Desarrollo de software para modelado y fabricación de objetos de madera usando un robot SCARA

Roberto Loaeza Valerio

12 de diciembre de 2008

Índice general

1. Introducción **6**

1.1. Motivación 6

1.2. Antecedentes 7

1.3. Objetivos 7

1.4. Alcances 8

1.5. Metodología 8

1.6. Contribuciones esperadas 8

1.7. Descripción de los capítulos 9

2. Ambiente de programación de robots manipuladores **10**

2.1. Robot 10

2.1.1. Clasificación 11

2.1.2. Estructura de los robots manipuladores 11

2.1.2.1. Tipos de articulaciones 11

2.1.2.2. Estructuras básicas 12

2.2. Revisión del Estado del Arte 13

2.2.1. Programación 13

2.2.2. Desarrollo tecnológico 14

3. Modelado de objetos **15**

3.1. Introducción 15

3.2. Algoritmos de graficación 15

3.2.1. Puntos 15

3.2.2. Lineas 15

3.2.3. Circulos . . . 15
3.2.4. Elipses . . . 15
3.2.5. Curvas de Bezier . . . 15
3.2.6. Elementos compuestos . . . 15
3.3. Almacenamiento y recuperación . . . 15
3.3.1. Almacenamiento en formato XML . . . 15
3.3.2. Estructura de almacenamiento . . . 16
3.4. Importacion de otros formatos . . . 17
3.4.1. Imágenes PGM . . . 17
3.4.2. Archivos CAD (dxf) . . . 17

4. Interfaces de software **18**
4.1. Introducción . . . 18
4.2. Lenguaje de programación . . . 18
4.2.1. Java . . . 19
4.3. Diseño de la interfaz gráfica . . . 20
4.4. Elección de sistema de renderización. . . . 20
4.4.1. OpenGL . . . 21
4.4.1.1. GLUT . . . 21
4.4.1.2. JOGL (JSR-231) . . . 22

5. Interfaz entre modelado y el robot SCARA **23**
5.1. Introducción . . . 23
5.2. Robot SCARA . . . 23
5.2.1. Características . . . 23
5.2.2. Diseño . . . 23
5.2.3. Limitantes . . . 23
5.3. Conversión de modelado a acciones SCARA . . . 23
5.3.1. Cinemática inversa . . . 23
5.4. Conectividad mediante puerto serie . . . 23

6. Pruebas **24**

6.1. Introducción 24

6.2. Funcionalidad multiplataforma del software 3D 24

6.2.1. Windows 24

6.2.2. Linux 24

6.3. Conectividad multiplataforma entre software y el robot 24

6.3.1. Windows 24

6.3.2. Linux 24

6.4. Pruebas en campo 24

6.4.1. Realizacion de primitivas 24

6.4.2. Imagenes PGM 24

6.4.3. Archivos CAD 24

7. Conclusiones **25**

7.1. Conclusiones 25

7.2. Trabajos futuros 25

Anexo **27**

Índice de figuras

1.1. Diagrama representativo del objetivo de la tesis 7

2.1. Tipos de articulaciones: a) rotación b) prismática c) cilíndrica d) planar e) esférica . 12

2.2. Estructura: a) 3D b) 2D c) 1D d) 3G 12

3.1. Estructura de almacenamiento 16

3.2. PGM a Robot 17

4.1. Diseño de la interfaz gráfica 20

4.2. Aplicación utilizando OpenG y aceleración gráfica por hardware 21

Índice de tablas

4.1. Lenguajes Sistemas Operativos 19
4.2. Comparativa entre Direct3D, Java3D y OpenGL 20

Capítulo 1

Introducción

En la actualidad las empresas madereras requieren tener procesos automatizados tanto en su administración como en la fabricación de sus productos para poder competir en el actual modelo de mercado. Desafortunadamente en la región maderera existen muy pocas empresas que cuenten con una automatización en sus procesos. Para automatizar una empresa dedicada a la producción de artefactos de madera se requiere principalmente de:

- Software tipo ERP - CRM para la adminstración.
- Robots industriales para la fabricación de productos.

Por parte la adminstración parece estar automatizada hasta cierto grado haciendo uso de paquetes de software como compiere, contpaq, entre otros; en cambio muy pocas empresas cuentan con robots industriales ya que estos requieren una inversion superior a los 1'000'000.00 pesos y en la mayoría son de origen extranjero lo cual implica que si llega a fallar es muy poco probable que se solucione en poco tiempo lo que conlleva a perdida de dinero y tiempo.

Observando la necesidad de la automatización en los procesos de fabricación de productos se optó por desarrollar una aplicación multiplataforma para el modelado de productos de madera haciendo uso de un robot SCARA realizado en la Facultad de Ingenieria Eléctrica de la Universidad Michoacana de San Nicolas de Hidalgo.

1.1. Motivación

En la actualidad la utilización de robots manipuladores en la mayoría de las pequeñas y medianas fábricas, en específico las fábricas de la región maderera del estado de Michoacán, son factor de que estas tengan un menor costo en su proceso de producción y con esto puedan así competir en el mercado saturado por productos extranjeros.

Por otra parte el diseño innovador de nuevos productos de madera se ve estancado por la falta de una herramienta para su modelado e implementación en un robot manipulador.

1.2. Antecedentes

Los antecedentes previos realizados en la División de Estudios de Posgrado de la Facultad de Ingeniería Eléctrica, de la Universidad Michoacana de San Nicolás de Hidalgo, es la construcción de un robot manipulador tipo SCARA de cuatro grados de libertad de plataforma abierta, para el cual se desarrollará la aplicación de la presente tesis.

1.3. Objetivos

El objetivo general es realizar investigación a fondo en los aspectos computacionales relacionados con el modelado de objetos físicos y desarrollar el software necesario para que el robot SCARA pueda realizarlos, de forma que se desarrolle tecnología propia que promueva una industria robótica y se generen diversas aplicaciones que tengan un importante impacto social como económico.

Se realizarán también estudios sobre implementación de planeadores de movimientos en brazos robóticos para que las acciones se realicen lo más rápido posible.

Finalmente se realizarán pruebas del software desarrollado con el robot SCARA.

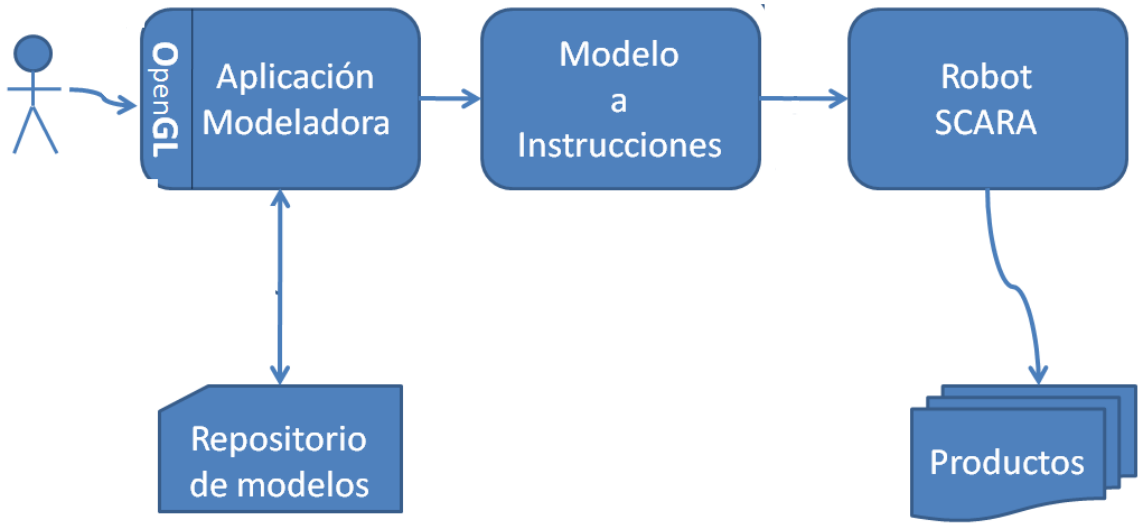

Figura 1.1: Diagrama representativo del objetivo de la tesis

En la figura 1.1 representa gráficamente el objetivo, donde se puede apreciar que el presente proyecto de tesis tiene como entrada el diseño por parte de una persona el cual es interpretado para su futura producción en un robot SCARA.

Objetivos específicos

- Generar una plataforma 3D para el modelado de objetos de madera.
- Implementar algoritmos de gráficación sobre la plataforma 3D los siguientes elementos:
  - Puntos
  - Lineas
  - Curvas de Bezier
  - Curvas Spline
- Desarrollar software para modelado implementando los algoritmos anteriores.
- Desarrollar software para comunicación entre el software de modelado y el robot SCARA.

1.4. Alcances

Los alcances previstos son:

- Desarrollo de una aplicación multiplataforma capaz de modelar objetos de madera para su producción utilizando un robot SCARA.
- Desarrollo de interfaz entre la aplicación multiplataforma para modelar objetos de madera y el robot SCARA para la producción.

1.5. Metodología

Para lograr los objetivos mencionados anteriormente se propone desarrollar una aplicación multiplataforma para el modelado de objetos de madera en el lenguaje Java.

Para esto se desarrollarán prototipos de la aplicación, hasta lograr una estabilidad considerable, la cual tenga una velocidad alta a la hora de generar los modelos en diferentes vistas, esto se deberá hacer mediante hardware usando la API de OpenGL.

Con el fin de impulsar este tipo de proyectos, se propone que la aplicación final sea evaluada en los talleres de Paracho.

1.6. Contribuciones esperadas

A continuación se describen brevemente las contribuciones de este trabajo:

- Aplicación multiplataforma para la construcción de modelos físicos de madera. Se utiliza la API OpenGL para acelerar tanto cálculo como regeneración de vistas 3D.

- Aplicación multiplataforma para la conexion entre el modelador y el robot SCARA.
- Código fuente, éste se distribuirá bajo la "*GNU General Public License v2*" y se puede acceder a él en la siguiente url: modelando-madera.googlecode.com

1.7. Descripción de los capítulos

En el capítulo 2 se presentarán una revisión del estado del arte asociado con la paqueteria de software existente para el modelado/fabricación de objetos de madera. Finalmente se describirá el contenido de cada capítulo.

En el capítulo 4 se mostrarán las principales interfaces de software para la implementación de software 3D, se mostrarán pros y contras de cada una de estas y finalmente se detallará el funcionamiento de la interfaz OpenGL sobre la cual se realizará la implementación del software.

En el capítulo 3 aborda las partes del software modelador tanto los algoritmos de graficación como la forma de interacción con el usuario final.

En el capítulo 5 trata las características del robot SCARA, la forma de conversión del modelo a acciones que el robot entienda así como un algoritmo de planeación para que el robot sea lo más optimo posible. También se describirá la interfaz entre el software de modelado y el robot.

En el capítulo 6 se mostrarán resultados de pruebas realizadas en una primera instancia al software modelador, seguido de los resultados de la conectividad entre el software y el robot, y finalmente se verificará que tanta funcionalidad doto el algoritmo de planeación implementado.

En el capítulo 7 se aportarán las conclusiones generales resultado de la investigación abordada e ideas para trabajo de investigación posterior a realizar en el mismo campo del conocimiento.

Capítulo 2

Ambiente de programación de robots manipuladores

2.1. Robot

Un robot, es un agente artificial mecánico o virtual. Los primeros son maquinas usadas para realizar un trabajo automaticamente y son controlados por una computadora; por otro lado tenemos a los robots virtuales también conocidos como "bots" que son agentes virtuales de software.

En general para considerar a un agente artificial como robot, este debe cumplir algunas propiedades:

- Manipular cosas de su entorno.
- Sentir cosas del entorno.
- Contar con cierta inteligencia para tomar decisiones basadas en el ambiente o en una secuencia previamente programada.
- Ser programable
- Moverse en uno o más ejes.
- Realizar movimientos coordinados.

A continuación se presentan algunas definiciones realizadas por diferentes institutos:

- La ISO 8373 lo define como "*un manipulador multipropósito automaticamente controlado y reprogramable de mas de tres ejes*".
- El Instituto de Robótica de América lo define como "*manipulador multifuncional y reprogramable diseñado para mover material, partes, herramientas o dispositivos especializados mediante varios movimientos programados para la realización de una variedad de tareas*".

2.1.1. Clasificación

La Asociación Japonesa de Robots Industriales (JIRA, por sus siglas en inglés) clasificó los robots desde manipuladores simples hasta sistemas avanzados incorporando inteligencia artificial. A continuación se lista dicha clasificación la cual fue realizada a finales del siglo veinte[CER, p. 267]:

- Manipuladores operados manualmente. Máquinas que requieren ser operadas por humanos.
- Manipuladores secuenciales. Dispositivos que realizan una serie de tareas en la misma secuencia cada vez que son activados.
- Manipuladores programables. Se incluyen los tipos simples de los robots industriales.
- Robots controlados numéricamente. Servo robots.
- Robots con sensores. Robots que utilizan sensores de cualquier tipo, proximidad, presión, tactil, etc.
- Robots adaptativos. Robots que ajustan la forma en que trabajan para compensar cambios en el entorno.
- Robots inteligentes. Robots con controladores de salida sofisticados que pueden considerarse para procesar inteligencia artificial.
- Sistemas mecatrónicos inteligentes. Computadoras que controlan un conjunto de robots o dispositivos robóticos.

2.1.2. Estructura de los robots manipuladores

Básicamente la estructura de un robot manipulador es la de un brazo articulado. De manera más específica un robot manipulador industrial es una cadena cinemática abierta formada por un conjunto de eslabones interrelacionados mediante articulaciones las cuales permiten el movimiento de esta dicha cadena.

2.1.2.1. Tipos de articulaciones

Existen diferentes tipos de articulaciones como podemos observar en la figura 2.1:

a) La articulación de rotación suministra un grado de libertad, es decir permite la rotación sobre el eje de la articulación.

b) La articulación prismática consiste en una translación a lo largo del eje de la articulación.

c) En la articulación cilíndrica existen dos grados de libertad, una rotación y una translación sobre el eje de la articulación.

d) La articulación planar se caracteriza por el movimiento de desplazamiento en un plano, tiene dos grados de libertad.

e) Por último la articulación esférica combina tres giros en tres direcciones perpendiculares en el espacio.

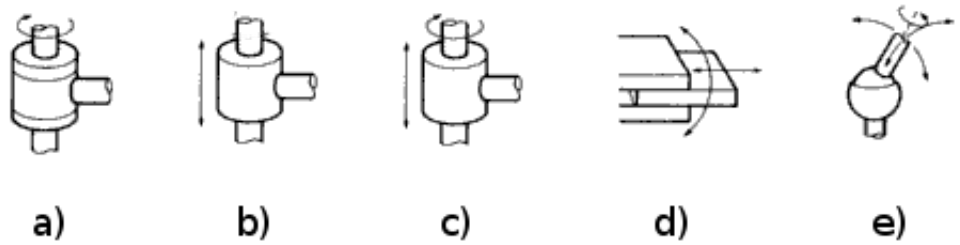

Figura 2.1: Tipos de articulaciones: a) rotación b) prismática c) cilíndrica d) planar e) esférica

2.1.2.2. Estructuras básicas

La estructura básica de un robot manipulador consiste en un brazo compuesto por elementos con articulaciones entre ellos. En el último enlace se coloca un efector final como una pinza o un dispositivo especial para realizar operaciones específicas. Estas estructuras tienen ciertas propiedades en cuanto a su espacio de trabajo y accesibilidad a posiciones determinadas. En la figura 2.2 se muestran algunas configuraciones básicas.

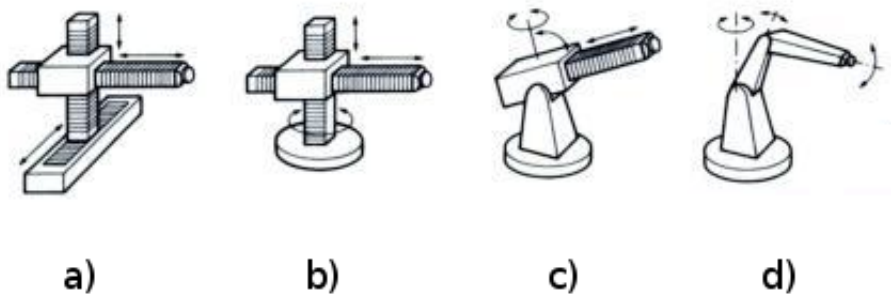

Figura 2.2: Estructura: a) 3D b) 2D c) 1D d) 3G

a) Configuración 3D o PPP tiene tres articulaciones prismáticas. Es bastante usual en la industria para el tansporte de cargas voluminosas.

b) Configuración cilíndrica presenta un volumen de trabajo parecido a un cilindro, sin embargo generalmente no tienen una rotación de 360º.

c) Configuración polar, los de brazo articulado y los modelos SCARA presentan un volumen de trabajo irregular.

d) Configuración angular.

2.2. Revisión del Estado del Arte

En la actualidad los robots son usados para realizar tareas peligrosas, difíciles, repetitivas y/o complicadas para los humanos. Esto usualmente toma la forma de un robot industrial usado en las líneas de producción. Algunas aplicaciones incluyen la limpieza de residuos tóxicos, minería y localización de minas terrestres. Sin embargo la manufactura continúa siendo el principal mercado donde los robots son utilizados, en particular, robots articulados son los más usados comúnmente. Las aplicaciones incluyen soldado, pintado y carga de maquinaria. En este ramo la industria automotriz ha tomado gran ventaja de esta nueva tecnología donde los robots han sido programados para reemplazar el trabajo de los humanos en muchas tareas repetitivas.

2.2.1. Programación

Un lenguaje de programación de robots sirve como interfaz entre el usuario humano y el robot industrial. Los manipuladores de robots se diferencian a sí mismos de la automatización fija por ser "flexibles", es decir que son programables. No solo son programables los movimientos de los manipuladores sino que, a través del uso de sensores y comunicación con otros tipos de automatización, los manipuladores pueden adaptarse a las variaciones a medida que realizan su tarea.

Generalmente la programación de los robots se realiza fuera de linea, a continuación se listan paquetes existentes para programar:

- MS Robotic Studio (~ $ 4 000.00 M.N)
  - Herramienta de programación visual para crear y depurar aplicaciones robóticas. El desarrollador puede interactuar con los robots mediante interfases basadas en web o nativas al sistema operativo (MS Windows).
  - Contiene simulación realística provista por el motor PhysX de AGEIA. Se posibilita la emulación por software o la aceleración por hardware.
  - Se permiten varios lenguajes como: Microsoft Visual Studio Express languages (Visual C#® y Visual Basic® .NET), JScript® y Microsoft IronPython 1.0 Beta 1, y lenguajes de terceros que se adecuen a la arquitectura basada en servicios.
  - Robots soportados: CoroWare's CoroBot ($3 200), Lego Mindstorms NXT, iRobot Create y Robosoft's robots (38 a 65 K€).

- KUKA.OfficeLite (>$55 000.00 USD)

    - Este sistema de programación posee las mismas características que el software de sistema KUKA: para el manejo y la programación se utiliza la interfaz de usuario Original KUKA y la sintaxis KRL: un lenguaje completo.
    - Disponibilidad de todo el repertorio de funciones de las respectivas ediciones del software de sistema. Sin embargo, no se pueden conectar dispositivos de hardware periféricos.
    - Comprobación de sintaxis mediante el compilador y el interpretador disponibles; creación de programas KRL de usuario ejecutables.
    - Control completo de la ejecución de un programa de aplicación de robot. Ello permite optimizar la duración de los ciclos.
    - El Techware de KUKA para optimización de programas se puede utilizar e instalar en todo momento. De este modo, en un PC estándar se puede disponer de todo el software de sistema Original, sin necesidad de emulaciones.
    - Las entradas originales se pueden simular.
    - KUKA.OfficeLite no se puede utilizar para controlar un robot.

2.2.2. Desarrollo tecnológico

Desafortunadamente el desarrollo actual en el país está enfocado generalmente a robots móviles dejando a los manipuladores de lado.

Universidades como el Tecnológico de Monterrey [Dirección de Vinculación y Desarrollo campus Guadalajara] se está enfocando más en la planeación de movimientos para robots, así como el Instituto Tecnológico Autónomo de México[Laboratorio de robótica, ITAM] se enfocan al área de robots móviles pequeños.

Capítulo 3

Modelado de objetos

3.1. Introducción

Para modelar

3.2. Algoritmos de graficación

3.2.1. Puntos

3.2.2. Lineas

3.2.3. Circulos

3.2.4. Elipses

3.2.5. Curvas de Bezier

3.2.6. Elementos compuestos

3.3. Almacenamiento y recuperación

El almacenamiento y recuperación es una de las operaciones más importantes en un sistema,

3.3.1. Almacenamiento en formato XML

El formato XML

Algunas de las ventajas de este formato son:

- Independiente de la plataforma.

- Soporta código Unicode.
- El mismo documento define la estructura y los campos asi como los valores respectivos.
- Es basado en estándares internacionales.

3.3.2. Estructura de almacenamiento

La estructura de almacenamiento puede observarse en la figura 3.1. Como puede observar la estructura es simple, compacta y fácil de entender.

```
<modelador>
    <fig tipo="1">
        <x0> val_x0 </x0>
        <y0> val_y0 </y0>
        <z0> val_z0 </z0>
            .  .  .
            .  .  .
            .  .  .
        <xN> val_xN </xN>
        <yN> val_yN </yN>
        <zN> val_zN </zN>
    </fig>
     .  .  .
    <fig tipo="M">
          ...
     </fig>
</modelador>
```

Figura 3.1: Estructura de almacenamiento

En la figura 3.1 puede observarse:

- No existe límite en la cantidad de primitivas usadas en un diseño.
- Los puntos de control de una primitiva pueden no tener límite, es decir una primitiva puede tener N puntos de control.
- Cada primitiva es identificada mediante mediante un número entero (tipo):
    - Punto (1)
    - Línea (2)
    - Poli-Línea (3)
    - Círculo (4)
    - Elipse (5)
    - Curva de Bezier (6)

3.4. Importacion de otros formatos

3.4.1. Imágenes PGM

Una imagen PGM es la representacion a escala de grises de una imagen.

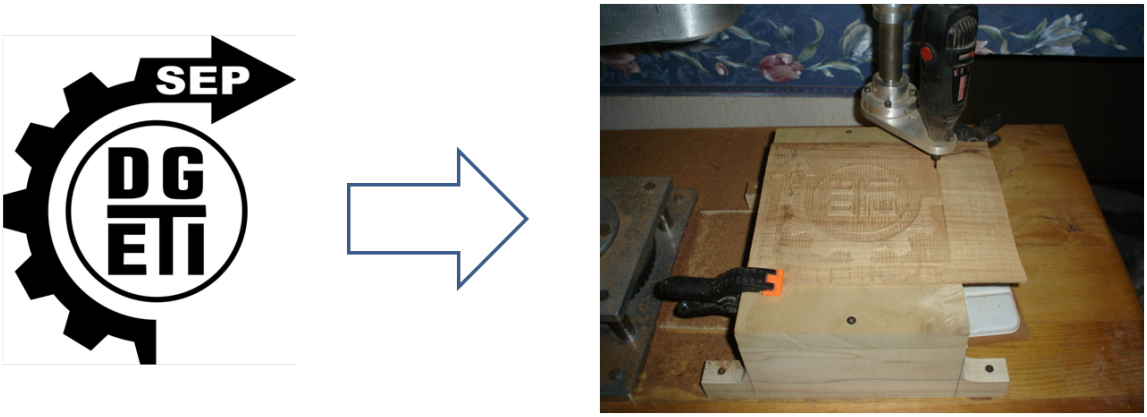

Figura 3.2: PGM a Robot

3.4.2. Archivos CAD (dxf)

Capítulo 4

Interfaces de software

4.1. Introducción

La interfaz de usuario es lo más importante en cualquier sistema de cómputo, en la actualidad todo sistema debe contener una herramienta con la que el usuario pueda ordenar al computador que hacer.

Una interfaz mal diseñada puede causar que el mejor sistema de cómputo sea ineficaz y por lo tanto sea deshechado, es por esto que se debe diseñar una interfaz de usuario, la cual incluye:

- Eleccion de lenguaje de programación.
- Diseño de la interfaz gráfica.
- Elección de sistema de renderización.

4.2. Lenguaje de programación

En la actualidad existe una gran cantidad de lenguajes de programación, la mayoría de propósito general y el resto de propósito específico, a continuación se listan los más populares de ambas categorías:

- C.
- Basic.
- Java.
- .Net.
- Perl.
- PHP.

Estos podrían ser los más utilizados hoy por hoy, pero debido a la heterogeneidad de sistemas operativos y arquitecturas, no todos los lenguajes de programación son aplicables.

En el caso de los lenguajes compilados es poco probable su funcionalidad en diferentes plataformas debido a la dependencia de tipo de procesador que adquieren al momento de compilar, por ejemplo si se compila en un procesador tipo RISC (Reduced-Intruction-Set Computing) es imposible ejecutarlo en un procesador tipo CISC (Complex-Intruction-Set Computing).

Por otro lado los lenguajes interpretados son muy portables a todas las arquitecturas y sistemas operativos debido a que no dependen de un tipo de arquitectura pero requieren de que exista un progama nativo que los interprete, es decir, que si la empresa desarrolladora del lenguaje interpretado no libera una versión de su intérprete para un sistema operativo en una arquitectura determinada los programas escritos en este lenguaje no se ejecutarán.

	Windows	Linux	MacOS
C	X	X	X
Basic	X		
Java	X	X	X
.Net	X	X	
Perl	X	X	X
PHP	X	X	

Tabla 4.1: Lenguajes Sistemas Operativos

En la tabla 4.1 se puede apreciar los alcances de los lenguajes de programación más desarrollados y más difundidos en la actualidad, dejando claro que Java es el lenguaje de programación con mayor portabilidad, debido a que se cuenta con intérpretes para todas las arquitecturas y además que no se requiere modificar el código fuente para ejecutarse en éstas.

4.2.1. Java

Java es una plataforma de software desarrollada por Sun Microsystem, de forma que los programas creados en ella puedan ejecutarse sin cambios en diferentes tipos de arquitecturas y dispositivos computacionales.

Esta plataforma está compuesta por:

- El Lenguaje de programación.
- La máquina virtual de Java (JVM).
- Un conjunto de bibliotecas estándar, conocidas como API.

El lenguaje de programación usa la sintaxis de C++, incorpora sincronización, manejo de tareas e interfaces como un mecanismo alternativo a la herencia múltiple de C++.

La máquina virtual de Java (JVM) es un programa nativo (un ejecutable de una plataforma específica) capaz de interpretar y ejecutar código binario especial (llamado Java Bytecode), el cual es

generado a partir del compilador de Java. Es por esto que Sun Microsystem ha liberado versiones de su JVM para las arquitecturas y sistemas operativos más utilizados, volviendo así a las aplicaciones desarrolladas en Java multiplataforma.

4.3. Diseño de la interfaz gráfica

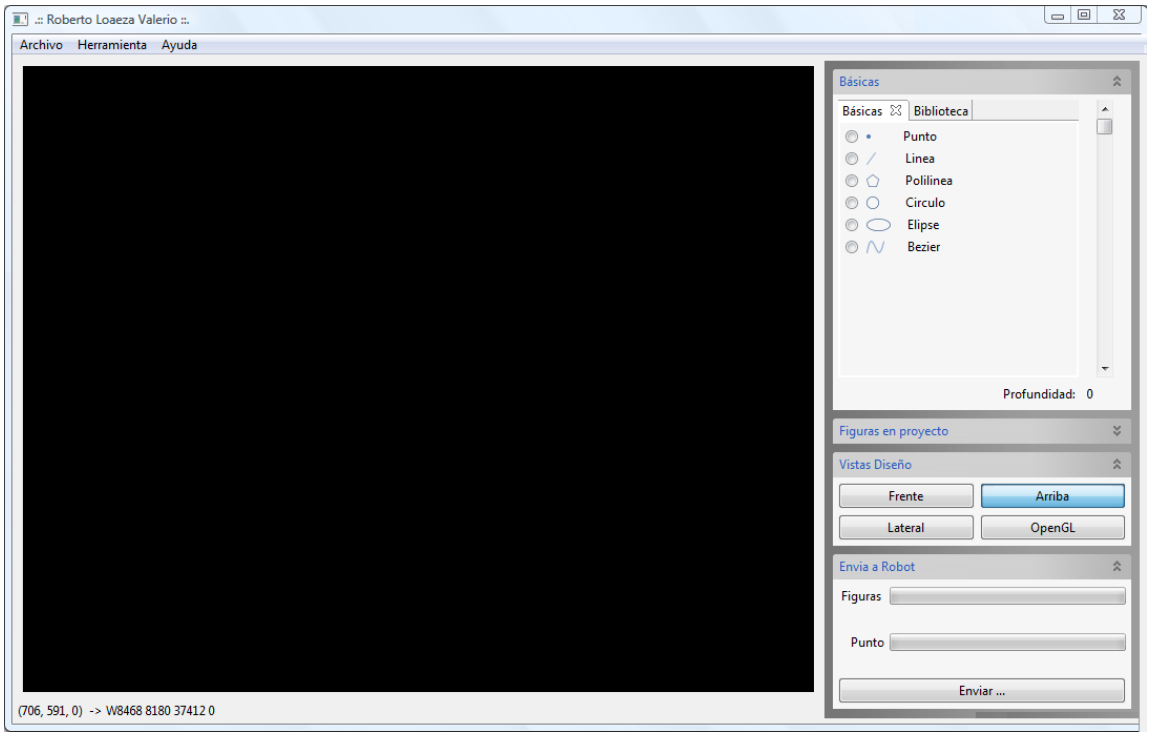

Figura 4.1: Diseño de la interfaz gráfica

4.4. Elección de sistema de renderización.

La elección debe realizarse tomando en cuenta ventajas y desventajas de las tecnologías disponibles las cuales se muestran en la tabla 4.2.

	Direct3D	Java3D	OpenGL
Plataformas soportadas	Windows	Multiplataforma	Multiplataforma
Soporte nativo	Si	No	Si
Licencia	EULA	GNU General Public	SGI

Tabla 4.2: Comparativa entre Direct3D, Java3D y OpenGL

Debido a que Direct3D solo soporta la plataforma MS-Windows fue deshechada y por otra parte Java3D tiene una dependencia con OpenGL o con Direct3D también fue deshecada dejando como sistema de renderización a OpenGL.

4.4.1. OpenGL

OpenGL es una especificación estándar que define una API multilenguaje y multiplataforma para escribir aplicaciones que produzcan gráficos 2D y 3D. Fue desarrollada por Silicon Graphics Inc. (SGI) en 1992 .

Las funciones y comandos de esta API están integrados en las tarjetas de aceleración gráfica 3D, o en su defecto son emuladas por el sistema operativo. Es claro que para el propósito específico de modelado 3D no se debe realizar emulación por software sino se debe tener acceso directo al hardware.

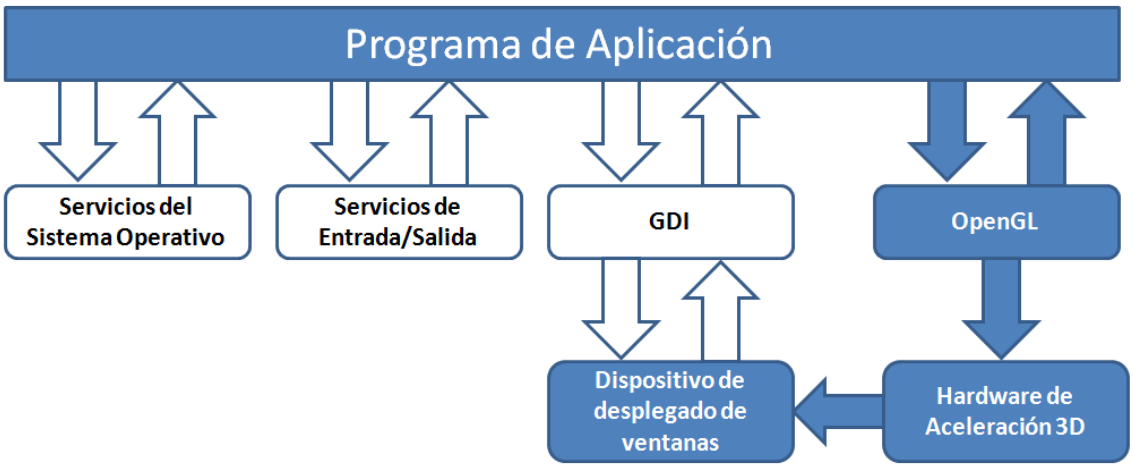

Figura 4.2: Aplicación utilizando OpenG y aceleración gráfica por hardware

En la figura 4.2 se muestra una aplicación utilizando OpenGL y aceleración gráfica por hardware; se puede observar que las llamadas a la interfaz de aplicación OpenGL pasan directamente al hardware de aceleración y no pasan a tavés de la interfaz de dispositivo de gráfico (GDI) que sería el caso si se tratara de simulación por software.

4.4.1.1. GLUT

Es un conjunto de herramientas OpenGL, un sistema de ventanas independientes para escribir programas OpenGL en lenguajes C y FORTRAN. Implementa un simple interfaz de programación para aplicaciones (API) para OpenGL. Por su parte GLUT hace considerablemente fácil la programación OpenGL.

GLUT provee un soporte para las plataformas Windows, Linux, Mac. Con la única desventaja de tener que recompilar para ser usado en otra plataforma.

4.4.1.2. JOGL (JSR-231)

Jogl es un enlace del lenguaje de programación Java para la API de OpenGL. Soporta integración con los estándares de Java AWT y Swing. Jogl provee aceso a las más recientes rutinas de OpenGL (OpenGL 2.0 con extenciones de fabricante) asi como una pataforma independiente para accesar a hardware acelerador.

Jogl tambien prvee algunas de las mas populares características introducidas por otros enlaces existentes para OpenGL como son GL4Java, LWJGL, GLUT entre otros.

Es importante mencionar que esta tecnologia es código abierto y fue iniciada por Game Technology Group.

Capítulo 5

Interfaz entre modelado y el robot SCARA

5.1. Introducción

La interfaz entre el software modelador y el robot es una parte muy importante, ya que si esta llega a colapsar o mal-funcionar se verá reflejado en un trabajo final(objeto de madera) no satisfactorio.

5.2. Robot SCARA

5.2.1. Características

5.2.2. Diseño

5.2.3. Limitantes

5.3. Conversión de modelado a acciones SCARA

5.3.1. Cinemática inversa

5.4. Conectividad mediante puerto serie

Capítulo 6

Pruebas

6.1. Introducción

6.2. Funcionalidad multiplataforma del software 3D

6.2.1. Windows

6.2.2. Linux

6.3. Conectividad multiplataforma entre software y el robot

6.3.1. Windows

6.3.2. Linux

6.4. Pruebas en campo

6.4.1. Realizacion de primitivas

6.4.2. Imagenes PGM

6.4.3. Archivos CAD

Capítulo 7

Conclusiones

7.1. Conclusiones

7.2. Trabajos futuros

Anexo Glosario

CRM Customer Relationship Management

ERP Enterprise Resource Planning.

ISO International Organization for Standardization

JIRA Japan Industrial Robot Association

OpenGL Open Graphics Library.

PGM Portable GrayMap.

SCARA Selective Compliant Articulated Robot Arm.

XML Extensible Markup Language

Bibliografía

[WIKI-ROBOT] http://en.wikipedia.org/wiki/Industrial_robot

[GLUT] http://www.opengl.org/resources/libraries/glut/

[JOGL] https://jogl.dev.java.net/

[RAH] Thomas R. Kurfess, Robotics and Automation Handbook, E.U.A.: ed. CRC PRESS, 2005.

[CER] Stan Gibilisco, Concise Encyclopedia of Robotics, E.U.A.: ed. McGraw-Hill, 2003

[PJGD] Andrew Davison, Pro Java 6 3D Game Development Java 3D, JOGL, JInput and JOAL APIs, E.U.A.: ed. Apress, 2007

[RMRM] Aníbal Ollero Baturone, Robótica: Manipuladores y robots móviles, Marcombo Editorial.