_3(S_{23}C_2 + S_2C_{23}) + 2m_3L_3^2S_{23}C_{23}\}\dot{\theta}_1\dot{\theta}_2 - \{2m_3L_2L_3S_{23}C_2 + 2m_3L_3^2S_{23}C_{23}\}$

$\frac{\partial L}{\partial\theta_1} = 0$

la ecuación de par torsor 1 es:

$$\tau_1 = \{(m_2+m_3)L_2^2C_2^2 + 2m_3L_2L_3C_2C_{23} + m_3L_3^2C_{23}^2\}\ddot{\theta}_1$$

$$-\{2(m_2+m_3)L_2^2S_2C_2+2m_3L_2L_3(S_{23}C_2+S_2C_{23})+2m_3L_3^2S_{23}C_{23}\}\dot{\theta}_1\dot{\theta}_2$$
$$-\{2m_3L_2L_3S_{23}C_2+2m_3L_3^2S_{23}C_{23}\}\dot{\theta}_1\dot{\theta}_3 \tag{3.53}$$

Para (3.30) tenemos que:

$\frac{\partial L}{\partial \dot{\theta}_2}=\{m_2L_2^2+m_3L_2^2+m_3L_3^2+2m_3L_2L_3C_3\}\dot{\theta}_2+\{m_3L_3^2+m_3L_2L_3C_3\}\dot{\theta}_3$

$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_2}=\{(m_2+m_3)L_2^2+m_3L_3^2+2m_3L_2L_3C_3\}\ddot{\theta}_2+\{m_3L_3^2+m_3L_2L_3C_3\}\ddot{\theta}_3$
$-\{2m_3L_2L_3S_3\}\dot{\theta}_2\dot{\theta}_3-\{m_3L_2L_3S_3\}\dot{\theta}_3^2$

$\frac{\partial L}{\partial \theta_2}=\{-(m_2+m_3)L_2^2S_2C_2-m_3L_2L_3(C_2S_{23}+C_{23}S_2)-m_3L_3^2C_{23}S_{23}\}\dot{\theta}_1^2$
$+m_2gLCG_2C_2+m_3g\{L_2C_2+LCG_3C_{23}\}$

la ecuación de par torsor 2 es:

$$\tau_2=\{(m_2+m_3)L_2^2+m_3L_3^2+2m_3L_2L_3C_3\}\ddot{\theta}_2+\{m_3L_3^2+m_3L_2L_3C_3\}\ddot{\theta}_3-\{2m_3L_2L_3S_3\}\dot{\theta}_2\dot{\theta}_3$$
$$+\{(m_2+m_3)L_2^2S_2C_2+m_3L_2L_3(C_2S_{23}+C_{23}S_2)+m_3L_3^2C_{23}S_{23}\}\dot{\theta}_1^2-\{m_3L_2L_3S_3\}\dot{\theta}_3^2$$
$$-m_2gLCG_2C_2-m_3g\{L_2C_2+LCG_3C_{23}\} \tag{3.54}$$

Finalmente, para (3.31) tenemos que:

$\frac{\partial L}{\partial \dot{\theta}_3}=\{m_3L_3^2\}\dot{\theta}_3+\{m_3L_3^2+m_3L_2L_3C_3\}\dot{\theta}_2$

$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_3}=\{m_3L_3^2\}\ddot{\theta}_3+\{m_3L_3^2+m_3L_2L_3C_3\}\ddot{\theta}_2-\{m_3L_2L_3S_3\dot{\theta}_3\}\dot{\theta}_2$

$\frac{\partial L}{\partial \theta_3}=-\{m_3L_2L_3C_2S_{23}+m_3L_3^2S_{23}C_{23}\}\dot{\theta}_1^2-\{m_3L_2L_3S_3\}\dot{\theta}_2^2-\{m_3L_2L_3S_3\}\dot{\theta}_2\dot{\theta}_3+m_3gLCG_3C_{23}$

la ecuación de par torsor 3 es:

$$\tau_3=\{m_32L_3^2+m_3L_2L_3C_3\}\ddot{\theta}_2+\{m_3L_3^2\}\ddot{\theta}_3+\{m_3L_2L_3C_2S_{23}+m_3L_3^2S_{23}C_{23}\}\dot{\theta}_1^2$$
$$+\{m_3L_2L_3S_3\}\dot{\theta}_2^2-m_3gLCG_3C_{23} \tag{3.55}$$

La representación matricial de Lagrange-Euler del modelo dinámico, se basa en la siguiente espresión:

$$\tau=H(\theta)\ddot{\theta}+c(\theta,\dot{\theta})\dot{\theta}+G(\theta)+F(\dot{\theta}) \tag{3.56}$$

Donde:

$H(\theta)$: Matriz de inercia
$c(\theta,\dot{\theta})\dot{\theta}$: Matriz de fuerza de coriolis
$G(\theta)$: Vector de fuerzas gravitatorias
$F(\dot{\theta})$: Fuerzas de fricción

La matriz de inercia es:

$$H(\theta) = \begin{pmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{pmatrix}$$

y sus elementos son:

$H_{11} = (m_2 + m_3)L_2^2C_2^2 + 2m_3L_2L_3C_2C_{23} + m_3L_3^2C_{23}^2$
$H_{12} = 0$
$H_{13} = 0$
$H_{21} = 0$
$H_{22} = (m_2 + m_3)L_2^2 + m_3L_2^2 + 2m_3L_2L_3C_3$
$H_{23} = m_3L_3^2 + m_3L_2L_3C_3$
$H_{31} = 0$
$H_{32} = m_3 + L_3^2 + m_3L_2L_3C_3$
$H_{33} = m_3 + L_3^2$

La matriz de fuerzas de coriolis y centrípetas:

$$C(\theta, \dot{\theta}) = \begin{pmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{pmatrix}$$

Los elementos de la matriz de fuerzas de coriolis y centrípetas son:

$C_{11} = 0$
$C_{12} = -\{2(m_2 + m_3)L_2^2S_2C_2 + 2m_3L_2L_3(S_{23}C_2 + S_2C_{23}) + 2m_3L_3^2S_{23}C_2\}$
$C_{13} = -\{2m_3L_2L_3S_{23}C_2 + 2m_3L_3^2S_{23}C_{23}\}\dot{\theta}_1$
$C_{21} = \{(m_2 + m_3)L_2^2S_2C_2 + m_3L_2L_3(C_2S_{23} + C_{23}S_2) + m_3L_3^2S_{23}C_{23}\}\dot{\theta}_1$
$C_{22} = -\{2m_3L_2L_3S_3\}\dot{\theta}_3$
$C_{23} = -\{m_3L_2L_3S_3\}\dot{\theta}_3$
$C_{31} = \{m_3L_2L_3C_2C_{23} + m_3L_3^2S_{23}C_{23}\}\dot{\theta}_1$
$C_{32} = \{m_3L_2L_3S_3\}\dot{\theta}_2$
$C_{33} = 0$

El vector de fuerzas gravitacionales $G(\theta)$está definido como:

$$G(\theta) = \begin{pmatrix} G_{11} \\ G_{21} \\ G_{31} \end{pmatrix}$$

y cuyos elementos son:

$G_{11} = 0$
$G_{21} = -m_2LCG_2C_2 + m_3\{L_2C_2 + LCG_3C_{23}\}$
$G_{31} = -m_3LCG_3C_{23}$

El MD ha tomado en cuenta la fricción viscosa y seca, este vector fue definido como F($\dot{q}$) el cual es la suma del vector de fricción seca y el vector de fricción viscosa que acontinuación se describe:

$$F(\dot{q}) = Fv(\dot{q}) + Fs(\dot{q}) \tag{3.57}$$

Donde:

$Fv(\dot{q})$: Fricción viscosa
$Fs(\dot{q})$: Fricción seca

$Fv(\dot{q})$se describe acontinuación:

$$Fv(\dot{q}) = \begin{pmatrix} Fv_1(\dot{q}) \\ Fv_2(\dot{q}) \\ Fv_3(\dot{q}) \end{pmatrix} = \begin{pmatrix} b_1\dot{\theta}_1 \\ b_2\dot{\theta}_2 \\ b_3\dot{\theta}_3 \end{pmatrix}$$

en donde b_i es el coeficiente de fricción viscosa o rozamiento dinámico y $\dot{\theta}_i$ es la velocidad angular.

$Fs(\dot{q})$se describe acontinuación:

$$Fs(\dot{q}) = \begin{pmatrix} Fs_1(\dot{q}) \\ Fs_2(\dot{q}) \\ Fs_3(\dot{q}) \end{pmatrix} = \begin{pmatrix} K_1 sgn(\dot{\theta}_1) \\ K_2 sgn(\dot{\theta}_2) \\ K_3 sgn(\dot{\theta}_3) \end{pmatrix} = \begin{pmatrix} K_1 tanh(\beta_1\dot{\theta}_1) \\ K_2 tanh(\beta_2\dot{\theta}_2) \\ K_3 tanh(\beta_3\dot{\theta}_3) \end{pmatrix}$$

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.

3.4.1. Propiedades dinámicas

Dado el M.D del R.M

$$H(\theta)\ddot{\theta} + c(\theta,\dot{\theta})\dot{\theta} + G(\theta) + F(\dot{\theta}) = \tau$$

1. $H(\theta) = H(\theta)^T$ *Simétrica(Propiedad 1)*

$$H(\theta) = \begin{pmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{pmatrix}$$

y su transpuesta :

$$H(\theta)^T = \begin{pmatrix} H_{11} & H_{21} & H_{31} \\ H_{12} & H_{22} & H_{32} \\ H_{13} & H_{23} & H_{33} \end{pmatrix}$$

Por lo tanto $H(\theta) = H(\theta)^T$

2. *Para cualquier* $X = [X, Y, Z]^T; X^T H(\theta) X > 0$ *definida positiva(Propiedad 2)*

3. $X^T\{H(\dot{\theta}) - 2C(\theta,\dot{\theta})\}X \equiv 0$ *Antisimétrica(Propiedad 3)*

3.4.2. Simulación digital

Para poder evaluar las propiedades dinámicas del robot Scorbot ER VII y saber que estas se cumplen hemos optado por representar su comportamiento que tiene físicamente, para proporcionar su visualización una idea de como lo vamos a visualizar, (ver figura 3.10), sus condiciones iniciales a evaluar son: $\theta_1 = \pi/2$, $\theta_2 = 0$, $\theta_3 = 0$.

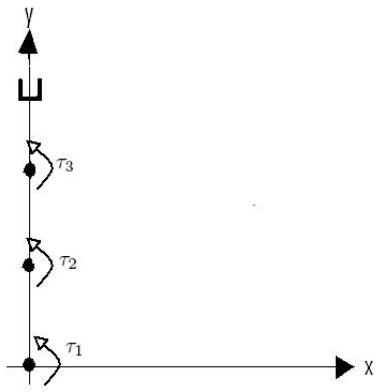


Figura 3.10: Condición inicial del Scorbot ER VII.

En donde obtendremos sus condiciones finales representadas de la forma siguiente $\theta_1 = -\pi/2$, $\theta_2 = 0$, $\theta_3 = 0$, figura 3.11.

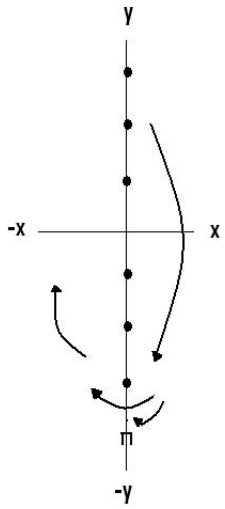


Figura 3.11: Condición final del Scorbot ER VII.

Una vez que hemos visualizado su comportamento fisicamente su simulación en MATLAB queda representado para las coordenadas generalizadas $q1$, $q2$ y $q3$, partiendo de sus condiciones iniciales: $q1$ =90°, $q2$ =0°, $q3$ =0°, figura 3.12.

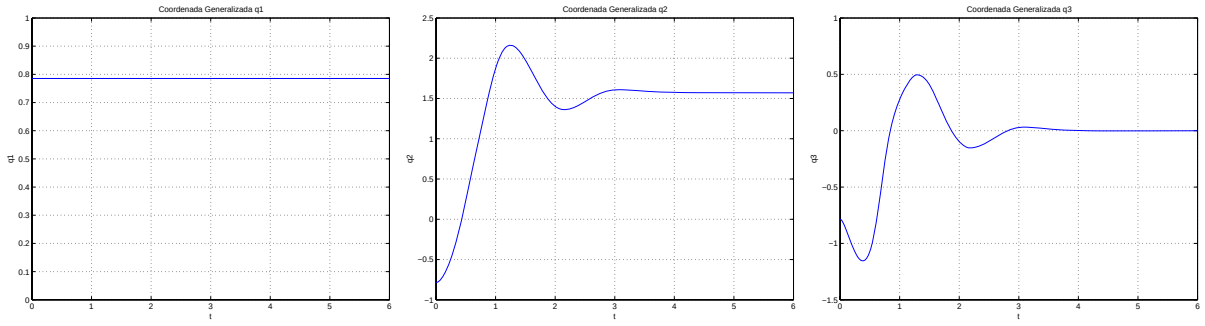


Figura 3.12: Coordenadas generalizadas q1, q2 y q3 en la simulación digital de las propiedades dinámicas

La figura 3.13 representa de la velocidad ángular dq1/dt, dq2/dt, dq3/dt, cuando en dq1/dt parte de una condición inicial de 0° oscila hasta llegar a la condición final es que un estado de reposo, de igual manera dq2/dt y dq3/dt.

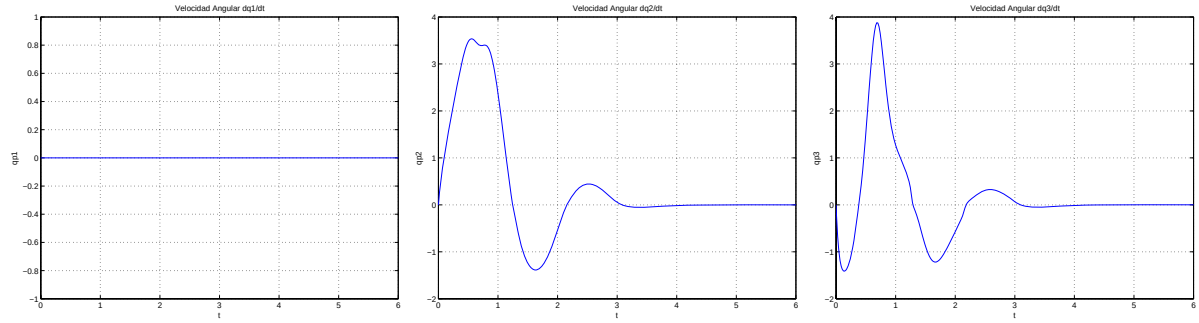


Figura 3.13: Velocidad angular dq1/dt, dq2/dt, dq3/dt en la simulación digital de las propiedades dinámicas.

La figura 3.14 representa la simulación de las coordenada operación X, Y, Z, haciendo su osilación hasta llegar al estado de reposo respectivamente.

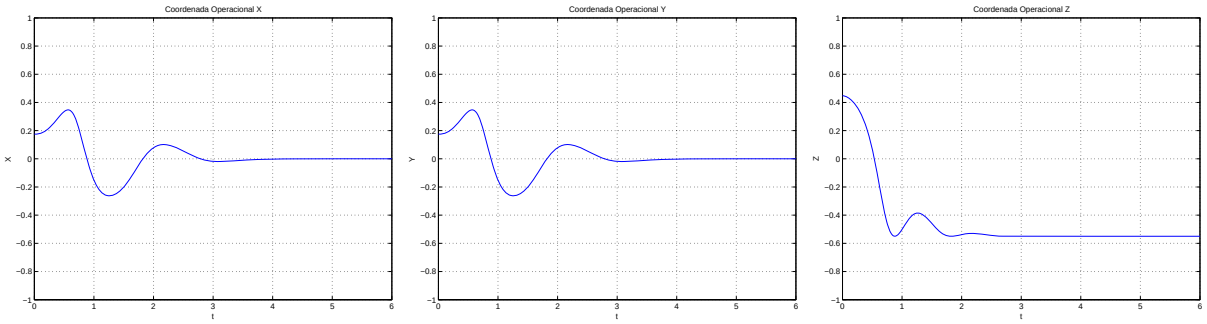


Figura 3.14: Coordenada operacional X, Y, Z en la simulación digital de las propiedades dinámicas.

La figura 3.15 representa la propiedad definida positiva ya que se cumple la condición:
$X = [X, Y, Z]^T$; $X^T H(\theta)\mathrm{X} > 0$, mostrandola positiva.

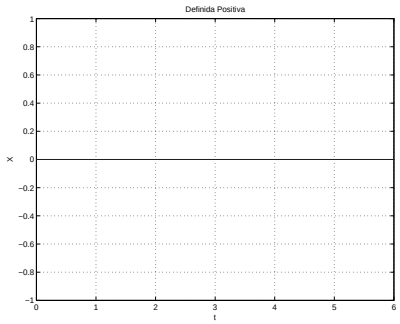


Figura 3.15: Propiedad definida positiva en la simulación digital de las propiedades dinámicas.

La figura 3.16 representa la propiedad definida como simétrica ya que se cumple $H(\theta) = H(\theta)^T$ mostrandonos su simulación.

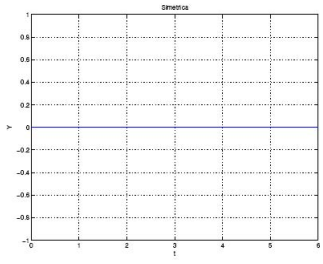


Figura 3.16: Propiedad definida simétrica en la simulación digital de las propiedades dinámicas.

La figura 3.17 representa la propiedad antisimétrica ya que se cumple:
$X^T\{H(\dot{\theta}) - 2C(\theta, \dot{\theta})\}X \equiv 0$ mostrandonos su simulación .

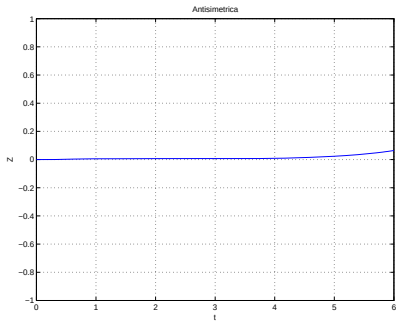


Figura 3.17: Propiedad definida antisimétrica en la simulación digital de las propiedades dinámicas.

3.5. Control

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

La metodología de diseño de los sistemas de control pueden resumirse a través de los pasos siguientes:

- Familiarización con el sistema físico a controlarse.
- Modelado.
- Especificaciones de control.

Familiarización con el sistema físico a controlarse
En esta etapa se deben determinar las variables físicas del sistema cuyo comportamiento se desea gobernar tales como temperatura, presición, desplazamiento, velocidad, etc. Estas variables reciben el nombre de salida del sistema. Además, también deben identificarse claramente aquellas variables físicas del sistema que se encuentran disponibles y que influyen en su evolución, y en particular de las salidas del sistema [35].

La figura 3.18 muestra el diagrama de bloques correspodiente al caso donde las posiciones y velocidades articulareas q y $\dot{q}$ son las salidas del robot, es decir:

$$Y = Y(q, \dot{q}, f) = \begin{bmatrix} q \\ \dot{q} \end{bmatrix} \tag{3.58}$$

mientras τ es su entrada. En esta situación, nótese que para robots con n articulaciones se tendrán que generar 2 salidas y n entradas[35].

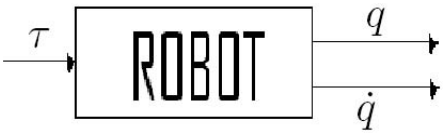


Figura 3.18: Diagrama de bloques [35].

Modelo dinámico
En esta etapa se procede a determinar la regla matemática que vincula las varibles de entrada y salida del sistema. Generalmente dicha caracterización matemática se manifiesta por medio de ecuaciones diferenciales. El modelo matemático del sistema a controlar se obtiene tradicionalmente por una de las dos técnicas siguientes: analítico y esperimental [35].

Especificaciones de control
En esta última etapa se procede a dictar las características que se desean para el sistema del control a través de los llamados objetivos de control tales como: estabilidad, regulación, seguimiento de trayectoria, optimación [35].

3.5.1. PID no lineal

Es un hecho conocido que muchas relaciones entre magnitudes físicas no son lineales, aunque frecuentemente se aproximan por medio de ecuaciones lineales por simplificación matemática. Esta simplificación puede ser satisfactoria miestras la solución resultante estén deacuerdo con los resultados experimentales. Una de las características mas importantes de sistemas no lineales, es la dependencia en el comportamiento de respuesta del sistema de la magnitud y tipo de entrada [36].

En aplicaciones de control de robots con tareas de regulación a una coordenada no singular del espacio de trabajo del manipulador, el controlador PD con compensación de gravedad y el control PID lineal tienen excelente desempeño. Para el diseño del control, es necesario diseñar una referencia nominal teta de referencia tal que en lazo cerrado el robot tenga un comportamiento estable. El control PID no lineal es de modo deslizante de segundo orden y no integrara el error si no que hara la integración de la tangente hiperbólica en el lugar de la función sgn.

Si se sabe que S es el manifold de error y que este lleva al error de posicion y velocidad, entonces tenemos que la formulación es la siguiente:

$$S = e + \varphi \frac{de}{dt} \tag{3.59}$$

Donde e esta representado de la siguiente forma:

$$e = \theta referencia - \theta real \tag{3.60}$$

Derivando al error tenemos la ecuación 3.47:

$$\frac{de}{dt} = \frac{d}{dt}(\theta referencia - \theta real) \tag{3.61}$$

Sustituyendo en S los valores 3.45 y 3.46 tenemos la siguiente:

$$S = \theta referencia - \theta real + \varphi(\theta referencia - \theta real) \tag{3.62}$$

De tal manera que obtenemos la ecuación final:

$$\tau = Kd * S + Ki \int sgn(s)dt \tag{3.63}$$

Donde:

$$lim\beta \longrightarrow \alpha \tanh(\beta S) = sgn(S) \tag{3.64}$$

Por lo tanto:

$$\tau = Kd * S + Ki \int sgn(\beta s)dt \tag{3.65}$$

Teniendo por condición las siguientes:

$$S = e + \alpha \dot{e} \tag{3.66}$$

Donde la constante:

$$\beta = 100 \tag{3.67}$$

Las ganancias estan definidas de la siguiente manera:

$$Kd = Wn^2 \tag{3.68}$$

$$\alpha = \frac{2}{Wn} \tag{3.69}$$

$$Ki << 2Wn^3 \tag{3.70}$$

Donde hay que mencionar que al refererirse a Wn es la frecuencia natural que todo cuerpo posee y se tiene como constante, dentro del modelo es una constante que determina los valores iniciales a las constantes de proporción y derivación.

3.5.2. Simulación digital del PID no lineal

Las simulaciones para evaluar el control PID no lineal se describen a continuación, parte de la Condiciòn Inicial: $q1 = 0{,}7854, q2 = -0{,}7854, q3 = -0{,}7854, qp1 = 0, qp2 = 0, qp3 = 0,\ X = 0{,}175, Y = 0{,}175 Z = 0{,}4475$, haciendo el seguimiento de una trayectoria siguiendo:

x(t)= h + rcos(wt)
y(t)=k+rsin(wt)

Donde:

h=0.3;
k=.2;
r=.1;
w=1;

La figura 3.19 representa la coordenada generalizada q1 en su espacio de trabajo donde osila en los 0.7854, aplicando el control de modos deslizantes, en donde se puede apreciar que cumple con la condición real y la deseada.

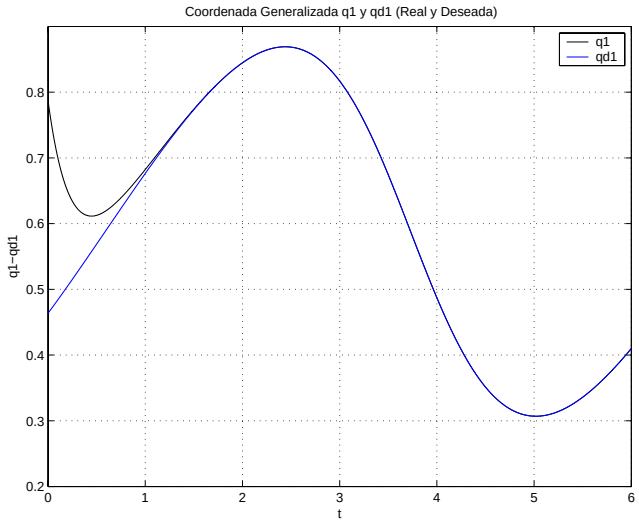


Figura 3.19: Coordenada q1 en su comportamiento PIDNL.

De igual forma la figura 3.20 representa la coordenada generalizada q2 en su espacio de trabajo en donde se puede apreciar que cumple con la condición real y la deseada, osilando en los -0.7854, aplicando el control de modos deslizantes.

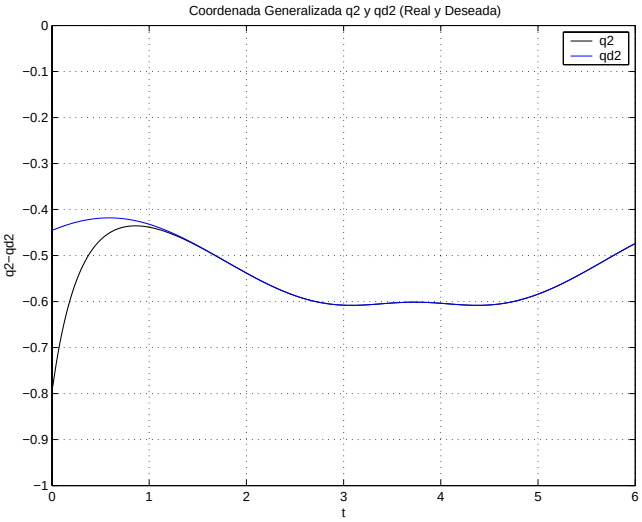


Figura 3.20: Coordenada q2 en su comportamiento PIDNL.

En la figura 3.21 hace la simulación de la coordenada generalizada q3, en donde hace una oscilación de -0.7854, aplicando el control de modos deslizantes y comportandose de acuerdo al comportamiento de la base y el hombro.

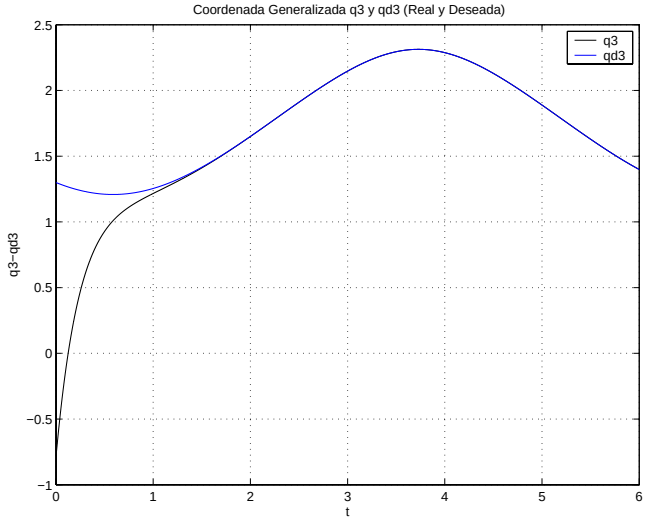


Figura 3.21: Coordenada q3 en su comportamiento PIDNL.

La figura 3.22 representa la coordenada operacional X en su espacio de trabajo haciendo que la simulación sea la deseada como lo muestra la figura, aplicando el control de modos deslizantes.

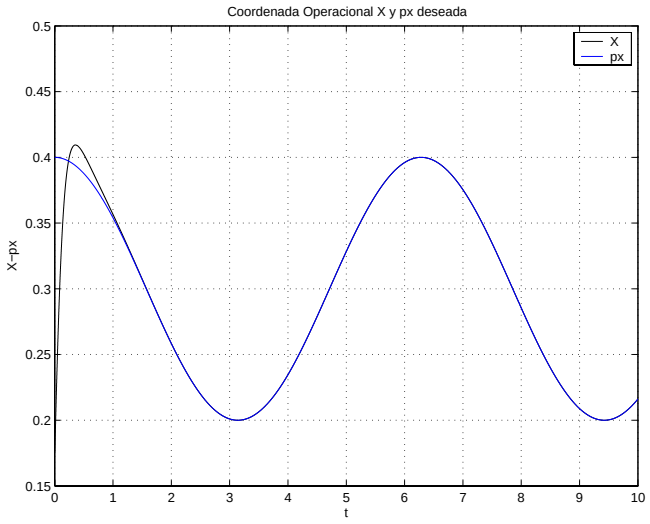


Figura 3.22: Coordenada operacional X en su comportamiento PIDNL.

La figura 3.23 representa la coordenada operacional Y en su espacio de trabajo, aplicando el control de modos deslizantes.

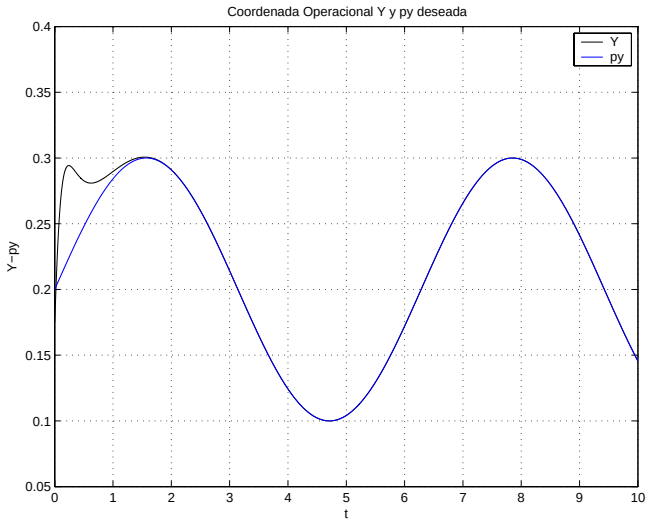


Figura 3.23: Coordenada operacional Y en su comportamiento PIDNL.

La figura 3.24 hace la simulación de la coordenada operacional Z en su espacio de trabajo, aplicando el control de modos deslizantes.

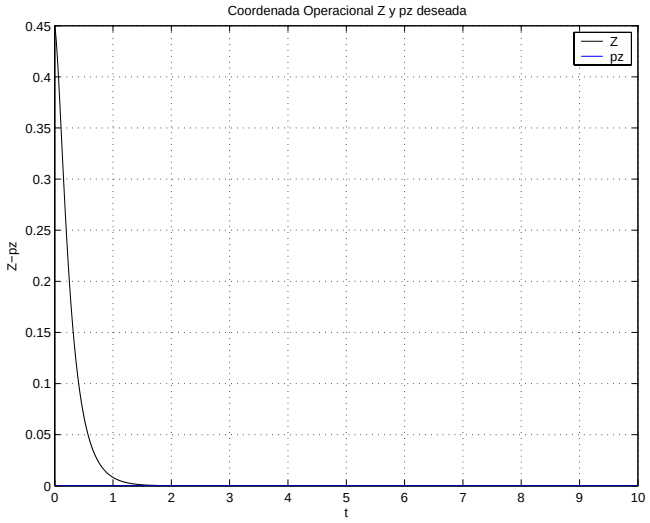


Figura 3.24: Coordenada operacional Z en su comportamiento PIDNL.

La figura 3.25 representa la simulación de error articular eq1, donde se demuestra que al aplicar el control PID no lineal este tiende a reducirse hasta casi llegar a 0, aplicando el control de modos deslizantes.

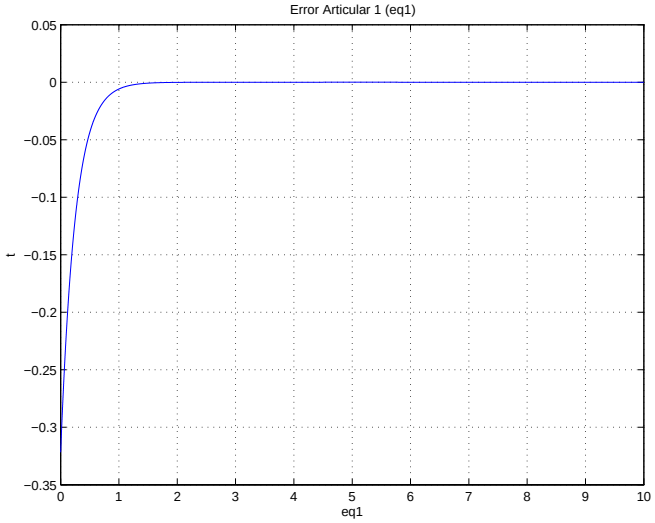


Figura 3.25: error articular 1 en su comportamiento PIDNL.

La figura 3.26 representa el error articular eq2, de igual manera tiende a llegar a 0, aplicando el control de modos deslizantes.

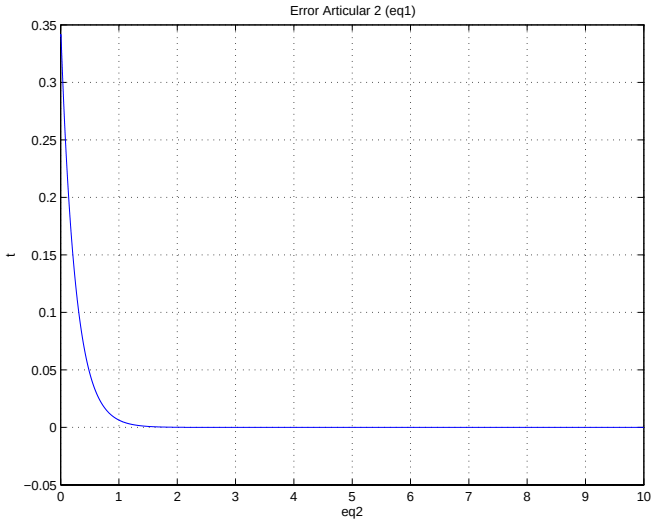


Figura 3.26: error articular 2 en su comportamiento PIDNL.

La figura 3.27 hace la simulación del error articular eq3 osilando a 0, aplicando el control de modos deslizantes.

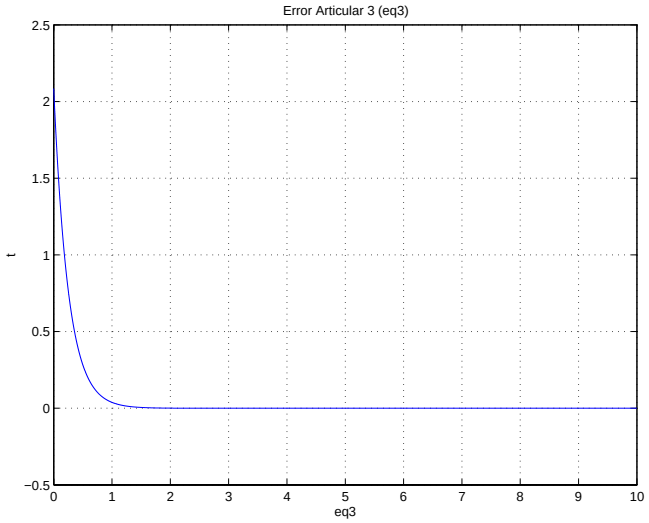


Figura 3.27: error articular 3 en su comportamiento PIDNL.

La figura 3.28 muestra la simulación de la trayectoria, mostrandonos una circunferencia en su espacio de trabajo, representando que no existen perturbaciones durante la simulación, aplicando el control de modos deslizantes.

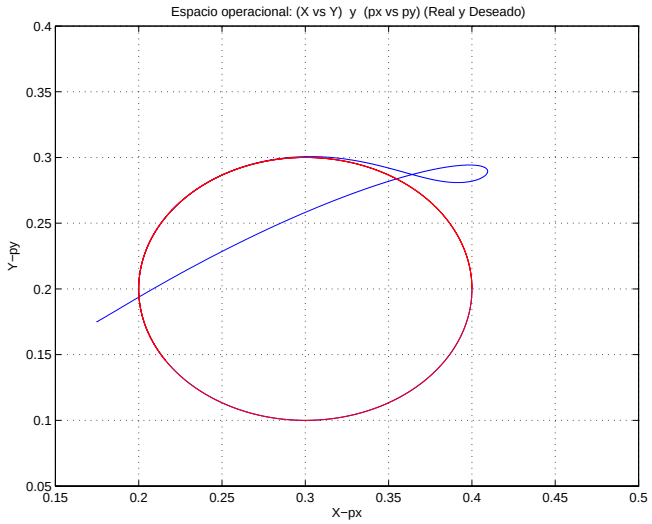


Figura 3.28: Circunferencia.

3.6. Conclusiones

Se concluyó que el robot Scorbot ER-VII es un robot de 5 grados de libertad los cuales para nuestro caso de estudios cosideramos 3 que son: base, hombro y codo ya que junto con el codo esta integrado el efector final, se derivó su cadena cinemática, se procedio a la identificación de los marcos ortonormales de referencia, posteriormente se obtuvieron los parámetros de Denavit-Hartenberg (PDH) que no son mas que la definición de posición y orientación de un marco respecto a otro, una vez obtenido estos parámetros se procedió a situarlos en una matriz de transformacion homogénea donde se obtubo la estructura de posición, posteriormente se hizo el análisis del modelo cinemático directo e inverso de posición obteniendo lo que son el posicionamiento de las coordenadas operacionales y sus variables articulares en su espacio de trabajo, haciendo la represntación matricial de Euler-Lagrange del modelo dinámico donde se definen: la matriz de inercia, la matriz de fuerza y coriolis y el vector de fuerzas gravitatorias, así como las fuerzas de fricción, para ello se mostraron su simulación digital en MATLAB y por ultimo se hizo el seguimiento de una cicunferencia aplicando el control de modos deslizantes obteniemdo un desempeño optimo.

Capítulo 4

Diseño e integración de un mundo virtual, Scorbot ER-VII.

4.1. Introducción.

En este cápitulo se describe la construcción de nuestro ambiente virtual, por medio de las diferentes herramientas, además mostrando la interfaz que se hizo con Matlab para la evaluación de la cinemática directa e inversa de posición, evaluando el control de modo deslizante, el cual es leido desde un archivo.mat en el cuadro de control de nuestro ambiente virtual.

Este ambiente virtual se desarrollo para la evaluación de controles y el metodo de enseñanza-aprendizaje de alumnos, docentes, o cualquier otro usuario que desee conocer o tenga inquietud respecto a estos temas que son de trivial interes, ya que por medio de ellos podemos simular y evaluar controles en los cuales se tenga inquietudes.

El mundo virtual fue modelado con herramientas de edición gráfica en 3D Studio Max las cuales nos permitieron crear el robot en 3D de manera similar que uno real sin ningún problema por medio de las vistas o visores, con Deep exploration generamos el codigo en Visual C++ y librerías de OpenGl para su manipulabilidad por medio de ventanas, teclado y mouse.

4.2. Herramientas de edición

En esta sección describiremos brevemente cada una de las herramientas de edición que ocupamos para la realización y construcción de nuestro ambiente virtual, ya que para nuestro caso de estudio decidimos primero utilizar 3D Studio Max 6 ya que es un modelador gráfico que nos proporciona una serie de herramientas para la construcción del robot Scorbot ER-VII, como son: primitivas estándar o extendidas, estas primitivas nos da la opción de manipularla deacuerdo a nuestra necesidad, proporcionandonos además una serie de vistas o visores. Deep Exploration nos permite hacer uso de las librerías OpenGl y Visual C++ para su manipulación ya que OpenGl y Visual c++ nos permite visualizar el gráfico 3D y manipularlo de acuerdo a nuestra necesidad.

4.2.1. 3D Studio Max 6.

3D Studio Max popular modelador de soluciones de render para cine, televisión, juegos y visualización de diseño. 3D Studio MAX cuenta con las herramientas para dar alta productividad necesaria para hacer animaciones para cine y televisión; los más vanguardistas juegos electrónicos y las más distintivas visualizaciones en diseño [38].

3d Studio max6®. Esta versión del software ofrece avanzadas posibilidades que permitirán a los profesionales

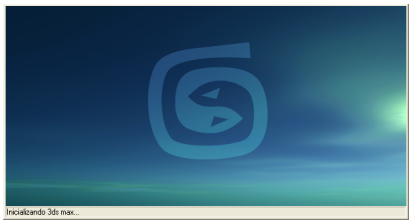


Figura 4.1: 3D Max 6 [37].

artífices de 3D, desarrolladores y diseñadores, evolucionar a un nivel más alto de calidad, sofisticación y productividad al crear los juegos más vendidos, los efectos visuales de películas renombradas o las visualizaciones complejas de diseños industriales, construcciones o simulaciones para gobierno.

Muchas de las características nuevas de 3ds max 6 han sido integradas por requerimiento directo de clientes que trabajan con efectos especiales de películas, visualización de diseños y desarrolladores de juegos, como SEGA Corporation de Japón. Los diseñadores y animadores a través de todas las industrias del diseño profesional, podrán disfrutar la profundidad y amplitud de las nuevas características de 3ds max 6, para crear sus diseños en tercera dimensión con mayor rapidez y herramientas más creativas. Las características de 3ds max incluyen: Representación esquemática avanzada para facilitar la visualización y la mejor administración de escenas complejas; el render mental ray®; vertex color painting; herramientas de diseño, visualización y de soporte interactivo con Autodesk y otras soluciones CAD y relacionadas con CAD. Integración con reactor® 2 y el sistema de física con simuladores de acrobacias y dinámica de vehículos; texturizador de malla distribuido y características que proveen gran accesibilidad al manejo de las aplicaciones a usuarios profesionales. También un sistema de partículas (Particle Flor) para crear de una forma realista fuentes, niebla, nieve, salpicado, estelas de humo, explosiones y otros efectos ambientales.

Discreet 3ds max es usado por el 80 por ciento de todos los desarrolladores de juegos, que han producido productos exitosos como: Grand Theft - Auto, Tom Clancey's Splinter Cell, Star Wars: Knights of the Old Republic, and Neverwinter Nights y continua liderando la industria de desarrollo de videojuegos. 3ds max es usado en un creciente listado de películas de gran taquilla que incluyen: X-Men II, Bulletproof Monk, The Core, Final Destination II, Jason vs. Freddy. 3ds max es la herramienta de elección para realizar visualizaciones especiales, usada por arquitectos y diseñadores de productos con la distinción de haber sido la primera herramienta de software de diseño 3D que ha sido usada por la fundación Frank Lloyd Wright en Taliesin West (Scottsdale, AZ), la figura 4.2 muestra el entorno de 3D Studio Max [37].

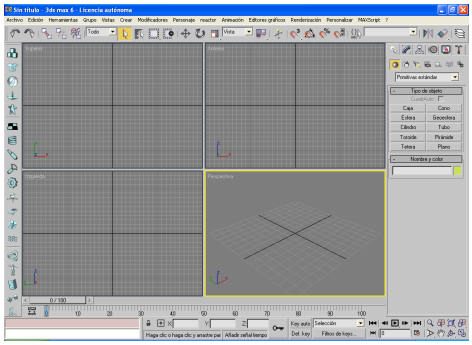


Figura 4.2: Entorno 3D Max 6.

4.2.2. Deep Exploration.

Deep Exploration se vende en dos formas: Edición CAD y Edición estandar. La Edición CAD abre e importa todos los tipos de archivo CAD, figura 4.3.

Figura 4.3: Deep Exploration [39].

La aplicación abarca la conversión del archivo principal, y un SDK está disponible para crear gráficos, dirección y herramientas de manipulación.

Deep Exploration usa la entada de datos para desplegar diversa información de los gráficos. Por ejemplo, puede ver el número de vertices, y los mapas de bits de la textura en un dibujo. Puede rotar y hace subir verticalmente para explorar los modelos 3d, agregar comentarios, cámaras, luces, pueden verse los detalles del objeto y las texturas.

Mientras que se desarrolla las aplicaciones multimedia puedes crear imagenes 3D y animaciones.

La publicación de pdf te deja rápidamente diseñar, crear, y publicar documentos interactivos pdf 3D sin los gráficos que usan de programación 3D proveídos por la edición estándar de Deep Exploration o la edición del cad, ya que permite publicar, ver y compartir los gráficos 3D usando Microsoft PowerPoint, Word, Excel y Adobe® Acrobat®.

El DAO de Exploración es el uso de escritorio de Dirección de Gráfica de Producto líder mundial más avanzado. Deep Exploration deja: Transformar modelos gráfica 2D y de 3D [2].

4.2.3. Visual C++.

Microsoft Visual C++ es un entorno de programación en el que se combinan la programación orientada a objetos (C++) y el sistema de desarrollo diseñado especialmente para crear aplicaciones gráficas para Windows (SDK).
Visual C++ es un paquete para desarrollar aplicaciones que incluye, como características más sobresalientes:

- Una bibliteca de clases, MFC, que da soporte a los objetos de Windows como ventanas, cajas de diálogo, controles, así como a los objetos GDI (Graphic Device Interface)tales como lápices, pinceles, fuentes y mapas de bits.
- Un entorno de desarrollo integrado (editor de texto, compilador, depurador, explorador de código fuente, administrador de proyectos, etc).
- El editor de textos ayuda a complementar cada una de las sentencias visualizando la sintaxis correspondiente a las mismas.

- Asistente para el desarrollo de aplicaciones como AppWizard, editores de recursos (editor de menús, de diálogo, de tablas de cadena de caracteres, de tablas de aceleradores y de objetos gráficos como mapas de bits o iconos), ClassWizard, ControlWizard, WizarBar y ATL COM Wizard (Activate Template Library Component Object Model Wizard).
- Galeria de objetos incrustados y vinculados (OLE - Object Linking and Embedding). Esto es, software para reutilizar en cualquier aplicación. Asi mismo, Visual C++ proporciona soporte para diseñar componentes software a medida.
- Visualización y manipulación de datos de otras aplicaciones Windows utilizando controles OLE.
- Una interfaz para múltiples documentos (MDI - Multiple Document Interface) que permite crear una aplicación con una ventana principal y múltiples ventanas de documento.
- Personalización AppWizard.
- Cabeceras precompiladas que reducen el tiempo de compilación.
- Editar y continuar. Durante una sesión de depuración, se pueden realizar modificadores en el código de la aplicación sin tener que salir de dicha sesión, recompilar y reiniciar la depuración. Los cambios son recompilados y aplicados a la aplicación que se está ejecutando.
- Nuevas clases para la programación de hilos (threads), para implementar páginas HTML, etc.
- Creación y utilización de bibliotecas dinámicas (DLL - Dynamic Link Libraries).
- A partir de la vesión 4.2, Visual C++ integra la biblioteca estándar de C++ e incorpora soporte para la programación de aplicaciones para internet; forma parte de este soporte la tecnología de componentes activos (ActiveX).
- Soporte para el estándar COM (Component Object Model - modelo de objeto componente; en otras palabras, componente software) al que pertenecen los componentes activos (ActiveX o formalmente contoles OLE).
- Objetos de acceso a datos (DAO) que permiten acceder a bases de datos a través del motor de Access o de controladores ODBC.
- OLE DB como un proveedor de datos y objetos ADO (ActiveX Data Objects - objetos ActiveX para acceso a datos), como tecnología de acceso a datos, para satisfacer los nuevos escenarios demandados por las empresas, tales como los sistemas de información basados en la Web.
- Soporte para aplicaciones que interaccionen con Internet a través de la API para Internet de Windows (biblioteca WinInet).

Visual ofrece un entorno de desarrollo de C/C++ integrado en Windows, que permite construir aplicaciones Windows (.EXE), aplicaciones de consola (.EXE), bibliotecas de enlace dinámico (:DLL), bibliotecas estáticas (.LIB), modelos AppWizard personalizados (.AWX), controles activos (ActiveX) y otros. La siguiente figura 4.4 muetra este entorno de desarrollo. [40]

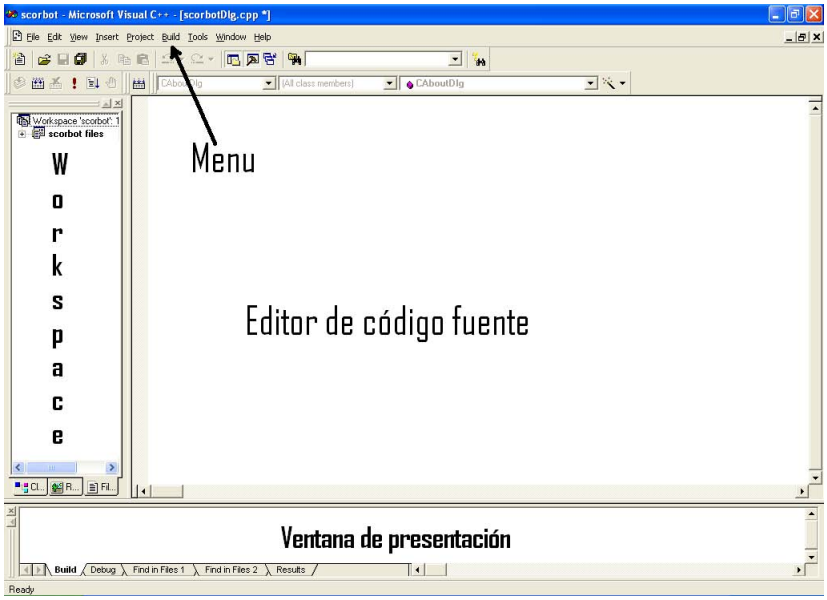


Figura 4.4: Entorno de desarrollo de Visual C++.

4.2.4. OpenGl.

OpenGL, desarrollado originalmente por Silicon Graphics Incorporated (SGI) para sus estaciones de trabajo de gráficos, permite que las aplicaciones creen imágenes en color de alta calidad independientes de los sistemas de ventanas, los sistemas operativos y el hardware, figura 4.5 [37].

Figura 4.5: OpenGl [41].

Estándares. La Organización del estándar de OpenGL es mantenido por un grupo independiente llamado Architectural Review Board (ARB)un consorcio independiente de estándares formado en 1992, gobierna el API de OpenGL, que incluye representantes de la DEC, IBM, Intel, Silicon Graphics y Microsoft. El ARB define pruebas de conformidad, aprueba las especificaciones para las nuevas características y extensiones de OpenGL [37].

Definición de OpenGL.
OpenGL es un interfaz de software que permite a programadores de gráficos crear imágenes tridimensionales en color de alta calidad con efectos gráficos como la iluminación, trazar un mapa de textura y transformaciones de matriz. OpenGL es un estándar abierto diseñado para correr sobre una variedad de ordenadores y de sistemas operativos.

Cuando es aplicable.
OpenGL se construye para la compatibilidad a través del hardware y de los sistemas operativos. Esta arquitectura hace fácil los programas de OpenGL de un sistema a otro. Mientras que cada sistema operativo tiene requisitos

únicos, el código de OpenGL en muchos programas puede ser utilizado como es para el uso de OpenGL en sistemas operativos de Microsoft Windows NT, Windows 2000, Windows 95, y Windows 98, tendrá que modificar sus programas para trabajar con Windows NT / Windows 2000 y Windows 95/98.

Developer Audience
Diseñada para el empleo por programadores C/C ++, OpenGL requiere la familiaridad con Windows el interfaz de usuario gráfico así como la arquitectura conducida por mensaje. OpenGL requiere Windows NT de Microsoft, Windows 2000, o Windows 95/98. [38]

GLUT de OpenGL.
El GLUT es una herramientas para uso general de OpenGL, una herramientas independiente del sistema para la escritura de los programas de OpenGL. El GLUT proporciona un API portable así que se puede escribir un solo programa de OpenGL que trabaje a través de todas las plataforma del SO de la PC y del sitio de trabajo. [41]

4.3. Construcción del escenario virtual

En la construcción del escenario virtual es empleado Visual C++ y OpenGL se requiere de instrucciones tanto para poder hacer gráficos como la programación del funciones que nos permitiran interactuar en el ambiente y manipularlo, para ello mencionaremos las sentencias utilizadas.

La siguiente instrucción nos permite crear nuestro escenario virtual en donde posicionaremos nuestro robot Scorbot ER-VII:

```
glBegin( Tipo );
        glVertex3f(x, y, z);
        glVertex3f(x, y, z);
        glVertex3f(x, y, z);
        glVertex3f(x, y, z);
glEnd();
```

Esta función dibuja un polígono que se encapsula entre las funciones **glBegin** y **gLEnd**. El parámetro que recibe la primera sirve para decirle a OpenGL que tipo de polígono deseamos crear, que pueden ser:

- `GL_POINTS:`

  para que todos los vértices indicados entre ambas funciones se dibujen por separado a modo de puntos "libres".

- `GL_LINES:`

  para dos vértices definidos, se traza automáticamente una línea que los une.

- `GL_POLYGON:`

  se unen todos los vértices formando un polígono.

- `GL_QUADS:`

  cada 4 vértices se unen para formar un cuadrilátero.

- `GL_TRIANGLES:`

  cada 3 vértices se unen para formar un triángulo.

- `GL_TRIANGLE_STRIP:`

  crea un triángulo con los 3 primeros vértices entoces sucesivamente crea un nuevo triángulo unido al anterior usando los dos últimos vértices que se han definido y el actual.

- GL_QUAD_STRIP:

  igual que el anterior pero con los cuadriláteros.

GL_POINTS,GL:LINES,GL_POLYGON,GL_QUADS,GL_TRIANGLES,GL_TRIANGLE_STRIP,GL_QUAD_STRIP

son constantes definidas en la librería.

```
glVertex3f(0.0, 0.0, 0.0);
```

Esta función crea un vértice situado en el origen de coordenadas del mundo, es decir, x = y = z = 0.0. En el primer caso el nombre de la función termina en "3f"(3 floats). Esto significa que vamos a especificar el vértice con 3 valores o variables de tipo real, o sea float. Si trabajamos en 3D especificamos las coordenadas del vértice en este orden: X, Y, Z. Si deseamos trabajar en 2D solo tenemos que hacer una coordenada igual a 0.0, normalmente la Z.

Para poder colocar texturas a los objeto, lo primero que tenemos que hacer es crear un arreglo en el cual especificamos el nombre de la textura que deseamos colocar, las cuales debe estar en las mismas carpetas en donde se encuentra el proyecto, este arreglo es de tipo static, posteriormente se crea la función la cual nos permite asignar parámetros para la manipulación de las texturas en el ambiente virtual

La textura se guarda en un archivo de tipo .bmp en el mismo directorio del proyecto. Se escribe el nombre del archivo de la textura y el identificador de OpenGL al archivo de .cpp. La estructura de texture_maps de sample_TEXTURE se usa para guardar la información sobre los mapas de la textura. Todos los mapas de la textura se combinan en una sola serie llamado el texture_maps.

```
static sample_TEXTURE texture_maps [1] = {
    {"texture.bmp",0}
};

void LoadTexture(char*filename)
 {
  DIB2D dib;
  GLTXTLOAD load;
  if(LoadDIB(filename,&dib))
   {
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);

    if(ScaleImage(dib,load))
     {
   glTexImage2D(GL_TEXTURE_2D,0,load.perpixel,
                load.Width,load.Height,0,
                load.format,GL_UNSIGNED_BYTE,
                load.bits);
                delete load.bits;
     }else{
   glTexImage2D(GL_TEXTURE_2D,0,load.perpixel,
                dib.Info->biWidth,dib.Info->biHeight,
                0,load.format,GL_UNSIGNED_BYTE,dib.bits);
     }
   delete dib.Info;
   }
 };
```

4.3.1. Animación al ambiente virtual

Para poder animar nuestro ambiente virtual es necesario de la utilización de funciones como glTranslatef, glRotatef, glScalef. El glTranslatef(x,y,z) nos permite trasladar en x, y, z sobre la matriz actual, en la figura 4.6, el cubo original es de color rojo y está centrado en el origen. Al cubo verde se le ha aplicado una traslación de 30 unidades positivas siguiendo la dirección del eje X. Obviamente el tamaño se conserva pues no hemos aplicado ningún escalado.

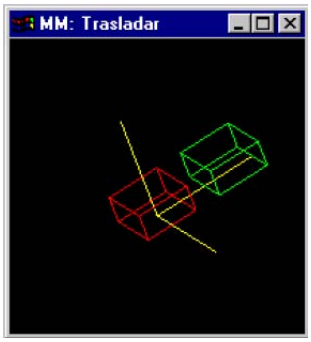


Figura 4.6: Trasladar un cubo.

glRotatef, se define mediante la primitiva glRotatef(alpha,x,y,z)en donde multiplica la matriz actual por una matriz de rotación de alpha grados respecto al eje x, y, z. En la figura 4.7, el cubo rojo sigue estático en el origen. El cubo verde tiene exactamente las mismas dimensiones pero se ha rotado 45 grados alrededor del eje vertical, que en este caso es el eje Y y no el Z.

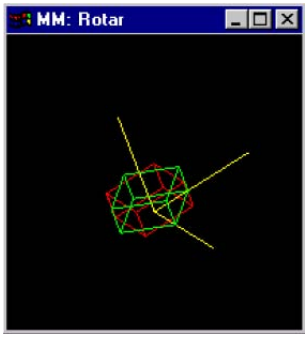


Figura 4.7: Rotar un cubo.

glScalef definiendose mediante la primitiva glScalef(x,y,z) que es el escalado de cada uno de los ejes. La figura 4.8, el cubo original es de color rojo y esta centrado en el origen. El cubo escalado es de color verde. Es 2 veces mayor que el origen al haber efectuado un escalado de 2.0 sobre todas sus componentes.

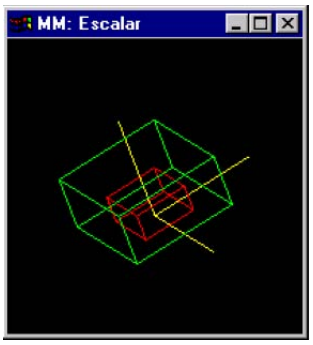


Figura 4.8: Escalar un cubo.

La función void glutPostRedisplay(void);

Esta función envía un evento de redibujado. En donde indica que la ventana actual necesita ser redibujada, si no se incluye esta rutina OpenGL no sabría que tiene que dibujar la ventana cuando se ha presionado una tecla. La función 'callback' registrada por glutDisplayFunc() será llamada. La última llamada a glutPostRedisplay() hace que tras la actualización se dibuje el frame.

La última función a la que se debe llamar en el main es el bucle de eventos glutMainLoop(void). Todas las ventanas que han sido creadas serán mostradas por esta función y el rendering de la ventana. Esta función es la encargada de pasar el control de flujo del programa a la GLUT y siempre que ocurra un evento seran llamadas las funciones callbacks hasta que se cierre la ventana.

La función display definida como función callback para dibujar la ventana cada ves que sea necesario es donde mandamos llamar a los objetos que conforman el escenario virtual, tanto los creados en 3d Studio Max como los creados con OpenGL además de asignar la posición y tamaño del escenario, aunque este podría estar definido en el momento de crearse, sin embargo por comodidad y su fácil manipulación aquí es donde están asignadas esas propiedades, para lo cual utiliza las siguientes funciones: glRotatef, glTranslated, glScaled.

Como estamos utilizando una ventana en modo de doble buffer utilizaremos la función glutSwapBuffers(), esta función es la encargada de intercambiar el buffer posterior con el buffer anterior. Cuando se dibuja cualquier figura, esta es dibujada en el buffer posterior es decir el que está atrás y cuando el dibujo está terminado los dos buffers se intercambian.

La función glutDisplayFunc(display); Es el primero y mas importate evento de una función callback, define que la función display que es pasada como argumento sea ejecutada cada vez que GLUT determine que la ventana debe ser dibujada. Siempre que OpenGL determina el contenido de una ventana que necesita ser revisualizada, la función callback registrada por glutDisplayFunc() es ejecutada. Por tanto, se debe poner en todas las rutinas en las que se necesite redibujar la escena. Cuando glutPostRedisplay() es llamado hace que glutMainLoop() pase el control de flujo del programa a la GLUT y comienza un bucle infinito así, llame a display() y renderise la escena después de que los demás eventos han sido procesados. La figura 4.19 muestra todas estas aplicaciones.

4.3.2. Construcción del robot virtual

La construcción del robot se hizo de la siguiente manera, como se muestra en la figura 4.9.

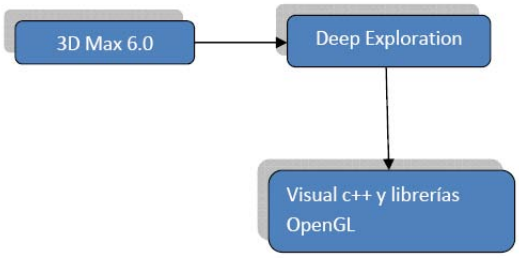


Figura 4.9: Diagrama de construcción del robot.

Modelado en 3D Max 6

Para poder hacer la construcción del robot Scorbot ER-VII utilizamos las primitivas extendidas y las primitivas estándar como lo muestra la figura 4.10.

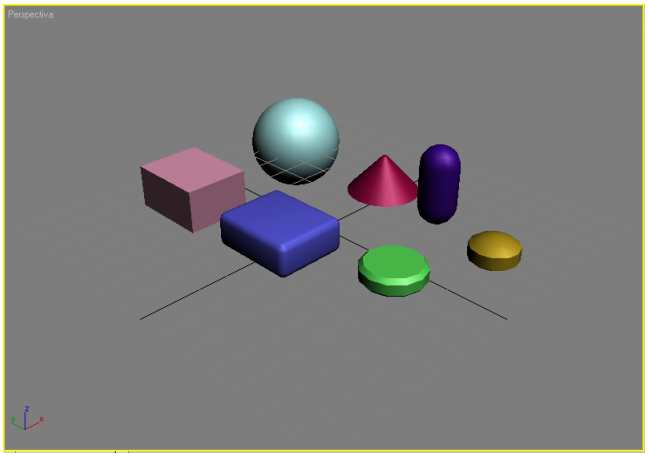


Figura 4.10: Primitivas estándar y primitivas extendidas.

Estas figuras tiene la opción de deformarción, tamaño, color, luz, etc., que son herramientas para darles una mejor presentación y renderizado al diseño. Estos cuerpos geométricos fueron deformados y en el mayor de los casos utilizamos las primitivas extendidas ya que nos permiten darle mas renderizado al ambiente por medio de los parámetros de empalme y segmentos de empalme. Primero antes que nada hicimos la construccion y modelado de la base, las medidas las tomamos del robot Scorbot ER-VII en escala 1:1 figura 4.11.

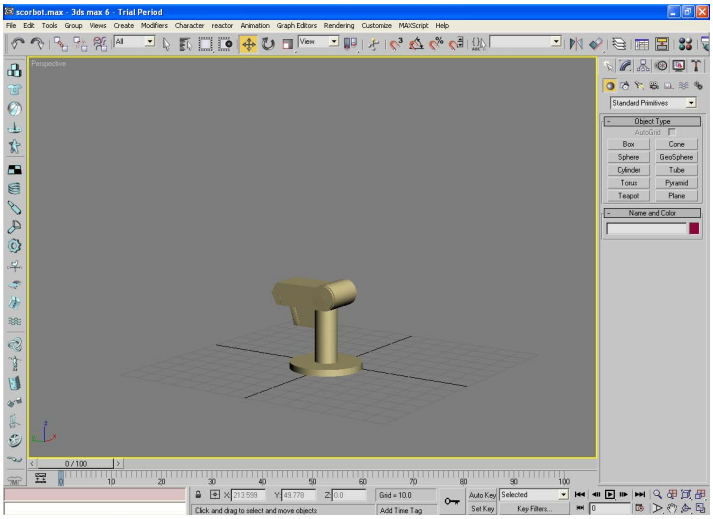


Figura 4.11: Construcción de la base del Scorbot ER-VII.

Así mismo modelamos y contruimos el hombro y codo como lo muestra la figura 4.12, utilizando las herramientas de diseño y guiandonos por medio de las vistas.

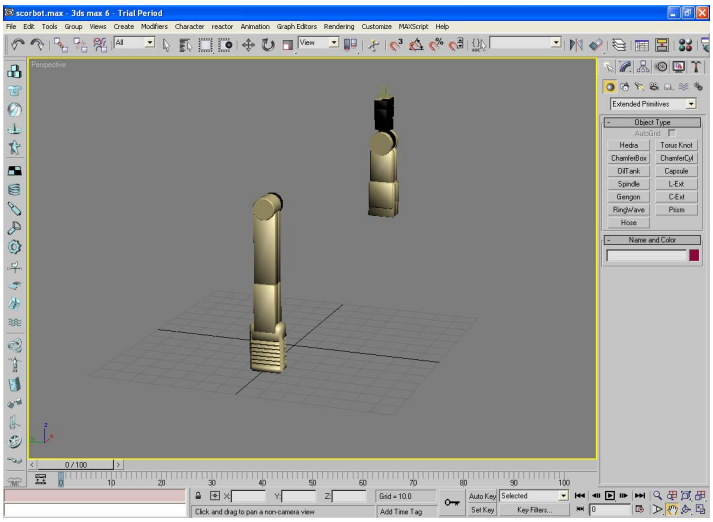


Figura 4.12: Hombro y codo del robot.

Una vez modelado y manipulado las piezas que componene al robot unimos las piezas, con la ayuda de las diferentes vistas o visores como lo muestra la figura 4.13 podemos verlo de diferentes ángulos, superior izquierdo muestra una vista posterior del robot, superior derecha muestra una vista superior, la parte inferior izquierda muestra una vista de izquierda y la inferior derecha una vista de perspectiva para que quedemos completamente convencidos de que en realidad esta totalmente unido para poder exportarlo ya que podemos orientarnos al momento de estar modelando el robot, y ver cada uno de los cambios en el ambiente virtual en las otras tres.

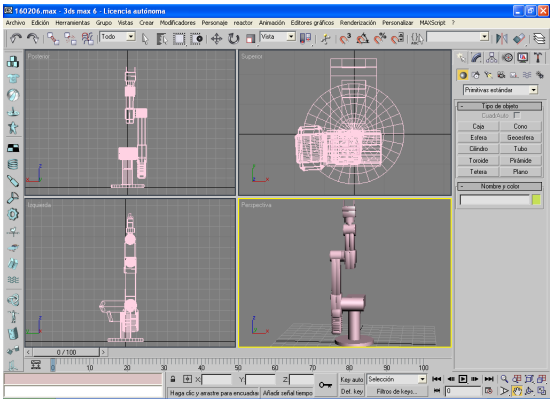


Figura 4.13: Vistas de 3D Max 6.

Deep Exploration

Por medio de este software abrimos la exportación realizada en 3D Studio Max 6, y generamos una aplicación en Visual C++. Además de que se pueden agregar texturas, colores y diseños detallados de entorno del robot. Una vez ya hecha la exportación, generar un Archivo de código fuente de C++, un Archivo de proyecto y un Espacio de trabajo de proyecto, el cual nos va a permitir manipular el robot de acuerdo a nuestras necesidades y la utilización de las librerias de OpenGL para la realización de gráficos y eventos de la ventana, teclado y mouse, figura 4.14.

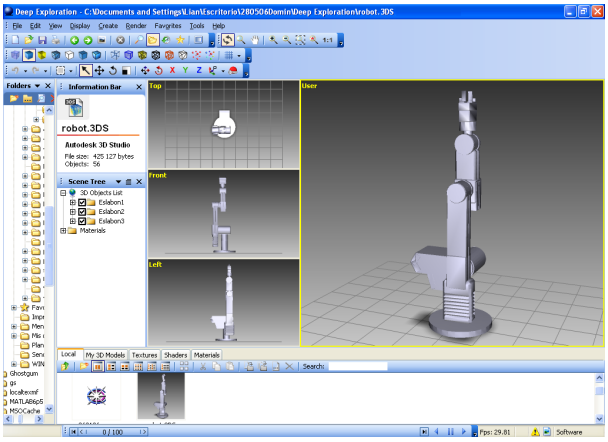


Figura 4.14: Deep Exploration.

OpenGl y visual C++

OpenGL es un API en donde se siguen una serie de pasos para lograr un efecto, este API incluye 120 comandos y funciones. El lenguaje de programación de Visual C++ permite la programación de OpenGL. Para ello se requiere del llamado de las librerías de OpenGL.

Y la instalación de los glut, estos son glut32.lib, glut32.dll y glut.h, estas deben de ser instaladas en el sistema ya que permitirán el manejo de ventanas e interacción por medio del teclado y el ratón. La figura 4.15 muestra el ambiente virtual como lo arroja por primera vez, el robot de un inicio, el cual no tiene texturas, efectos, etc., además de que el evento WM_TIMER hace que el robot tenga movimiento continuo.

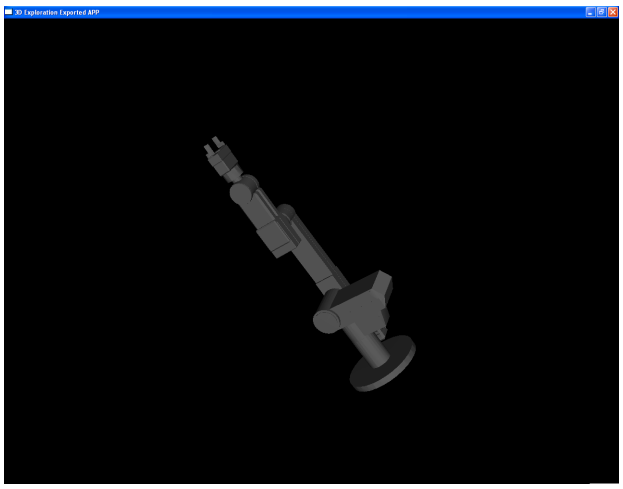


Figura 4.15: Ejecución del programa.

Una vez creado el codigo Scorbot.dsw procedemos a su manipulación. El código que fue generado por medio de Deep Exploration es muy extenso por lo que para un mejor manipulación del programa del robot este fue agrupado en tres eslabones y los guardamos como librerias, llamandolos, de la siguiente forma:
#include "1.h", para la base
#include "2.h", para el hombro
#include "3.h", para el codo, pich y roll

Creadas estas librerías son llamamadas para ser integradas y hacer la construcción del robot, void Base(); esta compuesta por todos los vertices que integran a la base del robot, void eslabon1(); esta compuesto por todos los vertices que integran al hombro del robot y void eslabon2(); esta compuesto por todos los vertices que integran al codo, pich y roll.

4.3.3. Animación al robot virtual

Para poder dar animación al robot como ya se dijo anteriormente se hace por medio del glTranslatef(x,y,z) y glRotatef(alpha,x,y,z), estas funciones se colocan variables que controlan la función y orientación de cada objeto en el ambiente virtual, para manipular estas variables se hace de dos formas:

- Desde el teclado.
- Desde un archivo.

Desde el teclado

Para hacer una interacción desde el teclado y utilizar las 256 teclas y sus posibles combinaciones, se hace lo siguiente:

```
  void keyboard(char key, int x, int y)
{
    switch(key)
    {
        case ´´:
    }
    glutPostredisplay():

 }
```

La interacción por medio de mouse, lo primero que debemos tener en cuenta es el poder saber cuando esta o no presionado el botón, una vez que ya sabemos cuando esta presionada mandamos llamar a la función display-MouseClick();, donde se realizara la actualización de las coordenadas, como se muestra:

```
  void DisplayMouseMotion(int x, int y)
{
    ......................

 }


  void DisplayMouseClick(int button, int state, int x, int y)
{
    ......................
    glutPostRedisplay():

 }
```

Desde un archivo

Para manipular las variables que controlan la translación y orientación desde un archivo, es necesario que ya se tenga, el protocolo para leer los datos de un archivo es el siguiente:

- El nombre del archivo debe ser circunferencia.mat
- El tipo de archivo es de texto, ASCII.
- El archivo debe de contener 7 datos separados por un espacio en blanco.
- El orden de dato debe de ser: tiempo, Xd = X deseada, Yd = Y deseada, Zd = Z deseada, teta d1 θ_1 deseada, teta d2 θ_2 deseada, teta d3 θ_3 deseada.

Cuando se selecciona la opción de visualización en el ambiente virtual y posteriormente se presiona el botón circunferencia los datos se descargan cuando se selecciona el botón graficar, los datos se van leyendo de 7 en 7 verificando siempre que el primero no sea un dato nulo, el siguiente codigo muestra la lectura del archivo:

```
  if( Aentrada.eof() )
{
    Aentrada.close();
    Aentrada.open("circunferencia.mat");
    Aentrada.close();
}
```

4.3.4. Construcción del cuadro de control

El cuadro de control fue realizando en un nuevo proyecto en Visual C++, el cual fue programado de la siguiente manera: lado superior izquierdo corresponde a X, Y, Z, teta 1, teta 2 y teta 3 que son los valores instantaneos correspondientes al MCDP, lado superior derecho Xd, Yd, Zd, teta d1, teta d2 y teta d3 son los valores deseados, correspondientes al MCIP. Los errores instantaneos que son: ex, ey, ez, eteta 1, eteta 2, eteta 3 son son errores mínimos que el robot debe de tener al aplicarle los controles. La dimánica del robot que es m1, m2 y m3 corresponde a la masa de cada uno de los eslabones, L1, L2 y L3 son las longitudes de los eslabones, LCG1, LCG2, LCG3 son las longitudes a los centros de gravedad de las masas, b1, b2 y b3 es la fricción viscosa del robot, en esta parte que es la dinámica del robot los valores son fijos ya que describe las propiedades que el robot tiene. Por último los botones que son: Circunferencia, Gráficar, Borra y Salida. El boton Circunferencia llaman un archivo .mat correspondiente a su nombre, el boton Gráfica ejecuta la acción y simula en el ambiente virtual la selección, el boton Borrar, borra la trayectoria realizada y por último el botón Salir, nos saca de la aplicación. El cuadro de contol se muestra en la figura 4.16:

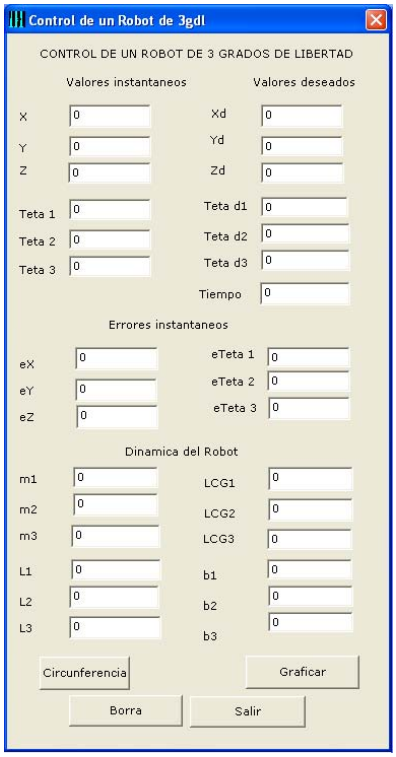


Figura 4.16: Cuadro de control.

4.3.5. Integración y validación de los modelos matemáticos y de control en el robot virtual

Las formulas del MCDP, MCIP, parametros Denavit Hartenberg y el control de modos deslizantes que hemos desarrollado en el capítulo 3, procederemos ahora a ejecutarlos en visual C++ y visualizarlas en el cuadro de control para que el robot Scorbot ER-VII haga su simulación.

Dentro de la función void mcdp();, programaremos los modelos matemático del robot y la lectura del control, siendo visualizados en el cuadro de control, el siguiente codigo muestra la programación de los valores instantaneos que son: X, Y, Z, Teta 1, Teta 2, Teta 3 y el control, lo que representara el bloque que corresponde a los valores deseados que son xd, yd, zd, teta d1, teta d2 y teta d3 quedando definido de la forma siguiente:

```
void mcdp()
{
    for(int paso=0;paso<350;paso++)
    {
        mcdpX = cos (rotabase) * (a3 * (cos (rotaesla1)
        + cos (rotaesla2) ) + a2 * cos (rotaesla1) + a1);
        mcdpY = sin (rotabase) * (a3 * (cos (rotaesla1)
        + cos (rotaesla2) ) + a2 * cos (rotaesla1) + a1);
        mcdpZ = d1 - a2 * sin (rotaesla1) - a3 * ( sin (rotaesla1)
        + sin (rotaesla2) );

        Aentrada>>n;
        s->m_xd=n;
        s->m_X=mcdpX;

        Aentrada>>n;
        s->m_yd=n;
        s->m_Y=mcdpY;

        Aentrada>>n;
        s->m_zd=n;
        s->m_Z=mcdpZ;

        if( Aentrada.eof() )
        {
            Aentrada.close();
            Aentrada.open("circunferencia.mat");
            Aentrada.close(); //bien

        }

        s->m_slabon1=rotabase;
        s->m_slabon2=rotaesla1;
        s->m_slabon3=rotaesla2;

        mcdpX=s->m_xd;
        mcdpY=s->m_yd;
        mcdpZ=s->m_zd;
        posIni=sqrt(pow((finX-mcdpX),2)+pow((finY-mcdpY),2)+pow((finZ-mcdpZ),2));
        if(posIni>0.01 && arr<LV)
        {
          arr++;
```

```
          arrX[arr]=mcdpX;
          arrY[arr]=mcdpY;
          arrZ[arr]=mcdpZ;
          finX=mcdpX;
          finY=mcdpY;
          finZ=mcdpZ;
        }
        error();
    }
    dinamica();
    s->UpdateData(false);
    glutPostRedisplay();
}
```

Los errores instantaneos son programados por medio de una función void error();, de la manera siguiente:

```
void error()
{
        s->m_et11= (s->m_td1) - (s->m_slabon1);
        s->m_et2= (s->m_td2) - (s->m_slabon2);
        s->m_et3= (s->m_td3) - (s->m_slabon3);

        s->m_eX = (s->m_xd) - (s->m_X);
        s->m_eY = (s->m_yd) - (s->m_Y);
        s->m_eZ = (s->m_zd) - (s->m_Z);

}
```

La dinámica que representa las caracteristicas de los eslabones que definen su masa, longitud, longitud de gravedad de los centros de masas y la fricción biscosa, se programa a través de una función representada a continuación:

```
void dinamica()
{

    s->m_ma1=0.45;
    s->m_ma2=0.25;
    s->m_ma3=0.15;

    s->m_long1=0.35;
    s->m_long2=0.35;
    s->m_long3=25.0;

    s->m_lcg1=0.5;
    s->m_lcg2=0.5*12;
    s->m_lcg3=0.5*13;
    s->m_b1=0.3;
    s->m_b2=0.15;
    s->m_b3=0.05;
}
```

4.3.6. Programación de los botones del cuadro de control

La programación de los botones que son Circunferencia, Graficar, Borrar y Salir nos van a permitir manipular y ejecutar aplicaciones en el ambiente virtual, el botón circunferencia llama el archivo circunferencia.mat, el botón

Grafica va a graficar la acción leida desde el botón Circunferencia es decir va hacer el seguimiento de la trayectoria de la circunferencia, el botón borrar permite borrar la tarea programada y por último el boton salir al pulsarlo cerrara la plicación.

```
void CScorbotVIIDlg::OnCircunferencia()
{
    // TODO: Add your control notification handler code here

        init=TRUE;
        dibT= FALSE;
        arr=0;
        init=FALSE;
        dibT= TRUE;
        strcpy(nomArch,"circunferencia.mat");

}

void CScorbotVIIDlg::OnGraficar()
{

    init=TRUE;
    dibT= FALSE;
    arr=0;
    init=FALSE;
    dibT= TRUE;
    s =this;

    UpdateData(true);
    UpdateData(false);

    Aentrada.open(nomArch);
    if(Aentrada.fail()) {
        return;
    }

    //////glutDisplayFunc( display);
    // inicialización de los datos del programa
    inicializar();
    glutIdleFunc( mcdp );
    glutMouseFunc(DisplayMouseClick);
    glutMotionFunc(DisplayMouseMotion);
    glEnable(GL_DEPTH_TEST);
    glutKeyboardFunc(key);
    glutSpecialFunc( specialKeys);

}


void CScorbotVIIDlg::OnBorra()
{
    // TODO: Add your control notification handler code here
```

```
        init=TRUE;
        dibT= FALSE;
        arr=0;
        init=FALSE;
        dibT= TRUE;
}

void CScorbotVIIDlg::OnSalir()
{
    // TODO: Add your control notification handler code here
    Aentrada.close();
    exit(0);
}
```

4.3.7. Modelo conceptual

La interfaz de visualización se planeó para observar el desempeño de los modelos matemáticos y de control, la figura 4.17 muestran los modelos conceptuales y el modo de mover el robot Scorbot ER-VII virtual.

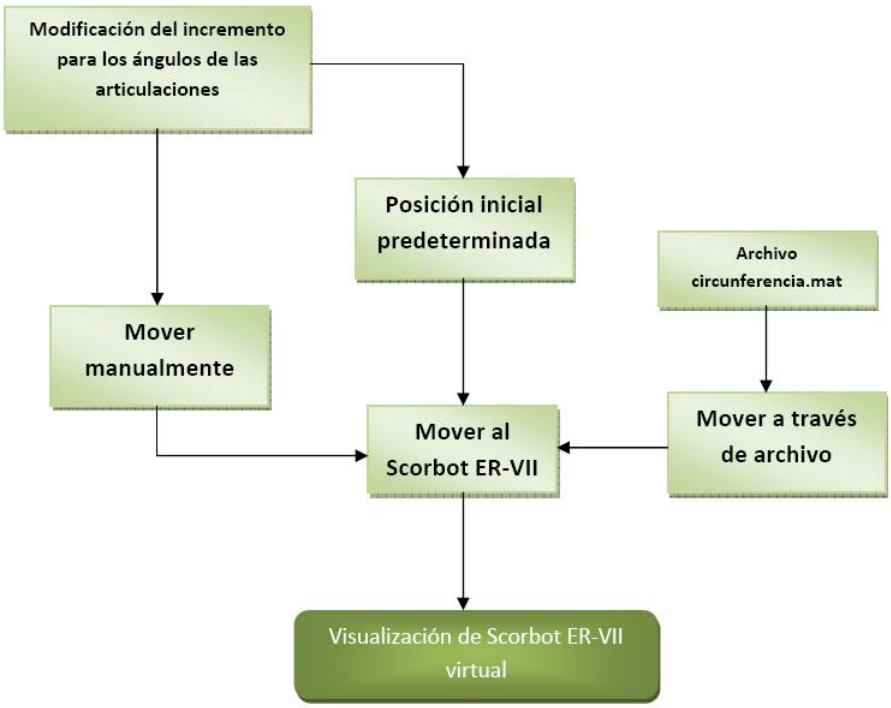


Figura 4.17: Opciones para mover al Scorbot ER-VII virtual.

Como lo representa la figura anterior, el modo de manipular al robot Scorbot se hace por medio de:

- Archivo
- Manualmente
- Regresarlo a su posición inicial predeterminada una vez que ya se hizo la tarea programada.

A través del archivo; esto se puede llevar a cabo mediante el archivo circunferencia.mat en donde se hace la trayectoria de una circunferencia y en este archivo solo se guardan como ya se dijo anteriormete valores de 7 en 7 que representan los datos de x, y, z, teta 1, teta 2, teta 3.

Posición inicial predeterminada; esta opción regresa al Scorbot a la posición inicial y se hace por medio de la tecla F2.

Mover al robot manualmente; es donde el usuario a través del teclado puede rotar las articulaciones del robot y manipularlas.

La figura 4.18 muestra un diagrama de visualización del enlace de MATLAB con la interfaz de visualización para la generacion de archivos.mat y su lectura desde el ambiente virtual simulando la trayectoria de la circunferencia, Como lo muestra la figura primero se programa el control de modos deslizantes utilizando los modelos matematicos que describen la cinematica directa e inversa de posició, velocidad y aceleración obteniendo además la dinámica de Euler-Lagange haciendo su validación en el control de modos deslizantes simulando el seguimiento de una trayectoria de una circunferencia, todo esto se obtiene con MATLAB, ya que MATLAB genera un archivo circunferencia.mat que contiene 7 columnas que son de tiempo, X, Y, Z, teta1, teta2, teta3 que son leidos por visual c++ y librerías de OpenGL haciendo el seguimiento de la trayectoria en el ambiente virtual, este archivo tiene que estar dentro de la carpeta del 1robot.

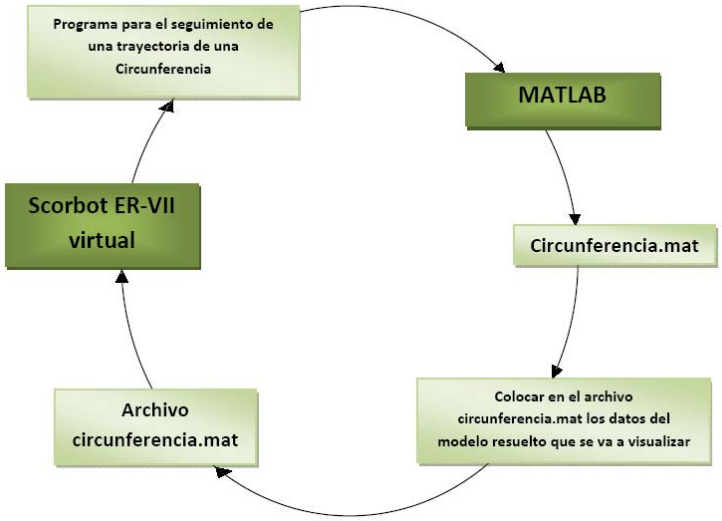


Figura 4.18: Enlace de MATLAB con la interfáz de visualización.

La figura 4.19 muestra las diferentes formas de manipular el ambiente virual es decir modificar la posición y orientación de las cámaras y la manipulación de los eslabones reflejada en el aumento o disminución de las

articulaciones. Como por ejemplo al presionar la tecla f: maximiza la ventana de visualización, F7: Al presionar esta tecla permite rotar la base a la izquierda.

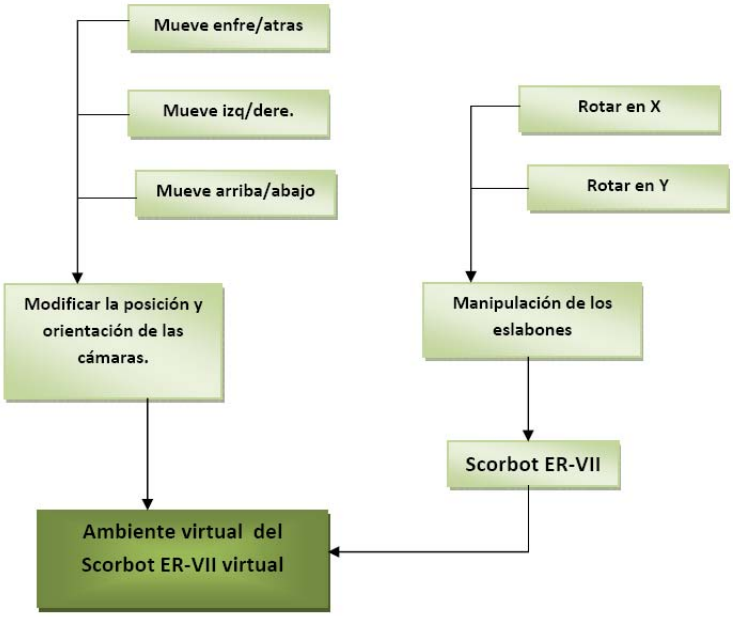


Figura 4.19: Modificación del escenario y el robot.

4.4. Conclusiones

En este trabajo de investigación y desarrollo tecnológico se implementó, a partir del modelo cinemático y dinámico del robot Scorbot ER-VII, un ambiente virtual para validar el desempeño de un controlador no lineal, misma situación que puede efectuarse con cualquier otro controlador. Las contribuciones de un visualizador virtual y evaluador de estrategias de control de movimiento para robots manipuladores permiten resolver problemas de diseño, tanto del robot como de las técnicas de control. Este sistema fue realizado a partir de su modelado geométrico en 3D Studio Max, y su exportación a código OpenGL cuyo código no es de trivial interpretación, con esto se establecen las técnicas de graficación convencionales con base en los modelos cinemáticos para lograr los movimientos articulares, dichos movimientos responden a una ley de control basada en el modelo dinámico y su evaluación en Matlab. La metodología empleada en la contrucción del sistema su referencia se encuentra en el Anexo G.

Capítulo 5

Conclusiones y perspectivas

5.1. Conclusiones generales

La creación de los ambientes virtuales son de gran estrategia de los robots manipuladores ya que en ocasiones se pueden manipular hasta en situaciones remotas haciendo a que no sean forzados a realizar singularidades, como medios de pruevas o como metodos de enseñanza-aprendizaje. Para hacer la implementación del ambiente virtual del robot Scorbot ER VII se hizo a partir del modelo cinemático y dinámico, haciendo uso del control de modos deslizantes, hacendo el seguimiento de una circunferencia, los controladores ayudan a que los robots se muevan de forma estable y en su mayoria son utilizados en la industria, es necesario que estos tengan un comportamiento preciso de sus movimientos, debido a que en muchas ocasiones son utilizados en actividades complicadas que requieren gran exactitud y por su dificultad no pueden ser realizados por el operador humano . Las contribuciones de un visualizador virtual y evaluador de estrategias de control de movimiento para robots manipuladores permiten resolver problemas de diseño, tanto del robot como de las técnicas de control. Este sistema fue realizado a partir de su modelado geométrico en 3D Studio Max, y su exportación a código Visual C++ con librerias OpenGL cuyo código no es de trivial interpretación, con las librerias de OpenGl se desarrollo el escenario en 3D, con Visual C++ se hizo la programación robusta y consistente,con esto se establecen las técnicas de graficación convencionales con base en los modelos cinemáticos para lograr los movimientos articulares, dichos movimientos responden a una ley de control basada en el modelo dinámico y su evaluación en Matlab. Esta interfaz de visualización esta diseñada para manipular al robot con el teclado y el mouse con motivo de una mejor apeciación en cuanto a su tarea encomendada.

5.2. Perspectivas

La virtualización del Scorbot ER-VII es una estrategia utilizada principalmente para aplicaciones en la industria y mas si son para aplicaciones remotas en donde se requiere su manipulación e interpretación en tiempo real, lo que implica que para virtualización instantanea o fuera de linea de cualquier sistema electromecanico, el metodo propuesto permite lograr eficientemente otras perspectivas que son:

- Desarrollar un visualizador virtual via web.
- Establecer virtualización en tiempo real de sistemas dinámicos reales.
- Establecer el método de Runge-Kutta de orden superior en linea, es decir no emplear MATLAB.

Apéndice A

Apéndices

A.1. Glosario

Aceleración Angular: es la razón con la cual la velocidad angular cambia con el tiempo [42].

Actuador o elementos motrices: son los encargados de producir el movimiento de las articulaciones, bien directamente o a través de poleas, cables, cadenas, etc. Clasificados en tres grandes grupos neumáticos, hidráulicos, electrónicos [43].

Articulación: son aquellas que conectan los enlaces conocidas como pares cinemáticos, cada par tiene dos elementos cada uno conectado a cada uno de los enlaces que une [44].

Articulación prismática: las que tienen asociado un desplazamiento. Para crear una articulación prismática o de rotación, es necesario, escoger una figura (cilindro, cubo) que va a representar a la articulación y agregarla al objeto robot. Nombrar a la figura como articulación n donde n es el número de articulación. Después de colocar la figura en la posición deseada, bloquear el movimiento y rotación de la articulación en todos los ejes de coordenadas. Si la articulación es de rotación se deja desbloqueado el eje Z para rotación si la articulación es prismática se bloquea el eje Z para movimiento [45].

Articulación de revolución: articulaciones que tienen como variable generalizada un ángulo [45].

Cadena cinemática: Es la combinación de articulaciones rotativas y/o de traslación, o ejes de movimiento [46].

Centro de gravedad: de un cuerpo es el punto donde se encuentra aplicada la resultante de la suma de todas las fuerzas gravitatorias que actúan sobre cada una de las partículas del mismo [47].

Centro de Masa: de un cuerpo es el punto donde se encuentra aplicada la resultante de la suma de todas las fuerzas gravitatorias que actúan sobre cada una de las partículas del mismo [47].

Cinemática: Parte de la mecánica, que estudia las diferentes clases de movimiento de los cuerpos sin atender las causas de lo producen [47].

Cinemática inversa: dado el punto final de la estructura, queremos determinar los ángulos y desplazamientos de la estructura que la lleven a ese punto. Este problema puede tener múltiples soluciones [45].

Detección de colisiones: sirve para verificar si hay ó no colisiones entre el robot y los demás objetos del entorno [45].

Dinámica: la parte de la mecánica que estudia conjuntamente el movimiento y las fuerzas que lo originan [48].

Efector final u órgano terminal: a la muñeca del manipulador se acopla una garra o una herramienta, que será la encargada de materializar el trabajo previsto [43].

Energía Cinética Ek: es la energía que posee un cuerpo en virtud de su movimiento [49].

Energía potencial Ep: es la energía que posee un cuerpo en virtud de su posición o condición [49].

Eslabón: Pieza o cuerpo rígido en una cadena cinemática [46].

Espacio de trabajo: Espacio de Trabajo: Es la zona donde el robot puede posicionarse y está limitada por las dimensiones físicas del manipulador [46].

Frecuencia: es el numero de vueltas, revoluciones o ciclos que efectúa un móvil en un segundo [47].

Fricción: se llama fricción a la fuerza que actúa entre dos materiales que se tocan mientras se deslizan uno al lado del otro [50].

Fuerza de Fricción: es una fuerza tangencial sobre una superficie que se opone al deslizamiento de la superficie a través de una superficie adyacente. La fuerza de fricción es paralela a la superficie y opuesta, en sentido a su movimiento, un objeto empezara a resbalar solo cuando la fuerza aplicada sobrepase la fuerza máxima de fricción estática [42].

Fricción seca: si se trata de mover un bloque sobre una superficie rugosa por aplicación de una fuerza, éste no se moverá si la fuerza P no sobrepasa cierto valor. Esto quiere decir que si el bloque no se mueve, la superficie debe ejercer una fuerza igual a la aplicada [51].

Fricción Dinámica: fuerza de fricción en movimiento [52].

Fricción Viscosa: presenta una relación lineal entre la fuerza aplicada y la velocidad del movimiento: f(t)=Bv(t), donde B es la constante de viscosidad [53].

Fuerza de fricción estática: las fuerzas de fricción que actúan entre superficies que están en reposo entre sí [54].

Fuerza centrifuga: a veces se atribuye al movimiento circular una fuerza dirigida hacia fuera que se conoce como fuerza centrifuga. Centrifuga significa que huye del centro o que se aleja del centro [50].

Fuerza de Coriolis: fuerza ficticia que parece actuar sobre un cuerpo cuando se observa éste desde un sistema de referencia en rotación. Así, un objeto que se mueve sobre la Tierra a velocidad constante con una componente de dirección Norte-Sur se ve desviado en relación con la Tierra que gira. En el hemisferio norte se desvía en el sentido de las agujas del reloj, y en el hemisferio sur en el sentido opuesto [50].

Grados de libertad: son los parámetros que se precisan para determinar la posición y la orientación del elemento terminal del manipulador. También se pueden definir los grados de libertad, como los posibles movimientos básicos independientes [43].

Gravedad: es una de las fuerzas universales de la naturaleza. Es una fuerza de atracción que existe entre toda materia, y es muy débil en comparación con otras fuerzas de la naturaleza [52].

Inercia: se le da el nombre de inercia a la tendencia de un cuerpo a permanecer en reposo o en movimiento

lineal uniforme y a la primera ley de Newton se le llama ley de la inercia [54].

Jacobiano: Es una matriz que se puede ver como la versión vectorial de la derivada de una función escalar. Es importante en el análisis y control de movimiento de un robot (planificación de trayectorias suaves, determinación de configuraciones singulares, ejecución de movimientos coordinados, derivación de ecuaciones dinámicas). Permite conocer las velocidades del extremo del robot a partir de los valores de las velocidades de cada articulación [51].

Lagragiano: es una función diseñada para resumir un sistema entero, el dominio apropiado del lagrangiano es un espacio de fase y debe obedecer las ecuaciones de Euler-Lagrange [53].

Manipulador: son sistemas mecánicos multifuncionales, con un sencillo sistema de control, que permite gobernar el movimiento de sus elementos, de los siguientes modos: Manual: Cuando el operario controla directamente la tarea del manipulador. De secuencia fija: Cuando se repite, de forma invariable, el proceso de trabajo preparado previamente. De secuencia variable: Se puede alterar algunas características de los ciclos de trabajo. También recibe el nombre de manipulador o brazo de un robot, el conjunto de elementos mecánicos que propician el movimiento del elemento terminal (aprehensor o herramienta) [43].

Masa: cantidad de materia contenida en un cuerpo [54].

Modelo: proceso de simplificar e idealizar [48].

Modelo matemático: un modelo matemático proporciona una útil perspectiva sobre el comportamiento de un actuador hidráulico [55].

Momento de inercia: queda determinado a partir del periodo de oscilación de un eje de torsión, en el que se ha insertado el cuerpo de prueba y que está unido con el soporte mediante un resorte espiral. El sistema es excitado para obtener oscilaciones armónicas. A partir del periodo de oscilación T y con el factor direccional angular D se calcula el momento de inercia I del cuerpo de prueba según la fórmula: $I = D(T/2\pi)$ [53].

Par electromagnético: es la fuerza aplicada para el moviemto del robot [51].

Periodo: tiempo que tarda un cuerpo en dar la vuelta completa o en completar un ciclo [52].

Perturbación: Desviación de un objeto que esta en orbita de su trayectoria normal debido a la presencia de una fuerza gravitacional adicional [52].

Planificador de trayectorias: es el diseño de tareas en un entorno sin intervención humana [51].

Programación del espacio de trabajo: la programación gestual y textual, controlan diversos aspectos del funcionamiento del manipulador: Control de la velocidad y la aceleración. Saltos de programa condicionales. Temporizaciones y pausas. Edición, modificación, depuración y ampliación de programas. Funciones de seguridad. Funciones de sincronización con otras maquinas. Uso de lenguajes específicos de Robótica [2].

Realidad virtual: combinación de diversas tecnologías e interfaces que permiten al usuario interactuar de forma intuitiva con un entorno inmersivo y dinámico generado por una computadora [56].

Robot: es un dispositivo mecánico que realiza acciones basadas en movimientos. Sus acciones más comunes son moverse autonomicamente entre otras manipulaciones demasiado precisas, pesadas, repetitivas o riesgosas para el humano [57].

Robot antropomórfico: también conocido como brazo articulado y es un robot con todas las articulaciones

rotativas y movimientos similares a los del brazo de una persona [58].

Robot redundante: un robot es redundante cuando el número de columnas es mayor que el de renglones en la matriz Jacobiana. Es decir, el número de grados de libertad es menor que el número de grados de movimiento [59].

Robótica: Ciencia de diseño, construcción y aplicación de robots [58].

Servomotor: aparato mecánico gobernado por la rueda del timón y que a su vez acciona la caña del timón en los buques grandes [52].

Simulación digital: es el análisis del comportamiento lógico y en el tiempo de los dispositivos y circuitos digitales [60].

Singularidad: Puntos y zonas que son inadmisibles dentro y fuera del espacio de trabajo respectivamente para el robot [61].

Torque: es la fuerza que producen los cuerpos en rotación .[62]

Trayectoria o camino: Está formada de puntos que realiza el robot o pasa a través de una operación dependiendo de la programación [58].

Velocidad angular: (W) de un objeto es la razón con la cual la coordenada angular, el desplazamiento angular Θ cambia con el tiempo, si Θ cambia de Θ_0 a Θ_f en un tiempo t [42].

Virtualización: proceso mediante el cual un humanointerpreta una impresión sensorial como un objeto en un entorno de un objeto en un entorno distinto al entorno en el que el objeto existe fisicamente [56].

Apéndice B

Acrónimos

E-L Euler - Lagrange.
ex Error en x.
ey Error en y.
ez Error en z.
eteta1 Error en teta1.
eteta2 Error en teta2.
eteta3 Error en teta3.
IH Interfaz háptica.
J Matriz Jacobiana.
K Energía cinética.
MTH Matriz de transformación homogénea.
MD Modelo Dinámico.
MCDP Modelo Cinemático Directo de Posición.
MCIP Modelo Cinemático Inverso de Posición.
MCDV Modelo Cinemático Directo de Velocidad.
MCIV Modelo Cinemático Inverso de Velocidad.
MCDA Modelo Cinemático Directo de Aceleración.
MCIA Modelo Cinemático Inverso de Aceleración.
PIDNL : Control proporcional-integral-derivativo no lineal.
PDH Parámetros de Denavit- Hartenberg.
PREVI software de realidad virtual cuyos escenarios virtuales se han diseñado en función del trastorno mental.
RV Realidad virtual.
Sw Software.
teta d1 Teta deseada1.
teta d2 Teta deseada2.
teta d3 Teta deseada3.
V Velocidad de desplazamiento.
VR Virtual reality.
xd X deseada.
yd Y deseada.
zd Z deseada.
ω Velocidad angular.
Z Impedancia.
2D : Dos dimensiones.
3D : Tres dimensiones.

Apéndice C

Manual de usuario

La interfaz de visualización del robot Scorbot ER-VII se implemento con fines de enseñanza-aprendizaje en donde se visualiza el seguimiento de una tayectoria de una circunferencia empleando para ello el control de modos deslizantes, ademas tiene diferentes formas de manipular las coordenadas operacionales y variables articulares y asimismo tambien diferentes formas de modificación de posiciones y orientaciones de las camaras, esto se puede llevar acabo a través del teclado y/o mouse. Estas interfaces se desarrollaron empleando librerás de OpenGL para Visual c++.

C.0.1. Introducción

El objetivo de este manual es dar a conocer cuales son las estrategias y la formar de manipular nuestro mundo virtual e interactuar con el, además de conocer y apreciar de cerca la simulacion del control de modos deslizantes aplicados al robot Scorbot ER-VII.

C.0.2. Requerimientos

Para poder hacer un buen uso y visualización del ambiente vitual se tiene que instalar las librerías:

```
glut32.dll en %WinDir%\System,
glut32.lib en $(MSDevDir)\..\..\VC98\lib, y
glut.h     en $(MSDevDir)\..\..\VC98\include\GL.
```

Como ya se ha mencionado en el capítulo 4 estas librerías nos van a permitir la manipulación de la ventana por medio del teclado y el mouse, otro de los requerimientos es que no se necesita tener instalado Visual c++.

C.0.3. Ejecución de la aplicación

La aplicación se ejecuta al pulsar el ícono Scorbot.exe, al ejecutar la aplicación nos muestra del lado izquierdo el cuadro de control y de lado derecho su aplicación. Los datos que se van mostrando en el cuadro de cotrol son extraidos como ya se menciono en el capítulo 4 del archivo circunferencia.mat por lo que no hay ninguna necesidad de teclear algo, para salir de la aplicación solo hay que presionar el boton de salir, como lo muestra la figura C.1.

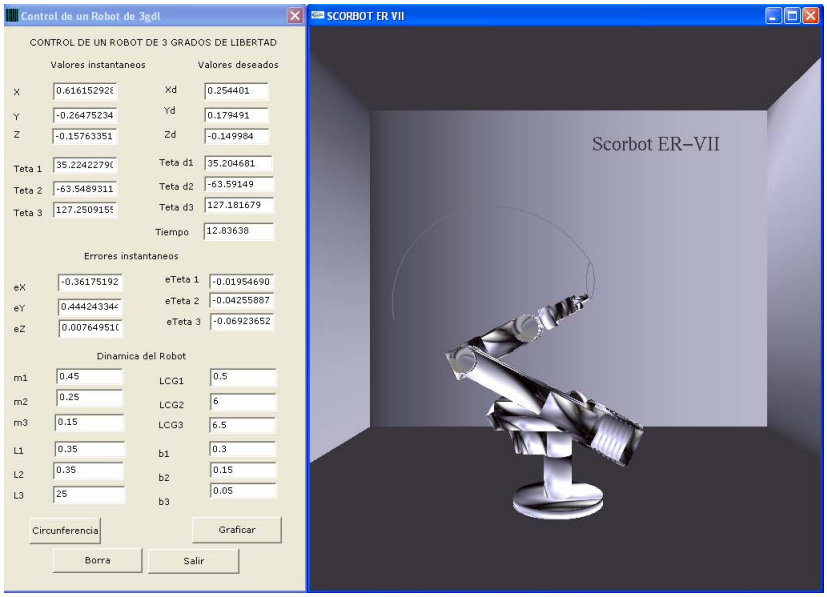


Figura C.1: Simulación en el espacio de trabajo del Scorbot ER-VII..

El cuadro de control muestra los botones: Circunferncia, Graficar, Borrar y Salir. Estas opciones nos permite gráficar y visualizar el control de modos deslizantes haciendo el seguimiento de una *Circunferecia*, pero primero antes para visualizar esta simulación se tiene que presionar el boton Circunferencia y posteriormente presionar el botón graficar, ya que este boton a gráfica la trayectoria seleccionada en el ambiente virtual.

C.0.4. Manipulación en el ambiente virtual

Para poder interactuar con el teclado es necesario conocer la programación de las teclas, para ello se describen a continuación:

- d: Al presionar esta tecla borra la trayectoria que el robot haya hecho.
- f: Al presionarla maximiza la ventana de visualización.
- w: Restaura la ventana.
- -: Hacerca la visualización.
- +: Aleja la visualización.
- F1: Al presionar esta tecla nos permite posicionar al fondo del Scorbot.
- F2: Regresa a la visualización inicial.
- F3: Permite visualizarlo esquinado hacia la dereecha.
- F4: Apresiación por la parte de arriba.
- F5: Apresiacón por la parte de abajo.
- F6: Visualización en posición en y.

- F7: Al presionar esta tecla permite rotar la base a la izquierda.
- F8: Al presionar esta tecla permite rotar la base a la derecha.
- F9: Al presionar esta tecla permite rotar el hombro hacia abajo.
- F10:Al presionar esta tecla permite rotar el hombro hacia arriba.
- F11:Al presionar esta tecla permite rotar el codo hacia abajo.
- F12: Al presionar esta tecla permite rotar el codo hacia arriba.

Otra de las opciones que podemos realizar es por medio del mouse ya que al presionado el mouse dentro de la simulación y deslizarnos podemos navegar sin ningún problema hacia cuarquier ángulo.

Las siguientes figuras muestra algunas de las opciones realizadas con el teclado y el mouse. La figura C.2 muestra la simulación del Scorbot ER-VII ya que al presinar las teclas F2 y "w"nos proporciona una vista de frente y de manera como la vemos desde un inicio.

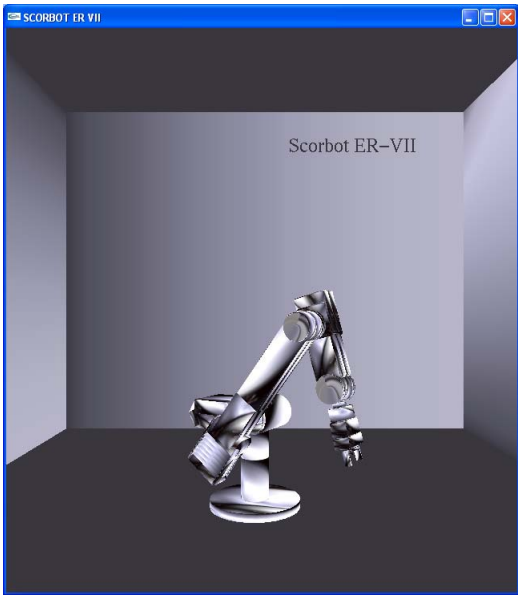


Figura C.2: Simulación del Scorbot ER-VII al presionar F2 y w.

La figura C.3 muestra la simulación del Scorbot ER-VII ya que al presinar las teclas f y F10 nos proporciona una vista de tamaño máximo y la manipulación del hombro hacia arriba.

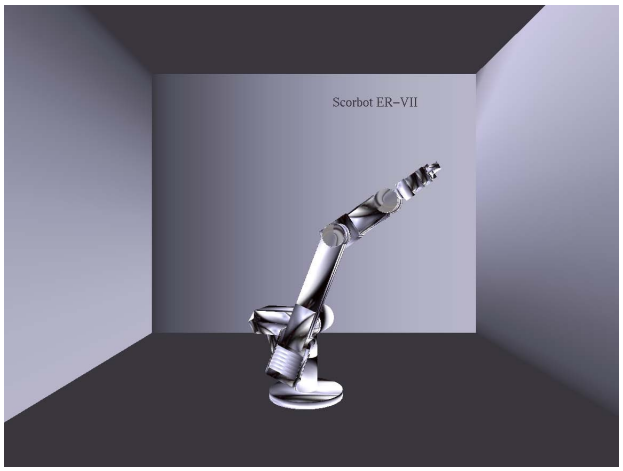


Figura C.3: Simulación del Scorbot ER-VII al presionar f y F10.

C.0.5. Simulación del seguimiento de una trayectoria de una Circunferencia

Para poder apreciar el seguimiento de la circunferencia en el ambiente virtual, una vez ejecutada la aplicación del robot scorbot.exe, se presiona el botón circunferencia que este nos va a permitir dibujar la circunferencia que se calculo en MATLAB , posteriormente se presiona el botón Graficar que nos permitirá como su nombre lo dice graficar la circunferencia que al presionar este botón cargara el archivo circunferencia.mat y leerá a través del cuadro de control el archivo de 7 en 7 que estos datos representan a las coordenadas operacionales X, Y, Z y sus variables articulares teta 1, teta 2, teta 3 y el tiempo, si se desea volver a graficar la circunferencia lo único que se tiene que hacer es presionar el botón borrar y seguir los pasos que anteriormente ya se describieron. El programa ejecutable generado por MATLAB circunferncia.mat se encuentran dentro de la carpeta Scorbot VII.

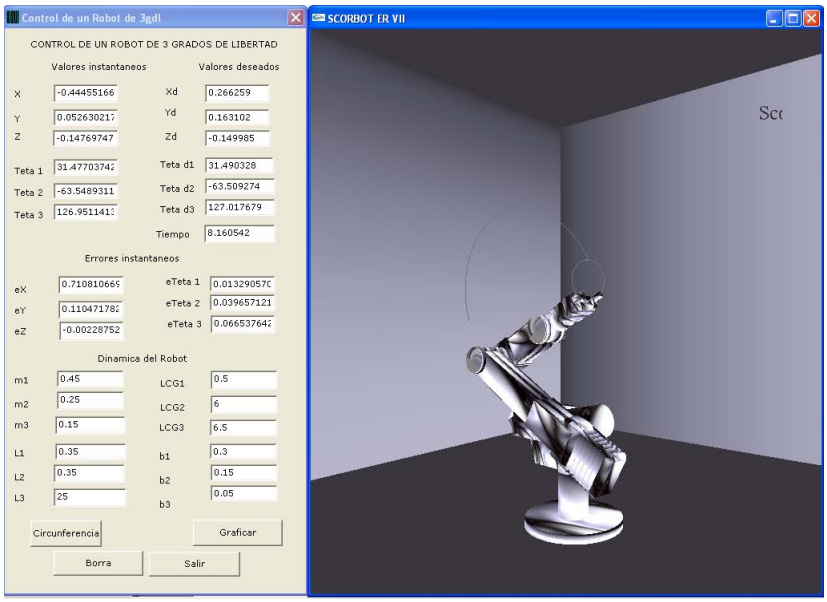


Figura C.4: Graficación de la circunferencia.

Apéndice D

Programa del Scorbot ER-VII

El programa que se presenta a continuación, es el codigo fuente en donde se desarrolla el ambiente virtual del Scorbot ER-VII haciendo las simulaciones de las trayectorias de los controles aplicados. Este ambiente virtual fue programado en visual C++ con librerías de OpenGL y en el cual se puede hacer la manipulación del Scorbot ER-VII así como también su navegación con el teclado y el mouse.

```
#include "stdafx.h"
#include "Scorbot VII.h"
#include "Scorbot VIIDlg.h"
#include <string.h>  // Texto
#include <malloc.h>
#include "dos.h"
#include <malloc.h>
#include <math.h>
#include <stdio.h>
#include <iostream.h>
#include <math.h>
#include <stdlib.h>
#include <ctype.h>
#include <fstream.h>
#include "gl/glaux.h"  // LIBRERIA PARA TEXTURAS
#include "1.h"
#include "2.h"
#include "3.h"
#include "textura.h"
#pragma comment (lib, "glaux.lib")
#define LV 850000
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE static char THIS_FILE[] = __FILE__;
#endif
#define ANCHOL 6

ifstream Aentrada; CScorbotVIIDlg *s;

double mcdpX, mcdpY, mcdpZ;
double d1=0.35, a3=0.25, a2=0.30,a1=0.05;
double rotaX = 0;
double rotaY = 0;
```

```
double anteriorX = 0;
double anteriorY = 0;
double RatonPresionado = GL_FALSE;
double distancia=7.2;
double n, ang, paso, dtx, dtz;
double  x = 0, y = 0.7, z = -5;
double posIni, ciclo=1;
double finX=4.434, finY=0.0, finZ=0.134;
double arrX[2000], arrY[2000],arrZ[2000];
double ValorX,ValorY,ValorZ;    //variables de posicion
bool dibT=TRUE;
bool init=TRUE;
int arr=0;
char nomArch[100]="circunferencia.mat";
float rotabase,rotaesla1,rotaesla2;
float t_eslabon1,t_eslabon2,t_eslabon3,posicion,velocidad2,velocidad3;


void dinamic()
{
    s->m_ma1=0.45; //13.13;
    s->m_ma2=0.25; //12.1;
    s->m_ma3=0.15; //3.25;

    s->m_long1=0.35; //35.85;
    s->m_long2=0.35; //30.0;
    s->m_long3=25.0;

    s->m_lcg1=0.5; //0.0;
    s->m_lcg2=0.5*12; //-0.18;
    s->m_lcg3=0.5*13; //-0.125;
    s->m_b1=0.3; //1.0;
    s->m_b2=0.15; //1.0;
    s->m_b3=0.05; //1.0;
}

void cir()
{
        glBegin(GL_LINE_STRIP);
        glColor3f(0.5f,0.01f,0.0f);
        for(int va=1;va<=GuarDato-1;va++)
        // int va=1;  +1 para mostrar el seguimiento en ela circunferencia
        {
            glVertex3f(AnteX[va],AnteZ[va],AnteY[va]);
        }
        glEnd ();
}

void error() {

        s->m_et11= (s->m_td1) - (s->m_slabon1);
        s->m_et2= (s->m_td2) - (s->m_slabon2);
```

```
        s->m_et3= (s->m_td3) - (s->m_slabon3);

        s->m_eX = (s->m_xd) - (s->m_X);
        s->m_eY = (s->m_yd) - (s->m_Y);
        s->m_eZ = (s->m_zd) - (s->m_Z);

}

void mcdp() {
    for(int paso=0;paso<350;paso++)
    {
        mcdpX = cos (rotabase) * (a3 * (cos (rotaesla1) + cos (rotaesla2) )
        + a2 * cos (rotaesla1) + a1);
        mcdpY = sin (rotabase) * (a3 * (cos (rotaesla1) + cos (rotaesla2) )
        + a2 * cos (rotaesla1) + a1);
        mcdpZ = d1 - a2 * sin (rotaesla1) - a3 * ( sin (rotaesla1) + sin (rotaesla2) );

        Aentrada>>n;
        s->m_xd=n;
        s->m_X=mcdpX;

        Aentrada>>n;
        s->m_yd=n;
        s->m_Y=mcdpY;

        Aentrada>>n;
        s->m_zd=n;
        s->m_Z=mcdpZ;


        if( Aentrada.eof() )
        {
            Aentrada.close();
            Aentrada.open("circunferencia.mat");
            Aentrada.close();

        }

        s->m_slabon1=rotabase;
        s->m_slabon2=rotaesla1;
        s->m_slabon3=rotaesla2;

        mcdpX=s->m_xd;
        mcdpY=s->m_yd;
        mcdpZ=s->m_zd;
        posIni=sqrt(pow((finX-mcdpX),2)+pow((finY-mcdpY),2)+pow((finZ-mcdpZ),2));
        if(posIni>0.01 && arr<LV)
        {
          arr++;
          arrX[arr]=mcdpX;
          arrY[arr]=mcdpY;
          arrZ[arr]=mcdpZ;
```

```
          finX=mcdpX;
          finY=mcdpY;
          finZ=mcdpZ;
        }
        error();
    }
    dinamica();
    s->UpdateData(false);
    glutPostRedisplay();
}

void Base() {
    glMatrixMode (GL_MODELVIEW);
    int mcount=0;
    int mindex=0;

    for(int i=0;i<sizeof(base)/sizeof(base[0]);i++)
    {
        if(!mcount)
        {
            SelectMaterial(material_ref[mindex][0]);
            mcount=material_ref[mindex][1];
            mindex++;
        }
        mcount--;
        for(int j=0;j<3;j++)
        {
            int vi=base[i][j];
            int ni=base[i][j+3];
            int ti=base[i][j+6];
            glNormal3f (normals[ni][0],normals[ni][1],normals[ni][2]);
            glTexCoord2f(textures[ti][0],textures[ti][1]);
            glVertex3f (vertices[vi][0],vertices[vi][1],vertices[vi][2]);
        }
    }
}


void eslabon1() {
    glMatrixMode (GL_MODELVIEW);
    int mcount2=0;
    int mindex2=0;

    for(int y=0;y<sizeof(hombro)/sizeof(hombro[0]);y++)
    {
        if(!mcount2)
        {
            SelectMaterial(material_ref[mindex2][0]);
            mcount2=material_ref[mindex2][1];
            mindex2++;
        }
        mcount2--;
```

```
        for(int z=0;z<3;z++)
        {
            int vi2=hombro[y][z];
            int ni2=hombro[y][z+3];
            int ti2=hombro[y][z+6];
            glNormal3f (normals[ni2][0],normals[ni2][1],normals[ni2][2]);
            glTexCoord2f(textures[ti2][0],textures[ti2][1]);
            glVertex3f (vertices[vi2][0],vertices[vi2][1],vertices[vi2][2]);
        }
    }
}


void eslabon2() {
    glMatrixMode (GL_MODELVIEW);
    int mcount3=0;
    int mindex3=0;

    for(int u=0;u<sizeof(eslabon_3)/sizeof(eslabon_3[0]);u++)
    {
        if(!mcount3)
        {
            SelectMaterial(material_ref[mindex3][0]);
            mcount3=material_ref[mindex3][1];
            mindex3++;
        }
        mcount3--;
        for(int ul=0;ul<3;ul++)
        {
            int vi3=eslabon_3[u][ul];
            int ni3=eslabon_3[u][ul+3];
            int ti3=eslabon_3[u][ul+6];
            glNormal3f (normals[ni3][0],normals[ni3][1],normals[ni3][2]);
            glTexCoord2f(textures[ti3][0],textures[ti3][1]);
            glVertex3f (vertices[vi3][0],vertices[vi3][1],vertices[vi3][2]);
        }
    }
}


void display(void) {

    GLfloat glfLightAmbient[] = { 7.8f, 7.8f, 7.8f, 8.8f };
    GLfloat glfLightDiffuse[] = { 2.2f, 2.2f, 2.2f, 2.0f };
    GLfloat glfLightSpecular[] = { 0.9f, 0.9f, 2.9f, 1.0f };

    // Add a light to the scene.
    glLightModeli( GL_LIGHT_MODEL_TWO_SIDE, GL_TRUE);
    glLightfv (GL_LIGHT0, GL_AMBIENT, glfLightAmbient);
    glLightfv (GL_LIGHT0, GL_DIFFUSE, glfLightDiffuse);
    glLightfv (GL_LIGHT0, GL_SPECULAR, glfLightSpecular);
    glEnable (GL_LIGHTING);
```

```
glEnable (GL_LIGHT0);
// Enable depth testing and backface culling.
glEnable (GL_DEPTH_TEST);

glEnable (GL_CULL_FACE);
GLuint texture_name;
glGenTextures(1,&texture_name);
texture_maps[0].id=texture_name;
glBindTexture(GL_TEXTURE_2D,texture_name);
LoadTexture(texture_maps[0].name);

// limpia la escena (ventana)
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
// verfica superficies visibles
glEnable( GL_DEPTH_TEST );
// inicializa la Matriz de transformación de coordenadas (Matriz del Modelo)
glLoadIdentity();
// traslada la escena al centro del espacio (Hortiz)
glRotatef( ang, 0,1,0 );
glTranslatef( -ANCHOL/2.0+x, y, z );
// construcción de la escena
glTranslatef(2.7f, 0.0f,distancia);
glScalef(1, 1, 1);
glRotatef(rotaX, 0.0, 1.0, 0.0);
glRotatef(-rotaY, 1.0, 0.0, 0.0);

glPushMatrix();
  room();
  /// Texto
  glColor3f (1.0F, 0.0F, 1.0F);
  sprintf (texto, " Scorbot ER-VII");
  glRasterPos3f (-0.1F, -0.9F, 0.6F);
  escribe (texto);
  glEnd();
glPopMatrix();
    //  glDisable(GL_TEXTURE_2D);
glPushMatrix();
glPushMatrix();

  glTranslatef (0.0, -01.3f,-12.88f);
  glRotatef (-135.0f, 15.0f, -22.9f, -23.0f);
  glRotatef (rotabase, 0.0f, 0.0f, -0.01f);
  glScalef (6.5f,6.5f,6.5);
  Base();
  glEnd();
  glPopMatrix();

glPushMatrix();

  glTranslatef (0.0f, -01.3f,-12.88f);
  glRotatef (-135.0f, 15.0f, -22.9f, -23.0f);
  glRotatef (rotabase, 0.0f, 0.0f, -0.01f);
```

```
      glRotatef (rotaesla1, 0.01f, 0.0f, 0.0f);
      glScalef (6.5f,6.5f,6.5);
      eslabon1();
      glEnd();
    glPopMatrix();

    glPushMatrix();

      glTranslatef (0.0f, -01.3f,-12.88f);
      glRotatef (-135.0f, 15.0f, -22.9f, -23.0f);//21.0f
      glRotatef (rotabase, 0.0f, 0.0f, -0.01f);
      glRotatef (rotaesla1, 0.01f, 0.0f, 0.0f);
      glRotatef (rotaesla2, 0.01f, 0.0f, 0.0f);
      glScalef (6.5f,6.5f,6.5);
      eslabon2();
      glEnd();
    glPopMatrix();
      cir();
    glPopMatrix();
    glutSwapBuffers(); /* Volcar el contenido del buffer sobre la memoria de video */

}

void DisplayMouseMotion(int x, int y) {
    int dx = anteriorX - x;
    int dy = y - anteriorY;

    if( RatonPresionado )
    {
       rotaX += (dx * 180.0f) / 600.0f;
       rotaY += (dy * 180.0f) / 600.0f;
       anteriorX = x;
       anteriorY = y;
       glutPostRedisplay();
    }
}


/*| Se llama al hacer clic del ratón. Control de clic & drag... *|
rotación escena.... */ void DisplayMouseClick(int button, int
state, int x, int y) {
  if( state == GLUT_DOWN )
  {
    if(button == GLUT_LEFT_BUTTON)
    {
       anteriorX = x;
       anteriorY = y;
       RatonPresionado = TRUE;
       DisplayMouseMotion(x,y); // se mueve al hacer clic
    }
  }
  else if( state == GLUT_UP )
```

```
  {
    glutPostRedisplay();
    RatonPresionado = FALSE;
  }
}

void key(unsigned char key, int x, int y) {
  switch (key)
  {
    case 'd':
        init=TRUE;
        dibT= FALSE;
        arr=0;
        init=FALSE;
        dibT= TRUE;
        break;

    case 'f':
        glutFullScreen ();//Solicita que la ventana actual sea cambia a tamaño completo
        break;

    case 'w':
        glutReshapeWindow (645,705); //Cambia el tamaño de la ventana
        glutPositionWindow(385,30); //Cambio de posición de la ventana
        break;

    case '-':
        distancia += 0.3;
        break;

    case '+':
        distancia -= 0.3;
        break;

    case 27:
        exit(0); // tecla de escape
        break;
  }
    glutPostRedisplay();
}

void muestraAmbiente() {
    int argc=1;
    char **argv;
    argv = (char **)malloc( 10 * sizeof( long ) );
    argv[0] = (char *)malloc( 80 * sizeof( long ) );
    strcpy(argv[0], "Prueba");
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DEPTH | GLUT_DOUBLE | GLUT_RGBA);
    //glutInitDisplayMode() define el modo en el que se debe dibujar la ventana
    //GLUT_RGB indica el tipo de modelo de color con el que se dibujará (Red-Green-Blue),
    //GLUT_DEPTH indica que se debe incluir un buffer de profundidad y
```

```
    //GLUT_DOUBLE dibuja la escena en un buffer fuera de la
    //pantalla, mientras que la imágen de la pantalla no se toca para nada. Una vez que la
    //imágen se ha dibujado en el buffer de fondo, se intercambia por el de la pantalla, de
    //manera que eliminamos el parpadeo.
    glutInitWindowPosition(385,0);//Posicion de la ventana
    glutInitWindowSize(638,705);//Tamaño de la ventana
    glutCreateWindow("SCORBOT ER VII");//Nombre la ventana
    glutDisplayFunc( display);
    glutReshapeFunc (changeSize);
    glutMainLoop();  //espera eventos
}


void CScorbotVIIDlg::OnSalir() {
    // TODO: Add your control notification handler code here
    Aentrada.close();
    exit(0);
}


void CScorbotVIIDlg::OnGraficar() {

    init=TRUE;
    dibT= FALSE;
    arr=0;
    init=FALSE;
    dibT= TRUE;
    s =this;

    UpdateData(true);
    UpdateData(false);

    Aentrada.open(nomArch);
    if(Aentrada.fail()) {
        return;
    }

    //////glutDisplayFunc( display);
    // inicialización de los datos del programa
    inicializar();
    glutIdleFunc( mcdp );
    glutMouseFunc(DisplayMouseClick);
    glutMotionFunc(DisplayMouseMotion);
    glEnable(GL_DEPTH_TEST);
    glutKeyboardFunc(key);
    glutSpecialFunc( specialKeys);

}

void CScorbotVIIDlg::OnPaint() {
    if (IsIconic())
    {
```

```
        CPaintDC dc(this); // device context for painting

        SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0);

        // Center icon in client rectangle
        int cxIcon = GetSystemMetrics(SM_CXICON);
        int cyIcon = GetSystemMetrics(SM_CYICON);
        CRect rect;
        GetClientRect(&rect);
        int x = (rect.Width() - cxIcon + 1) / 2;
        int y = (rect.Height() - cyIcon + 1) / 2;

        // Draw the icon
        dc.DrawIcon(x, y, m_hIcon);
    }
    else
    {
        CDialog::OnPaint();
    }
    muestraAmbiente();
}

// The system calls this to obtain the cursor to display while the user drags
//  the minimized window.
HCURSOR CScorbotVIIDlg::OnQueryDragIcon() {
    return (HCURSOR) m_hIcon;
}


void CScorbotVIIDlg::OnCircunferencia() {
    // TODO: Add your control notification handler code here

        init=TRUE;
        dibT= FALSE;
        arr=0;
        init=FALSE;
        dibT= TRUE;
        strcpy(nomArch,"circunferencia.mat");

}

void CScorbotVIIDlg::OnBorra() {
    // TODO: Add your control notification handler code here

        init=TRUE;
        dibT= FALSE;
        arr=0;
        init=FALSE;
        dibT= TRUE;
}
```

Apéndice E

Programas en MATLAB

MATLAB es un lenguaje de alto desempeño en el cual se ocupa de resolver matrices y hacer cálculos números orientados a matrices y vectores, proporcionando una herramienta para su visualización y comportamiento en tiempo real, es por ello que hemos optado la opción de realizar y programar nuestros programas en esta plataforma, ya que además puede trabajar bajo las plataformas de unix, windows y mac.

El programa de modos deslizantes generan un archivo.mat en el cual fueron calculados: tiempo, ángulos teta1, teta2, teta3 y sus coordenadas operacionales x, y, z, mostrandose en el archivo en columnas de acuerdo al orden mencionado anteriormente.

E.0.6. Dinamica.m

El programa Dinamica.m es el programa fuente en donde se evaluan las propiedades dinámicas descritas en el capítulo 3

```
Parte de la Condición inicial: q1=0.7854, q2=-0.7854,
q3=-0.7854,qp1=0, qp2=0, qp3=0, X=0.175, Y=0.175 y Z=0.4475
```

Y evaluando las propiedades dinámicas quedando representado de la forma siguiente:

```
% Programa que Valida las Propiedades del Modelo Dinamico dinamico
% Parte de la Condiciòn Inicial: q1=0.7854, q2=-0.7854, q3=-0.7854, qp1=0, qp2=0, qp3=0
%                                X=0.175, Y=0.175 y Z=0.4475


function dx = Dinamica(t,x) global dx

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

l1=0.35; % longitud de los eslabones
l2=0.35; l3=0.2;

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

g=9.81; % Coeficiente de gravitación

% Friccion Seca o Coulomb
k1=0.1; k2=0.05; k3=0.01;
```

```
beta=10; % k*tanh(beta*x')

% Fricción Viscosa
b1=0.3; b2=0.15; b3=0.05;

% Matriz de Inercia
H11=(m2+m3)*l2^2*(cos(x(3)))^2+2*m3*l2*l3*cos(x(3))*cos(x(3)+x(5))+m3*l3^2*(cos(x(3)+x(5)))^2;
H12=0; H13=0; H21=0;
H22=(m2+m3)*l2^2+m3*l3^2+2*m3*l2*l3*cos(x(5));
H23=m3*l3^2+m3*l2*l3*cos(x(5)); H31=0;
H32=m3*l3^2+m3*l2*l3*cos(x(5)); H33=m3*l3^2;

%Vector de Fuerzas de Coriolis y de Fuerzas Centripetas
et_1=sin(x(3))*cos(x(3));
et_2=sin(x(3)+x(5))*cos(x(3))+sin(x(3))*cos(x(3)+x(5));
et_3=sin(x(3)+x(5))*cos(x(3)+x(5)); et_4=sin(x(3)+x(5))*cos(x(3));

C11=0;
C12=-(2*(m2+m3)*l2^2*et_1+2*m3*l2*l3*et_2+2*m3*l3^2*et_3)*x(2);
C13=-(2*m3*l2*l3*et_4+2*m3*l3^2*et_3)*x(2);

C21=((m2+m3)*l2^2*et_1+m3*l2*l3*et_2+m3*l3^2*et_3)*x(2);
C22=-(2*m3*l2*l3*sin(x(5)))*x(6); C23=-(m3*l2*l3*sin(x(5)))*x(6);

C31=(m3*l2*l3*et_4+m3*l3^2*et_3)*x(2);
C32=(m3*l2*l3*sin(x(5)))*x(4); C33=0;

%Matriz de Gravedad
G1=0; G2=-(m2*g*lcg2+m3*g*l2)*cos(x(3))-m3*g*lcg3*cos(x(3)+x(5));
G3=-m3*g*lcg3*cos(x(3)+x(5));

%Matriz de Fricción Viscosa y Seca
Fr1=b1*x(2)+k1*tanh(beta*x(2)); Fr2=b2*x(4)+k2*tanh(beta*x(4));
Fr3=b3*x(6)+k3*tanh(beta*x(6));

% Variables Articulares y Derivadas
q=[x(1);x(3);x(5)]; qp=[x(2);x(4);x(6)];

%Matriz Jacobiana del Robot

% Matriz Jacobiana

J11=-l3.*sin(x(1)).*cos(x(3)+x(5))-l2.*sin(x(1)).*cos(x(3));
J12=-l3.*cos(x(1)).*sin(x(3)+x(5))-l2.*sin(x(3)).*cos(x(1));
J13=-l3.*cos(x(1)).*sin(x(3)+x(5));
J21=l3.*cos(x(1)).*cos(x(3)+x(5))+l2.*cos(x(1)).*cos(x(3));
J22=-l3.*sin(x(1)).*sin(x(3)+x(5))-l2.*sin(x(3)).*sin(x(1));
J23=-l3.*sin(x(1)).*sin(x(3)+x(5)); J31=0;
J32=-l3.*cos(x(3)+x(5))-l2.*cos(x(3)); J33=-l3.*cos(x(3)+x(5));


J=[J11,J12,J13;J21,J22,J23;J31,J32,J33];
```

```
% Modelo Cinemático Directo de Velocidad (MCDV)

Xp=J*qp;

% Matrices del robot
H=[H11,H12,H13;H21,H22,H23;H31,H32,H33];
C=[C11,C12,C13;C21,C22,C23;C31,C32,C33]; G=[G1;G2;G3];
Fr=[Fr1;Fr2;Fr3]; N=Fr+C*qp+G;

% Ecuación de control
T=0;

% Ecuaciones de Estado
qpp=inv(H)*(T-N);


              %Propiedades del modelo dinámico de un robot

%Matriz de Inercia Definida Positiva.

%Extrae el determinante de H y si este es negativo imprime dicho resultado
dmi=det(H);

if (dmi<=0),
   ['Deteminante de la Matriz de Inercia y Propiedad Positiva']
   dmi=det(H)
end

%Matriz de Inercia Simétrica

%Imprime la diferencia de H con su transpuesta
%si sus determinantes son diferentes
while(det(H)~=det(H')),
     ['Simetría de la Matriz de Inercia']
     ms=H-H'
end

%Imprime la diferencia de H con su transpuesta
%si son diferentes
while(H~=H'),
   ['Simetría de la Matriz de Inercia']
   ms=H-H'
end


%Propiedad de la Matriz Antisimétrica P'*(DH-2C)*P=0

%Derivada de la Matriz de Inercia

DH11a=-2*(m2+m3)*l2^2*cos(x(3))*sin(x(3))*x(4)-2*m3*l3^2*cos(x(3)+x(5))*sin(x(3)+x(5))*(x(4)+x(6));
DH11b=-2*m3*l2*l3*(cos(x(3))*sin(x(3)+x(5))*(x(4)+x(6))+
```

```
cos(x(3)+x(5))*sin(x(3))*x(4)); DH11=DH11a+DH11b; DH12=0; DH13=0;
DH21=0; DH22=-2*m3*l2*l3*sin(x(5))*x(6);
DH23=-m3*l2*l3*sin(x(5))*x(6); DH31=0;
DH32=-m3*l2*l3*sin(x(5))*x(6); DH33=0;

DH=[DH11,DH12,DH13;DH21,DH22,DH23;DH31,DH32,DH33];

% Un punto espacial cualquiera
P=[-.1;-.2;.7];
% PROP debe ser cero
PROP=P'*(DH-2*C)*P;

% Imprime PROP cuando su valor absoluto es mayor que 1e-6
if(abs(PROP)>1e-6),
   ['Propiedad de la Matiz Antisimétrica']
   PROP=P'*(DH-2*C)*P;
   x(12)=abs(PROP);
end
   x(10)=det(H);
if(det(H)>=0),
   ['Propiedad definida positiva']
else
   ['Propiedad no definida positiva']
end if(H-transpose(H)==0),
   ['Simétrica']
   x(11)=0;
else
   ['No simétrica']
   x(11)=1;
end


dx(1,1)=x(2); dx(2,1)=qpp(1); dx(3,1)=x(4); dx(4,1)=qpp(2);
dx(5,1)=x(6); dx(6,1)=qpp(3); dx(7,1)=Xp(1); dx(8,1)=Xp(2);
dx(9,1)=Xp(3); dx(10,1)=x(10); dx(11,1)=x(11); dx(12,1)=x(12);
```

E.0.7. Dinamica1.m

El programa scrip de Dinamica.m es Dinamica1.m, este programa es el complemento en donde se programa las simulaciones evaluadas de las propiedades dinámicas y se visualiza su comportamiento.

```
% Script

% Parte de la Condiciòn Inicial: q1=0.7854, q2=-0.7854, q3=-0.7854, qp1=0, qp2=0, qp3=0
%                                X=0.175, Y=0.175 y Z=0.4475

clear

global dx dx=zeros(12,1);

options=odeset('MaxStep',0.1,'InitialStep',0.1);
[t,x]=ode45('Dinamica',[0 10],[1.57 0 -1.57 0 -1.57 0 0.175 0.175
```

```
0.4475 0 0 0],options);

figure(1)

plot(t,x(:,1),'b'); title('Coordenada Generalizada q1');
xlabel('t'); ylabel('q1'); grid axis([0 6 0 1])


figure(2)

plot(t,x(:,3),'b'); title('Coordenada Generalizada q2');
xlabel('t'); ylabel('q2'); grid axis([0 6 -1 2.5])


figure(3)

plot(t,x(:,5),'b'); title('Coordenada Generalizada q3');
xlabel('t'); ylabel('q3'); grid axis([0 6 -1.5 1])


figure(4)

plot(t,x(:,2),'b'); title('Velocidad Angular dq1/dt');
xlabel('t'); ylabel('qp1'); grid axis([0 6 -1 1])


figure(5)

plot(t,x(:,4),'b'); title('Velocidad Angular dq2/dt');
xlabel('t'); ylabel('qp2'); grid axis([0 6 -2 4])


figure(6)

plot(t,x(:,6),'b'); title('Velocidad Angular dq3/dt');
xlabel('t'); ylabel('qp3'); grid axis([0 6 -2 4])

figure(7)

plot(t,x(:,7),'b'); title('Coordenada Operacional X');
xlabel('t'); ylabel('X'); grid axis([0 6 -1 1])

figure(8)

plot(t,x(:,8),'b'); title('Coordenada Operacional Y');
xlabel('t'); ylabel('Y'); grid axis([0 6 -1 1])

figure(9)

plot(t,x(:,9),'b'); title('Coordenada Operacional Z');
xlabel('t'); ylabel('Z'); grid axis([0 6 -1 1])
```

```
figure(10)

plot(t,x(:,10),'b'); title('Definida Positiva'); xlabel('t');
ylabel('X'); grid axis([0 6 -1 1])

figure(11)

plot(t,x(:,11),'b'); title('Simetrica'); xlabel('t'); ylabel('Y');
grid axis([0 6 -1 1])

figure(12)

plot(t,x(:,12),'b'); title('Antisimetrica'); xlabel('t');
ylabel('Z'); grid axis([0 6 -1 1])

end
```

E.0.8. Circunferencia.m

Este programa hace la simulación del control PID no lineal, partiedo de su condición inicial

```
q1=0.7854, q2=-0.7854, q3=-0.7854,qp1=0, qp2=0, qp3=0, X=0.175,
Y=0.175 y Z=0.4475

% Programa que Efectúa un Control PD para Seguimiento de una Circunferencia en el
% Dispositivo Háptico Phantom en su Estructura de Posición 3 grados de libertad.

% Parte de la Condiciòn Inicial: q1=0.7854, q2=-0.7854, q3=-0.7854, qp1=0, qp2=0, qp3=0
%                                X=0.175, Y=0.175 y Z=0.4475

% Seguimiento de una Circunferencia en el Plano XY con Ecuación (x-0.3)^2+(y-0.2)^2=(0.1)^2
% Elaboro: Omar Arturo Domínguez Ramírez.


function dx = Circunf_pidnl1(t,x)

%Parámetros supuestos del Robot Phantom y Ganancias

global dx

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

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

l1=0.35;%0.35; % longitud de los eslabones
l2=0.35;%0.35;
l3=0.35;%0.2;

lcg1=0.5*l1; % longitud al centro de masas
lcg2=0.5*l2; lcg3=0.5*l3;
```

```
g=9.81; % Coeficiente de gravitación

% Friccion Seca o Coulomb
k1=0.1; k2=0.05; k3=0.01;
beta=10; % k*tanh(beta*x')

% Fricción Viscosa
b1=0.3; b2=0.15; b3=0.05;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                         % Parámetros del Controlador

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

kp=[100,0,0;0,81,0;0,0,64]; kv=[20,0,0;0,18,0;0,0,16];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


                      % Referencia Deseada ( Seguimiento )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Referencia Deseada en el Plano Operacional: Circunferencia C(h,k) y Radio r

lambda=.05; r=.05; h=0.3; k=.2; w=2*pi/5;;

%fi=lambda*t;
%ro=r*cos(3*fi);

px=h+r*cos(w*t); py=k+r*sin(w*t); pz=0; pd=[px;py;pz];

% Primera derivada de la referencia
dpx=-r*w*sin(w*t); dpy=r*w*cos(w*t); dpz=0; dpd=[dpx;dpy;dpz];

% Segunda derivada de la referencia
ddpx=-r*w^2*cos(w*t); ddpy=-r*w^2*sin(w*t); ddpz=0;
ddpd=[ddpx;ddpy;ddpz];

% MCIP: Modelo Cinemático Inverso de Posición Deseado


qd1=atan2(py,px); c3=(px.^2+py.^2+pz.^2-l2.^2-l3.^2)/(2.*l2.*l3);
s3=sqrt(1-c3.^2); qd3=atan2(s3,c3); rz=sqrt(px.^2+py.^2);
aux1=l3.*s3; aux2=l2+l3.*c3; qd2=-atan2(pz,rz)-atan2(aux1,aux2);

qd=[qd1;qd2;qd3];

% Modelo Cinemático Inverso de Velocidad (MCIV)
```

```
%Matriz Inversa del Jacobiano

J11id=-sin(qd1)/(l2.*cos(qd2)+l3.*cos(qd2+qd3));
J12id=cos(qd1)/(l2.*cos(qd2)+l3.*cos(qd2+qd3)); J13id=0;
J21id=(cos(qd1).*cos(qd2+qd3))/(l2.*sin(qd3));
J22id=(sin(qd1).*cos(qd2+qd3))/(l2.*sin(qd3));
J23id=-(sin(qd2+qd3))/(l2.*sin(qd3));
J31id=-(l2.*cos(qd1).*cos(qd2)+l3.*cos(qd1).*cos(qd2+qd3))/(l2.*l3.*sin(qd3));
J32id=-(l2*sin(qd1).*cos(qd2)+l3.*sin(qd1).*cos(qd2+qd3))/(l2.*l3*sin(qd3));
J33id=(l2*sin(qd2)+l3.*sin(qd2+qd3))/(l2.*l3.*sin(qd3));

Jid=[J11id,J12id,J13id;J21id,J22id,J23id;J31id,J32id,J33id];

qdp=Jid*dpd;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                              % Modelo Dinámico del Robot

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Matriz de Inercia
H11=(m2+m3)*l2^2*(cos(x(3)))^2+2*m3*l2*l3*cos(x(3))*cos(x(3)+x(5))+m3*l3^2*(cos(x(3)+x(5)))^2;
H12=0; H13=0; H21=0;
H22=(m2+m3)*l2^2+m3*l3^2+2*m3*l2*l3*cos(x(5));
H23=m3*l3^2+m3*l2*l3*cos(x(5)); H31=0;
H32=m3*l3^2+m3*l2*l3*cos(x(5)); H33=m3*l3^2;

%Vector de Fuerzas de Coriolis y de Fuerzas Centripetas
et_1=sin(x(3))*cos(x(3));
et_2=sin(x(3)+x(5))*cos(x(3))+sin(x(3))*cos(x(3)+x(5));
et_3=sin(x(3)+x(5))*cos(x(3)+x(5)); et_4=sin(x(3)+x(5))*cos(x(3));

C11=0;
C12=-(2*(m2+m3)*l2^2*et_1+2*m3*l2*l3*et_2+2*m3*l3^2*et_3)*x(2);
C13=-(2*m3*l2*l3*et_4+2*m3*l3^2*et_3)*x(2);

C21=((m2+m3)*l2^2*et_1+m3*l2*l3*et_2+m3*l3^2*et_3)*x(2);
C22=-(2*m3*l2*l3*sin(x(5)))*x(6); C23=-(m3*l2*l3*sin(x(5)))*x(6);

C31=(m3*l2*l3*et_4+m3*l3^2*et_3)*x(2);
C32=(m3*l2*l3*sin(x(5)))*x(4); C33=0;

%Matriz de Gravedad
G1=0; G2=-(m2*g*lcg2+m3*g*l2)*cos(x(3))-m3*g*lcg3*cos(x(3)+x(5));
G3=-m3*g*lcg3*cos(x(3)+x(5));

%Matriz de Fricción Viscosa y Seca
Fr1=b1*x(2)+k1*tanh(beta*x(2)); Fr2=b2*x(4)+k2*tanh(beta*x(4));
Fr3=b3*x(6)+k3*tanh(beta*x(6));

% Variables Articulares y Derivadas
```

```
q=[x(1);x(3);x(5)]; qp=[x(2);x(4);x(6)];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                             % Modelado Cinemático del Robot

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Matriz Jacobiana del Robot

% Matriz Jacobiana

J11=-l3.*sin(x(1)).*cos(x(3)+x(5))-l2.*sin(x(1)).*cos(x(3));
J12=-l3.*cos(x(1)).*sin(x(3)+x(5))-l2.*sin(x(3)).*cos(x(1));
J13=-l3.*cos(x(1)).*sin(x(3)+x(5));
J21=l3.*cos(x(1)).*cos(x(3)+x(5))+l2.*cos(x(1)).*cos(x(3));
J22=-l3.*sin(x(1)).*sin(x(3)+x(5))-l2.*sin(x(3)).*sin(x(1));
J23=-l3.*sin(x(1)).*sin(x(3)+x(5)); J31=0;
J32=-l3.*cos(x(3)+x(5))-l2.*cos(x(3)); J33=-l3.*cos(x(3)+x(5));


J=[J11,J12,J13;J21,J22,J23;J31,J32,J33];

% Modelo Cinemàtico Directo de Velocidad (MCDV)

Xp=J*qp;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                   % Ecuaciones del Modelo Dinamico del Robot

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Genrador de Base de Tiempo
% función suave que va de 0 a 1 en tiempo tb

tb = .4; alfa_0 = 1.001; delta = 0.001;

  Si = 1/2+1/2*sin(pi/tb*(t-tb/2));
 dSi = 1/2*(pi/tb)*cos(pi/tb*(t-tb/2));
ddSi = -1/2*(pi/tb)^2*sin(pi/tb*(t-tb/2));

if t >= tb
    Si = 1;
   dSi = 0;
  ddSi = 0;
end

%Ganancia del TBG
 sigma = alfa_0*dSi/(1-Si+delta);
dsigma = alfa_0*(((1-Si+delta)*ddSi+dSi^2)/(1-Si+delta)^2);
```

```
 if t >= tb
    sigma = 4;
   dsigma = 0;
 end

% Ecuaciones de Errores

alfa=[4,0,0;0,4,0;0,0,4]; kd=[20,0,0;0,18,0;0,0,16];
ki=[25,0,0;0,25,0;0,0,25];

qrp=qdp-sigma*(q-qd); S=qp-qrp;

Sq=S; Sr=Sq;

eq=qd-q; eqp=qdp-qp;

% Atractor
S_A1=3; S_A2=3; S_A3=3; k=500; A1=S_A1*exp(-k*t);
A2=S_A2*exp(-k*t); A3=S_A3*exp(-k*t);

A=[A1;A2;A3];



% Matrices del robot
H=[H11,H12,H13;H21,H22,H23;H31,H32,H33];
C=[C11,C12,C13;C21,C22,C23;C31,C32,C33]; G=[G1;G2;G3];
Fr=[Fr1;Fr2;Fr3]; N=Fr+C*qp+G;

I=[x(10);x(11);x(12)];

% Ecuación de control
T=-kd*Sr-ki*I+kd*A;
%T=kp*eq+kv*eqp;

% Ecuaciones de Estado
qpp=inv(H)*(T-N);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                                  % Integradores

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

dx(1,1)=x(2); dx(2,1)=qpp(1); dx(3,1)=x(4); dx(4,1)=qpp(2);
dx(5,1)=x(6); dx(6,1)=qpp(3); dx(7,1)=Xp(1); dx(8,1)=Xp(2);
dx(9,1)=Xp(3); dx(10,1)=tanh(100*Sq(1)); dx(11,1)=tanh(100*Sq(2));
dx(12,1)=tanh(100*Sq(3));
```

E.0.9. Circunferencia1.m

Circunferencia1.m es el Scrip de Circunferencia.m en donde se programan las simulaciones haciendo el seguimiento de la trayectoria.

```
% Script 1 para ejecutar el Programa que Efectúa un Control PD para Seguimiento de
% una Circunferencia en el Dispositivo Haptico Phantom en su Estructura de Posicion
% 3 grados de libertad.
% Parte de la Condiciòn Inicial: q1=0.7854, q2=-0.7854, q3=-0.7854, qp1=0, qp2=0, qp3=0
%                                X=0.175, Y=0.175 y Z=0.4475
% Seguimiento de una Circunferencia en el Plano XY con Ecuación (x-0.3)^2+(y-0.2)^2=(0.1)^2
% Elaboraron: Omar Arturo Domínguez Ramírez/Herbert Lara Ordaz
clear global dx dx=zeros(9,1);
options=odeset('MaxStep',0.1,'InitialStep',0.1);
[t,x]=ode45('Circunf_pidnl1',[0 15],[0.7854 0 -0.7854 0 -0.7854 0
0.175 0.175 0.4475 0 0 0],options);

% Referencia Deseada en el Plano Operacional: Circunferencia C(h,k) y Radio r

%lambda=1; %Numero de frecuencia
r=.05; h=0.3; k=.2; w=2*pi/5;;


px=h+r*cos(w*t); py=k+r*sin(w*t); pz=0*(k+r*sin(w*t));

% Primera derivada de la referencia

dpx=-r*w*sin(w*t); dpy=r*w*cos(w*t); dpz=0; dp=[dpx;dpy;dpz];

% MCIP: Modelo Cinemático Inverso de Posición Deseado

l1=0.35; % longitud de los eslabones
l2=0.35; l3=0.35;

qd1=atan2(py,px); c3=(px.^2+py.^2+pz.^2-l2.^2-l3.^2)/(2.*l2.*l3);
s3=sqrt(1-c3.^2); qd3=atan2(s3,c3); rz=sqrt(px.^2+py.^2);
aux1=l3.*s3; aux2=l2+l3.*c3; qd2=-atan2(pz,rz)-atan2(aux1,aux2);

%Matriz Inversa del Jacobiano deseado

J11id=-sin(qd1)./(l2*cos(qd2)+l3*cos(qd2+qd3));
J12id=cos(qd1)./(l2.*cos(qd2)+l3.*cos(qd2+qd3)); J13id=0;
J21id=(cos(qd1).*cos(qd2+qd3))./(l2.*sin(qd3));
J22id=(sin(qd1).*cos(qd2+qd3))./(l2.*sin(qd3));
J23id=-(sin(qd2+qd3))./(l2.*sin(qd3));
J31id=-(l2.*cos(qd1).*cos(qd2)+l3.*cos(qd1).*cos(qd2+qd3))./(l2.*l3.*sin(qd3));
J32id=-(l2*sin(qd1).*cos(qd2)+l3.*sin(qd1).*cos(qd2+qd3))./(l2.*l3*sin(qd3));
J33id=(l2*sin(qd2)+l3.*sin(qd2+qd3))./(l2.*l3.*sin(qd3));

%Jid=[J11id,J12id,J13id;J21id,J22id,J23id;J31id,J32id,J33id];

%qdp=Jid*dp;
```

```
qdp1=J11id.*dpx+J12id.*dpy+J13id.*dpz;
qdp2=J21id.*dpx+J22id.*dpy+J23id.*dpz;
qdp3=J31id.*dpx+J32id.*dpy+J33id.*dpz;

% Errores Articulares
error1=qd1-x(:,1); error2=qd2-x(:,3); error3=qd3-x(:,5);

%Errores de velocidades

errorv1=qdp1-x(:,2); errorv2=qdp2-x(:,4); errorv3=qdp3-x(:,6);

% Error extendido
s1=error1+5*errorv1; s2=error2+5*errorv2; s3=error3+5*errorv3;


fid = fopen('cir8.mat','w'); ite=size(t,1);
%grabando datos en disco:
% tiempo, q1,q2 y q3,x,y,z
%los angulos en grados sexagecimales

for (i = 1:1:ite)
   i;
   fprintf(fid,'%f',t(i));%tiempo
   fprintf(fid,' ');
   Ang=x(i,1)*180/pi;
   fprintf(fid,'%f',Ang);%q1
   fprintf(fid,' ');
   Ang=x(i,3)*180/pi;
   fprintf(fid,'%f',Ang);%q2
   fprintf(fid,' ');
   Ang=x(i,5)*180/pi;
   fprintf(fid,'%f',Ang);%q3
   fprintf(fid,' ');
   fprintf(fid,'%f',x(i,7));%x
   fprintf(fid,' ');
   fprintf(fid,'%f',x(i,8));%y
   fprintf(fid,' ');
   fprintf(fid,'%f',x(i,9));%z
   fprintf(fid,'\n');

   end fclose(fid);


% Para graficar
figure (1)

plot(t,x(:,1),'b',t,x(:,3),'r',t,x(:,5),'k') title('Coordenadas
operacionales XYZ');


figure (2)
```

```
plot(t,x(:,2),'b',t,x(:,4),'r',t,x(:,6),'k') title('Variables
articulares');

figure (3)

plot(x(:,7), x(:,8),'r') title('Circunferencia en el espacio de
trabajo');

figure (4)

plot(x(:,7), x(:,9),'r') title('Circunferencia en el espacio de
trabajo');

figure (5)

plot(x(:,8), x(:,9),'r') title('Circunferencia en el espacio de
trabajo');
```

Apéndice F

Publicaciones

Productos de este trabajo presentados en congresos, nacionales e internacionales:

1. López Pacheco Liana, Martínez Olguín Verónica y Domínguez Ramírez, Omar A., "Scorbot ER-VII Virtual: Consideraciones para el Diseño e Integración", Primer Congreso Nacional de Mecatrónica y Tecnologías Inteligentes CONAMTI 2006.El Saucillo, Mpio. de Huichapan, Hgo. 13 de Mayo de 2006.
2. López Pacheco Liana, Martínez Olguín Verónica and Domínguez Ramírez, Omar A., "Scorbot ER-VII Virtual: Mathematical Model and Control of Movements", Internacional Symposium On Robotics And Automation, San Miguel Regla, Hidalgo, México, August, 25-28.

Scorbot ER-VII Virtual: Consideraciones para el Diseño e Integración

Liana López Pacheco, Verónica Martínez Olguín
Y Omar Arturo Domínguez Ramírez

Universidad Autónoma del Estado de Hidalgo
Ciudad Universitaria Carretera Pachuca – Tulancingo Km. 4.5 C.P. 42184
Col. Carboneras, Mineral de la Reforma, Hgo.
Tel. 017717172000 Ext. 6732, 6734, 6738 Fax ext. 6732
lizli_lopez@hotmail.com, verola_mar@hotmail.com

Resumen.

En el presente artículo se muestran las consideraciones para el diseño e integración de un ambiente virtual del robot Scorbot ER-VII considerando 3gdl: base, hombro y codo, omitiendo el pitch y el roll para su manipulación, siguiendo una metodología práctica para la deducción del modelo cinemático directo e inverso de posición, el modelo dinámico basado en la formulación de Euler-Lagrange, aplicando un control no lineal, incorporándolo al mundo virtual.

Palabras clave: **Ambiente virtual, cinemática, dinámica, control adaptable.**

1. Objetivos.

Diseñar e integrar al robot Scorbot ER-VII en un mudo virtual en el que sea posible interactuar con los modelos matemáticos, con el propósito de que este efectué la simulación utilizando herramientas computacionales de última generación para la asignación de comportamientos complejos en mundos virtuales dinámicos.

2. Justificación.

El propósito de la virtualización del robot Scorbot ER-VII es de brindar una metodología nueva utilizando herramientas computacionales y de control moderno que permiten eficientar el proceso de diseño y construcción de un sistema de automatización de procesos industriales. Adicionalmente, este sistema puede ser utilizado como herramienta didáctica en el proceso de enseñanza-aprendizaje de materias de licenciatura y posgrado como: robótica, mecatrónica, automatización, control, realidad virtual y programación orientada a objetos.

3. Planteamiento del problema.

Crear un ambiente virtual del robot Scorbot ER-VII utilizamos 3D Max 6.0 que es el que nos permite crear el modelo del robot en tres dimensiones a partir de primitivas estándar o primitivas extendidas, que son objetos paramétricos como cajas, esferas, conos, etc, orientándonos por medio de vistas de perspectiva, vista anterior, vista superior, vista izquierda, etc, permitiendo la facilidad de visualización y representación de los modelos del robot, 3D Exploration permite crear la aplicación en visual C++, para la manipulación del robot y el ambiente virtual se programa con Visual C++ y

librerías de OpenGL permitiendo la programación de gráficos en el espacio virtual del Scorbot ER-VII, controlados por medio de los modelos matemáticos cinemático directo e inverso de posición, permitiendo conocer las coordenadas operacionales y sus variables articulares que este tenga así como también la dinámica basada en la formulación de Euler-Lagrange aplicando un control PID no lineal para su validación.

MatLab se hacen pruebas de estos modelos matemáticos, para verificar el comportamiento antes de implementarlo a la virtualización del robot Scorbot ER-VII.

4. Antecedentes

La realidad virtual nación en USA, al principio se enfocó principalmente en el campo militar. La Realidad Virtual es una simulación de un ambiente tridimensional generada por computadoras, en el que el usuario es capaz tanto de ver como de manipular los contenidos de ese ambiente [15].

Aplicación de robótica virtual encontradas en Internet:

Aplicaciones de la cirugía asistida por robots y telecirugía:

Actualmente existe un solo robot quirúrgico que se comercializa en todo el mundo el 'Da Vinci', fabricado por la empresa Intuitive Surgical Systems localizada en California, Estados Unidos. Los cirujanos pueden emplear estos sistemas quirúrgicos de 1,3 millones de dólares (algo más de un millón de euros) en unos 300 hospitales de todo el mundo para retirar próstatas cancerosas, reparar válvulas cardiacas y efectuar otros procedimientos médicos con más seguridad y realizando el mínimo trauma en el paciente disminuyendo con ello los períodos de recuperación. [11].

Mundos virtuales creados con fotografías.

Entre los sistemas de dibujo de gráficas por computadora basados en fotografías destaca el desarrollado por la Universidad de Carolina del Norte. Este sistema utiliza una cámara para tomar las fotografías panorámicas y un sensor láser para obtener los valores de profundidad para cada uno de los píxeles dentro de la imagen, con los datos del escáner láser y las fotografías se generan pequeñas unidades de información (tile) las cuales contienen la ubicación tridimensional del píxel y su color. Posteriormente se utilizan los tiles para generar las imágenes. Para sus experimentos utilizaron una computadora Silicon Graphics Onyx2 con InfiniteReality2 logrando 12.14 cuadros por segundo [12].

5. Realidad Virtual y Robótica.

La robótica virtual es un sistema de simulación de robot que puede habilitar la tele-manipulación de robots reales via www.

En el Laboratorio de Robótica del CINVESTAV-IPN se dispone de un sistema de manipulación robotizada asistido por visión compuesto de las siguientes partes:

1) Robot UNIMATE S-103 con controlador, *teach pendant*, basado en el sistema de programación propietario C/ROS.
2) Sistema de adquisición de imágenes digitales OCULUS 200. Genera 30 imágenes por segundo de 256×256 pixeles en 128 niveles de gris.
3) Computadora personal pentium con varios sistemas desarrollados en el laboratorio para la programación y control del robot (ROBOCOP), diversas aplicaciones de control inteligente basadas en Redes Neuronales y Lógica Borrosa, así como sistemas de visión artificial para el robot (AVICON).
4) Sistema de comunicación vía ethernet con redes de computadoras (Internet) [13].

Robot URIS

El robot URIS está propulsado por 4 sistemas motor-hélice que le proporcionan 3 grados de libertad conocidos en la nomenclatura marítima como *surge* (movimiento frontal), *yaw* (movimiento de cambio de rumbo) y *heave* (movimiento en profundidad). El vehículo ha sido diseñado pasivamente estable en *roll* y *pitch* de manera que el sistema de control queda simplificado. Este programa tiene dos funciones: (1) simular el movimiento del vehículo debajo del agua utilizando su modelo dinámico, y (2) proveerlo de una representación virtual del mundo subacuático. La utilidad del mundo virtual es doble. En primer lugar, permite la visualización del vehículo dentro del entorno marino y en segundo lugar permite la simulación del los sensores que dependen de los alrededores del robot [14].

6. Robot Scorbot ER-VII.

Fabricado por la compañía Israelí Eshed Robotec Inc. El robot Scorbot ER-VII figura 6.1 es un robot de 5gdl (base, hombro, codo y efector final este último compuesto por dos articulaciones) y asimiento (con dos fases: apertura y cierre) [2].

Figura 6.1 Robot Scorbot ER-VII

El espacio de trabajo Fig.6.2, es la zona donde el robot puede posicionarse y esta limitado por las dimensiones físicas del manipulador, así como por los límites de giro y desplazamientos de cada una de las articulaciones el rango de operaciones es el siguiente [3]:

1) Eje 1: rotación de base 250°
2) Eje 2: rotación de hombro 170°
3) Eje 3: rotación de codo 225°
4) Eje 4: pitch 180°
5) Eje 5: roll 360°

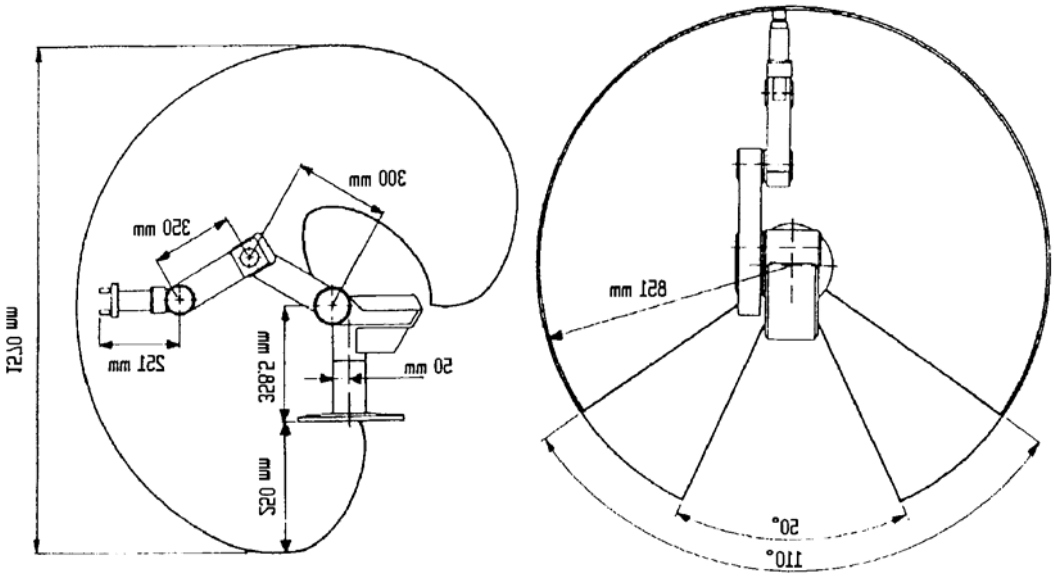


Figura. 6.2 Medidas de los eslabones y cadena cinemática

7. Modelos matemáticos.

La cinemática de robots se dedica al análisis y solución de los problemas derivados del posicionamiento de los elementos del manipulador. La cinemática de robot hace uso de tres conceptos fundamentales:

1) Estructura mecánica del manipulador.
2) Grados de libertad para el posicionamiento del elemento terminal.
3) Solución de los modelos directo e inverso.

El manipulador está constituido por tres elementos básicos: cuerpo, brazo y antebrazo, que se relacionan entre sí mediante articulaciones o pares cinemáticos de rotación o prismáticos. Cada elemento dispone de dos pares cinemáticos, estando el primero de ellos fijo a la base y el último con un extremo libre, en donde se sitúa el efector final.

El tipo de manipulador empleado más frecuentemente en la industria, tiene tres grados de libertad en sus elementos, más otros tres en el efector final. Con los grados de libertad propios de los elementos se consigue posicionar, en un punto de la zona operativa, el efector final y con los tres grados de libertad de esta última, se logra orientar en cualquier dirección el elemento terminal.

Mediante la cinemática se resuelven los dos problemas característicos en el posicionamiento del manipulador y que se comentan a continuación.

1) Cinemático directo de posición (MCDP) formula 7.1. Permite determinar las coordenadas operacionales x, y, z en función a las variables articulares θ_1, θ_2, θ_3, además de conocer la posición instantánea del efector final.

2) Cinemático inverso de posición (MCIP) formula 7.2. Es la función inversa del MCDP, permite conocer las variables articulares que el robot deberá adquirir para lograr una acción deseada del efector final.

La resolución de la cinemática directa e inversa de posición puede llevarse acabo aplicando las reglas de la trigonometría, o bien, de forma más exacta, usando el cálculo matricial y las transformadas homogéneas [4].

MCDP.- $x = f(q)$ Formula 7.1 Modelo cinemático directo de posición.

MCIP.- $q = f^{-1}(x)$ Formula 7.2 Modelo cinemático inverso de posición.

La dinámica del robot, trata con la formulación matemática de las ecuaciones del movimiento del brazo. Las ecuaciones dinámicas de movimiento de un manipulador son un conjunto de ecuaciones matemáticas que describen la conducta dinámica del manipulador. El modelo dinámico real de un brazo se puede obtener de leyes físicas conocidas tales como las leyes de Newton y la mecánica lagrangiana. Esto conduce el desarrollo de las ecuaciones dinámicas del movimiento para las distintas articulaciones del manipulador en términos de los parámetros geométricos e inerciales especificados para los distintos elementos. Se pueden aplicar sistemáticamente enfoques convencionales como las formulaciones de Euler- Lagrange y de Newton-Euler para desarrollar las ecuaciones de movimiento del robot.

7.1. Formulación de Euler-Lagrange.

Las ecuaciones de movimiento general de un manipulador se pueden expresar convenientemente mediante la aplicación directa de la formulación de Euler-Lagrange. Muchos investigadores utilizan la representación de Denavit-Hartenberg para describir el desplazamiento espacial entre los sistemas de coordenadas de elementos vecinos para obtener la información cinemática del elemento, y emplean la técnica dinámica lagrangiana para deducir las ecuaciones dinámicas de un manipulador. La aplicación directa se la formulación dinámica lagrangiana, junto con la representación de coordenadas de elementos de Denavit-Hartenberg, resulta en una descripción algorítmica conveniente y compacta de las ecuaciones de movimiento del manipulador [5].

$$\frac{d}{dt}\frac{\partial}{\partial \dot{\theta}} L - \frac{\partial}{\partial \theta_i} L = T_i \qquad i = 1,2, \ldots\ldots n$$

Formula 7.3 Euler-Lagrange

Donde:

T_i = Torque o par.

t = Tiempo.
L = Lagrangiano.
θ = Variable articular o coordenada generalizada.
$\dot{\theta}$ = Velocidad de la variable articular.

$L = \sum_{i=1}^{n} Ki - Pi$ Formula 7.4 Lagrangiano.

Donde:

$K_i = \frac{1}{2} m v_i^2$ Formula 7.5 Energía cinética.

$P_i = m_i g h_i$ Formula 7.6 Energia potencial

K_i: Energía cinética del i-esimo eslabón.
P_i: Energía potencial del i-esimo eslabón.
m_i: Masa constante del i-esimo eslabón.
g: Gravedad 9.81 m/s^2.
h_i: Altura del i-esimo eslabón.
v_i^2: Velocidad del i-esimo eslabón.

7.2. Control PID no lineal

En aplicaciones de control de robots, con tareas de regulación a una coordenada no singular del espacio de trabajo del manipulador, el control proporcional derivativo PD con compensación de gravedad y el control proporcional integral derivativo PID lineal tienen excelente desempeño. Sin embargo, estos controladores no tienen eficiente desempeño en tareas de seguimiento de trayectorias. Un control con eficiente desempeño en tareas de seguimiento, sin conocimiento de los parámetros dinámicos del robot, es el control PID no lineal.

$$\tau = K_d * S + K_i \int \mathrm{sgn}(s)dt$$

Formula 7.7 Control PID no lineal

Simulación del control PID no lineal en MatLab.

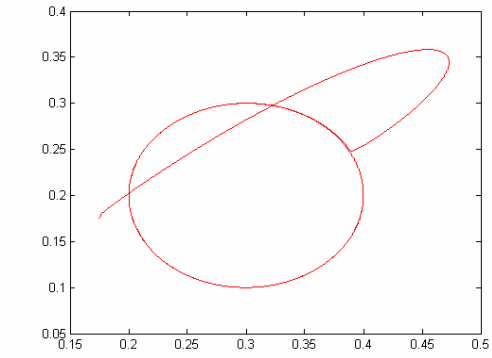


Figura 7.1 Comportamiento y simulacion del PID no lineal.

8. Modelado en 3D Studio Max.

El más popular modelador de soluciones de render para cine, televisión, juegos y visualización de diseño. 3D Studio Max cuenta con las herramientas para dar alta productividad necesaria para hacer animaciones de cine y televisión; los mas vanguardistas juegos electrónicos y las mas distintivas visualizaciones en diseño.

8.1. 3D Studio Max 6.

Esta versión del software ofrece avanzadas posibilidades que permitirán a los profesionales artífices de 3D, desarrolladores y diseñadores, evolucionar a un nivel mas alto de calidad, sofisticación y productividad al crear los juegos mas vendidos, los efectos visuales de películas renombradas o las visualizaciones complejas de diseños industriales, construcciones o simulaciones para gobierno.

Muchas de las características nuevas de 3D Studio Max 6 han sido integradas por requerimiento directo de clientes que trabajan con efectos especiales de películas, visualización de diseños y desarrolladores de juegos, como corporación SEGA de Japón. Los diseñadores y animadores a través de todas las industrias del diseño profesional, podrán disfrutar la profundidad y amplitud de las nuevas características de 3D Studio Max 6, para crear sus diseños en tercera dimensión con mayor rapidez y herramientas más creativas.

Las características de 3D Studio Max 6 incluyen: representación esquemática avanzada para facilitar la visualización y la mejor administración de escenas complejas; el render mental ray®; vertex color painting; herramientas de diseño, visualización y de soporte interactivo con Autodesk y otras soluciones CAD y relacionadas con CAD.

Integración con reactor® 2 y el sistema de física con simuladores de acrobacias y dinámica de vehículos; texturizador de malla distribuido y características que proveen gran accesibilidad al manejo de las aplicaciones a usuarios profesionales. También un sistema de partículas para crear de una forma realista fuentes, niebla, nieve, salpicado, estelas de humo, explosiones y otros efectos ambientales. Discreet 3ds max es usado por el 80 por ciento de todos los desarrolladores de juegos, que han producido productos exitosos como: Grand Theft - Auto, Tom Clancey's Splinter Cell, Star Wars:Knights of the Old Republic, and Neverwinter Nights y continua liderando la industria de desarrollo de videojuegos. 3ds max es usado en un creciente listado de películas de gran taquilla que incluyen: X-Men II, Bulletproof Monk, The Core, Final Destination II, Jason vs. Freddy. 3ds max es la herramienta de elección para realizar visualizaciones especiales, usada por arquitectos y diseñadores de productos con la distinción de haber sido la primera herramienta de software de diseño 3D que ha sido usada por la fundación Frank Lloyd Wright en Taliesin West (Scottsdale, AZ) [6].

8.2. 3D Exploration.

3D Exploration es un espectador fácil de usar para el Windows 95, 98, NT y 2000 con interfaz y apoyo de acelerador de hardware Open GL. Esto permite manejar muchos formatos de archivo populares de 3D: 3D Estudio ,Dirict X, Autocad y muchos otros así como los tipos de archivo comunes de 2D. 3D Exploration puede usarse para presentar una exposición de diapositivas de imágenes de una carpeta entera, en una ventana o la

pantalla entera para presentaciones. El programa también guarda la pista de las imágenes que se ha visto y deja seleccionar imágenes [7].

9. Visual C++ con OpenGL.

Microsoft Visual C++ es un entorno de programación en el que se combinan la programación orientada a objetos (C++) y el sistema de desarrollo diseñado especialmente para crear aplicaciones graficas para Windows (SDK).

Visual C++ es un paquete para desarrollar aplicaciones que incluye, como características mas sobresalientes:

1) Una biblioteca de clases, MFC, que da soporte a los objetos de Windows como ventanas, cajas de dialogo, controles, así como a los objetos GDI (Graphic Device Interface) tales como lápices, pinceles, fuentes y mapas de bits.
2) Un entorno de desarrollo integrado (editor de texto, compilador, depurador, explorador de código fuente, administrador de proyectos, etc).
3) Asistente para el desarrollo de aplicaciones como AppWizard.
4) Galería de objetos incrustados y vinculados (OLE - Object Linking and Embedding).
5) Visualización y manipulación de datos de otras aplicaciones Windows utilizando controles OLE.
6) Una interfaz para múltiples documentos (MDI - Multiple Document Interface) que permite crear una aplicación con una ventana principal y múltiples ventanas de documento.
7) Creación y utilización de bibliotecas dinámicas (DLL - Dynamic Link Libraries).
8) Soporte para el estándar COM (Component Object Model - modelo de objeto componente; en otras palabras, componente software) al que pertenecen los componentes activos (ActiveX o formalmente controles OLE).

Visual ofrece un entorno de desarrollo de C/C++ integrado en Windows, que permite construir aplicaciones Windows (.EXE), aplicaciones de consola (.EXE), bibliotecas de enlace dinámico (:DLL), bibliotecas estáticas (.LIB), modelos AppWizard personalizados (.AWX), controles activos (ActiveX) y otros [8].

9.1. OpenGL

Desarrollado originalmente por Silicon Graphics Incorporated (SGI) para sus estaciones de trabajo de gráficos, permite que las aplicaciones creen imágenes en color de alta calidad independientes de los sistemas de ventanas, los sistemas operativos y el hardware.

9.2. Estándares.

La Organización del estándar de OpenGL es mantenido por un grupo independiente llamado Architectural Review Board (ARB) un consorcio independiente de estándares formado en 1992, gobierna el API de OpenGL, que incluye representantes de la DEC, IBM, Intel, Silicon Graphics y Microsoft. El ARB define pruebas de conformidad, aprueba las especificaciones para las nuevas características y extensiones de OpenGL.

Permite crear imágenes tridimensionales en color de alta calidad con efectos gráficos como la iluminación, trazar un mapa de textura y transformaciones de matriz. OpenGL es un estándar abierto diseñado para correr sobre una variedad de ordenadores y de sistemas operativos.

9.3. Cuando es aplicable.

OpenGL se construye para la compatibilidad a través del hardware y de los sistemas operativos. Esta arquitectura hace fácil los programas de OpenGL de un sistema a otro. Mientras que cada sistema operativo tiene requisitos únicos, el código de OpenGL en muchos programas puede ser utilizado como es. Para el uso de OpenGL en sistemas operativos de Microsoft Windows NT, Windows 2000, Windows 95, y Windows 98, tendrá que modificar sus programas para trabajar con Windows NT / Windows 2000 y Windows 95/98.

9.4. Developer Audience

Diseñada para el empleo por programadores C/C ++ , OpenGL requiere la familiaridad con Windows el interfaz de usuario grafico así como la arquitectura conducida por mensaje. OpenGL requiere Windows NT de Microsoft, Windows 2000, o Windows 95/98 [9].

9.5. GLUT de OpenGL.

El GLUT es una herramienta para uso general de OpenGL, una herramienta independiente del sistema para la escritura de los programas de OpenGL. El GLUT proporciona un API portable así que se puede escribir un solo programa de OpenGL que trabaje a través de todas las plataforma del SO de la PC y del sitio de trabajo [10].

10. Integración del ambiente virtual

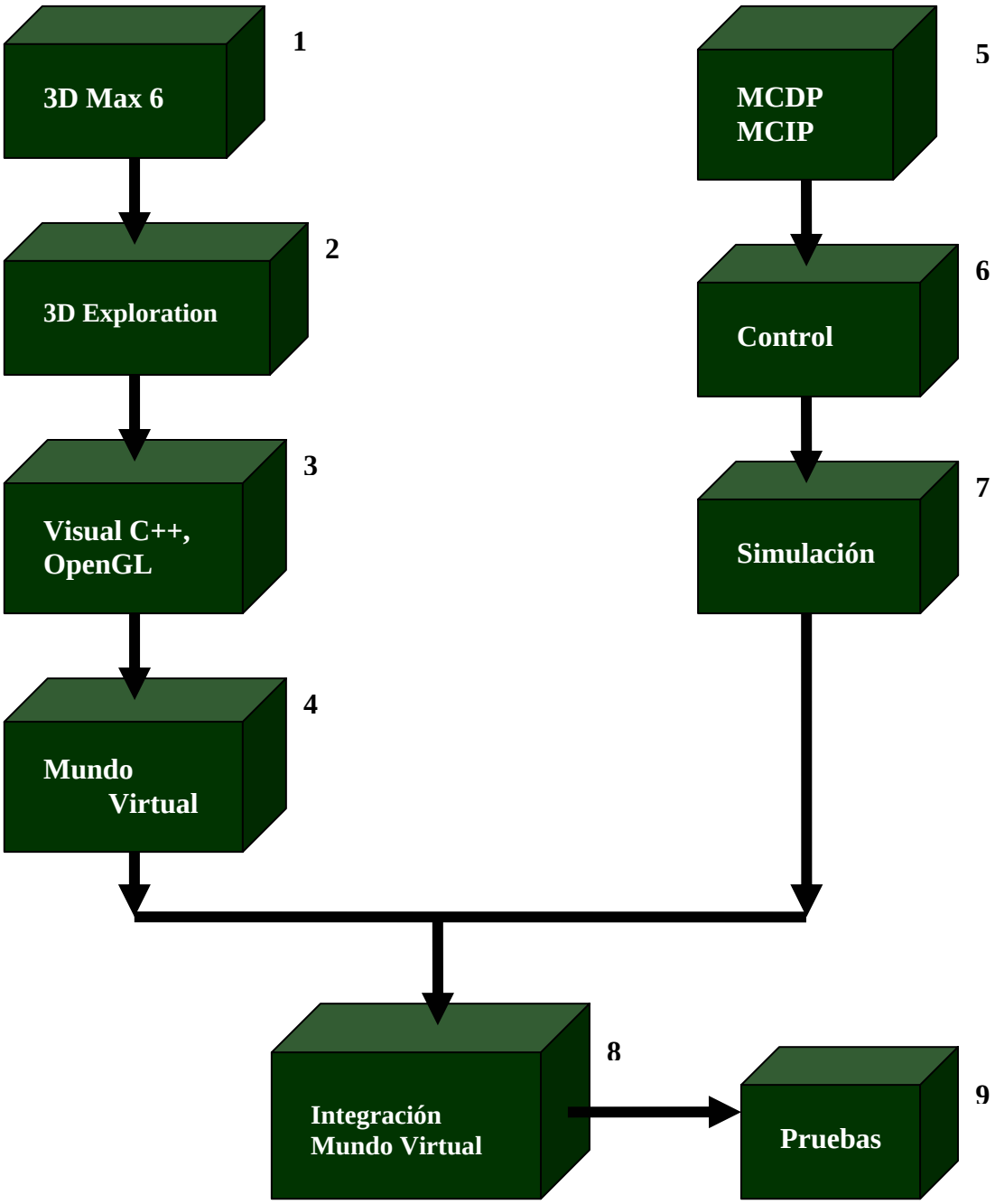


Figura 10.1 Diagrama de bloques.

10.1. Diagrama de bloques.

Bloque 1. 3D Studio Max 6.

El robot Scorbot ER-VII fue modelado en 3D Max 6 e una escala de 1:1, utilizando primitivas estándar y primitivas extendidas, que son objetos tridimensionales paramétricos como cajas, esferas, conos, etc., estos fueron deformados por medio de la lista de modificadores hasta obtener los parámetros y la forma deseada de los eslabones. Hasta obtener la integración de todo el robot Scorbot ER-VII. Posteriormente fue guardado y exportado con extensión .3Ds. La figura 10.1 muestra el modelado y entorno en 3D Studio Max 6 del robot.

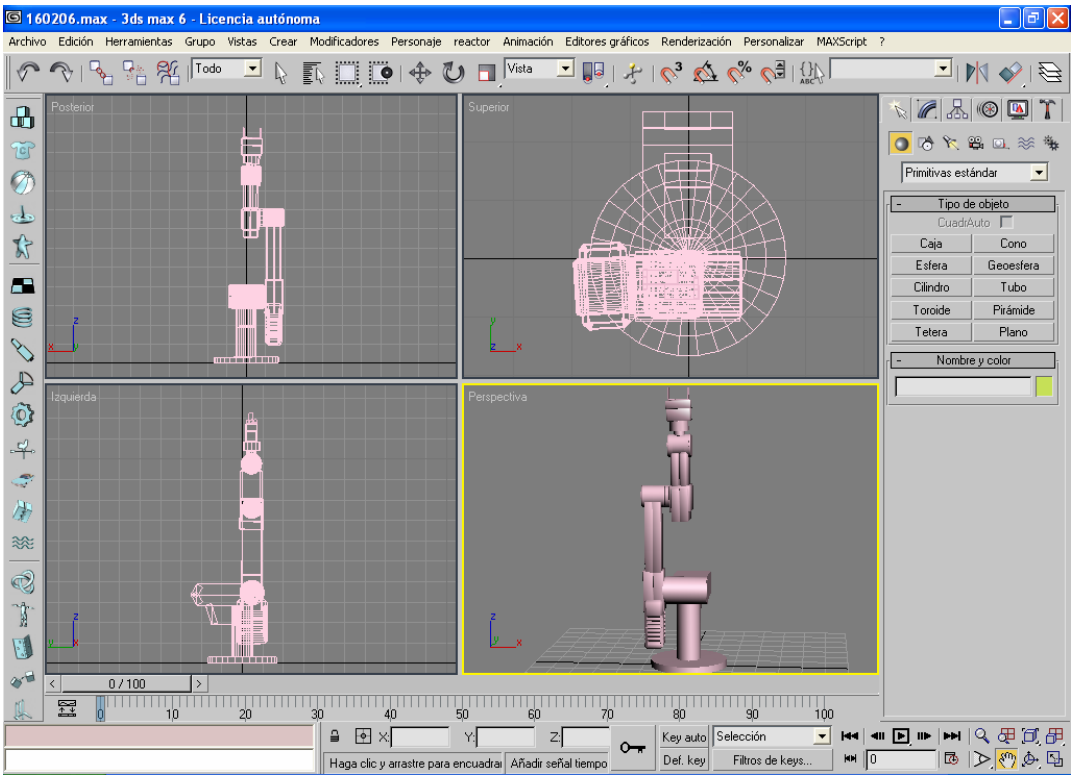


Figura 10.1 Modelado en 3D Studio Max 6

Bloque 2. 3D Exploration.

Por medio de este software abrimos la exportación realizada en 3D Studio Max 6, y generamos una aplicación en Visual C++. La figura 10.2 muestra la exportación en 3D Exploration.

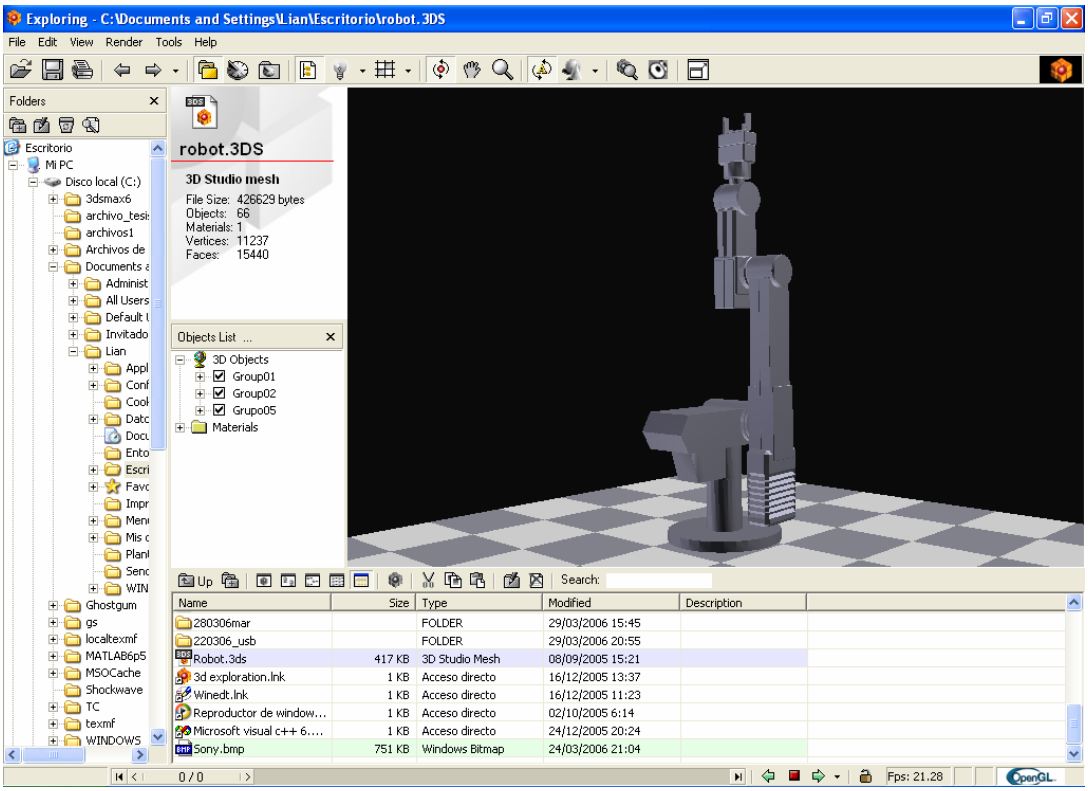


Figura 10.2 Exportación a 3D Exploration.

Bloque 3. Visual C++, OpenGL.

Instalamos los GLUT que nos permitirán el manejo de ventanas e interacción por medio del teclado y el ratón uso de librerías de OpenGL. glut32.dll, glut32.lib, glut.h. La figura 10.3 Muestra la aplicación en Visual C++.

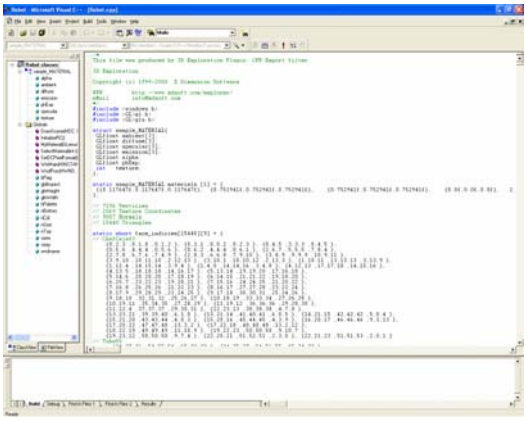


Figura 10.3 Aplicación Visual C++.

Bloque 4. Mundo virtual.

Al abrir la aplicación y ejecutar el programa se muestra el modelo del robot Scorbot ER-VII. El cuadro de control fue realizando en un nuevo proyecto en Visual C++ en el que programamos tres botones, uno que permite visualizar el ambiente virtual, otro para iniciar la animación del mismo y un tercero que sale del programa y diversas cajas de edición en donde se muestra la lectura de los modelos matemáticos que son MCDP, MCIP, los errores instantáneos y la dinámica.

La programación de los muros es realizada por glBegin(GL_QUADS); que genera una arista. Ejemplo

```
glBegin( GL_QUADS );
        glVertex3f(-1.04,0.95,0.83);
        glVertex3f(1.04,0.95,0.83);
        glVertex3f(1.04,0.95,-0.83);
        glVertex3f(-1.04,0.95,-0.83);
glEnd();
```

El código de los eslabones es muy extenso por lo que fue separado en tres partes, base, hombro y codo, sin separar el efector final de este último. La figura 10.4 Muestra la implementación de lo descrito anteriormente.

10.4 Mundo virtual.

Bloque 5. MCDP, MCIP.

Se derivo la cadena cinematica del robot Scorbot ER-VII, posteriormente se obtuvieron los parametros Denavit-Hartenberg que es la definicion de la posicion y orientacion de un marco respecto al otro. Una ves obtenidos prosedemos a sustituirlos en una matriz de transformacion homogenea donde se obtuvo la estructura de posicion, y con la cinematica se obtuvo el MCDP y el MCIP por medio de las formulas descritas en el apartado 7 obteniendo las ecuaciones de movimiento y se implemento en el simulador de MatLab.

Bloque 6. Control.

Una ves optenidos los MCDP y MCIP, se implemento la dinamica con la formulacion de Euler-Lagrange que permitio obtener las ecuacion de movimiento implementando un control no lineal, programandolo en el simulador MatLab.

Bloque 7. Simulación.

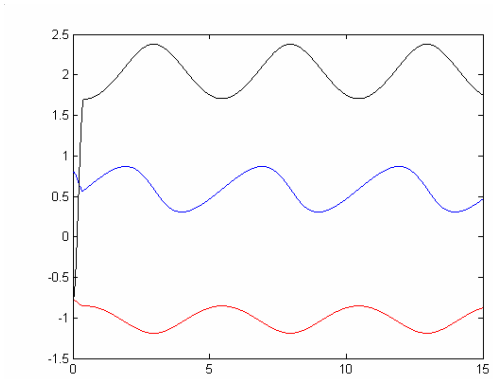


Figura 7.1 Coordenadas operacionales XYZ

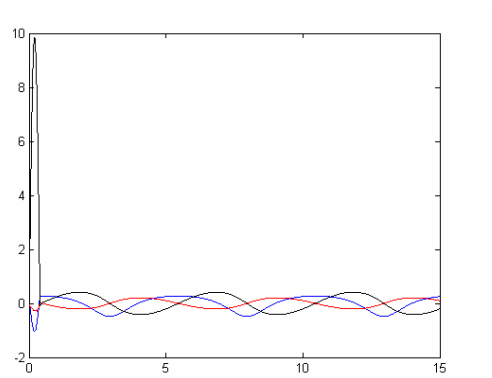


Figura 7.2 Variables aticulares

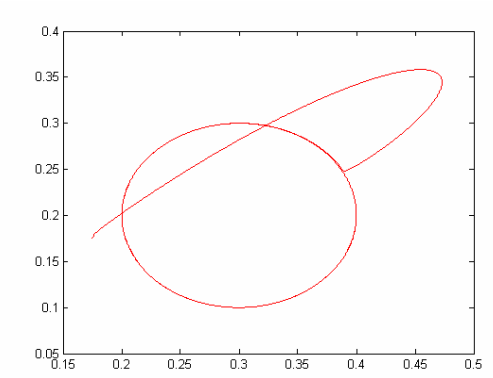


Figura 7.3 Circunferencia en el espacio de trabajo

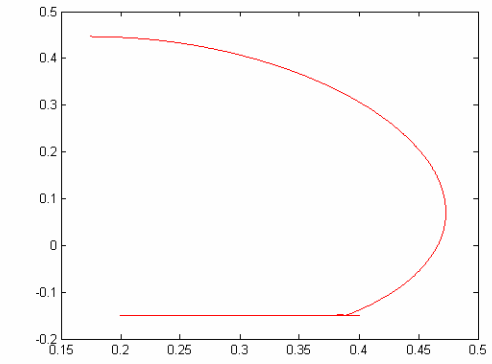


Figura 7.4 Circunferencia en el espacio de trabajo

Bloque 8. Integración al Mundo Virtual

Se evaluaron los valores instantaneso (MCDP), y los valores deseados (MCIP), y se aplico un control no lineal, logrando un margen de error minimo.

Las evaluaciones de estos modelos son guardados en un archivo .mat, que posteriormente son llamados desde Visual C++ por medio de archivos.

Bloque 9. Pruebas.

Los errores instantaneos al aplicar el control PID no lineal fueron minimos, el robot virtual Scorbot ER-VII cumplio con el seguimiento de la trayectoria deseada.

El efector final pinta la trayectoria que sigue la circunferencia. Como se muestra en las siguientes figuras.

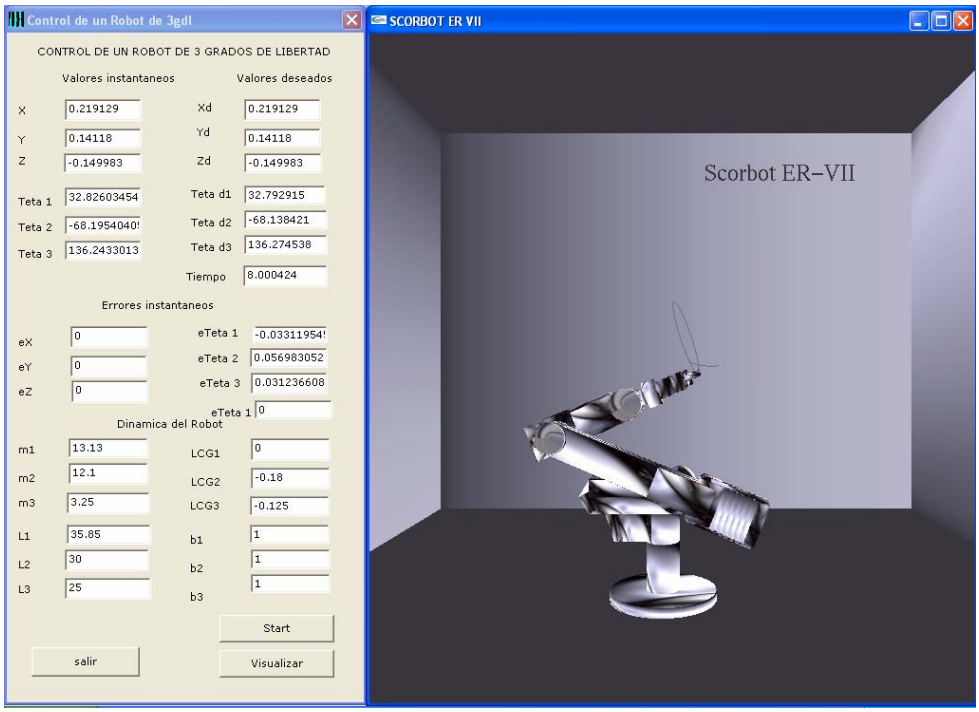


Figura 9.1 Seguimiento de trayectoria

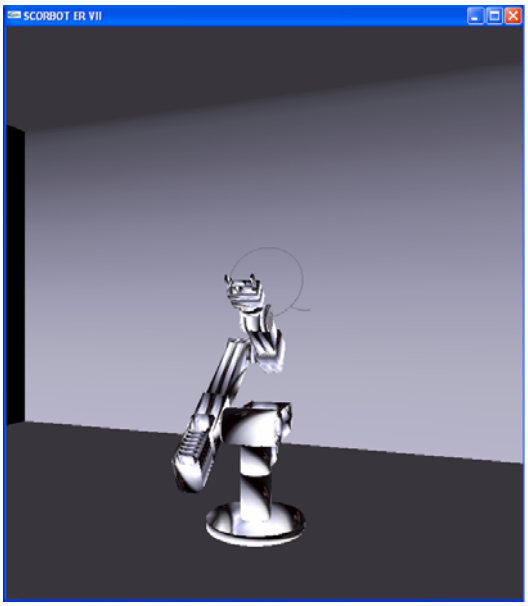


Figura 9.2 Seguimiento de trayectoria

Conclusiones.

En este trabajo de investigación y desarrollo tecnológico se implementó, a partir del modelo cinemático y dinámico del robot Scorbot ER-VII, un ambiente virtual para validar el desempeño de un controlador no lineal, misma situación que puede efectuarse con cualquier otro controlador. Las contribuciones de un visualizador virtual y evaluador de estrategias de control de movimiento para robots manipuladores permiten resolver problemas de diseño, tanto del robot como de las técnicas de control. Este sistema fue realizado a partir de su modelado geométrico en 3D Studio Max, y su exportación a código OpenGL cuyo código no es de trivial interpretación, con esto se establecen las técnicas de graficación convencionales con base en los modelos cinemáticos para lograr los movimientos articulares, dichos movimientos responden a una ley de control basada en el modelo dinámico y su evaluación en Matlab.

Referencias

[1] La Realidad Virtual, una tecnología educativa a nuestro alcance. Emilio R. Escartín. Instituto Superior Politécnico "José A. Echeverría". ISPJAE (Cuba). http://www.sav.us.es/pixelbit/articulos/n15/n15art/art151.htm

[2] Eshed Robotec, "Scorbot-ER VII User´s Manual",September 1991.

[3] González Fuentes Teresa, Rivera Martínez Erika del Roció, " Planificación de tareas basadas en la manilabilidad de robots manipuladores: caso de estudio robot Scorbot ER-VII" junio 2005.

[4] K. S. Fu, R. C. González, C. S. G Lee "Robótica: Control, detección, visión e inteligencia", Mc Graw - Hill, 1990 Primera edición.

[5] Tippens "Física Conceptos y Aplicaciones", Mc Graw - Hill, 1990 , 3ra edición.

[6] ©Copyright 2003, Autodesk, Inc. Todos los derechos reservados.

[7] Copyright c 2001 by Right Hemisphere, Inc. All Rights Reserved. 3D Exploration Author: Alexander A. Shelemekhov Publisher: Vladimir A. Noskov.

[8] Fco. Javier Ceballos, "Microsoft Visual C++ Aplicaciones para Win 32", Alfaomega - RA-MA, Segunda edición.

[9] http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnopen/html/msdn_gl1.asp

[10] http://www.opengl.org/resources/libraries/glut.html
[11] http://www.ruvid.org/actualidad.php?id=47
[12] McAllister, Nyland Popescu, Lastra McCue, "Real Time Rendering of Real World Enviroments", Rendering Techniques '99 Proc. of Eurographics Workshop o Rendering.

[13] Juan Manuel Ibarra Zannatha “Hacia un Laboratorio Virtual de Robótica” Centrode Investigación en Computación del IPN
[14] http://vicorob.udg.es/fitxers/projectes/uris_es.html

[15] Emilio R. Escarpín, “La realidad virtual, una tecnología educativa a nuestro alcance**”,** Instituto Superior Politécnico “José A. Echeverría”. ISPJAE (Cuba)

Scorbort ER-VII Virtual:
Mathematical Model and Control of Movements

Liana López Pacheco, Verónica Martínez Olguín and Omar Arturo Domínguez Ramírez
Research Center on Information and Systems Technologies

I. Summary

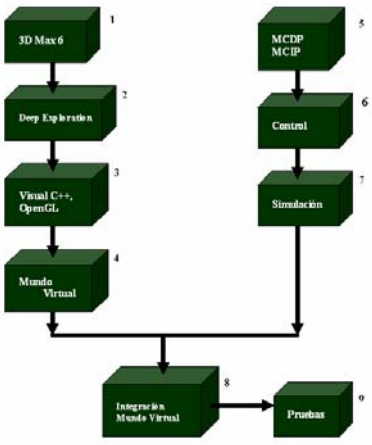
This project presents the design and integration of the robot's dynamic virtual environment as a result Scorbot ER-VII employee to evaluate strategies of control of movement with the use sliding mode control starting from the transitory respond of its geralized coordinate and its velocity. The system was elaborated in Visual C++ with tool box OpenGL and 3D Studio MAX for the modeling of the robot's structure with details fine Kematics and the export to OpenGL code is used Deep Exploration edition CAD. The time responds are solved with Matlab soft.

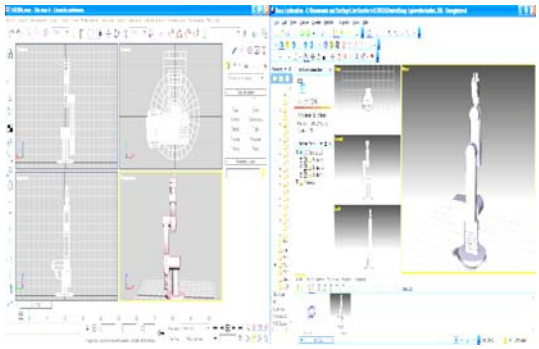
II. Introduction

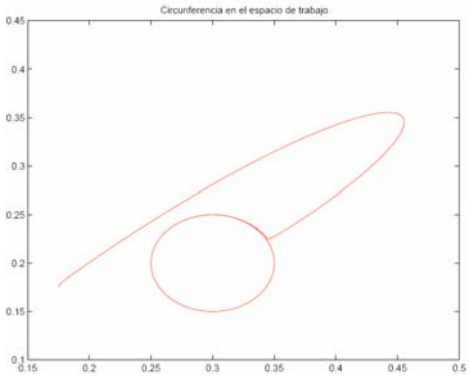
This project has the purpose of developing a system of scientific visualization with environment of virtual reality whose behaviors are based on the solutions of the mathematical models and control.

III. Study case

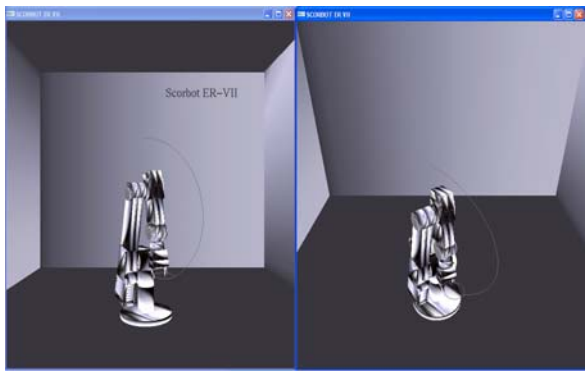
For the construction of the virtual environment following squeme is showing:







- Virtual environment



IV. Conclusions

In this research work it was implemented, starting from the forward kinematics and dynamic model of the robot Scorbot ER-VII, end virtual environment is validate with the non linear control in this case sliding mode control with time base generator for tracking in finite time .

lizli_lopez@hotmail.com, verola_mar@hotmail.com

ISBN 970-769-070-4, ISRA 2006 Proceedings

Apéndice G

Metodología

La metodoligia desarrollo de sistemas (SMD), se aborda en está sección como herramienta para el análisis, diseño, desarrollo, evaluación e implementación del software empleado.

G.1. Análisis del sistema

Para poder tener un buen uso del Software se recomienda como requisitos minimo: tener instalado Windiws 2000, procesador Pentium III a 266, 128MB de RAMuna tarjeta de los gráficos capaz de desplegar 24-bit color y un acelerador grafico de videos y los requisitos recomendados: tener windows XP, procesador Pentium III a 733GHz, 256MB en RAM y una targeta de acelerador de videos con 128Mb en RAM.

Además para poder hacer un buen uso y visualización del ambiente vitual se tiene que instalar las librerías:

```
glut32.dll en %WinDir%\System,
glut32.lib en $(MSDevDir)\..\..\VC98\lib, y
glut.h     en $(MSDevDir)\..\..\VC98\include\GL.
```

Por último no es necesario tener instalado Visual C++. Se hizo un estudio comparativo entre las diferentes herramientas de diseño en donde se determino utilizar para la construcción del escenario virtual:

- 3D Studio Max: es el mejor modelador en objetos 3D y su exportación en diferentes codigos.
- Deep Exploration: generar un archivo de código fuente de C++, un archivo de proyecto y un espacio de trabajo de proyecto para poder crear nuestro ambiente virtual utlizando Visual C++ y líbrerias de OpenGL
- Visual C++, OpenGL: se utilizó Visual c++ para nuestra comodidad con ambientes graficos en OpenGL e interacción con la ventana ya que VRML genera mucho espacio en memoria y Java 3D es demaciado complejo.
- MatLab: es utilizado para hacer calculos matriciales y vectoriales.

El sistema esta integrado por un cuadro de control y la visualización virtual del robot. El cuadro de control visualiza los valores instantaenos, los valores deseados, los errores instantaneos, la dinámica del robot y los botones, el ambiente virtual se visualiza al robot Scorbot ER-VII para ser manipulado por el cuadro de control, teclado y/o mause.

Valores instantaneos: muestran los valores instantaneos a través de los cuadros de edición nombrados X, Y, Z, que son la posición del efector final en el espacio virtual, Tata1, Teta2, Teta3 son los ángulos de las articulaciones de los eslabones, estos valores son adquiridos del robot en el ambiente virtual mientras va simulando el seguimiento de la trayectoria haciendo una circunferencia como tarea calculada.

Valores deseados: se encuentran representados en los cuadros de edición de la manera siguiente; Xd, Yd, Zd son las coordenadas operacionales deseadas del efector final, Teta d1, Teta d2, Teta d3 son las variables articulares deseadas del robot y el Tiempo es el lapso de tiempo en el que se calcula la tarea, estos valores deseados son leidos desde un archivo llamado circunferencia.mat y en donde en el ambiente virtual el robot sigue esa lectura de datos obteniendo como resultado el seguimiento de la circunferencia que queremos que el robot simule.

Errores instantaneos: eX, eY, eZ corresponde a los errores del efector final y los eTeta1, eTeta2 y eTeta 3 son los errores en las articulaciones, todos ellos son los errores mínimos que el robot va adquiriendo mientras se va aproximando a los valores deseados para hacer el seguimiento de la trayectoria.

Dinámica del robot: m1, m2, m3 son las masas de los eslabones, L1, L2, L3 son las longitudes de los eslabones, LCG1, LCG2, LCG3 son las longitud al centro de masas, b1, b2 y b3 son la frinción viscosa, estos valores que se representan son las caráteristicas que integran al robot y no son de longitud variable por eso son estaticos.

Botones: Circunferencia, Graficar, Borrar y Salir son las diferentes opciones que podemos elegir para que se ejecute la acción deseada y sea reflejada en el ambiente virtual del robot, el botón Circunferencia al seleccionar esta acción llama al archivo circunferencia.mat para ser leido desde los cuadros de edición de los valores deseados y que el robot ejecute la tarea, el botón Graficar hace que el robot Scorbot ER-VII grafique en el ambiente virtual la circunferencia y al terminar de leer los datos regrese a su posición inicial es decir la de reposo, el botón borrar permite borrar el seguimiento de la trayectoria que el robot simulo, el botón Salir nos permite cerrar el sistema.

La ventana del ambiente virtual del robot muestra el comportamiento que es leido desde el cuadro de control, obteniendo como resultado el seguimiento de una trayecroria. El robot se puede apreciar desde diferentes vistas, puede ser manipulado por medio del mouse y/o teclado [60].

La figura G.1 muestra la interfaz del ambiente virtual.

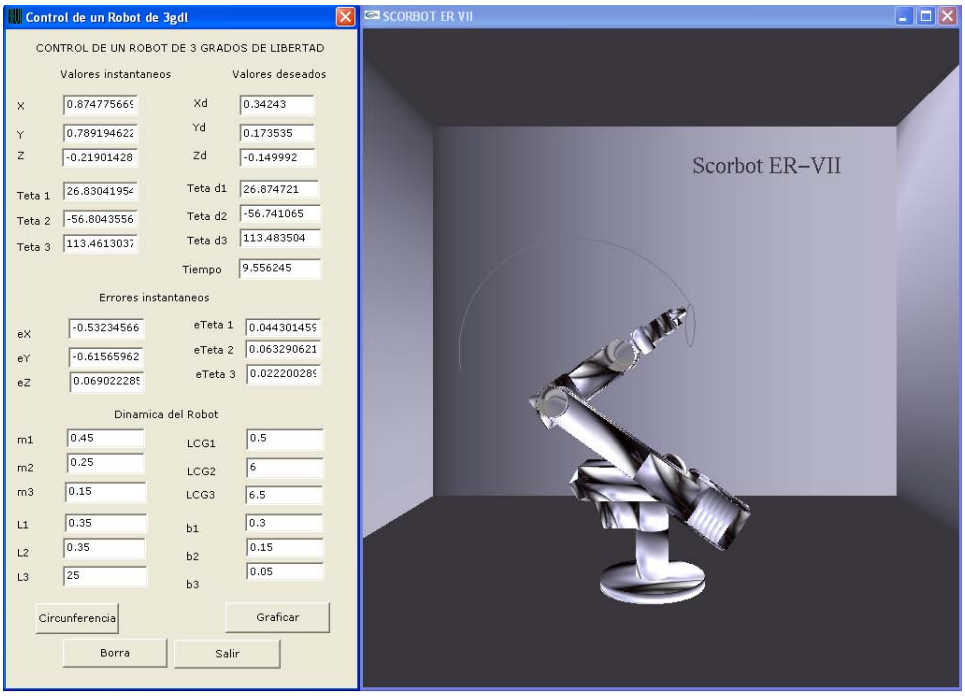


Figura G.1: Visualización de la interfaz del ambiente virtual.

G.2. Diseño general del sistema

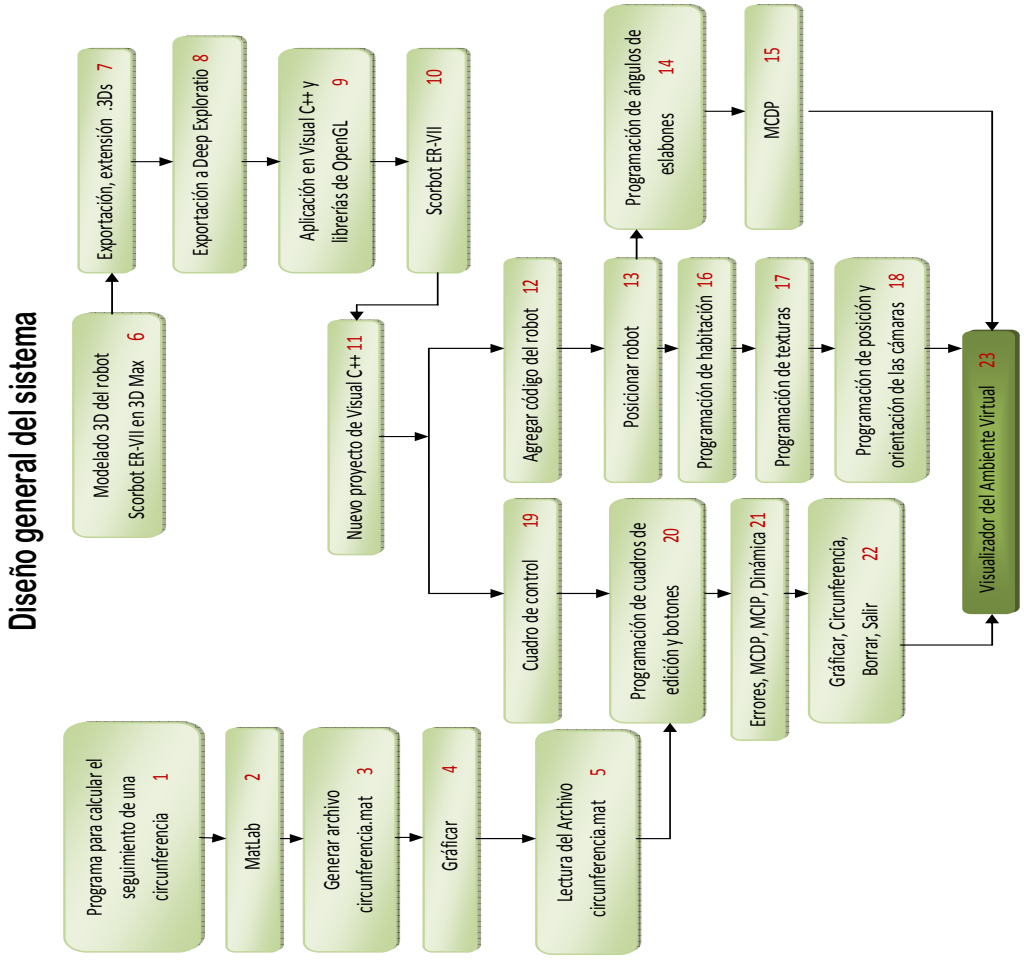


Figura G.2: Diseño general del sistema.

La figura G.2 representa de manera general como se fue construcyendo el ambiente virtual en donde:

- En el bloque 1 se programa el calculo de la circunferencia utilizando los modelos matemáticos y de control el de modos deslizantes para su validación.
- En el bloque 2 se utiliza MatLab para hacer el comportamiento de la simulación ya que este software es utilizado especialmente para los calculos matemáticos, matrices y vectores.
- En el bloque 3 una vez obtenido estos 2 pasos anteriores se procede a generar el archivo circunferencia.mat en donde se guardara en ese archivo 7 valores que corresponden a las coordenas operacionales, las variables articulares y el tiempo.
- Posteriormente en el bloque 4 se hace la simulación obteniendo como resultado la circunferencia.
- En el bloque 5 se hace la lectura del archivo desde el cuadro de control para que el robot llege a los valores deseados haciendo la simulación en el ambiente virtual.
- En el bloque 6 se modela al robot en 3D utilizando 3D Studio Max.
- En el bloque 7 se exporta al robot de 3D Studio Max a Deep Exploration.
- En el bloque 8 ya que se hizo la exportación se ejecuta en Deep Exploration el archivo para la generación de código en visual c++ y librerías de OpenGl.
- En el bloque 9 se genera la aplicación en Visual C++ y librerías de OpenGL para ser manipulado en el ambiente virtual.
- En el bloque 10 Arroja como resultado el código del robot Scorbot ER-VII asi como algunas características como propiedades de la ventana.
- En el bloque 11 se crea un nuevo proyecto de Visual C++ para la manipulación del robot desde un cuadro de control.
- En el bloque 12 en la parte que se refiere de agregar código del robot, separamos el código en 3 partes que son: base, hombro y codo con el objetivo de poder manipular las piezas por separado.
- En el bloque 13 posesionamos al robot en el ambiente virtual en donde se visualizara de manera definitiva por medio de glTranslatef, glRotatef, glScalef.
- bloque 14 en la programación de ángulos de los eslabones se hace esto para ser reflejado en el cuadro de control.
- En el bloque 15 se va obteniendo el MCDP mientras el robot va simulando en el ambiente virtual la tarea de la trayectoria, además que se hace la lectura del control de modos deslizantes para que el robot siga esos datos en el ambiente virtual.
- En el bloque 16 programación de la habitación, en esta parte se dibujan polígonos para simular la habitación.
- En el bloque 17 programación de texturas esto es para darle un mejor renderizado al robot y a las paredes de los muros.
- En el bloque 18 programación de la posición y orientación de las cámaras se realiza por medio de las funciones void keyboard(char key, int x, int y), void DisplayMouseMotion(int x, int y) para visualizar y manipular al robot en el ambiente por medio del teclado y/o mause.
- En el bloque 19 cuadro de control en esta sección se procede a diseñar y dibujar el cuadro de control.
- En el bloque 20 programación de cuadros de edición y botones, se hace la programación de ello, en donde se visualizara los datos y se ejecutaran las acciones que nosotros seleccionemos.

- En el bloque 21 en este bloque se procede a programar en Visual C++ los errores, MCDP, MCIP, dinámica que serán visualizados en el cuadro de control.
- En el bloque 22 Graficar, circunferencia, Borrar y Salir estos botones nos van a permitir ejecutar la acción que seleccionemos.
- En el bloque 23 visualizador del ambiente virtual, en este bloque se ve reflejado todos los demás bloques [60].

G.3. Desarrollo

El desarrollo del sistema se encuentra en el capítulo 4.

G.4. Implementación

En un proyecto a futuro se pretende crear un laboratorio virtual de robótica y una plataforma remota para teleoperaciones unilaterales, esta plataforma remota se pretende realizar de la forma siguiente:

- Como punto numero 1 se pretende tener una estación local(master) en donde sera manipula por medio del P5 Glove y el ambiente virtual de robot Scorbot ER-VII, la interfaz de visualización del P5 Glove.
- Como punto numero 2 sera manupulada por una estación remota(slave) en donde se visualizara el ambiente virtual del robot Scorbot ER-VII

Esto se llevara acabo por medio del IPV6, este paso pertenece al proyecto en el que la estación remota corresponde a un robot real. En este caso el robot Scorbot ER-VII virtual permite realizar todas las pruebas de control y comandos remotos, así como la caracterización de la plataforma [60].

G.5. Evaluación

Se procedio a la evaluación del software en donde se determina que es eficiente ya que se realizaron pruebas de autenticidad del sistema. Mencionaremos algunas pruebas que se realizaron:

- Se hieron otros calculos de la trayectoria de la circunferencia donde se cambio el el programa de Maltab los radios y espacio de trabajo para el seguimiento de la tarea donde el robot en el ambiente virtual se comporto de manera optima.
- En la interfaz de visualización del ambiente virtual se procedio a interactuar con los usuarios en donde apreciaron la autenticidad [60].

Bibliografía

[1] Omar Arturo Dominguez Ramirez, *"Interfaces Hápticas: Matemático y control de movimiento adaptativo"*, series de informes técnicos: sistemas mecatrónicos No. 2 de la serie.

[2] *"http://www.pergaminovirtual.com.ar/revista/cgi-bin/hoy/archivos/2006/00000848.shtml"*.

[3] *"http://msdn.microsoft.com/robotics/getstarted/simulation/default.aspx"*.

[4] *"http://www.bv.umsanet.edu.bo/revistas/bibliotecologia/numero3-4/articulos/realvirt.htm"*.

[5] *"http://telematica.cicese.mx/computo/super/cicese2000/realvirtual/Part2.html"*.

[6] *"http://www.uv.mx/rm/num anteriores/revmedicaarticulos/cirujano virtual.html"*.

[7] *"http://www.dte.eis.uva.es/Docencia/ETSII/Edig1/Practicas/ManPS.pdf"*.

[8] *"http://telematica.cicese.mx/computo/super/cicese2000/realvirtual/Part3.html"*.

[9] *"http://www.previsl.com/es/rvirtual/index.asp"*.

[10] Joe Gradecki, *"Realidad Virtual construcción de proyectos"*, ra-ma.

[11] Carlos David Correa, Miguel Ángel González, Juliana Restrepo, Christian Trefftz, Helmuth Trefftz, *"Realidad Virtual Distribuida para soportar la Educación a Distancia en Colombia"*, IV Congresso RIBIE, Brasilia 1998.

[12] Roxana E. Pintos, Sonia I. Mariño, Maria V. Godoy, *"La realidad virtual como herramienta en la enseñanza-aprendizaje de la anatomía humana para el nivel EGB II", Facultad de Humanidades"*, Universidad Nacional del Nordeste. Resistencia.

[13] Erika del Rocío Rivera Martínez, Victor Hugo Martínez González, Victor Hugo Pérez Barrera, Arturo Curiel Anaya, Omar Arturo Domínguez Ramírez, *"Desarrollo de una ambiente virtual para interactuar con el guante P5"*, Árticulo.

[14] Janet Saray Sosa Márquez, Francisco Ramos Ortiz y Arturo Curiel Anaya, *"Desarrollo de Software Educativo Caso de estudio: Matemáticas de Sexto de Primaria Área de Conocimiento: Computación Educativa"*, Árticulo.

[15] *"http://www.contactomagazine.com/realidadvirtual0417.htm"*.

[16] *"http://cybervox.org/j/oldsite/vrml/"*.

[17] *"Game Programming"*, MA, USA:Course Technology, 2002.

[18] Dirk Louis, *"Delphi 5"*, Alfaomega marcombo S.A. 2000.

[19] *"http://users.tiscali.es/acanudas/pdf/dp5entor.pdf"*.

[20] Rowe, Glenn, *"Java 3D"*, Palgrave Macmillan 2001.

[21] Murdock, Kelly, *"Maya"*, Course Technology Crisp 2004.

[22] *"http://stahlforce.com/games/info.html"*.

[23] Kennedy, Sanford, *"3D Studio Max"*, Charles River Media, 2004.

[24] Middlebrook, Mark, *"AutoCAD 2006 for Dummies"*, Hoboken, NJ, USA: John Wiley & Sons, Incorporated, 2004.

[25] *"http://www.sycode.com/products/terrain$_a$c/images/terrain$_a$c.gif"*.
"http : //ciberhabitat.gob.mx/hospital/robotica/conceptos.htm".

[26] *"http://ciberhabitat.gob.mx/hospital/robotica/conceptos.htm"*.

[27] *"http://www.mor.itesm.mx/∼lsi/redii/Lab_vir_der.html"*.

[28] *"http://www.tecnoedu.com/Denford/RobotVR.php"*.

[29] *"http://www.great-lakes-training.biz/robocell.htm"*.

[30] Francisco A. Candelas, Fernando Torres, Pablo Gil, Francisco Ortiz, Santiago Puente, Jorge Pomares, *"Laboratorio virtual remoto para robótica y evaluación de su impacto en la docencia"*, Dpto. de Física, Ingeniería de Sistemas y Teoría de la Señal Escuela Politécnica Superior, Universidad de Alicante. Alicante, España.

[31] Jose M^a Angulo Usategui *"Robótica Práctica, Tecnología y Aplicaciones"*, Raraninfo 4^a Edición 1996.

[32] Denavit, J. and R. Hartenberg, *"Kinematic notation for lower pair mechanisms based on matrices"*,Journal of Applied Mechanics, Vol. 77

[33] Murray, R., Li, Z., Sastry, S., *"A mathematical introduction to robotic manipulation"*, CRC press LLC, 1994.

[34] Rafael Kelly, Víctor Santibáñez, *"Control de Movimiento de Robots Manipuladores"*, Pearson Education, S. A., Madrid, 2003.

[35] Katshuhiko Ogata, *"Ingenieria de control moderno"*, University of Minnesota, Pretice-Hall Hispanoamericana, S.A.

[36] *"©Copyright 2003, Autodesk, Inc. Todos los derechos reservados"*.

[37] *"http://www1.discreet.com/site/latiname.nsf/products/ E38D3D66EB3A857A85256DD0004DD47C?opendocument"*.

[38] *"Copyright c 2006 Right Hemisphere"*.

[39] Fco. Javier Ceballos, *"Microsoft Visual C++ Aplicaciones para Win 32"*, Alfaomega - RA-MA, Segunda edición.

[40] *"http://support.microsoft.com"*.

[41] Frederick, j Buech *"Física general 1"*, Mc Graw Hill.

[42] J. Aguilar Peris *"Physica 1 Science Study Committe"*, editorial Reverte, s.a, 1970.

[43] José M. Angulo Usategui *"Robótica practica Etnología y aplicación"*, Editorial Paraninfo, 3ra. Edición.

[44] *"http://www.great-lakes-training.biz/robocell.htm"*.

[45] Nof, S. *"Handbook of Industrial Robotics "*, John Wiley Sons, INC 2da Edición, 1999

[46] Héctor Pérez Montiel *"Física general 1"*, Publicaciones Culturales.

[47] Francis W.Sears, Mark W. Zemonsky *"Fisica Universitaria"*, 6ta edición.

[48] Paul G. Hewitt *"Fisica Conceptual"*, editorial Addison Wesley Longman, 1999.

[49] Don.McCloy, D.M.J.Harris *"Robotica Industrial"*, Limusa, Grupo noriega editores 1993, Editorial Limusa, S.A de C.V

[50] Addison-Wesley Publishing *"Fisica conceptual"*, Segunda Edición Company, Menlon Park, California, E.U.A. 1992. Longmain.

[51] *"http://ieee.udistrital.edu.co/concurso/sistemas_dinamicos/Sistemas %20Dinamicos/SDVisual3.htm"*.

[52] Robert Resnick, David Halliday *"Fisica Parte 1"*, editorial Continental, 1980.

[53] Don McCloy y Michael Harris *"Robotica, una introducción"*, Limusa, 1993.

[54] Hector armando Gomez Hernandez *"Desarrollo virtual del conjunto arquitectonico del centro de autoacceso de la UAEH"*, , Tesis.

Mikell P. Groover, Mitchell Weiss, Roger N. Nagel y Nicolas G. Odrey. *"Robotica industrial, Tecnoligías, programación y aplicaciones"*,Derechos reservados 1990, interoamericana de México S.A de C.V.

[55] *"http://pronton.ucting.udg.mx/materias/robotica/r166/r66/r66.htm"*.

[56] *"http://www.depi.itch.edu.mx/apacheco/expo/html/ai10/page1"*.

[57] *"http://www.edu.aytolacoruna.es/aula/fisica/fisicaInteractiva/Ondasbachillerato /ondasCaract/ondas-Caract_indice.htm"*.

[58] Murray, R., Li, Z., Sastry, S. *"A Mathematical Introduction to Robotic Manipulation"*,CRC press LLC,1994.

[59] *"http://energiaycomputacion.univalle.edu.co/edicion20/revista20_1a.html"*.

[60] John G. Burch. Gary Grudnitski *"Diseño de sistemas de información teoría y práctica"*, Grupo Noriega Editores.