

**DESARROLLO DE UN MODELO VIRTUAL TRIDIMENSIONAL DE ROBOT
SCARA MEDIANTE LA PLATAFORMA PROCESSING 2.0**

LEONARDO DURAN MELO

UNIVERSIDAD PEDAGÓGICA NACIONAL

FACULTAD DE CIENCIA Y TECNOLOGÍA

LICENCIATURA EN ELECTRÓNICA

2017

**DESARROLLO DE UN MODELO VIRTUAL TRIDIMENSIONAL DE ROBOT
SCARA MEDIANTE LA PLATAFORMA PROCESSING 2.0**

Por:

LEONARDO DURAN MELO

Informe Final Proyecto

Para optar por el título de Licenciado en Electrónica

Tutor:

Prof. Diego Fernando Quiroga Páez

UNIVERSIDAD PEDAGÓGICA NACIONAL

FACULTAD DE CIENCIA Y TECNOLOGÍA


LICENCIATURA EN ELECTRÓNICA

2017

Nota de Aceptación

Firma del tutor de trabajo de grado

Bogotá D.C., agosto de 2017

 UNIVERSIDAD PEDAGÓGICA NACIONAL <i>Realizando lo Posible</i>	FORMATO	
	RESUMEN ANALÍTICO EN EDUCACIÓN - RAE	
Código: FOR020GIB	Versión: 01	
Fecha de Aprobación: 10-10-2012	Página 4 de 60	

1. Información General	
Tipo de documento	Trabajo de grado
Acceso al documento	Universidad Pedagógica Nacional. Biblioteca Central
Título del documento	Desarrollo de un modelo virtual tridimensional de robot Scara mediante la plataforma Processing 2.0
Autor(es)	DURÁN MELO, Leonardo
Director	QUIROGA, Diego Fernando
Publicación	Bogotá. Universidad Pedagógica Nacional, 2017. 60p.
Unidad Patrocinante	Universidad Pedagógica Nacional
Palabras Claves	INTERFAZ GRÁFICA, ROBOT SCARA, PROCESSING 2.0

2. Descripción
<p>Este trabajo desarrolla un prototipo virtual tridimensional de un robot SCARA industrial que incluye la implementación del modelado matemático de la cinemática directa e inversa, y una interfaz tridimensional interactiva, para la futura enseñanza e implementación de simulaciones 2D y 3D de robótica y mecanismos de eslabones a estudiantes de pregrado en áreas de electrónica, control y robótica.</p>

3. Fuentes
<p>Adep. 2016. <i>Technologies</i>. Recuperado de https:// www.adept.com/products/robots/scara/cobras600/</p> <p>Ardila, 2014. <i>Brazo robótico de un grado de libertad modelado como péndulo invertido y operado por computador</i>. Salamanca, España:</p>

Craig, 2016. *Robótica*. México: Prentice Hall

Muñoz, 2013. *Sistema de control para un robot self-balancing*

Pérez, 2009, *Diseño, modelamiento y simulación 3D de un robot móvil para exploración de terrenos*

Processing, 2017. Recuperado de <http://processing.org/>

Ramirez, 2017. *Cinemática directa de robot*. Recuperado de <http://www.kramirez.net/>.

Rodríguez, 2014. *Prototipo de robot submarino con la capacidad de seguimiento de trayectorias mediante tratamiento de imágenes*

Sun, 2017. *El Lenguaje de Programación Java*. Recuperado de <http://www.mmc.geofisica.unam.mx/>

Torres, 2016. *Diseño de trayectorias del efector final de un robot manipulador para la evitación de obstáculos*. Recuperado de <http://repositorio.upct.es/bitstream/handle/>

4. Contenidos

Para la modelación del robot en Processing inicialmente se trabajó con los modelos CAD que fueron adaptados a modelos OBJ y luego vinculados a una carpeta de datos para la elaboración de la interfaz. Paralelamente se desarrolla el modelo matemático, iniciando por establecer los sistemas de referencia del robot, los parámetros Denavit Hartenberg y el desarrollo de matrices de transformación homogénea.

Luego de este modelado se desarrolla el programa de la interfaz en lenguaje Java en la consola de processing, permitiendo la interacción del usuario con el robot utilizando dos modos de trabajo: cinemática directa y cinemática inversa.

5. Metodología

Para el desarrollo de la interfaz gráfica se desarrollaron tres actividades fundamentales descritas a continuación:

Desarrollo del modelo

En primer lugar se establecieron los sistemas de referencia y luego se procede a tabular los parámetros Denavit – Hartenberg; con estos parámetros se procede a desarrollar las matrices de transformación homogénea de manera consecutiva entre eslabones para obtener la posición final del actuador o último eslabón respecto de la base del robot.

Luego de tener las ecuaciones y las matrices de transformación se procede a generar el algoritmo que permite determinar los movimientos del robot SCARA.

Desarrollo de la programación en Processing

Se establece un diagrama de clases de acuerdo a los parámetros necesarios para que el robot realice diferentes movimientos en la interfaz de acuerdo a las necesidades del usuario. Se codifican todas las acciones y se establecen los vínculos para que el robot se visualice en la pantalla de la interfaz.

Después de este proceso se programa la ventana de interacción de los valores de las articulaciones y se configuran las teclas auxiliares para la manipulación total del robot como: Vista isométrica y en perspectiva, rotación y translación dentro de la pantalla, ejes de referencia y cambio de cinemática directa e inversa.

Prueba de la interfaz gráfica

Una vez se ha desarrollado el código se procede a correr el programa en la ventana de Proscene, que es una de las librerías de Processing que permite visualizar el robot, manipularlo y ver sus movimientos y la posición final del actuador mediante un sistema de coordenadas.

6. Conclusiones

Los resultados obtenidos a través del desarrollo de la interfaz gráfica, aseguran que la solución está acorde con los objetivos planteados y permitirá a estudiantes de pregrado tener un conocimiento básico de los movimientos y parametrización de un robot industrial.

El desarrollo de la programación en Processing, brinda una herramienta versátil y sencilla, basada en el lenguaje Java, para desarrollar modelos en 2D y 3D, de prototipos de máquinas y robots, que puede ser utilizada por jóvenes de secundaria y pregrado con la ventaja que es un software sin costo alguno y que constantemente está en procesos de actualización de sus librerías y aplicaciones.

Este trabajo es la base para futuros desarrollos en la programación y parametrización de robots en la universidad, permitiendo el fortalecimiento de la interacción entre la industria y la academia, así como la integración de nuevas tecnologías para la consolidación de la innovación

tecnológica en diferentes campos de aplicación.

El desarrollo de los modelos virtuales es muy importante en el contexto educativo ya que permite simular los objetos antes de desarrollarlos realmente, lo que nos da la certeza de su funcionamiento óptimo y de las limitaciones o posibles errores en el proceso de diseño, además del gran potencial de los sistemas computacionales en todos los campos del conocimiento.

Elaborado por:	DURÁN MELO, Leonardo
Revisado por:	QUIROGA, Diego Fernando

Fecha de elaboración del Resumen:	22	08	2017
--------------------------------------	-----------	-----------	-------------

Resumen

En este trabajo se presenta el desarrollo de una interfaz gráfica, en la cual se modela un robot SCARA, utilizado comúnmente en la industria, haciendo uso de la plataforma de diseño Processing 2.0, para visualizar sus movimientos básicos y los parámetros de sus articulaciones tanto en la cinemática directa como en la inversa.

El proceso se ha basado en el marco referencial de la aplicación de las TICs, para optimizar el trabajo en el diseño y parametrización de los robots, aprovechando el campo de la virtualidad para la simulación de los sistemas antes de llegar al desarrollo real de un prototipo.

Este modelo ha sido desarrollado utilizando herramientas de diseño CAD como Solid Works, para generar las diferentes partes del robot, que luego se han integrado al programa de diseño Processing, en donde finalmente se ha ejecutado la programación para que se visualice, mediante la librería de Proscene, la interfaz gráfica final.

Palabras clave: SCARA, Processing, robot, interfaz gráfica de usuario, simulación.

Abstract

This project will show the development of a graphic interface in which was model a SCARA robot, commonly used on the industry, in the design platform Processing 2.0, to visualize its basic movements and the parameters of its joints on the direct and inverse kinematic.

The process has been based on the referential framework of the application of the TICs, to optimize the work in the design and parameterization of the robots, taking advantage of the field of virtuality for the simulation of the systems before reaching the real development of a prototype.

This model has been developed using CAD design tools such as Solid Works, to generate the different parts of the robot, which are then integrated into the design program Processing, where finally the programming has been executed to be displayed, Proscene, the final graphic interface.

Keywords: SCARA, Processing, robot, graphical user interface, simulation.

Índice general

Resumen	8
Abstract	9
Introducción.....	11
Capítulo 1	13
Planteamiento del problema y justificación	13
1.1 Planteamiento del problema	13
1.2. Justificación.....	14
Capítulo 2	15
Objetivos.....	15
2.1. Objetivo general	15
2.2. Objetivos específicos.....	15
Capítulo 3	17
Marco teórico referencial.....	17
3.1. Antecedentes	17
3.2. Marco teórico	18
3.2.1. Referente Histórico.....	18
3.2.2. Marco Conceptual	20
Capítulo 4	25
Diseño metodológico	25
4.1. Dimensiones del Robot Adep Cobra 600	26
4.2. Desarrollo del modelo	27
4.3. Desarrollo de la programación en Processing	35
4.4. Diagrama de clase.	35
4.5. Interfaz gráfica desarrollada.	36
Capítulo 5	38
Parámetros para el trabajo con la interfaz gráfica.....	38
Conclusiones	46
Bibliografía	47
ANEXOS.....	48
Programa desarrollado en la consola de Processing.....	48

Introducción

La vida del hombre sufre profundas transformaciones a causa de los grandes cambios tecnológicos y de investigación que se dan de manera vertiginosa en todos los campos del saber, generando nuevas problemáticas en cuanto a su uso, desarrollo y beneficio. Es así que, la educación no es ajena a dichos transiciones y por el contrario ha tenido que revisar profundamente sus bases conceptuales y metodológicas para así fundamentar las soluciones a los problemas generados por el mismo desarrollo.

Es por esto que, Colombia como país entendió el adelanto tecnológico como base para su crecimiento y progreso económico, fundamentado en la educación como eje principal sobre el cual giran los argumentos de la cultura de la ciencia.

De manera que, en el marco de la Nueva Ley General de Educación y respondiendo a la necesidad de vincular la ciencia y la tecnología, en una forma más coherente, que brinde la posibilidad de un desarrollo científico y socio-cultural, a nuestro sistema educativo, se plantea en este proyecto el desarrollo de una herramienta virtual que de la posibilidad a los estudiantes de educación media, de involucrarse en el estudio de la robótica a través de la manipulación del modelo y a los estudiantes de la universidad de modelar mediante el algoritmo propuesto, sus propias ideas de robot y de mecanismos de eslabones.

En consecuencia, En este proyecto se le da relevancia a la facilidad que se tiene hoy en día de acceder a plataformas virtuales para simular los diferentes modelos de artefactos diseñados por el hombre, en procura de verificar el funcionamiento que ofrezca la certeza de idoneidad de los parámetros técnicos y económicos en un dispositivo, antes de sacarlo al mercado en forma real, con el fin de dar solución a una necesidad del campo de la ingeniería.

Para este proyecto, el modelo virtual se desarrollará tomando como base un robot SCARA (Selective Compliant Assembly Robot Arm o Selective Compliant Articulated Robot Arm) y usando el programa de diseño digital orientado a objetos Processing 2.0, que nos permite representarlos en 2D o 3D y utiliza como base de programación lenguaje JAVA, generándose una interfaz gráfica de usuario (GUI) para su trabajo; gran parte del desarrollo del modelo consiste en realizar el estudio de la cinemática del mismo para entregar los algoritmos de movimiento de acuerdo a lo que el usuario quiera hacer con el robot.

Capítulo 1

Planteamiento del problema y justificación

1.1 Planteamiento del problema

Hoy en día, las aulas virtuales permiten al estudiante realizar prácticas de laboratorio que eliminan los riesgos a la hora de la ejecución de los proyectos, obteniendo excelentes resultados, debido a que se puede controlar todas las variables objeto de estudio; la electrónica y más específicamente la robótica se pueden servir de estas herramientas para desarrollar sus modelos, sin la necesidad de hacer grandes inversiones de dinero antes de obtener la solución definitiva a un problema.

Gracias al desarrollo de las herramientas virtuales, se han generado nuevas formas de afrontar el proceso de enseñanza - aprendizaje en la educación y han surgido nuevas relaciones entre el estudiante, el profesor y la máquina a la hora de abordar los problemas del conocimiento.

Problema

¿Cómo ayudar a los estudiantes de electrónica a desarrollar modelos de robot y mecanismos de eslabones en una aplicación virtual gratuita?, y surgió como respuesta el desarrollo de un modelo virtual tridimensional de robot, aprovechando la plataforma Processing 2.0 y tomando como ejemplo un caso real de un robot industrial, el robot SCARA Omron i600.

1.2. Justificación

El incremento del uso de las TIC en los procesos de formación en todos los niveles de la educación de las personas ha generado la posibilidad de crear numerosas aplicaciones para contribuir al desarrollo del conocimiento científico y tecnológico en el país.

Es por ello que, el presente proyecto aporta con la creación de un modelo de robot virtual, con el cual los estudiantes de electrónica pueden simular sus aplicaciones de robótica y así dar solución a numerosos problemas que se plantean a diario, a nivel de la ciencia y el desarrollo industrial del país, en lo referente a la automatización de los procesos y al diseño e implementación de máquinas herramientas que simplifican las tareas del hombre bajo parámetros de seguridad y eficiencia.

Contar con este modelo de robot virtual permitirá a los estudiantes avanzar en el desarrollo de sus aplicaciones, debido a que el prototipo puede ser manejado de manera sencilla, mediante la interfaz gráfica de usuario y mediante el cambio de parámetros en el lenguaje de programación Java (estudiantes de electrónica), teniendo como base los algoritmos y ecuaciones de la cinemática directa e inversa, facilitándoles la simulación y evitando que se generen gastos innecesarios al realizar modelos físicos de robots; además que componente virtual permite un mayor número de usuarios del modelo, contrario a lo que sucede cuando se tiene robots físicos de alto costo.

Por otro lado, los docentes contarán con el beneficio de una nueva herramienta, que les permitirá el desarrollo de sus clases con la ayuda del modelo virtual, para hacer más eficiente y dinámico sus procesos de enseñanza-aprendizaje y así alcanzar los objetivos propuestos en las diferentes asignaturas.

Capítulo 2

Objetivos

2.1. Objetivo general

Desarrollar un prototipo virtual tridimensional de un robot SCARA industrial que incluya la implementación del modelado matemático de la cinemática directa e inversa, y una interfaz tridimensional interactiva, para la futura enseñanza e implementación de simulaciones 3D de robótica y mecanismos de eslabones a estudiantes de pregrado en áreas de electrónica, control y robótica.

2.2. Objetivos específicos

- Desarrollar el modelo matemático de la cinemática directa e inversa del robot SCARA industrial adept Cobra i600, mediante el uso de los métodos de modelado matrices de transformación y método geométrico.
- Implementar el modelado matemático de la cinemática directa e inversa del robot SCARA mediante algoritmos computacionales orientados a objetos, bajo el lenguaje de programación Java.
- Desarrollar una interfaz gráfica de usuario tridimensional interactiva para el robot SCARA modelado, mediante el uso de la plataforma Processing 2.0, basada en el lenguaje Java.
- Integrar la interfaz gráfica con el modelado matemático de la cinemática directa e inversa del robot SCARA, para la construcción del modelo base de simulación de robots 3D.

- Generar la documentación del proyecto, para futuros desarrollos por parte de estudiantes de ingeniería, mediante un documento de Tesis de grado.

Capítulo 3

Marco teórico referencial

3.1. Antecedentes

A nivel internacional algunas de las investigaciones en las que se puede sustentar son las siguientes:

Diseño, modelamiento y simulación 3D de un robot móvil para exploración de terrenos. (Bogotá D.C, marzo 12 de 2009) T44.09 P415d.

En este proyecto se propone el diseño estático y dinámico de un robot móvil para exploración y monitoreo de terrenos, así como la simulación 3D de su comportamiento dentro de un espacio de trabajo virtual. Se emplea diferente software como Solid Edge para el modelado mecánico, Algor FEA para el análisis estático de la estructura mecánica y Marilou Robotics Studio 2008 para el modelado, la simulación y la visualización en 3D del comportamiento del robot en tiempo real. (Pérez, 2009, p.17)

Brazo robótico de un grado de libertad modelado como péndulo invertido y operado por computadora. (Bogotá D.C., agosto de 2014) TE-17234.

El problema propuesto consiste en el diseño e implementación de un sistema de control para un brazo robótico de una sola articulación, modelado como péndulo invertido, mediante la comunicación entre el computador y el sistema físico a través de una tarjeta de adquisición de datos desde una interfaz en Matlab ®, como herramienta de apoyo para la enseñanza en el área de control del programa de Licenciatura en Electrónica de la Universidad Pedagógica Nacional. (Ardila, 2014, p.16)

Sistema de control para un robot self-balancing. (Bogotá D.C., 2013) TE – 16784

Este proyecto brinda herramientas que permiten ampliar los conocimientos sobre dicho tema, para la elaboración de diseños que hagan uso de sistemas de control difuso en modelos no lineales. Dichas herramientas no son solo datos teóricos y gráficos sino también un hardware que permite constatar los análisis previamente realizados. Esto ayuda desde la didáctica de la pedagogía a tener otros puntos de vista y análisis sobre un determinado tema, puesto que ya no se iría de lo teórico a lo práctico, sino que se analizaría un sistema real. En la educación en tecnología y electrónica el hecho de poder verificar los análisis teóricos en cosas reales hace que el aprendizaje sea significativo. (Muñoz, 2013, p.10)

Prototipo de robot submarino con la capacidad de seguimiento de trayectorias mediante tratamiento de imágenes. (Bogotá D.C., 2014) TE – 17007

El prototipo que se presenta en este trabajo de grado, permitirá al docente en formación de la Licenciatura en Electrónica un acercamiento directo al trabajo que se puede realizar con robots acuáticos; sus diferentes aplicaciones, manipulación y posterior recolección de datos. Esto se realiza con el fin de conocer las ventajas y desventajas del uso de robots en este medio en específico. Estos desempeños son los que nos conducen al desarrollo de artefactos, que permiten ser empleados como herramientas para impartir un conocimiento más acorde a las necesidades actuales en la Licenciatura en Electrónica. (Rodríguez, 2014, p.15-16)

3.2. Marco teórico

3.2.1. Referente Histórico

En casi todas las épocas y culturas, los hombres han intentado construir máquinas automáticas que faciliten su trabajo, hagan más cómoda su existencia, satisfagan su curiosidad y su afán de aprender e investigar, o simplemente les sirvan de entretenimiento.

Los primeros prototipos de robots datan de la antigua Grecia donde construyeron modelos que trataban de imitar movimientos humanos, facilitando sus labores diarias, a estos los llamaron autómatas; en la Edad Media y en el Renacimiento se desarrollan los mecanismos como los usados en los relojes, que son los primeros modelos de máquinas complejas que cumplen una función específica.

Durante los siglos XVII y XVIII se crearon ingenios mecánicos de mayor complejidad que tenían alguna de las características de los robots actuales; así por ejemplo Jacques de Vaucanson (1709-1782) construyó varios autómatas, uno de los más conocidos es un pato mecánico, que bebe, come, grazna, chapotea en el agua y digiere su comida (como un pato verdadero). Estos primeros autómatas estaban destinados fundamentalmente a ser exhibidos en las ferias y servir de entretenimiento en las Cortes y entre la nobleza. (Torres, 2015, Recuperado de <http://repositorio.upct.es>)

Debido al desarrollo industrial de la época se vio la necesidad de construir máquinas que realizaran las labores humanas con mayor eficiencia y eficacia para aumentar los niveles de productividad, bajando los costos, por esta razón el desarrollo robótico comenzó por el desarrollo de las máquinas industriales. Bajo esta creciente demanda se empezaron a desarrollar máquinas de vapor que funcionaban con reguladores los cuales dieron origen al concepto de realimentación y la regulación automática

La palabra robot se empleó por primera vez en 1920 en una obra de teatro llamada "Robots Universales Rossum" escrita por el dramaturgo checo Karel Capek, se deriva de la palabra checa "robotnik" y significa, siervo, servidor o trabajador forzado. Es en el

siglo XX cuando se empieza a hablar de robots y se produce el desarrollo de estos, que va ligado con el desarrollo de los microprocesadores. (Torres, 2016, Recuperado de <http://repositorio.upct.es>)

Durante varios años, se sientan las bases de la robótica industrial, posteriormente irá aumentando el empleo y la sofisticación de los robots con el aumento de las prestaciones de los microprocesadores y las posibilidades de la informática.

El auge de los robots, además de al desarrollo de la electrónica, se debe a la necesidad industrial de fabricar productos con variaciones en función de los gustos y necesidades de los clientes, lo cual ha hecho que las máquinas y dispositivos automáticos de fabricación específicos para desarrollar un producto único, solo sean rentables para grandes series; con la robótica y la automatización flexible, se consiguen fabricar distintos productos de una misma familia, con pocos o ningún cambio estructural en las líneas de producción, pues estos sistemas se adaptan por programa a las condiciones variables de fabricación.

3.2.2. Marco Conceptual

3.2.2.1. Robot industrial

Para la Federación Internacional de Robótica (IFR):

Por robot industrial de manipulación se entiende a una máquina de manipulación automática, reprogramable y multifuncional con tres o más ejes que pueden posicionar y orientar materias, piezas, herramientas o dispositivos especiales para la ejecución de

trabajos diversos en las diferentes etapas de la producción industrial, ya sea en una posición fija o en movimiento. (IFR, 2016, Recuperado de <https://ifr.org/img/oficce>)

La definición de robot industrial de la I.S.O., proviene de la "Robotic Industries Association (RIA) y es: (ISO, 2016, Recuperado de <https://ifr.org/obp/>)“Manipulador multifuncional reprogramable con varios grados de libertad, capaz de manipular materias, piezas, herramientas o dispositivos especiales según trayectorias variables programadas para realizar tareas diversas”.

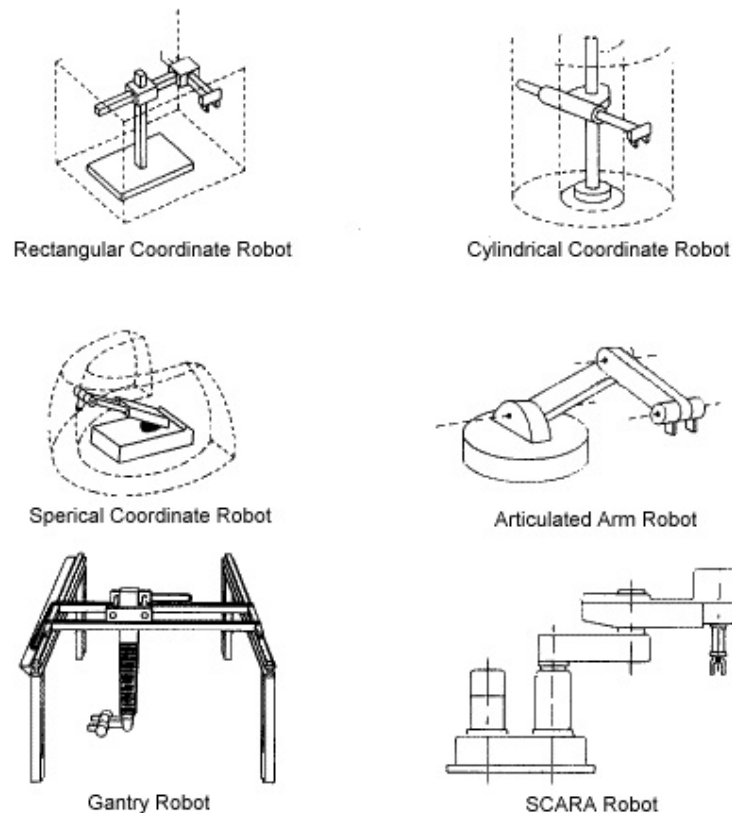


Figura 3.1: Ejemplos de robots contruidos con articulaciones de revolución y prismáticas [5]

Para la Asociación Japonesa de Robótica Industrial (JIRA): Los robots son

Dispositivos capaces de moverse de modo flexible análogo al que poseen los organismos vivos, con o sin funciones intelectuales, permitiendo operaciones en respuesta a las órdenes humanas. Esta definición es algo más ambigua que las anteriores; en Japón el concepto de robot es menos restrictivo que en otros países. (J.I.RA, 1974, Recuperado de <https://doi.org/10.1108>)

3.2.2.2. Robot SCARA. (Selective Compliance Arm for Robotic Assambly)

Es un robot con dos articulaciones R y una P, con las dos R se controla la posición respecto al plano X-Y y con la P la coordenada Z. Es rápido, barato y preciso, pero solo tiene accesibilidad a zonas de trabajo que estén en planos perpendiculares a su eje vertical. Se emplea fundamentalmente en operaciones de ensamblado o inserción de componentes electrónicos y en otros trabajos similares.



Figura 3.2: Ejemplos de robots SCARA siendo utilizados en la industria moderna. (Izquierda) Tarea de paletizado; (Derecha) Tarea de reposicionado.

Es originario de Japón y es allí donde más se emplea, su inconveniente inicial era la potencia de cálculo necesaria para determinar posiciones por combinación de giros,

pero este problema se ha resuelto para este y otros robots, gracias al desarrollo de los microprocesadores. (Craig, 2016, p.406)

3.2.2.3. Modelo virtual

Es una representación en dos o tres dimensiones de un objeto o sistema, previo desarrollo de las funciones matemáticas que sustentan el modelo de acuerdo a cómo se comporta o como se ve en el mundo real, mediante un software especializado.

3.2.2.4. Interfaz Gráfica de usuario (GUI – Graphic User Interface)

Conjunto de formas y métodos que posibilitan la interacción de un sistema con los usuarios utilizando formas gráficas e imágenes, como botones, iconos, ventanas, fuentes, etc., los cuales representan funciones, acciones e información.

3.2.2.5. Java

Java es un lenguaje de desarrollo de propósito general, y como tal es válido para realizar todo tipo de aplicaciones profesionales. Incluye una combinación de características que lo hacen único y está siendo adoptado por multitud de fabricantes como herramienta básica para el desarrollo de aplicaciones comerciales de gran repercusión.

Una de las características más importantes es que los programas “ejecutables”, creados por el compilador de Java, son independientes de la arquitectura. Se ejecutan indistintamente en una gran variedad de equipos con diferentes microprocesadores y sistemas operativos. Es intrínsecamente orientado a objetos. (Sun, 2017, Recuperado de <https://www.wmc.geofisica.unam.mx/>)

3.2.2.6. Processing 2.0

Es un software flexible basado en java, para aprender a codificar dentro de las artes visuales, ideal para realizar modelos en 2D, 3D o PDF en aplicaciones artísticas, de diseño o de investigación de prototipos en tecnología; es sencillo de manejar, de descarga libre y código abierto, fácil de actualizar, ya que cuenta con soporte en línea desde la página oficial, que lo documenta con más de 100 bibliotecas que extienden las aplicaciones del software. Además trabaja en plataformas GNU / Linux, Mac OS X, Windows, Android y ARM. Posee en su documentación gran cantidad de ejemplos desarrollados por colaboradores del software que permiten evidenciar claramente las diferentes funciones, atributos, comandos y aplicaciones. (Processing.org, 2017, en línea)

Dentro de las librerías del software se encuentra **Proscene**; una biblioteca que facilita la creación de escenas interactivas.

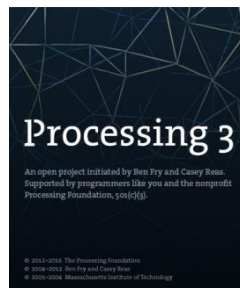


Figura 3.3: Inicio de la plataforma Processing [8]

Capítulo 4

Diseño metodológico

El trabajo desarrollado presenta en este documento el análisis del robot SCARA industrial referencia Adept Cobra 600, cuya demanda en el mercado es muy alta debido a su gran capacidad de trabajo y rapidez en las funciones móviles, que constan de seis ejes, y ofrece gran versatilidad de uso en los procesos finales de manufactura por su velocidad y precisión; dicho análisis da como resultado el desarrollo de una interfaz gráfica de usuario, para fines educativos, en la cual podemos parametrizar algunos movimientos del robot y definir sus trayectorias utilizando conceptos de cinemática directa e inversa.



Figura 4.1: Robot Adept Cobra 600 [9]

Respecto a la cinemática directa podemos decir que es determinar la posición final del actuador (extremo final del robot) una vez se han establecido los parámetros de rotación y traslación de las articulaciones, referenciados a un sistema de coordenadas; en este caso se utilizó para el desarrollo y solución de la cinemática directa las matrices de transformación

homogéneas y el método de la representación de los parámetros de Denavit – Hartenberg, que no solo nos permiten conocer la posición final del actuador, sino también la posición de cada una de las articulaciones del robot; también se manejan en esta cálculo los métodos numéricos.

4.1. Dimensiones del Robot Adep Cobra 600

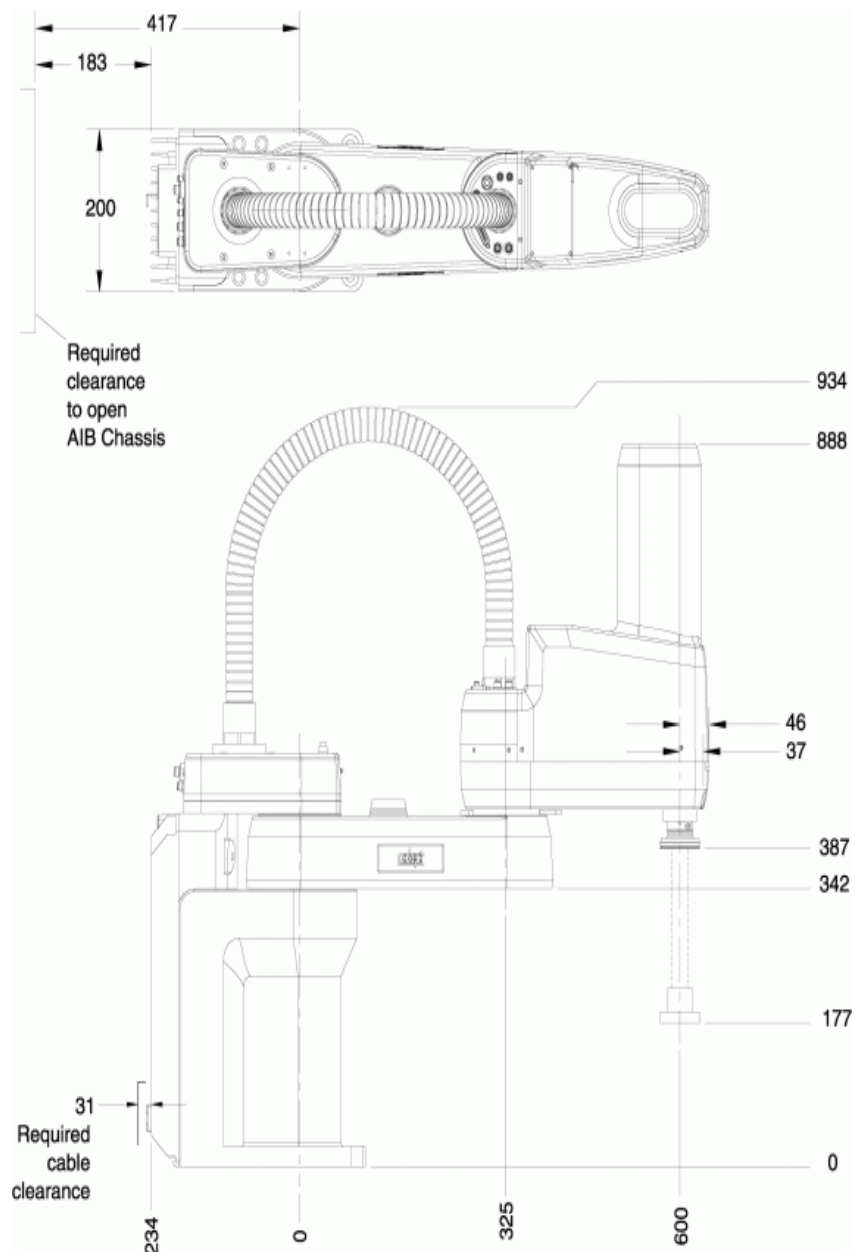


Figura 4.2: Vista superior y lateral del Robot [9]

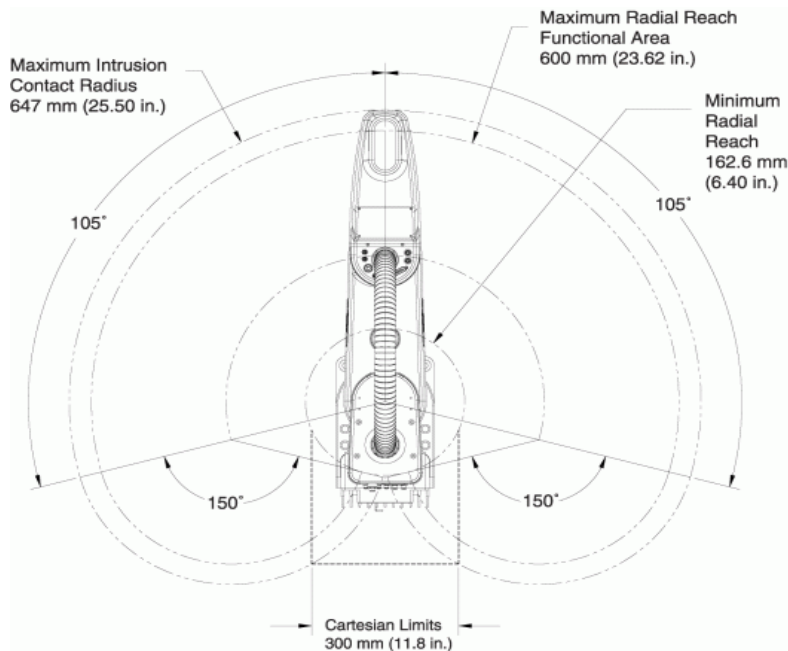


Figura 4.3: Ángulos máximos de desplazamiento del Robot [8]

4.2. Desarrollo del modelo

En primer lugar se establecieron los sistemas de referencia como se muestra en la gráfica 4.4, y luego se procedió a tabular los parámetros Denavit – Hartenberg para desarrollar las ecuaciones y matrices de transformación.

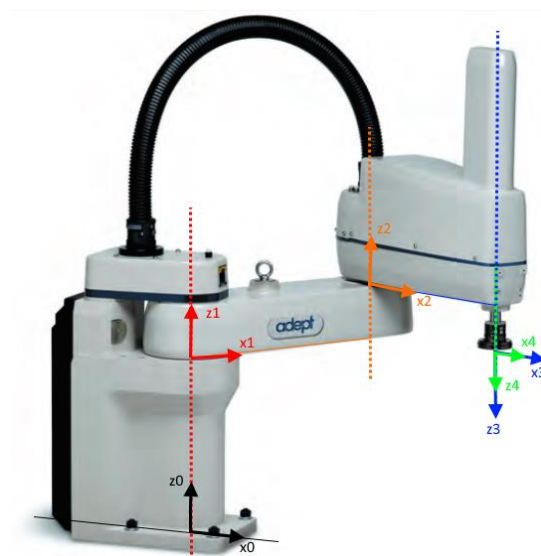


Figura 4.4: Sistemas de referencia del sistema [8]

Vínculos	a i-1	α i-1	d i	θ i
0	0	0	0	0
1	0	0	0,342	θ_1
2	0,325	0	0,0975	θ_2
3	0,275	180	0.52+d3	0
4	0	0	0	θ_4

Figura 4.5: parámetros Denavit - Hartenberg

Denavit-Hartenberg propusieron en 1955 un método matricial que permite establecer de manera sistemática un sistema de coordenadas (S_i) ligado a cada enlace i de una cadena articulada, determinando las ecuaciones cinemáticas de la cadena completa. (Ramírez, 2017, Recuperado de <https://www.kramirez.net/>)

Una vez que se han definido los parámetros de acuerdo al sistema de referencia del robot, se hace el modelo matemático utilizando las matrices de transformación homogénea, en la cual un punto en el espacio se define como un vector de la forma:

$${}^A\mathbf{R} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix}$$

Y la orientación de un cuerpo se representa con una matriz de la forma:

$${}^A_B\mathbf{R} = \begin{bmatrix} {}^A\hat{X}_B & {}^A\hat{Y}_B & {}^A\hat{Z}_B \end{bmatrix} = \begin{bmatrix} X_B \cdot X_A & Y_B \cdot X_A & Z_B \cdot X_A \\ X_B \cdot Y_A & Y_B \cdot Y_A & Z_B \cdot Y_A \\ X_B \cdot Z_A & Y_B \cdot Z_A & Z_B \cdot Z_A \end{bmatrix}$$

Además definimos los siguientes modelos que nos sirven para trabajar las matrices homogéneas:

Trama

$$\{B\} = \{ {}^A_R, {}^A P_{BORG} \}$$

Operador rotacional

$$\{B\} = \{ {}^A_R, {}^A P_{BORG} \}$$

Operador de transformación

$${}^A P_2 = T {}^A A_1$$

Matriz de rotación

$${}^A_R = \begin{bmatrix} {}^A X_B & {}^A Y_B & {}^A Z_B \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

Tomamos como ejemplo rotaciones de 30°, con lo cual se establecen las siguientes relaciones y los parámetros que se asocian para completar la matriz de 4x4:

R_Z(30°)

	X	Y	Z
X	CΘ	-SΘ	0
Y	SΘ	CΘ	0
Z	0	0	1

$R_Y(30^\circ)$

	X	Y	Z
X	$C\theta$	0	$S\theta$
Y	0	1	0
Z	$-S\theta$	0	$C\theta$

$R_X(30^\circ)$

Traslación

Escalado Global

Perspectiva robótica

	X	Y	Z	
X	1	0	0	1
Y	0	$C\theta$	$-S\theta$	2
Z	0	$S\theta$	$C\theta$	3
	0	0	0	1

Una vez se han establecido estas condiciones de trabajo general y tomando los parámetros Denavit – Hartenberg, se procede a desarrollar las matrices de transformación homogénea de manera consecutiva entre eslabones para obtener la posición final del actuador o último eslabón respecto de la base del robot.

Matrices base:

$${}^0_1T = \begin{bmatrix} C\theta_1 & -S\theta_1 & 0 & 0 \\ S\theta_1 & C\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

$${}^1_2T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -d_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

$${}^0_3T = {}^0_1T {}^1_2T {}^2_3T \quad (3)$$

$${}^1_3T = \begin{bmatrix} C\theta_3 & -S\theta_3 & 0 & 0 \\ 0 & 0 & -1 & -L_2 - d_2 \\ S\theta_3 & C\theta_3 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

$${}^2_3T = \begin{bmatrix} C\theta_3 & -S\theta_3 & 0 & 0 \\ S\theta_3 & C\theta_3 & 0 & 0 \\ 0 & 0 & 1 & L_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5)$$

$${}^0_4T = \begin{bmatrix} C\theta_1 C\theta_3 & -C\theta_1 S\theta_3 & S\theta_1 & S\theta_1(L_2 + d_2) \\ S\theta_1 C\theta_3 & S\theta_1 S\theta_3 & -C\theta_1 & C\theta_1(-L_2 - d_2) \\ S\theta_3 & C\theta_3 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6)$$

Desarrollo de las matrices con los datos numéricos de la tabla Denavit – Hartenberg

$${}^0_4T = {}^0_1T {}^1_2T {}^2_3T {}^3_4T \quad (1)$$

$${}^0_1T = \begin{bmatrix} C\theta_1 & -S\theta_1 & 0 & 0 \\ S\theta_1 & C\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 3.42 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

$${}^0_2T = \begin{bmatrix} C\theta_2 & -S\theta_2 & 0 & 0.325 \\ S\theta_2 & C\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0.0975 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

$${}^2_3T = \begin{bmatrix} 1 & 0 & 0 & 0.275 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & -0.52 - d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

$${}^3_4T = \begin{bmatrix} C\theta_4 & -S\theta_4 & 0 & 0 \\ S\theta_4 & C\theta_4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5)$$

$${}^0_4T = \begin{bmatrix} C\theta_1 C\theta_2 C\theta_4 + S\theta_1 S\theta_2 C\theta_4 - S\theta_1 S\theta_2 C\theta_4 + S\theta_1 S\theta_2 C\theta_4 & -C\theta_1 C\theta_2 S\theta_4 + C\theta_1 S\theta_2 S\theta_4 + S\theta_1 C\theta_2 S\theta_4 + S\theta_1 S\theta_2 S\theta_4 & 0.275 C\theta_1 C\theta_2 + 0.325 C\theta_1 - 0.275 S\theta_1 S\theta_2 & 0 \\ S\theta_1 C\theta_2 C\theta_4 + S\theta_1 S\theta_2 S\theta_4 + S\theta_1 C\theta_2 C\theta_4 - S\theta_1 C\theta_2 C\theta_4 & -C\theta_1 S\theta_2 S\theta_4 + S\theta_1 S\theta_2 C\theta_4 - C\theta_1 S\theta_2 S\theta_4 - C\theta_1 C\theta_2 C\theta_4 & 0 & 0.275 C\theta_1 S\theta_2 + 0.325 S\theta_1 + 0.275 S\theta_1 C\theta_2 \\ 0 & 0 & -1 & 0.342 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Ecuación final

En la cinemática inversa podemos establecer a partir de la posición final del actuador, las diferentes configuraciones de las articulaciones y las referencias a los sistemas coordenados.

Luego de tener las ecuaciones y las matrices de transformación se procede a generar el algoritmo que permite determinar los movimientos del robot SCARA.

En cuanto a la parte de diseño del robot, se hizo una adaptación de los modelos CAD de ADEPT: se convirtieron las partes en formato STEP a modelos de superficie utilizando el

software FreeCAD, luego se agregó color a las partes en formato OBJ desde el software 3D Builder agregándoles la biblioteca de materiales mtl.

Se almacenaron todos los archivos OBJ con su mtl en la carpeta Data y desde allí se cargan a Processing, cada objeto y eslabón con la clase Pshape.



Figura 4.6: Parte 1 del Robot Scara en formato OBJ

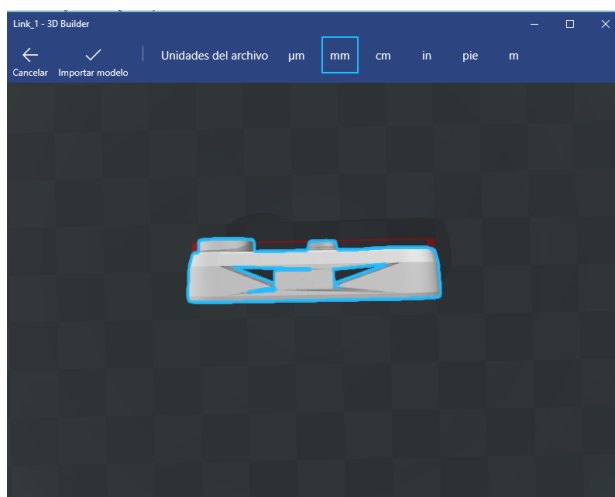


Figura 4.7: Parte 2 del Robot Scara en formato OBJ

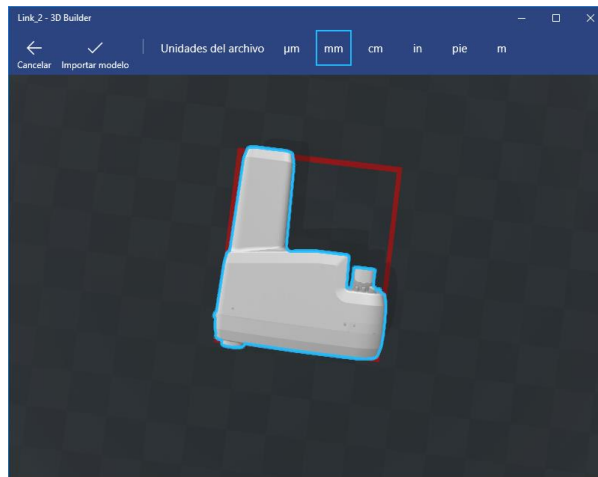


Figura 4.8: Parte 3 del Robot Scara en formato OBJ

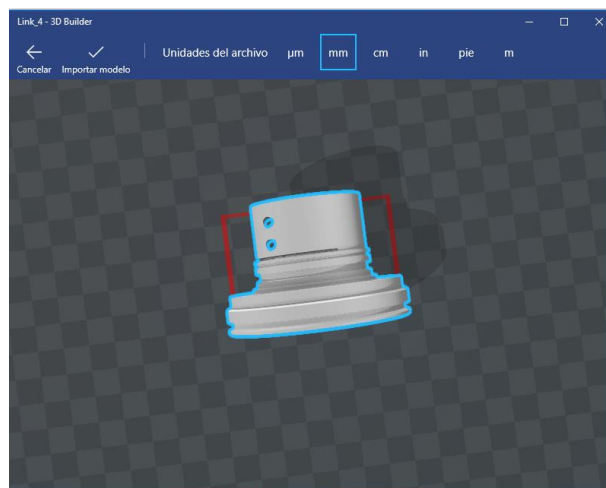


Figura 4.9: Parte 4 del Robot Scara en formato OBJ

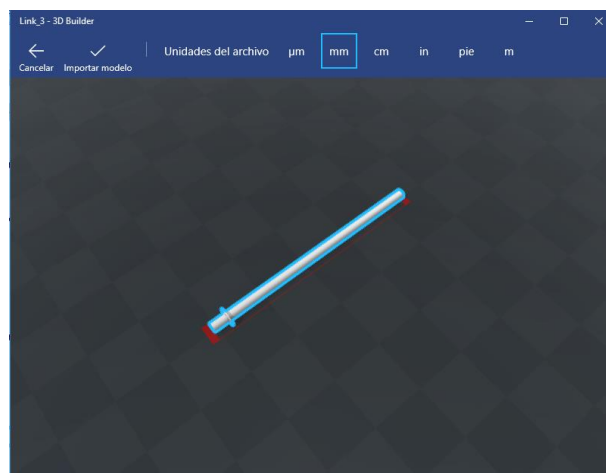


Figura 4.10: Parte 5 del Robot Scara en formato OBJ

Mediante el software Processing se genera la imagen 3D del robot y la interfaz para que los estudiantes puedan manipular los valores de las ecuaciones y puedan ver el movimiento del robot.

4.3. Desarrollo de la programación en Processing

El programa presenta la ventana inicial, donde se hace toda la codificación respecto de la interfaz gráfica. Luego del inicio del programa aparece la ventana de programación.

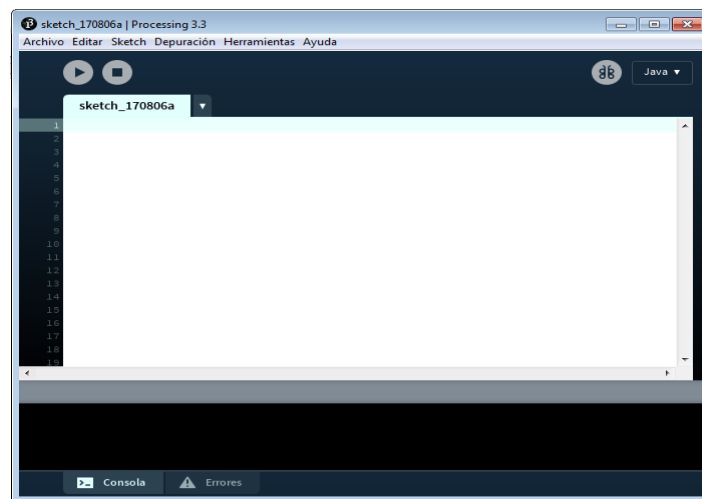
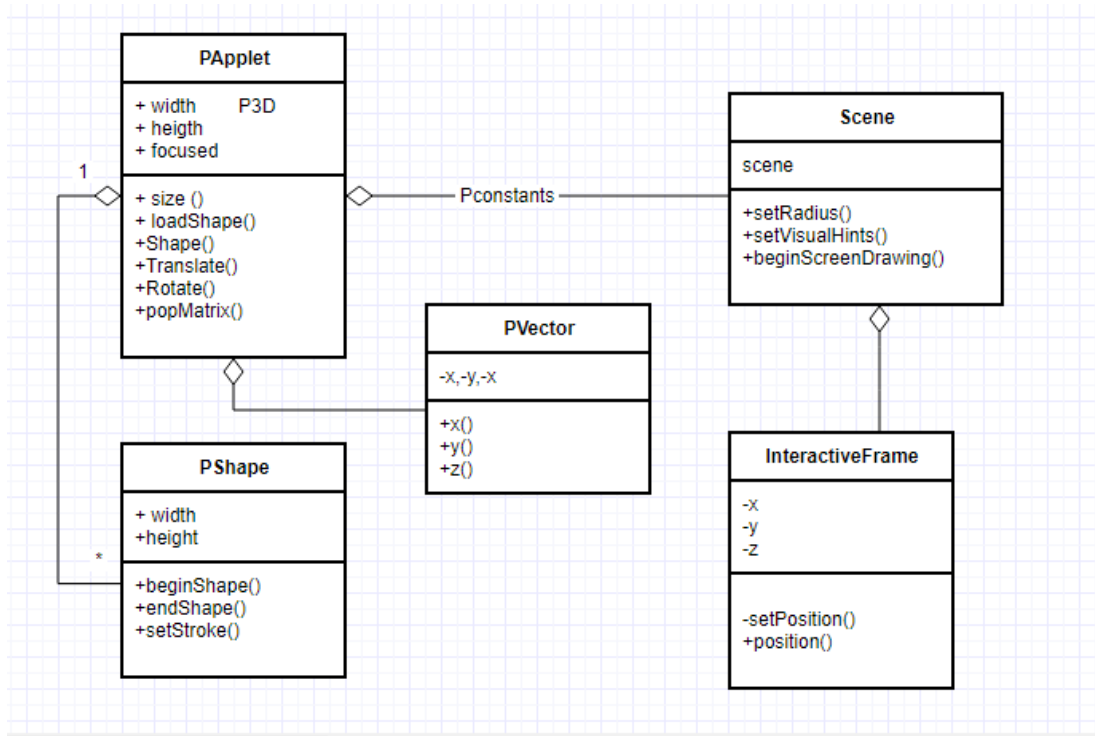


Figura 4.11: Ventana de programación Processing. [8]

4.4. Diagrama de clase.

Dentro del desarrollo del código de Processing se ha establecido el siguiente diagrama de clase, haciendo referencia a la programación orientada a objetos.



4.5. Interfaz gráfica desarrollada.

Una vez se ha desarrollado el código se procede a correr el programa en la ventana de Proscene, donde aparece la Interfaz Gráfica que se muestra a continuación.

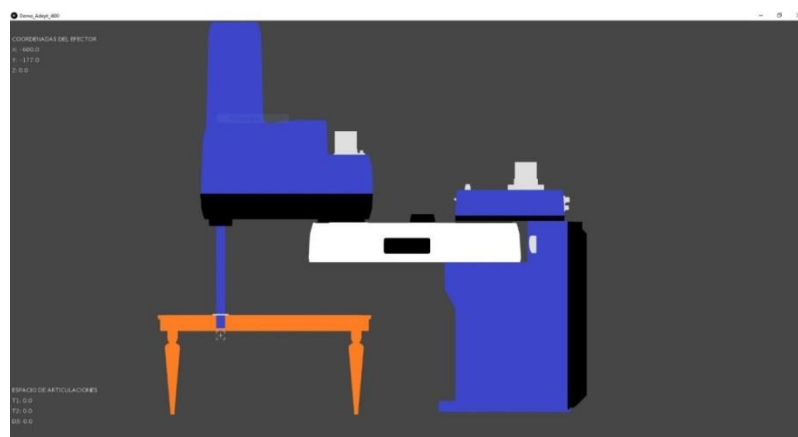


Figura 4.12: Interfaz Gráfica 1 en Proscene (Processing).

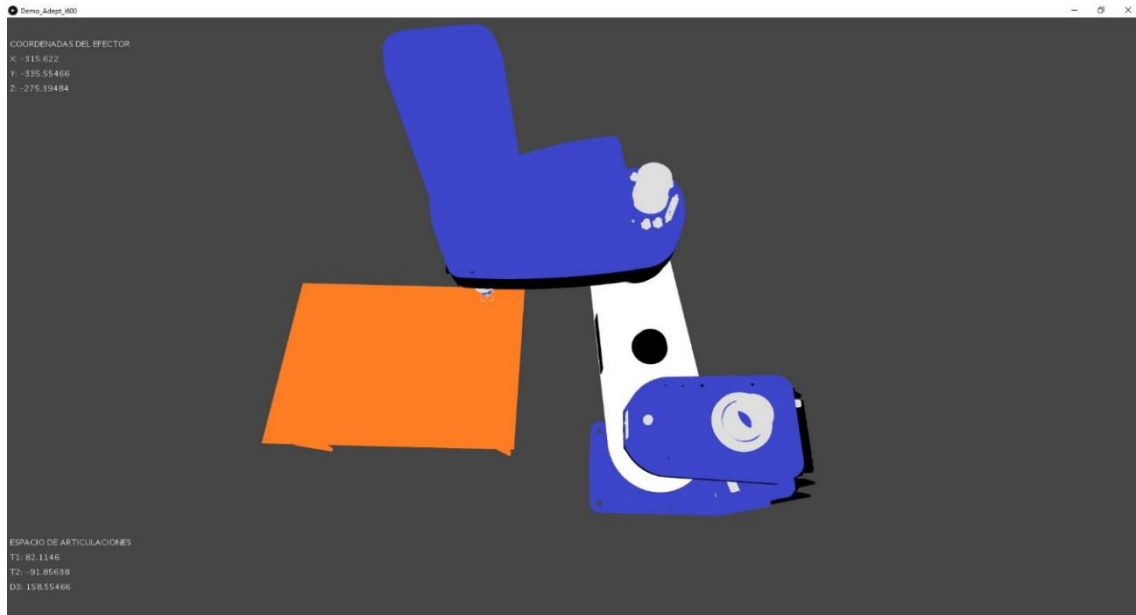


Figura 4.13: Interfaz Gráfica 2 en Proscene (Processing).

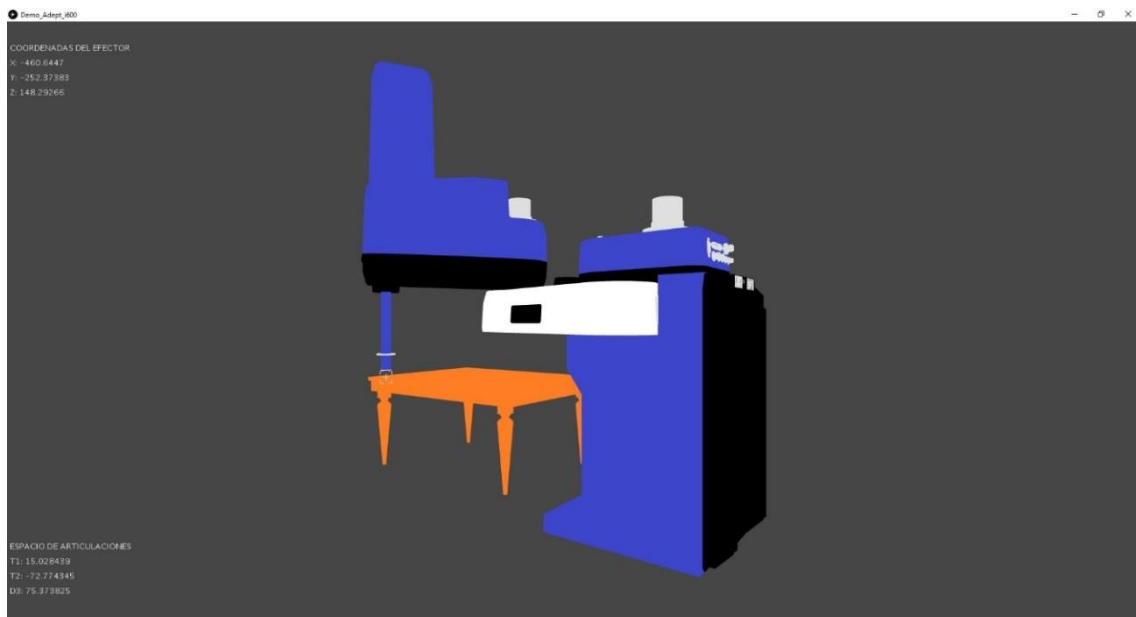


Figura 4.14: Interfaz Gráfica 3 en Proscene (Processing).

Capítulo 5

Parámetros para el trabajo con la interfaz gráfica

Para abrir la interfaz grafica se inicia el programa en la consola de processing y se observara la siguiente imagen.

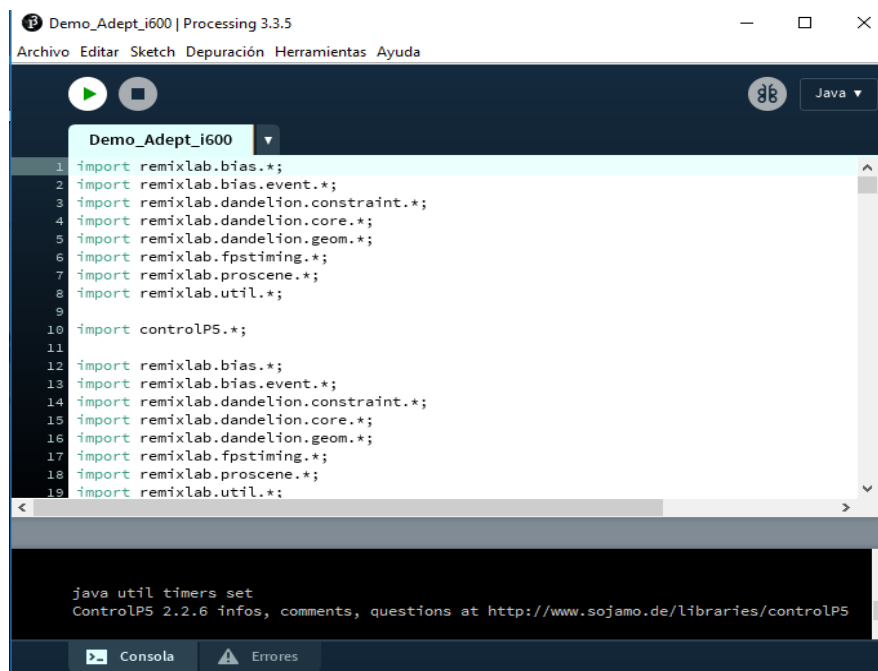


Figura 5.1: Pantalla de inicio con el programa.

Se ejecuta el programa dando click en el botón circular blanco que aparece en la parte superior izquierda con un triángulo en color verde en el centro y aparece la siguiente pantalla de la interfaz:

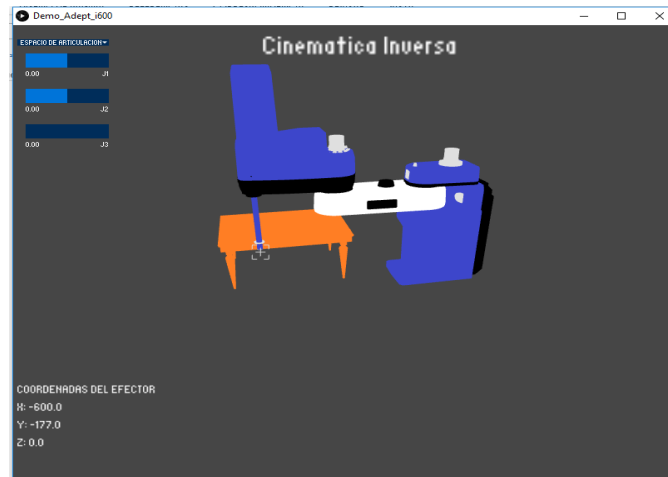


Figura 5.2: Inicio de la interfaz.

Para realizar el trabajo de manipulación del robot en la plataforma se usaran los botones del mouse y algunas letras del teclado. Con el botón scroll del mouse del PC, se puede alejar o acercar el robot dentro de la ventana de la interfaz para observar en mayor detalle la posición de las articulaciones.

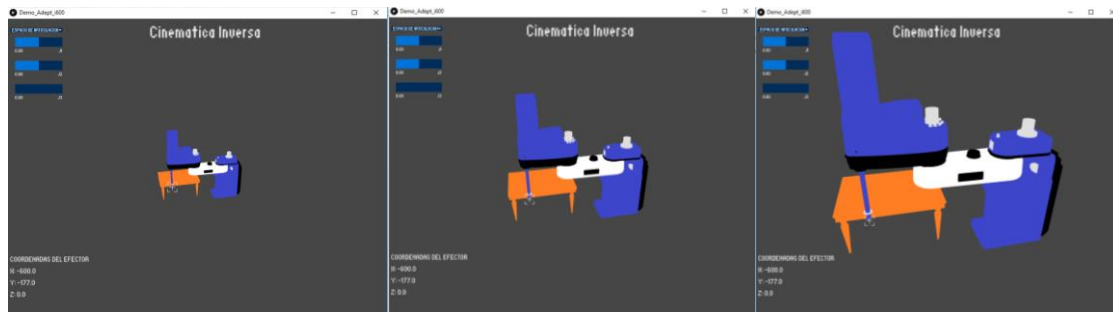


Figura 5.3: Secuencia de zoom.

Con el click izquierdo del mouse se rota el robot dentro de la interfaz lo cual permite tener en detalle todas las vistas para el análisis de cada uno de los movimientos establecidos.

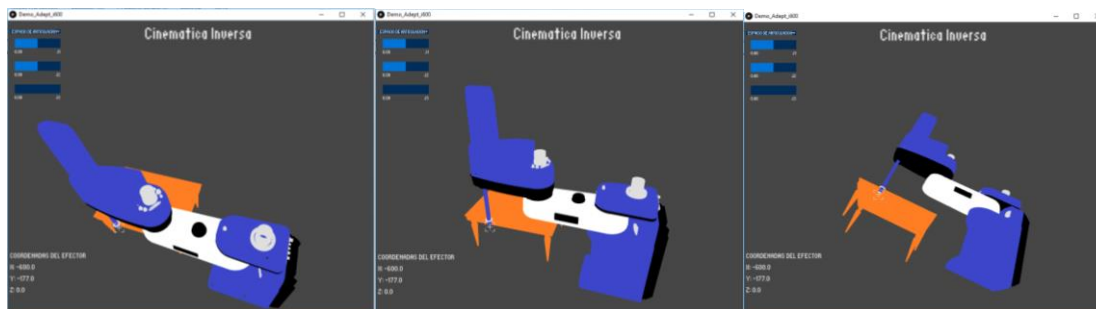


Figura 5.4: Secuencia de rotación.

Con el click derecho del mouse el robot se puede ubicar en diferentes posiciones de la pantalla de la interfaz, de acuerdo con las necesidades del usuario.

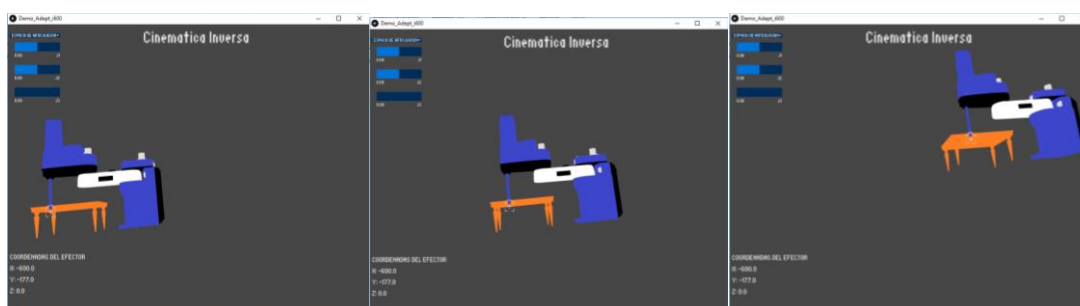


Figura 5.5: Secuencia de ubicación.

Con la tecla “e” el robot pasa a vista en perspectiva.

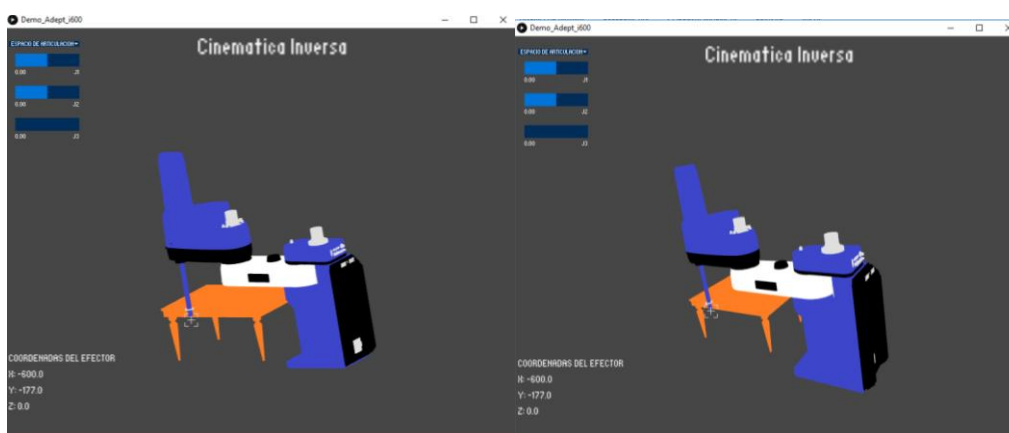


Figura 5.6: Cambio a vista en perspectiva.

Al oprimir la tecla “n” el robot, desde cualquier posición que se encuentre, queda en vista de planta

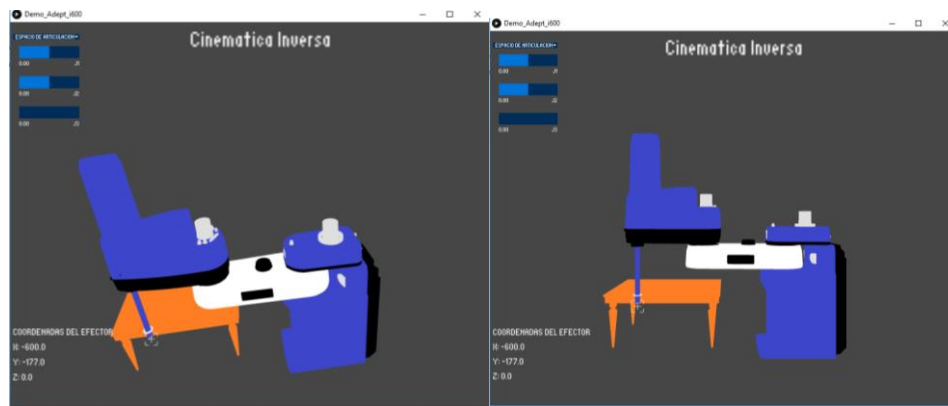


Figura 5.6: Cambio a vista de planta.

Al oprimir la tecla “g” aparece en la interfaz la grilla de puntos de referencia del robot en el espacio.

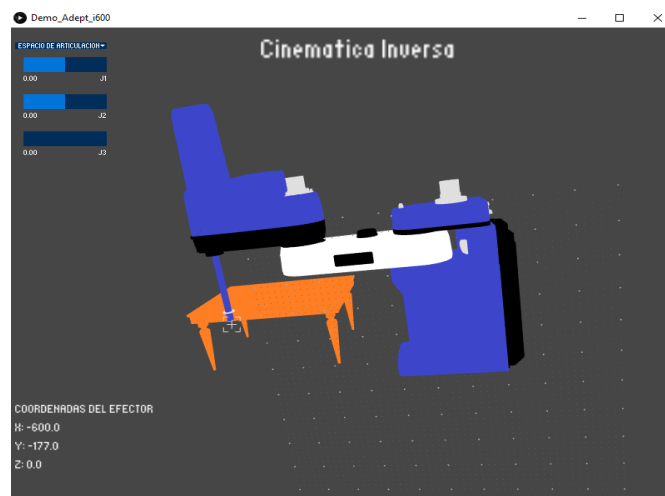


Figura 5.7: Grilla de referencia en el espacio.

Al oprimir la tecla “a” aparece en la interfaz los ejes x , y , z , de referencia del robot en el espacio que nos permiten observar permanentemente sobre que plano se está desplazando el robot, respecto a la base como punto inicial de la cadena de movimientos.

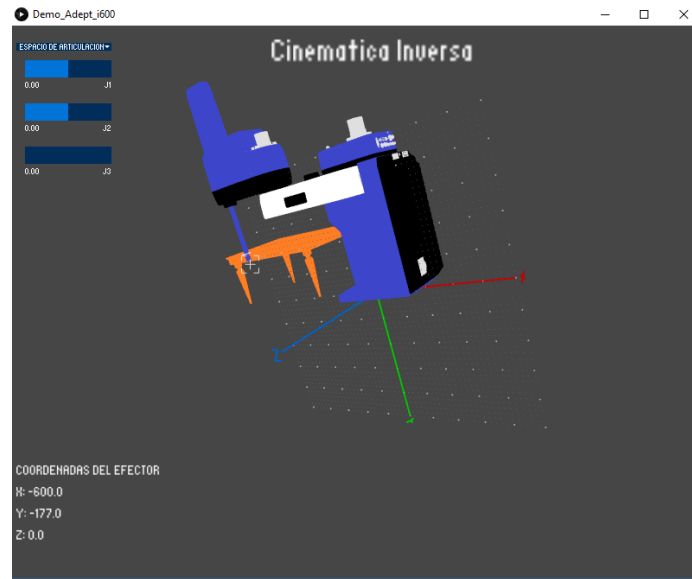


Figura 5.7: Robot con ejes de referencia.

Al oprimir la barra espaciadora se cambia la interfaz de cinemática directa a cinemática inversa, de acuerdo a lo que el usuario necesite trabajar.

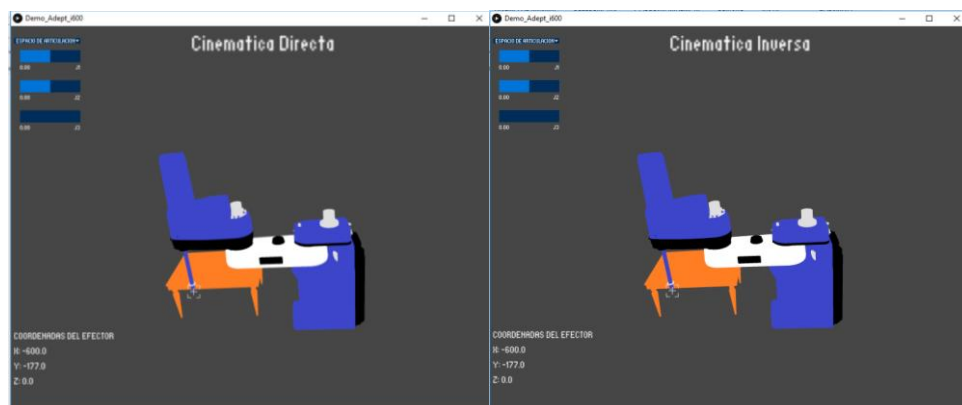


Figura 5.8: Cambio de cinemática directa a inversa

En modo Cinemática Directa se da click en cada una de las barras que indican la posición de las articulaciones y el robot se desplaza en sus niveles máximos de rotación y desplazamiento como se muestra en las siguientes acciones:

Posicion inicial de las dos articulaciones rotacionales y de la prismática que se encuentra en su punto mas bajo respecto a su plano de trabajo. Las barras se desplazan a izquierda y derecha con el click izquierdo del mouse como se muestra en las figuras 5.9 y 5.10.

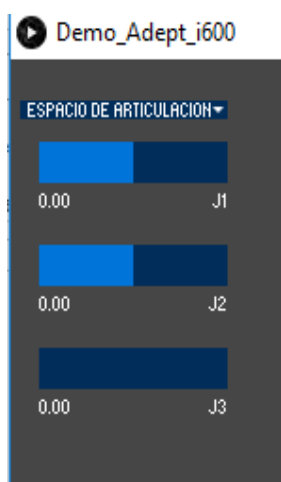


Figura 5.9: Barra de articulaciones en detalle.

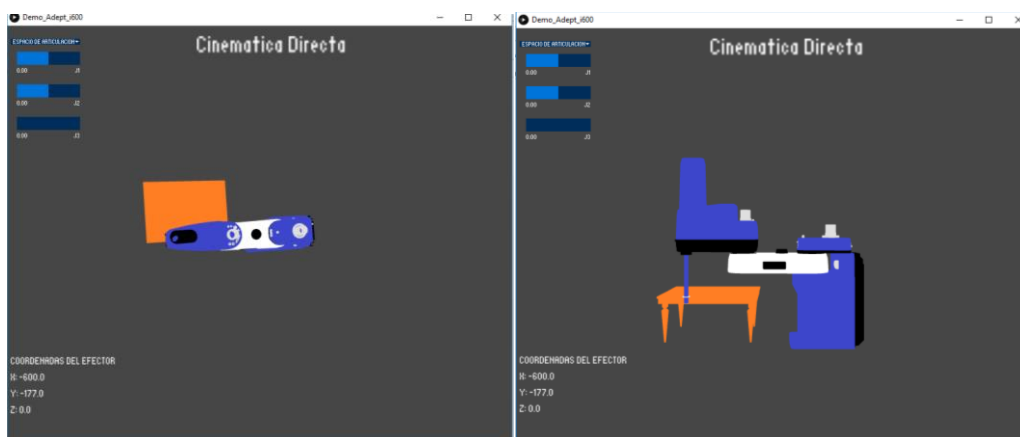


Figura 5.10: Posiciones cero del robot.

La rotación de la articulación 1 varía desde -105° a 105° , la rotación de la articulación 2 varía desde -150° a 150° y el actuador se desplaza verticalmente desde cero (0), que es su posición

más baja, hasta doscientos (200), que es su posición más alta en el eje Y. En todos los casos se observa en la parte inferior derecha las coordenadas finales del actuador.

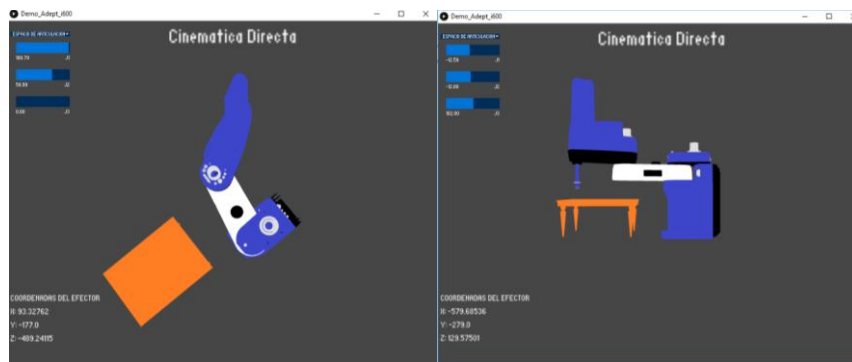
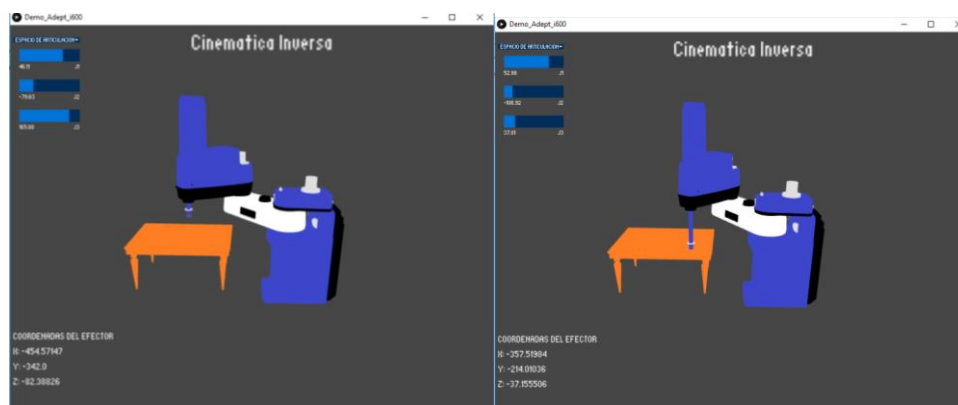


Figura 5.11: Movimientos en cinemática directa.

En modo Cinemática Inversa se toma con el click derecho del mouse el punto final del actuador y a medida que se desplaza por la pantalla automáticamente las posiciones de las articulaciones varían en la ventana de barras superior izquierda y en la parte inferior izquierda también se va ubicando la posición final del actuador.



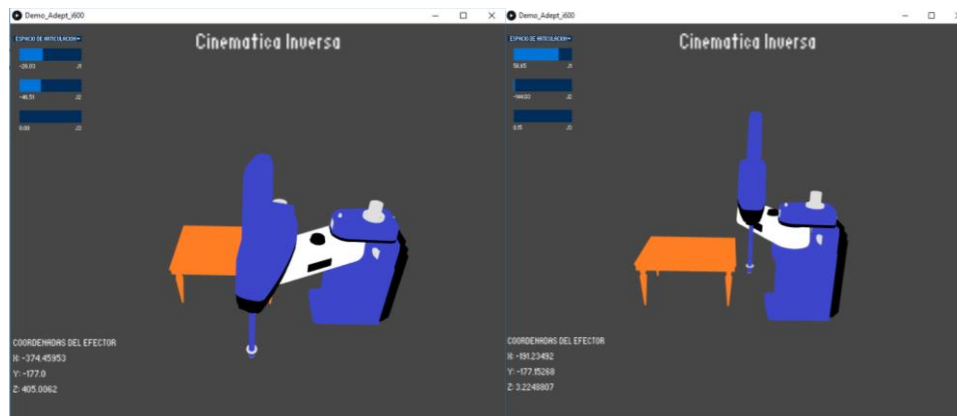


Figura 5.12: Movimientos en cinemática inversa.

A demás de la manipulación del robot, los usuarios pueden modificar directamente en el programa las ecuaciones de posición del robot fijando otros parámetros de tal manera que la matriz de transformación homogénea se reorganiza y genera una nueva vista del robot.

Conclusiones

Los resultados obtenidos a través del desarrollo de la interfaz gráfica, aseguran que la solución está acorde con los objetivos planteados y permitirá a estudiantes de pregrado tener un conocimiento básico de los movimientos y parametrización de un robot industrial.

El desarrollo de la programación en Processing, brinda una herramienta versátil y sencilla, basada en el lenguaje Java, para desarrollar modelos en 2D y 3D, de prototipos de máquinas y robots, que puede ser utilizada por jóvenes de secundaria y pregrado con la ventaja que es un software sin costo alguno y que constantemente está en procesos de actualización de sus librerías y aplicaciones.

Este trabajo es la base para futuros desarrollos en la programación y parametrización de robots en la universidad, permitiendo el fortalecimiento de la interacción entre la industria y la academia, así como la integración de nuevas tecnologías para la consolidación de la innovación tecnológica en diferentes campos de aplicación.

El desarrollo de los modelos virtuales es muy importante en el contexto educativo ya que permite simular los objetos antes de desarrollarlos realmente, lo que nos da la certeza de su funcionamiento óptimo y de las limitaciones o posibles errores en el proceso de diseño, además del gran potencial de los sistemas computacionales en todos los campos del conocimiento.

Bibliografía

Adep. 2016. *Technologies*. Recuperado de <https://>

www.adept.com/products/robots/scara/cobra-s600/

Ardila, 2014. *Brazo robótico de un grado de libertad modelado como péndulo invertido y operado por computador*. Salamanca, España:

Craig, 2016. *Robótica*. México: Prentice Hall

Muñoz, 2013. *Sistema de control para un robot self-balancing*

Pérez, 2009, *Diseño, modelamiento y simulación 3D de un robot móvil para exploración de terrenos*

Processing, 2017. Recuperado de <http://processing.org/>

Ramirez, 2017. *Cinemática directa de robot*. Recuperado de <http://www.kramirez.net/>.

Rodríguez, 2014. *Prototipo de robot submarino con la capacidad de seguimiento de trayectorias mediante tratamiento de imágenes*

Sun, 2017. *El Lenguaje de Programación Java*. Recuperado de

<http://www.mmc.geofisica.unam.mx/>

Torres, 2016. *Diseño de trayectorias del efector final de un robot manipulador para la evitación de obstáculos*. Recuperado de <http://repositorio.upct.es/bitstream/handle/>

ANEXOS

Programa desarrollado en la consola de Processing

```
import remixlab.bias.*;

import remixlab.bias.event.*;

import remixlab.dandelion.constraint.*;

import remixlab.dandelion.core.*;

import remixlab.dandelion.geom.*;

import remixlab.fpstiming.*;

import remixlab.proscene.*;

import remixlab.util.*;
```

```
import controlP5.*;
```

```
import remixlab.bias.*;

import remixlab.bias.event.*;

import remixlab.dandelion.constraint.*;

import remixlab.dandelion.core.*;

import remixlab.dandelion.geom.*;

import remixlab.fpstiming.*;

import remixlab.proscene.*;

import remixlab.util.*;
```

```
/**
```

```
 * Simulador de cinemática inversa
```

```
 * por Leonardo Duran Melo
```


* Universidad Pedagógica Nacional

*

* Simulador de cinemática inversa de robots

*/

//Carga de la librería ProScene

import remixlab.proscene.*;

import controlP5.*;

//Objetos de geometría de eslabones

PShape link0;

PShape link1;

PShape link2;

PShape link3;

PShape link4;

PShape mesa;

//Objeto de escena, base de la simulación

Scene scene;

ControlP5 cp5;

//Atributos para la cinemática inversa del robot SCARA.

PVector posicion; //Vector de posición

float posicionX; // Componente X del vector de posición

float posicionY; //Componente Y del vector de posición

```

float posicionZ; //Componente Z del vector de posición

//Atributos de cinemática inversa

float hipotenusa; //Hipotenusa, desde el origen de la base hasta la posición del efector

float alfa; //Angulo de orientación de la hipotenusa, desde el origen del eslabón base

float beta; //Angulo de orientación de la articulación 1

float gamma; // Angulo de orientación de la articulación 2

float theta1; //Angulo de orientación de la articulación 1

float theta2; //Angulo de orientación de la articulación 2

float d3; //Distancia de articulación 3

float a1=0; //Longitud del eslabón 1

float a2=325; //Longitud del eslabón 2

float a3=275; //Longitud del eslabón 3

String tipoC = "";

//Objeto iFrame, con posición del efector

InteractiveFrame punto;

//

boolean tipoCinematica = true; //True=inversa, False=directa.

//CONFIGURACIÓN DE LA ESCENA (se ejecuta solo una vez, durante la carga del
programa)

void setup() {

```

```
//Definición del tamaño de ventana, y parámetros de la escena
```

```
size(800, 640, P3D);
```

```
scene = new Scene(this);
```

```
scene.setRadius(500);
```

```
scene.setVisualHints( Scene.PICKING );
```

```
//Definición de controles graficos
```

```
cp5 = new ControlP5(this);
```

```
cp5.setAutoDraw(false);
```

```
float minJ1=-105;
```

```
float maxJ1=105;
```

```
float minJ2=-150;
```

```
float maxJ2=150;
```

```
float minJ3=0;
```

```
float maxJ3=200;
```

```
Group controles = cp5.addGroup("ESPACIO DE ARTICULACION")
```

```
.setPosition(5, 30)
```

```
.setBackgroundColor(color(70))
```

```
.setSize(110, 170);
```

```
cp5.addSlider("J1")
```

```

.setPosition(10,10)

.setRange(minJ1,maxJ1)

.setSize(100,20)

.setGroup(controles);

cp5.addSlider("J2")

.setPosition(10,60)

.setRange(minJ2,maxJ2)

.setSize(100,20)

.setGroup(controles);

cp5.addSlider("J3")

.setPosition(10,110)

.setRange(minJ3,maxJ3)

.setSize(100,20)

.setGroup(controles);


cp5.getController("J1").getValueLabel().align(ControlP5.LEFT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);

cp5.getController("J1").getCaptionLabel().align(ControlP5.RIGHT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);

cp5.getController("J2").getValueLabel().align(ControlP5.LEFT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);

cp5.getController("J2").getCaptionLabel().align(ControlP5.RIGHT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);

cp5.getController("J3").getValueLabel().align(ControlP5.LEFT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);

```

```

    cp5.getController("J3").getCaptionLabel().align(ControlP5.RIGHT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);

//Definición del iFrame del efector

punto = new InteractiveFrame( scene );

punto.setPosition(-a2-a3, -177,0);

//punto.setClickBinding(RIGHT, 1, "activarInversa");


//Carga de modelo 3D de los eslabones y objetos de la escena

link0 = loadShape("Link_0.obj");

link1 = loadShape("Link_1.obj");

link2 = loadShape("Link_2.obj");

link3 = loadShape("Link_3.obj");

link4 = loadShape("Link_4.obj");

mesa = loadShape("table.obj");

}


//DIBUJO DE LA ESCENA (se ejecuta cíclicamente durante el uso del programa)

void draw() {

    if (keyPressed) {

        if (key == ' ' || key == ' ') {

            tipoCinematica = !tipoCinematica;

            delay(100);

        }

    }

}

```

```
}
```

```
//Definición de vector del efector, para usar en cinemática inversa
```

```
posicion = new PVector(punto.position().x(), punto.position().y(), punto.position().z());
```

```
posicionX = posicion.x;
```

```
posicionY = posicion.y;
```

```
posicionZ = posicion.z;
```

```
if (tipoCinematica){
```

```
//Cálculo de la cinemática inversa del robot
```

```
hipotenusa = sqrt (pow(posicionX,2)+pow(posicionZ,2));
```

```
alfa = atan2(-posicionZ, -posicionX);
```

```
//Inclusión de la restricción del espacio de trabajo del robot en el plano XZ
```

```
if (hipotenusa<=600 && hipotenusa >162.6) {
```

```
    beta = acos((pow(hipotenusa,2)+pow(a2,2)-pow(a3,2))/(2*hipotenusa*a2));
```

```
    gamma = acos((pow(a2,2) + pow(a3,2) - pow(hipotenusa,2))/(2*a2*a3));
```

```
}
```

```
//Calculo de ángulos de articulación
```

```
if ((alfa+beta)*180/PI<=105 && (alfa+beta)*180/PI>=-105){
```

```
    theta1=alfa+beta;
```

```
}
```

```

theta2=gamma-PI;

//Inclusión de la restricción del espacio de trabajo del robot en XZ
if (theta1<=(-105*PI/180) && theta1>(105*PI/180) ) {
    punto.setPosition(600*cos(-105*PI/180),posicionY,600*sin(-105*PI/180));
    //posicionY=-177;
}

d3=posicionY;

//Inclusión de la restricción del espacio de trabajo del robot en Y
if (d3<=-342) {
    punto.setPosition(posicionX,-342,posicionZ);
}

if (d3>-177) {
    punto.setPosition(posicionX,-177,posicionZ);
}

cp5.getController("J1").setValue(theta1*180/PI);
cp5.getController("J2").setValue(theta2*180/PI);
cp5.getController("J3").setValue(-posicionY-177);

}

//Borrar pantalla para actualizar
background(70);

```

```
//Llamado a funciones de dibujo de los eslabones del robot y objetos de escena
```

```
drawBaseRobot();
```

```
drawLinkOne();
```

```
drawLinkTwo();
```

```
drawLinkThree();
```

```
//Dibujo de coordenadas del efector en pantalla
```

```
scene.beginScreenDrawing();
```

```
cp5.draw();
```

```
scene.endScreenDrawing();
```

```
scene.beginScreenDrawing();
```

```
pushStyle();
```

```
textSize(14);
```

```
fill(255);
```

```
text("COORDENADAS DEL EFECTOR ", 5, height-125);
```

```
text("X: " + posicionX , 5, height-100 );
```

```
text("Y: " + posicionY, 5, height-75);
```

```
text("Z: " + posicionZ, 5, height-50);
```

```
if(tipoCinematica){
```

```
    tipoC = "Inversa";
```

```
    } else{
```

```
        tipoC = "Directa";
```

```
    }
```

```
pushStyle();
```



```

        textSize(32);

        text("Cinematica " + tipoC, width-500, 30);

        popStyle();

        popStyle();

        scene.endScreenDrawing();

    }

```

```

//BASE_ROBOT Rendering

public void drawBaseRobot() {

    //translate(0, 50, 0);

    scale(1);

    rotateX(PI);

    fill(255);

    shape(link0);

    pushMatrix();

    translate(-500, 110, 0);

    scale(2);

    fill(0);

    shape(mesa);

    popMatrix();

}

```

```

//LINK_ONE Rendering

public void drawLinkOne() {

```

```
translate(0,0,0);  
  
scale(1);  
  
rotateY(theta1);  
  
fill(0);  
  
shape(link1);  
  
}
```

//LINK_TWO Rendering

```
public void drawLinkTwo() {  
  
    translate(-a2, 15, 0);  
  
    scale(1);  
  
    rotateY(theta2);  
  
    translate(a2, -15, 0);  
  
    shape(link2);  
  
}
```

//LINK_THREE Rendering

```
public void drawLinkThree() {  
  
    translate(0, -d3-177, 0);  
  
    //noStroke();  
  
    fill(24, 184, 19);  
  
    scale(1);  
  
    shape(link3);  
  
}
```

```
//void activarInversa(InteractiveFrame i) {  
  
//  tipoCinematica = true;  
  
//}
```

```
//Sets color according to selection  
  
public void setColor(boolean selected) {  
  
    if (selected)  
  
        fill(200, 200, 0);  
  
    else  
  
        fill(200, 200, 200);  
  
}
```

```
void J1(float j1){  
  
    //tipoCinematica=false;  
  
    theta1=j1*(PI/180);  
  
    cinematicaDirecta();  
  
}
```

```
void J2(float j2){  
  
    //tipoCinematica=false;  
  
    theta2=j2*(PI/180);  
  
    cinematicaDirecta();  
  
}
```

```
void J3(float j3){
```

```

//tipoCinematica=false;

d3=-j3-177;

cinematicaDirecta();

}

void cinematicaDirecta(){

    if (!tipoCinematica){

        punto.setPosition(-(a2+a3*cos(theta2))*cos(theta1),d3,-(a3*cos(-
theta2)*sin(theta1)+a2*sin(theta1)));

    }

}

```