**UNIVERSIDAD NACIONAL DE INGENIERÍA**

**FACULTAD DE ELECTROTECNIA Y COMPUTACIÓN**

INFORME DE TRABAJO MONOGRÁFICO

SISTEMA OPERATIVO PARA ROBOTS(ROS): APLICACIÓN EN EL DESARROLLO DE UN LABORATORIO VIRTUAL PARA EL ESTUDIO DE LOS FUNDAMENTOS DE LA ROBÓTICA INDUSTRIAL

Presentado por: Br. Yeser Alfredo Morales Calero

Tutor: MSc. Alejandro Alberto Méndez Talavera

Diciembre 12, 2018

**DEDICATORIA**

**RESUMEN**

**ABSTRACT**

**INDICE**

[INTRODUCCIÓN 1](#_Toc498515051)

[OBJETIVOS 3](#_Toc498515052)

[JUSTIFICACIÓN 4](#_Toc498515053)

[CAPÍTULO I: MARCO TEÓRICO 6](#_Toc498515054)

[1. CONTROLADORES DE LOGICA PROGRAMABLE 7](#_Toc498515055)

[1.1 INTRODUCCIÓN 7](#_Toc498515056)

[1.2 TIPOS DE PLCs 8](#_Toc498515057)

[1.3 GESTIÓN DE LA ALIMENTACIÓN DE LA ENERGÍA DEL SISTEMA. 10](#_Toc498515058)

[1.4 SECCIÓN DE ENTRADAS Y SALIDAS. 12](#_Toc498515059)

[1.4.1 Entradas digitales. 13](#_Toc498515060)

[1.4.1.1 Aislamiento Galvánico. 1](#_Toc498515060)5

[1.4.2 Salidas digitales 16](#_Toc498515061)

[1.4.2.1 Tipos de salidas digitales 1](#_Toc498515061)7

[1.4.3 Entradas y salidas digitales tipo Sinking y Sourcing 18](#_Toc498515062)

[1.5 MÓDULOS DE ENTRADAS Y SALIDAS. 19](#_Toc498515063)

[1.5.1 Módulos de entradas analógicas 21](#_Toc498515064)

[1.5.2 Módulos de salidas analógicas 22](#_Toc498515065)

[1.6 PROTOCOLOS DE COMUNICACIÓN. 23](#_Toc498515066)

[1.7 CONTROLADOR. 24](#_Toc498515067)

[1.8 PROGRAMACIÓN DEL CONTROLADOR. 25](#_Toc498515068)

[1.9 INTERFAZ HUMANO MÁQUINA (HMI). 27](#_Toc498515069)

[CAPÍTULO II: DISEÑO E IMPLEMENTACIÓN DEL CONTROLADOR Y MÓDULOS DE EXPANSIÓN 29](#_Toc498515070)

[2 MODELO DEL CONTROLADOR IMPLEMENTADO 30](#_Toc498515071)

[2.1 DETERMINACIÓN DE LOS COMPONENTES DEL CONTROLADOR. 30](#_Toc498515072)

[2.1.1 Determinación del microcontrolador. 31](#_Toc498515073)

[2.1.2 Acondicionamiento de señales digitales de entrada. 32](#_Toc498515074)

[2.1.3 Determinación del driver para los Relés. 34](#_Toc498515075)

[2.1.4 Implementación del circuito impreso. 38](#_Toc498515076)

[2.2 MÓDULO DE ENTRADAS-SALIDAS DIGITALES 39](#_Toc498515077)

[2.2.1 Implementación del circuito impreso. 43](#_Toc498515078)

[2.3 MÓDULO DE ENTRADAS ANALÓGICAS 43](#_Toc498515079)

[2.3.1 Determinación del ADC 44](#_Toc498515080)

[2.3.2 Acondicionamiento de las señales de entrada analógicas 47](#_Toc498515081)

[1.4.2 Receptor de corriente (4 – 20) mA](#_Toc498515061) 47

[1.4.2 Receptor de voltaje (0 – 5) V, (0 – 10) V](#_Toc498515061) 49

[2.3.3 Implementación del circuito impreso. 51](#_Toc498515082)

[2.4 MÓDULO DE SALIDAS ANALÓGICAS 52](#_Toc498515083)

[2.4.1 Determinación del DAC 53](#_Toc498515084)

[2.4.2 Salidas de voltaje de (0 – 5) V, (0 – 10) V. 55](#_Toc498515085)

[2.4.3 Salidas de corriente de 4 – 20 mA. 56](#_Toc498515086)

[2.4.4 Implementación del circuito impreso 58](#_Toc498515087)

[2.5 IMPLEMENTACIÓN DEL HMI 59](#_Toc498515088)

[2.5.1 Lectura del teclado vía *I2C.* 61](#_Toc498515089)

[2.5.2 Implementación del circuito impreso 64](#_Toc498515090)

[2.6 REGULADORES DE VOLTAJE PARA 5 VDC Y 12 VDC. 64](#_Toc498515091)

[2.7 BOOTLOADER Y PROGRAMACIÓN. 65](#_Toc498515092)

[2.8 SIMULACIÓN DE LOS SISTEMAS. 69](#_Toc498515093)

[2.8.1 Interfaz de entradas de corriente. 69](#_Toc498515094)

[2.8.2 Interfaz de entradas de voltaje 71](#_Toc498515095)

[2.8.3 Salidas de voltajes analógicos 73](#_Toc498515096)

[2.9 RESULTADOS OBTENIDOS 74](#_Toc498515097)

[2.10 COSTOS 77](#_Toc498515098)

[CAPÍTULO III: CoNCLUSIONES Y RECOMENDACIONES 8](#_Toc498515070)2

[BIBLIOGRAFÍA 84](#_Toc498515100)

[ANEXOS 86](#_Toc498515101)

[A- Construcción de prototipo del controlador y sus módulos. 87](#_Toc498515102)

[B- Códigos de las pruebas del semáforo y lectura de corriente.](#_Toc498515102) 91

**Lista de Figuras**

**Capítulo I**

Figura [1.1: Diagrama de bloques del controlador y módulos de expansión 6](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672338)

Figura [1.2: Ejemplo de un PLC tipo Nano. Siemens Logo! OBA7](#_Toc473672339) 8

Figura [1.3: Ejemplos de PLCs Compactos. Omron CP1L, Allen Bradley L23E y Siemens S7-200](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672340) 9

Figura [1.4: Ejemplos de PLCs modulares. Allen Bradley SLC-500 y Siemens S7-300](#_Toc473672341) 10

Figura [1.5: Diagrama de bloques de entrada discreta. En “Programmable Logic Controller” (P.23), por Frank Petruzella](#_Toc473672342) 13

Figura [1.6: Diagrama de bloques de salida discreta. En “Programmable Logic Controller: An emphasis on desing and application” (P.158), por Kelvin T. Erickson](#_Toc473672342) 16

Figura [1.7: Conexiones en sourcing y sinking (a) PLC entrada Sinking (b) Entrada Sourcing. En “Programmable Logic Controller: An emphasis on desing and application” (p. 150), por Kelvin T. Erickson](#_Toc473672343) 19

Figura [1.8: Conexiones en sourcing y sinking (a) PLC salida Sourcing (b) Salida Sinking. En “Programmable Logic Controller: An emphasis on desing and application” (p. 150), por Kelvin T. Erickson](#_Toc473672344) 19

Figura [1.9: Aislamiento entre dispositivos de entrada/salida y el PLC. En “Programmable Logic Controller: An emphasis on desing and application” (p. 149), por Kelvin T. Erickson 20](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672338)

Figura [1.10: HMI. En “Programmable Logic Controller” (P.40), por Frank Petruzella 28](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672340)

Figura 1.11: HMI implementado en PC. Tomado de: http://setecindca.com/descargas/software-winccflex.pdf 28

**Capítulo II**

Figura [2.1: Sistema para el controlador y módulos implementados](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672338) 30

Figura [2.2: Optoacoplador: En “Programmable Logic Controller” (p. 8). por W. Bolton](#_Toc473672339) 32

Figura [2.3: Interfaz de entradas digitales. (Captura de pantalla) 33](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672340)

Figura [2.4: ULN2803A. Tomado de: http://www.ti.com/lit/ds/symlink/uln2803a.pdf](#_Toc473672341) 35

Figura 2.5: ULN2803A, Diodo Clamp. Tomado de: http://www.ti.com/lit/ds/symlink/uln2803a.pdf 36

Figura [2.6: ULN2803A como driver para relés (Captura de pantalla)](#_Toc473672342) 37

Figura [2.7: Vista del plano inferior y superior del PCB del controlador. (Captura de pantalla)](#_Toc473672343) 39

Figura 2.8: PCF8574. Tomado de: http://www.ti.com/lit/ds/symlink/pcf8574.pdf 40

Figura [2.9: Esquemático. Módulo digital de entradas y salidas. (Captura de pantalla)](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672338) 42

Figura [2.10: Vista del plano inferior y superior del módulo digital. (Captura de pantalla) 43](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672340)

Figura 2.11: MCP3424. Tomado de: http://ww1.microchip.com/downloads/en/DeviceDoc/22088c.pdf 46

Figura [2.12: Convertidor de corriente a voltaje (Captura de pantalla) 48](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672338)

Figura [2.13: Receptor de corriente (4 – 20) mA. (Captura de pantalla) 48](#_Toc473672339)

Figura [2.14: Divisor de voltaje (Captura de pantalla) 50](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672340)

Figura [2.15: Esquema de entrada analógica (0 – 5) V, (0 – 10) V. (Captura de pantalla)](#_Toc473672341) 51

Figura [2.16: Vista del plano inferior y superior del módulo de entradas analógicas. (Captura de pantalla)](#_Toc473672342) 52

Figura 2.17: MCP4728. Tomado de: http://ww1.microchip.com/downloads/en/DeviceDoc/22187E.pdf 54

Figura [2.18: Esquema de salida analógica (0 – 5) V, (0 – 10) V. (Captura de pantalla)](#_Toc473672343) 55

Figura 2.19: XTR111. Tomado de: http://www.ti.com/lit/ds/symlink/xtr111.pdf 57

Figura 2.20: Circuito de aplicación XTR111. Tomado de: http://www.ti.com/lit/ds/symlink/xtr111.pdf 57

Figura [2.21: Vista del plano inferior y superior del módulo de salidas analógicas. (Captura de pantalla)](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672340) 59

Figura [2.22: LCD *I2C*. (Captura de pantalla)](#_Toc473672341) 60

Figura [2.23: Teclado *I2C*. (Captura de pantalla) 6](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672338)0

Figura [2.24: Lectura del teclado por *I2C*, paso #1. (Captura de pantalla) 61](#_Toc473672339)

Figura [2.25: Lectura del teclado por *I2C*, paso #2. (Captura de pantalla) 6](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672340)2

Figura [2.26: Lectura del teclado por *I2C*, paso #3. (Captura de pantalla)](#_Toc473672341) 62

Figura [2.27: Vista del plano inferior y superior del módulo HMI. (Captura de pantalla)](#_Toc473672342) 64

Figura [2.28: Regulador 7805. (Captura de pantalla)](#_Toc473672342) 65

Figura [2.29: Regulador 7812. (Captura de pantalla)](#_Toc473672343) 65

Figura [2.30: Hardware para BootLoader. (Captura de pantalla)](#_Toc473672344) 66

Figura [2.31: Interfaz gráfica de Siow. (Captura de pantalla)](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672338) 67

Figura 2[.32: Buscar archivo a grabar. (Captura de pantalla)](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672340) 68

Figura [2.33: Selección de archivo a grabar. (Captura de pantalla)](#_Toc473672341) 68

Figura [2.34: Grabación del PIC completada. (Captura de pantalla) 6](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672338)8

Figura [2.35: Simulación receptor de corriente 4 mA. (Captura de pantalla) 70](#_Toc473672339)

Figura 2[.36: Simulación receptor de corriente 12 mA. (Captura de pantalla) 70](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672340)

Figura [2.37: Simulación receptor de corriente 20 mA. (Captura de pantalla)](#_Toc473672341) 71

Figura [2.38: Simulación receptor de voltaje 5 V a 2 V. (Captura de pantalla)](#_Toc473672342) 72

Figura [2.39: Simulación receptor de voltaje 10 V a 2 V. (Captura de pantalla)](#_Toc473672342) 72

Figura [2.40: Salida analógica de 5 V. (Captura de pantalla)](#_Toc473672343) 73

Figura [2.41: Salida analógica de 10 V. (Captura de pantalla)](#_Toc473672344) 74

Figura [2.42: Prueba de funcionamiento de Semáforo con el controlador](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672338) 75

Figura [2.43: Prueba de funcionamiento del controlador, módulo de entrada analógica y módulo de salida analógica 7](file:///C:\Users\Vilma\Dropbox\Monografia%20Incubadora\avance_doc_monografico_Final.docx#_Toc473672340)6

**Anexos**

**Anexo A**

Figura A.1: Prueba de funcionamiento XTR111 en tabla de nodos 87

Figura A.2: Prueba de funcionamiento módulo HMI en tabla de nodos 88

Figura A.3: Elaboración de circuitos impresos 89

Figura A.4: Primera versión de PCB del controlador y módulo den entradas y salidas digitales 90

Figura A.5: Prueba de funcionamiento con el controlador y módulo HMI. 90

Figura A.6: Prototipo final. Controlador, módulo de entradas y salidas digitales, módulo de salidas analógicas, módulo de entradas analógicas, módulo HMI 91

Figura A.7: Prototipo final, instalado en un gabinete. 92

**Lista de Tablas**

**Capítulo I**

Tabla 1.1: Requerimientos de la fuente de alimentación a usar 11

Tabla 1.2: Rango de voltaje y corriente analógicas para señales industriales 13

Tabla 1.3: Rango de voltaje para entradas industriales 14

Tabla 1.4: Dispositivos industriales de entrada 14

Tabla 1.5: Dispositivos finales de control 17

Tabla 1.6: Rangos de voltaje para salidas industriales 17

Tabla 1.7: Dispositivos generadores de señales analógicas 21

Tabla 1.8: Dispositivos analógicos de salidas 22

Tabla 1.9: Comparación de protocolos de comunicación 23

**Capitulo II**

Tabla 2.1: Comparación de ADCs 46

Tabla 2.2: Comparación de DACs 54

Tabla 2.3: Costos de elaboración del controlador y módulos 77

Tabla 2.4: Comparación de precios. Logo! OBA8 vs prototipo desarrollado 80

Tabla 2.5: Precios fabricando cien tarjetas electrónicas de cada subsistema desarrollado 81

**INTRODUCCIÓN**

La robótica es uno de los campos de la tecnología que se desarrolla velozmente y encuentra cada día nuevas áreas de aplicación de tal forma que la clasificación de los robots debe ser ajustada constantemente. En la actualidad encontramos aplicaciones de la robótica en el campo personal y doméstico, servicios profesionales, investigación y desarrollo. Unos de los primeros campos donde los robots entraron en acción fue la industria en la cual destaca el uso de los mismos en el sector automotriz (soldadura y/o manejo de materiales). A pesar de que el uso de los robots en la industria data de la década de los 60s, en Nicaragua el uso de los mismos es relativamente bajo. El hecho mencionado tiene un impacto negativo en el desarrollo de la industria nicaragüense. Son pocas las industrias que poseen robots industriales y, de igual forma, son pocas las instituciones de educación superior que cuentan con la infraestructura adecuada, y el recurso humano, para la enseñanza e investigación de la robótica. Una de las razones es que un laboratorio físico, para experimentar en los diferentes aspectos asociados con los robots industriales, requiere de una inversión relativamente alta dado que es necesario garantizar el espacio físico, adquirir los robots y su mantenimiento.

La falta de infraestructura para el desarrollo e investigación de la robótica ha sido experimentada en la mayoría de las instituciones de educación superior latinoamericana muchas de las cuales han recurrido a la utilización laboratorios virtuales, diseñados e implementados por dichas instituciones o adquiridos en el mercado internacional. MATLAB, software ampliamente utilizado en el campo de las ingenierías, cuenta con un toolbox para robótica (Corke, 1996) mediante el cual se puede modelar y simular el comportamiento de un manipulador, sin embargo, el mismo tiene limitaciones. Simscape Multibody (antes llamado SimMechanics), de MATLAB y Simulink, permite la simulación de sistemas mecánicos en 3D, tales como robots. El software mencionado requiere de una licencia y el costo de la misma aumenta en dependencia del número de usuarios. También se puede utilizar el software de programación de robots, versión evaluación, de algunos fabricantes, tales como Robot Studio de ABB. Dicho software, además de limitaciones temporales, solo puede ser utilizado en los robots fabricados por la empresa. Para trabajar con diferentes robots, es necesario adquirir y aprender los lenguajes de programación de cada uno de ellos.

En la actualidad se busca estandarizar el trabajo con los robots y una de las herramientas disponibles para tal fin es el middleware “Sistema Operativo para Robot (ROS, por sus siglas en inglés)” el cual se ha convertido en el estándar de facto a nivel industrial tanto así que ya existe una versión denominada ROS Industrial (2017). ROS (s.f.) es un middleware, open-source, para el desarrollo a gran escala de complejos sistemas robóticos. Se trata de una colección de herramientas, librerías y convenciones que tienen como objetivo simplificar la tarea de crear un comportamiento complejo y robusto en una amplia variedad de plataformas robóticas.

Para dar a conocer ROS y mostrar la importancia de este, para el desarrollo de la robótica, se desarrollo un laboratorio virtual de robótica (LVR) para el estudio de los fundamentos de la robótica industrial.

Para el desarrollo del laboratorio, además de ROS, se utilizaron herramientas de simulación tales como GAZEBO, visualizadores como RVIZ y RQT. La integración de los diferentes nodos se logró utilizando el lenguaje de programación C++, para la modelación de los robots se utilizó el formato de descripción unificado de robots (URDF, por sus siglas en inglés), y para la creación de ambientes y escenarios fue empleado el formato de descripción de la simulación (SDF, por sus siglas en inglés).

El laboratorio virtual cuenta con los recursos necesarios para que los estudiantes estudien y realicen prácticas relacionadas a la morfología de los robots, herramientas matemáticas necesarias para el estudio de los robots, cinemática de los robots y programación de manipuladores industriales. Un elemento de mucha importancia en el LVR, es el simulador que incorporará el cual permitirá manipular el modelo de un robot en un escenario particular, en respuesta a un programa utilizando instrucciones parecidas a las que incorporan los lenguajes de programación de fabricantes tales como ABB.

El laboratorio virtual de robótica cuenta con un manual en el cual es presentada información de los recursos disponibles en el LVR, de forma tal que los docentes puedan diseñar prácticas que permitan a los estudiantes afianzar los fundamentos de la robótica industrial, e información general sobre ROS y herramientas asociadas, a un nivel tal que el usuario podrá implementar un ambiente de desarrollo básico. El profesor podrá elaborar prácticas de laboratorio relacionadas con los cuatro aspectos mencionados utilizando para ello los modelos de robots presentes en el LVR. Por ejemplo, el profesor podrá diseñar una práctica para que los estudiantes apliquen el procedimiento de Denavit y Hartenberg para determinar la matriz de transformación homogénea. En su versión inicial, el LVR cuenta con dos guías para realizar algunas actividades que permitan a los futuros usuarios apropiarse del procedimiento necesario para utilizar efectivamente los recursos del LVR.

En el presente documento son presentados los principales aspectos relacionados con el desarrollo del laboratorio virtual, así como los resultados obtenidos.

**OBJETIVO****S**

**Objetivo general**

* Desarrollar un laboratorio virtual para el estudio de los fundamentos de la robótica industrial utilizando, para su implementación, el sistema operativo para robots (ROS).

**Objetivos específicos**

* Identificar las principales características/recursos/estructura de ROS enfatizando aquellas aplicables directamente a la robótica industrial.
* Identificar las herramientas de software requeridas, además de ROS, para el desarrollo del laboratorio virtual para el estudio de los fundamentos de la robótica industrial.
* Desarrollar la interfaz gráfica de usuario (GUI) del laboratorio virtual, según los requerimientos establecidos.
* Verificar la efectividad de los recursos del laboratorio virtual mediante la realización de dos prácticas de laboratorio.
* Elaborar un manual para el instructor donde se describa los recursos, y las características, del laboratorio virtual

**JUSTIFICACIÓN**

El campo de la robótica ha tenido, principalmente en las últimas décadas, un desarrollo vertiginoso y está impactando de forma positiva, principalmente en países desarrollados, en áreas como la medicina, la industria, y la investigación, entre otras. En Nicaragua, país en vías de desarrollo, el uso de la robótica es relativamente bajo, sin embargo, dicha tecnología tendrá mayor presencia en el futuro cercano y por consiguiente es necesario contar con profesionales con las competencias requeridas para realizar efectiva y eficientemente las diferentes tareas asociadas con la aplicación, el desarrollo y la investigación de la robótica.

La formación de profesionales en el campo de la robótica requiere de recursos humanos e infraestructura apropiada para tal fin (Jara, 2011a). En Nicaragua pocas instituciones de educación superior cuentan con recursos básicos mínimos para la formación de los estudiantes en el campo de la robótica (Universidad Tecnológica La Salle, 2017), siendo uno de los principales obstáculos para tener mejores condiciones la imposibilidad de garantizar la alta inversión requerida para la adquisición de los robots, software, accesorios asociados, instalación y mantenimiento de acuerdo a un estudio de Castellanos, & Martínez (2010). Compañías como FESTO ofrecen Kits denominados modulares, para prácticas de un proceso automatizado utilizando un robot Mitsubishi, que cuesta alrededor de 41,500 euros.

En el campo industrial, muchos fabricantes ofrecen manipuladores industriales los cuales tienen precios altos y, además, cada uno ofrece su propio software para la programación de los robots (Owen, 2016). Lo anterior significa que los estudiantes o especialistas en robótica industrial tendrían que aprender a utilizar el software correspondiente al manipulador en turno lo cual es imposible lograr si no se cuenta con la infraestructura adecuada. Se han realizado intentos para estandarizar los aspectos relacionados con la programación de los robots, hacerla independiente del fabricante, y uno de los resultados más destacado en la última década es el middleware “Sistema Operativo para Robot” (ROS, por sus siglas en inglés). En la actualidad es considerado el estándar de facto para la aplicación, desarrollo e investigación de la robótica. Una muestra de la importancia de ROS es que en el campo industrial crearon el ROS-Industrial.

El desarrollo de un laboratorio virtual en robótica contribuiría en la mejora de las condiciones requeridas para el estudio de los fundamentos básicos de esta ya que brindaría a los estudiantes, y docentes, una herramienta que podría ser utilizada en cualquier lugar y en cualquier momento.

El laboratorio virtual sería desarrollado tomando como elemento principal el middleware ROS lo cual, entre otras cosas, permitiría a los estudiantes experimentar con modelos de robots de diferentes fabricantes tales como ABB, KUKA, FANUC, entre otros.

La inversión en el desarrollo e implementación del laboratorio virtual siempre será inferior a la requerida para tener un laboratorio físico ya que la mayoría de las herramientas utilizadas, tales como C++, ROS, GAZEBO, Qt creator son de código abierto (Open Source).

Los principales beneficiaros de los resultados de este proyecto son los docentes y los estudiantes. Los primeros dispondrán de una herramienta a partir de las cuales podrán diseñar experimentos, con su guía apropiada, relacionados con los fundamentos básicos de los robots industriales. Los estudiantes, por su parte, contarán con una herramienta flexible, que podrán utilizar en cualquier momento y en cualquier lugar, la cual les posibilitará verificar los fundamentos teóricos relacionados con los robots industriales abordados en programas de asignatura relacionados así programar los robots para que realicen tareas básicas.

El laboratorio virtual de robótica desarrollado debe ser considerado como el primer paso hacia un laboratorio virtual con mayores recursos en el cual se pueda experimentar con diversos tipos de robots, no solo los antropomórficos. Lo anterior contribuiría considerablemente en el estudio del funcionamiento de los robots y sus aplicaciones y presentaría un espacio para el desarrollo e investigación en el campo de la robótica.

**CAPÍTULO I: MARCO TEÓRICO**

Las instituciones académicas, principalmente las de educación superior (IES), buscan constantemente mejorar la calidad de la educación y una forma de hacerlo es integrando nuevas técnicas y herramientas orientadas a mejorar los resultados del proceso enseñanza-aprendizaje. En la actualidad, en muchas IES, a la tradicional clase presencial se suman otras formas de enseñanza-aprendizaje las cuales se fundamentan en las TIC, permitiendo de esta manera nuevas formas de enseñar, aprender, generar y compartir conocimiento.

Un recurso de mucho impacto en la formación de los futuros profesionales y cuya presencia crece día a día en los procesos de enseñanza aprendizaje es el laboratorio virtual (LV), el cual puede presentar altos niveles de flexibilidad y la inversión es muy inferior a la requerida para instalar un laboratorio real.

Uno de los campos en el cual el desarrollo y utilización de un laboratorio virtual reviste mucha importancia, dada la inversión alta requerida para tener un laboratorio físico, es el de la robótica en general y en particular, la robótica industrial.

En la introducción del presente documento se estableció que el trabajo de monografía propuesto pretende dar a conocer el middleware ROS, su importancia en el desarrollo de la robótica industrial, y dar muestra de sus posibilidades mediante el diseño y construcción de un laboratorio virtual que sirva de base para el estudio de los fundamentos de los manipuladores industriales tipo serie.

En los siguientes apartados se presentan los elementos teóricos-tecnológicos requeridos para el desarrollo del laboratorio virtual.

* 1. **Tipos de laboratorios**

La experimentación de un fenómeno físico, o mediante la simulación, requiere de entornos de trabajos que reúnan los equipos o herramientas necesarias para el desarrollo efectivo de una práctica. En los ambientes académicos podemos encontrar, en la actualidad, diferentes tipos de laboratorios

* + 1. **Laboratorio Físico**

Es ampliamente utilizado en universidades con modelos clásicos de enseñanza convirtiendo estos sitios de trabajo como el único sitio concebido para elaborar un experimento el cual involucra la presencia física tanto del tutor como del alumno. Cabe destacar que la interacción directa con los equipos apropiados de un laboratorio físico aporta una experiencia difícil de igualar debido a que los alumnos perciben los resultados de carácter palpable entrando en juego los cinco sentidos (Vista, tacto, audición, olfato e incluso, a veces el gusto) Calvo, (2009a).

* + 1. **Laboratorio Virtual**

Es una alternativa al laboratorio físico y utiliza recursos computacionales haciendo uso de modelos matemáticos y recursos de visualización como modelos CAD y animaciones gráficas. Se pretende, mediante de este tipo de laboratorio, que el estudiante pueda estudiar el comportamiento de la realidad mediante el uso de modelos, sencillos o complejos, de la misma. Los recursos de un laboratorio virtual pueden ser ampliados en el tiempo permitiendo la realización de prácticas asociadas a temas de mayor complejidad.

* + 1. **Laboratorio Remoto online**

Nacen bajo la necesidad de dar acceso a los estudiantes a un espacio de trabajo, de forma online, que combina los recursos de un laboratorio físico con recursos de software. El estudiante accede al laboratorio por medio de una página web pudiendo de esta forma controlar los recursos disponibles y llevar a cabo el experimento de interés. Su aprovechamiento está basado en la accesibilidad por parte de un usuario en horarios flexibles.

* + 1. **Realidad virtual 3D**

Es la integración de recursos de hardware y software para acercar aún más al usuario a la experimentación de una teoría estimulando los sentidos tanto de la vista al exponerlos a entornos virtuales 3D y audición bajo altavoces con sonido envolvente.

En la actualidad se puede encontrar diferentes recursos para la enseñanza de la robótica Industrial tales como kits de robótica industrial, laboratorios virtuales de robótica de categoría libre o de pago para ejecutar cierta práctica. El desarrollo de un laboratorio remoto requiere contar con acceso a un laboratorio físico y enlazarlo a un servidor para que el estudiante pueda acceder vía remota garantizándole una plataforma amigable y multiplataforma como pueden ser los navegadores WEB. Candelas, F. et al. (2004) analizan las ventajas del uso de un laboratorio remoto ROBOLAB, proyecto desarrollado por ellos mismos bajo el nombre de grupo de investigación AUROVA. El proyecto utiliza herramientas gráficas para modelado y visualización de objetos 3D de un brazo robótico que coincide con el mismo modelo de un robot físico, permitiendo al estudiante realizar sus practica al observar y configurar el modelo virtual y una vez alcanzado un nivel de aprendizaje aceptable, proceder a interactuar con el robot físico utilizando la misma plataforma web. Ellos concluyen que al utilizar el laboratorio Remoto ROBOLAB “*La mayoría de alumnos prefieren disponer de un laboratorio en la universidad dónde trabajar con la ayuda de los compañeros y el apoyo didáctico del profesor, pero también hay muchos alumnos que reciben con agrado la opción de un laboratorio virtual que les ofrezca unos horarios flexibles en los que realizar los experimentos”.* (Candelas, et al, 2004b).

En resumen, un laboratorio virtual es un elemento importante en el proceso enseñanza-aprendizaje ya que destacan, entre otras, las siguientes ventajas según Calvo, et al (2009b):

1. El estudiante se familiariza con el experimento evitando acudir al aula sin conocimiento previo.
2. Comparación del comportamiento de modelos matemáticos ante una simulación permitiendo extraer sus propias conclusiones de cierta práctica.
3. Manejo de herramientas informáticas contemporáneas para la formación integral de un estudiante.
4. Repetitividad de los experimentos realizados por el estudiante que permitirá reproducir cuantas veces desee hasta consolidar el conocimiento.
5. Disminución de riesgos y accidentes que pueden ocasionar una mala práctica o configuración de un equipo físico.
6. Multiplicidad de experimentos simultáneos realizados ya que cada estudiante podrá ejecutar la práctica indicada en su computador asignado, además de esta forma se favorecen los procesos colaborativos como el de “Lluvia de ideas” al opinar cada alumno sobre su percepción adquirida al ejecutar la práctica.

Considerando lo anterior, se decidió, para dar a conocer la importancia y posibilidades de ROS, desarrollar un laboratorio virtual el cual tendrá a ROS como elemento integrador. Es importante destacar la inversión relativamente baja requerida para implementar el laboratorio virtual, la posibilidad de ampliación del mismo y la posibilidad que el mismo ofrecerá a los docentes para diseñar nuevas prácticas a partir de los recursos disponibles.

* 1. **Sistema operativ****o para robot (ROS)**

El sistema operativo para robot (ROS), elemento fundamental del presente trabajo, es un middleware ampliamente utilizado en el mundo creciente de la robótica. Originalmente fue desarrollado en el 2007 en los laboratorios de Inteligencia Artificial de Stanford y en el 2008, cedieron el derecho de desarrollo al instituto de investigación de robótica Willow Garage (2010), donde la filosofía es proporcionar bibliotecas y herramientas libres para ayudar a los desarrolladores de software a crear aplicaciones para robots

En este trabajo monográfico, se promueve la importancia de ROS en el campo de la robótica y se hace mediante el diseño y construcción, utilizando los recursos proporcionados por este, de un Laboratorio Virtual de Robótica, en el cual será posible realizar experimentos sobre los fundamentos de la robótica industrial específicamente de los manipuladores industriales tipo serie.

Una característica importante de ROS es su código abierto (Open Source) lo que ha llevado al desarrollo, por muchos colaboradores a nivel mundial, de una amplia colección de herramientas, librerías y convenciones. Los recursos que ofrece ROS permiten el desarrollo, a gran escala, de sistemas robóticos complejos, tanto físicos como simulados.

Los desarrolladores de ROS han establecido un conjunto de convenciones y buenas prácticas en el desarrollo de software de forma tal que se fomente la reutilización de código para robots, trabajando sobre una arquitectura robótica totalmente sólida y funcional. De esta forma se garantiza, entre otras cosas, que el proyecto no sea fallido y que sus resultados satisfagan la calidad requerida en este tipo de trabajos.

* + 1. **Conceptos básico****s de ROS**

El sistema operativo para robot (ROS) fue diseñado bajo una estructura distribuida y modular con el propósito de que los usuarios puedan usar los recursos requeridos según la aplicación a desarrollar. Es decir, el usuario puede seleccionar los recursos de software necesarios para implementar la solución.

En su operación, ROS presenta una red de ***procesos*** que se ejecutan simultáneamente en una estructura ***peer-to-peer*** donde cada nodo o elemento del sistema puede actuar al mismo tiempo como cliente y como servidor. Para una aplicación distribuida de ROS se requiere que los nodos peer to peer estén sincronizados bajo un middleware el cual genera un orden temporal para los eventos generados en el sistema, además de utilizar la comunicación XML/RPC que define la información de registro de cada nodo al ROSMaster, una vez registrado los nodos se procede a ejecutar la comunicación peer to peer entre los nodos utilizando el protocolo TCP/IP y, entre los muchos conceptos básicos relacionados en la misma, destacan los siguientes:

1. **Nodos** (nodes): Son procesos que realizan cálculos. Por ejemplo, en un sistema robótico, un nodo realiza la planificación de la trayectoria y otro puede suministrar una vista grafica del sistema.
2. **Mensajes** (messages): Los nodos se comunican entre ellos enviándose mensajes. Un mensaje es simplemente una estructura de datos, que contiene campos de un solo tipo de datos. Tipos primitivos de datos (enteros, booleanos, punto flotante, etc.) son soportados, así como lo son arreglos de tipos primitivos.
3. **Tópico** (topic): El tópico es un nombre usado para identificar el contenido de un mensaje. Un nodo envía un mensaje publicándolo en un tópico dado. Los mensajes son enrutados vía un sistema de transporte con una semántica de publicación/suscripción. es un nombre utilizado para identificar el contenido de un mensaje. Un nodo que está interesado en cierta clase de datos debe suscribirse al tópico apropiado.
4. **Maestro** (master): El master proporciona los elementos necesarios para que los nodos se encuentren los unos a los otros, intercambien mensajes o invoquen servicios.
5. **Servicios** (services): Es un modelo de comunicación cliente-servidor, es una manera en el que los nodos se pueden comunicar. Estos permiten que se envíen solicitudes y se reciban respuestas
6. **Bolsas** (bags): Formato para guardar y reproducir datos de un mensaje proveniente de un nodo de ROS. Son un importante mecanismo para almacenar datos, tales como datos de sensores, que son difíciles de obtener pero que son necesarios para el desarrollo y prueba de algoritmos.

La arquitectura básica de comunicación de ROS, usando 3 nodos como ejemplo simplificado es mostrada en la figura 2.1. Para establecer comunicación con diferentes nodos es necesario la utilización de registros, cuyos nombres son suministrados por el maestro.

Como se muestra en la Figura 2.1 el Nodo 1, el Nodo 2 y el Nodo 3 se registran al Máster usando comunicación XML/RPC el cual, luego de conocer cada nodo registrado, permite a los diferentes nodos intercambiar la información entre ellos utilizando para esto el protocolo TCP/IP.

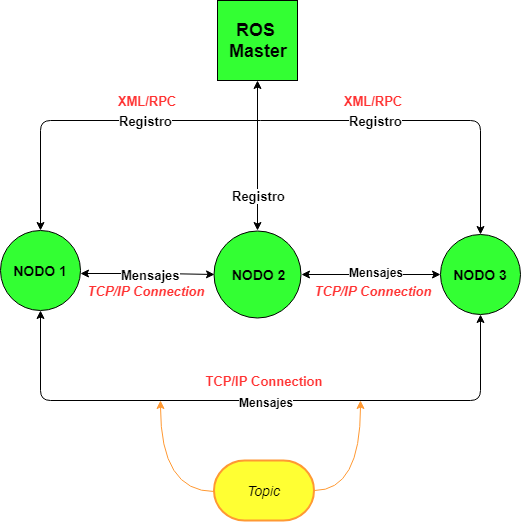


Fig.2.1.- Esquema de funcionamiento de la Arquitectura de ROS

Dado que cada nodo puede compartir múltiples mensajes, a estos se les clasifica como tópicos (topics) lo cual permite la centralización de información proveniente de varios procesos de un sistema robótico ya sea este físico o simulado. En la figura 2.2 podemos observar un comportamiento básico de un nodo y sus tópicos.

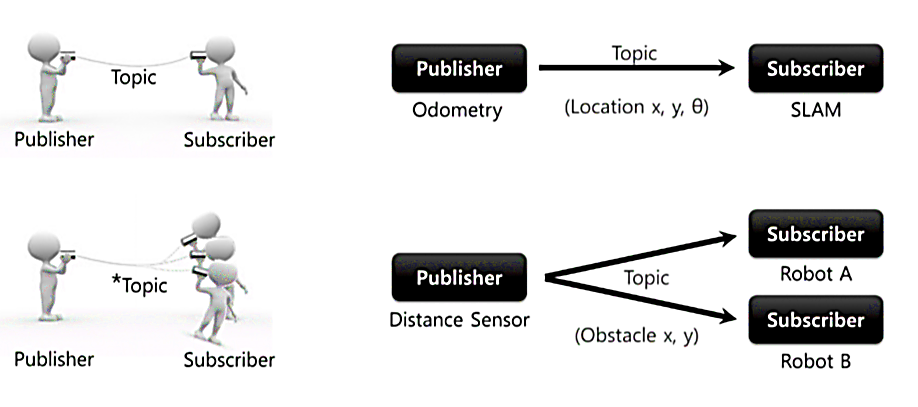


Figura 2.2. – Comunicación utilizando Tópicos. Copyright 2017 por ROBOTIS Co., Ltd.

Un ejemplo del funcionamiento de la arquitectura de ROS es mostrado en la figura 2.3 en la cual se identifica el ***nodo*** “robot\_localization.” El nodo mencionado recibe mensajes desde varios nodos bajo los ***tópicos***: IMU, Camera, GPS, Detección 3D (Sensing 3D) y odometria (odometry), los datos suministrados en los ***mensajes*** son procesados por el nodo y como resultado comparte bajo un único tópico, y varios mensajes, al visualizador 3D RVIZ las coordenadas de localización del objeto físico para que este pueda ser representado de manera virtual.

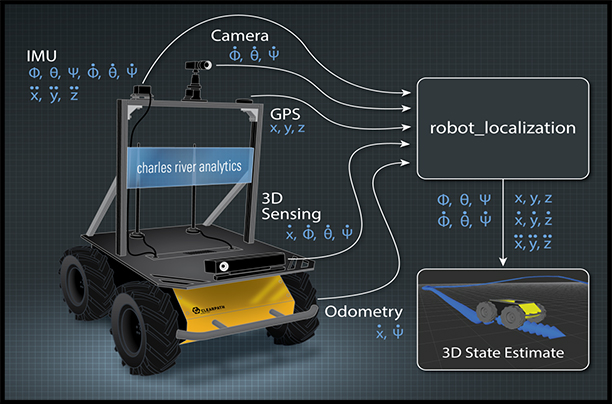


Fig.2.3 - Nodo robot\_localization recibiendo tópicos de un robot físico y mostrando su localización en el ambiente virtual. Copyright 2017 por Charles River Analytics Inc.

Un nodo podemos considerarlo como un ejecutable dentro del paquete de ROS y utiliza la librería cliente, lista de nodos disponibles de ROS, para comunicarse con otros nodos, y estos a su vez se les configura para publicar o subscribirse a un tópico, o algunos procesos de ROS tales como servicios o almacenamiento de datos (Bags).

* + 1. **Librería****s de ROS**

El sistema operativo para robots (ROS) presenta una serie de características que lo hacen atractivo, para las personas dedicadas a la robótica, a la hora de diseñar e implementar sistemas basados en robots. Una de las características principales es que puede acceder a una amplia colección de librerías las cuales varían de acuerdo con la distribución de ROS (turtle, kinetic, etc.) con la que se desee trabajar. Las librerías de las distribuciones más recientes cuentan con librerías con códigos más estables e incorporan nuevas librerías probadas y recomendadas por la comunidad ROS.

Para implementar un nodo en ROS se pueden utilizar diferentes librerías tales como roscpp para C++ o rospy para Phyton.

ROS dispone de las librerías necesarias para la implementación de las actividades que deben ser desarrolladas en un sistema robótico tales como el intercambio de mensajes entre nodos o la estructuración de datos para descripción de trayectorias.

En el desarrollo del trabajo monográfico se utilizó la distribución (versión) Kinetic dado que es una de las más estables hasta la fecha.

* + 1. **Herramientas** **de visualización de datos en ROS**

Un elemento importante en cualquier laboratorio virtual de robótica es la posibilidad de visualizar el comportamiento del modelo de un robot en un escenario dado. También es fundamental que el mismo cuente con los recursos necesarios para facilitar el análisis de los datos generados por el sistema robótico bajo estudio. El sistema operativo para robots (ROS) dispone de una serie de herramientas para la visualización de los mensajes de los sistemas robóticos ya sean estos físicos o virtuales. Entre el sinnúmero de herramientas disponibles mencionamos RVIZ, RQT, y MoveIt, nodos en el sistema de suscripción de ROS con sus tópicos y servicios para permitir la entrada y salida de los datos a procesar. Las mismas herramientas mencionadas son utilizadas en el desarrollo del laboratorio virtual propuesto.

1. **RVIZ**: es una herramienta de visualización 3D que permite combinar en una misma pantalla modelos de robots, datos de sensores (cámara, láser, etc.) y otros datos provenientes de mensajes 3D. RVIZ es utilizado en el desarrollo del laboratorio virtual principalmente para visualizar la morfología y estudiar la cinemática de los robots. La figura 2.4 muestra una imagen generada utilizando RVIZ.

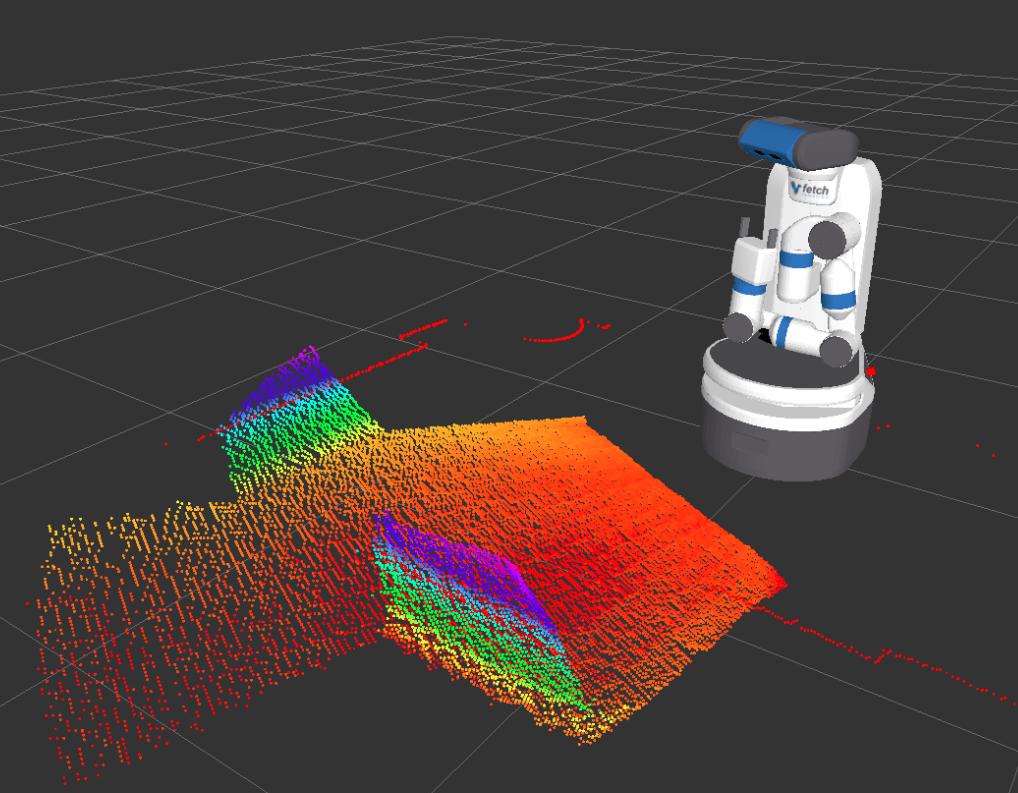


Fig.2.4. – RVIZ y visualización de modelo de robot y datos provenientes de mensajes 3D.

1. **RQT**: La herramienta rqt ofrece varios plugins, entre los cuales destacan ‘rqt\_image\_view’, ‘rqt\_graph’, ‘rqt\_plot’ y ‘rqt\_bag’, los cuales pueden ser utilizados para la introspección y visualización de datos provenientes de procesos de ROS. Por ejemplo, si es necesario visualizar los mensajes provenientes de una cámara en un sistema robótico, podríamos utilizar el plugin rqt-image\_view. rqt expone los nodos, y las conexiones entre ellos, lo que permite entender la estructura el sistema, su funcionamiento, así como su fácil depuración. En la figura 2.5 se muestra el resultado de utilizar rqt\_plot para visualizar los datos provenientes de un nodo.

En el laboratorio virtual de robótica RQT se utilizó para graficar la estructura de datos correspondiente a la posición del robot durante su movimiento.

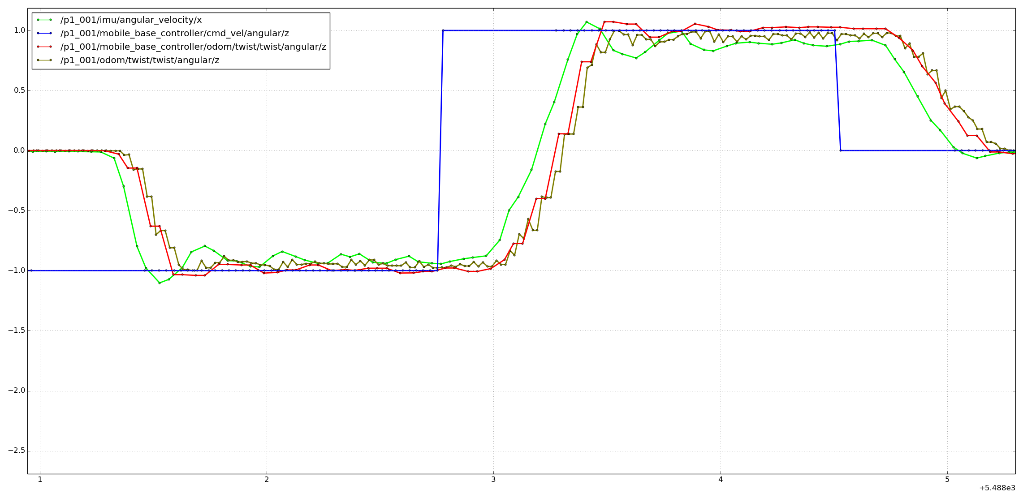


Fig.2.5. - Gráfico de línea XY de datos provenientes de nodos.

1. **MoveIt!:** Es una herramienta de software, escrita en C++, que permite la planeación de trayectorias para un robot industrial, así como la visualización de los datos y del modelo 3D del mismo, valiéndose de un plugin que es incorporado a la aplicación de visualización de datos RVIZ. Actualmente, MoveIt (s. f.) es utilizado en diversas industrias para hacer la planeación de las trayectorias para diferentes tipos de manipuladores industriales, adoptando la visión de ROS de facilitar la programación de diferentes tipos de robots con los mismos recursos de Software.

Al sistema en desarrollo se le puede suministrar mayor funcionalidad al utilizar ROS ya que el mismo permite la integración y utiliza la información generada por aplicaciones externas de código abierto muy populares tales como GAZEBO y OpenCV. Por ejemplo, es posible utilizar OpenCV para hacer reconocimiento de imágenes y tomar decisiones, a partir de los datos suministrados. Más adelante en el documento se presentarán las características de GAZEBO dado que es el simulador que se incorporo en el LVR en la practicas de programación del robot.

* + 1. **Lenguajes de pro****gramación soportados por ROS**

En el desarrollo del software para una aplicación de robótica, ROS permite el uso de distintos lenguajes de programación. De manera oficial soporta Python, C++ y LISP. Java podría ser soportado en el futuro, pero en la actualidad se encuentran en una fase experimental, apoyada por Google. De igual forma C# se encuentra en una fase experimental utilizando UNITY 3D, el cual es un motor de video juegos multiplataforma que permite simular robots importando el URDF como un GameObject en UNITY3D y obtener de este una simulación más realista con el renderizado de UNITY3D.

Con ROS es posible utilizar nodos creados en diferentes lenguajes de programación tales como Phyton y C++ y garantizar el intercambio de información entre estos.

En el desarrollo de la GUI y el intercambio de información entre los nodos ROS se utilizará el lenguaje C++ y el compilador CMake el cual fue diseñado para construir, probar y empaquetar software. CMake es el compilador oficial soportado por la comunidad ROS y el mismo desplazó al anterior compilador llamado rosbuild.

El ambiente integrado de desarrollo Qt-Creator posee un plugin para ROS (ROS Industrial, 2016) que cuenta con los recursos necesarios (por ejemplo, templates) para crear nodos, servicios, y archivos URDF, entre otros. Qt-Creator permite al desarrollador construir los recursos de software necesarios, desde el IDE, para un sistema robótico.

* 1. **Laboratorio virtual** **para robótica industrial**

En una institución de educación superior, que dispone de un laboratorio físico para la enseñanza de la robótica, el robot es parte de un escenario que simula un ambiente industrial y las prácticas generalmente están relacionadas con la programación del robot para que realice una tarea determinada. El sistema no presenta la flexibilidad necesaria para que los estudiantes mejoren su comprensión acerca de los fundamentos de la robótica industrial. De hecho, aprenden a programar el robot utilizando el lenguaje específicamente diseñado para este.

El laboratorio virtual para robótica que se ha desarrollado contempla la incorporación de los recursos necesarios para que los estudiantes estudien o realicen experimentos relacionados con temas tales como la morfología del robot, la cinemática del robot, la programación de robots de diferentes fabricantes independientemente del lenguaje especifico desarrollado para cada uno. Se incorporarán recursos que permitirán al estudiante entender las herramientas matemáticas utilizadas en el campo de la robótica industrial, específicamente manipuladores industriales tipo serie.

En el desarrollo de software es común el uso de los términos Backend y Frontend los cuales aportan una perspectiva de la clasificación de tipo de Software a utilizar en la construcción del propio laboratorio virtual, se uso ROS para manejo de datos en Backend y el uso de librerías de software libre para desarrollar GUI del lado de Frontend.

En el LVR el estudiante, para la realización de los experimentos, puede utilizar modelos de robots de fabricantes tales como ABB o KUKA. De igual forma, cuenta con las herramientas y la guía para desarrollar sus modelos propios.

Los principales elementos del laboratorio virtual son:

* Interfaz gráfica de usuario (GUI) para (interacción, programación, visualización, etc.)
* Modelos de robots industriales de ABB, KUKA, y DELTA
* Escenarios para la realización de tarea específicas
* Documentación sobre aspectos fundamentales sobre robótica industrial
* Otros

La interacción entre los diferentes elementos del laboratorio virtual es garantizada mediante el middleware ROS.

A continuación, son presentados algunos de los conocimientos o recursos tecnológicos requeridos para el desarrollo del LVR, específicamente para la implementación de la interfaz gráfica de usuario (GUI) y la construcción de algunos recursos para el estudio de los fundamentos de los manipuladores industriales.

* + 1. **Interfaz gráfica de usuari****o (GUI)**

Una interfaz gráfica de usuario (GUI, por sus siglas en inglés) facilita la explotación de una aplicación mediante la incorporación de elementos destinados para tal fin. La GUI debe comportarse de una manera comprensible y predecible, de modo que un usuario sepa qué esperar cuando realiza una acción. La interfaz gráfica de usuario del laboratorio virtual cuenta con los elementos necesarios (botones, sliders, editor de texto, pantalla de visualización, etc.) para realizar las prácticas sobre los fundamentos de la robótica industrial, así como para acceder a documentos de interés tales como libros, guías de laboratorio, entre otros. La GUI está diseñada de tal forma que el estudiante podrá usar los recursos de manera transparente sin necesidad de tener que configurar aspectos básicos de ROS tales como su inicialización y el lanzamiento de nodos.

En la actualidad existen muchos lenguajes que pueden ser utilizados para desarrollar una interfaz de usuario y entre ellos destacan C++, C#, Java, Phyton, Visual Basic. Un criterio para utilizar un lenguaje en particular es que el mismo sea multi-plataforma ya que la interfaz podría ser utilizada bajo diferentes sistemas operativos.

Hoy día están disponibles un sinnúmero de ambientes integrados de desarrollo (IDE, por sus siglas en inglés) que facilitan la programación ya que aglomeran todas las herramientas necesarias tales como conjuntos de librerías, preprocesador, depurador, y compiladores. Entre los IDEs más conocidos destacan, entre otros, Visual Studio, Eclipse, NetBeans, CLion, y Qt Creator. Para la construcción de la GUI del laboratorio virtual de robótica industrial se utilizó el IDE Qt Creator, debido, principalmente, a que este posee un plugin de soporte para ROS.

Qt Creator se utilizó para crear las aplicaciones o nodos de ROS y se exploro las capacidades que posee QT Creator para la creación de aplicaciones de escritorio. Qt Creator cuenta con QML, un estándar propio del IDE, que permite crear aplicaciones dinámicas y más elaboradas capaces de incorporar recursos WEB. Esa línea de creación se utilizó para incorporar un lector de PDF con el objetivo que los estudiantes dispongan en el LVR de documentación de interés tales como libros o guías de laboratorio.

En el desarrollo de la GUI del LVR se utilizaron los widgets de Qt Creator (slider, botones, etc.) los cuales están escritos en C++. Esta línea de creación es muy importante para lograr la visualización en el LVR ya que es muy fácil incorporar las librerías de ROS, escritas en C++, a una aplicación utilizando widgets. Un ejemplo es el desarrollo de una aplicación de escritorio personalizada que se acople a los requerimientos del LVR al incorporar las librerías de RVIZ para visualizar los modelos de los robots.

Es importante mencionar que en el laboratorio virtual se realizan múltiples procesos en el Backend, con el objetivo de garantizar la transparencia en el uso de los sus recursos por los usuarios, que darán inicio a la ejecución de las interacciones en la interfaz de usuario tales como la visualización de los robots y datos, programación y simulaciones de los robots. Dichos procesos están basados en nodos que son lanzados o eliminados de acuerdo con la lógica implementada en el propio LVR.

* + 1. **Fundamentos de** **la robótica industrial.**

El laboratorio de robótica virtual que se desarrollo tiene un alcance limitado ya que de los temas considerados en un curso básico de robótica industrial el mismo solo contempla recursos para la experimentación en temas como herramientas matemáticas, morfología del robot, cinemática y programación de robots. El trabajo pretende mostrar la potencialidad del sistema operativo para robots (ROS) y lo hace poniendo en marcha el diseño, la programación y el uso del LVR. La información suministrada acerca de ROS y el laboratorio propuesto deben servir de base para ampliar los recursos de este.

* + - 1. **Morfología** **del robot**

El laboratorio virtual de robótica cuenta con modelos creados utilizando URDF y los mismos podrán ser visualizados con la herramienta RVIZ. El estudiante puede visualizar el espacio de trabajo, para una morfología dada, con la ayuda de plugins desarrollados para la visualización. Interactuar con diferentes tipos de articulaciones ya sean prismáticas o de revolución y robots de diferentes grados de libertad. El estudiante tendrá la posibilidad de experimentar con modelos construidos por él mismo utilizando el formato URDF.

**RVIZ** es una abreviación para ROS visualización y es una herramienta poderosa para la visualización en 3D. Permite al usuario ver el modelo simulado del robot, obtener información de los sensores del robot, y reproducir la información obtenida de los sensores. Mediante la visualización de lo que ve y hace el robot el usuario puede eliminar los errores de la aplicación robótica desde entradas de sensores hasta acciones planificadas o no planificadas.

* + - 1. **Herramientas m****atemáticas**

El modelado cinemático de un robot busca las relaciones entre las variables articulares y la posición (expresada normalmente en forma de coordenadas cartesianas) y orientación del extremo del robot (expresada como matrices de rotación, ángulos de Euler o algún otro de los métodos establecidos para tal fin.

Entender las herramientas matemáticas utilizadas para determinar la localización espacial de los diferentes elementos de un robot industrial, especialmente del efector final, presenta cierta dificultad a los estudiantes, principalmente cuando se utiliza solamente la pizarra para su explicación. Para contribuir al entendimiento y aplicación de dichas herramientas el laboratorio virtual de robótica incorpora los recursos necesarios para que los estudiantes puedan analizar la posición de un objeto utilizando coordenadas esféricas y cilíndricas, así como la orientación de este mediante la matriz de rotación, los ángulos de Euler, y los cuaternios. Las operaciones requeridas para el desempeño de los recursos estarán soportadas por la librería TF2, disponible para C++ y Phyton, propia de la distribución de ROS, que permite hacer conversiones de sistemas de coordenadas.

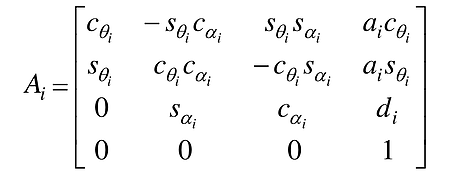
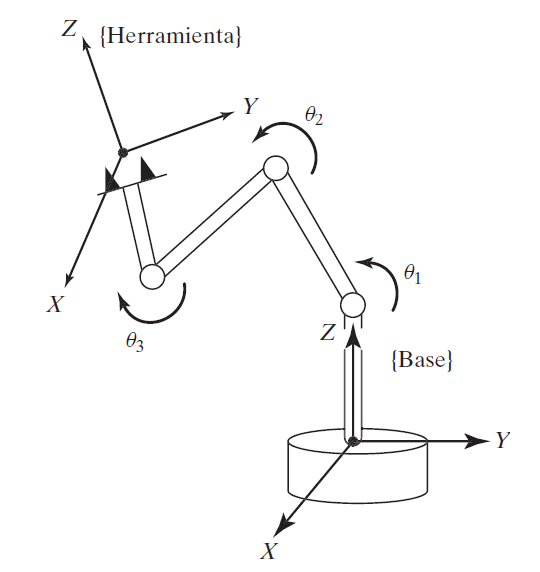


Fig.2.6. – Modelo de robot, sistemas de referencia.

* + - 1. **Cinemática directa e i****nversa**

La cinemática del robot estudia el movimiento de este con respecto a un sistema de referencia sin considerar las fuerzas que intervienen. Así, la cinemática se interesa por la descripción analítica del movimiento espacial del robot como una función del tiempo, y en particular por las relaciones entre la posición y la orientación del extremo final del robot con los valores que toman sus coordenadas articulares. Dentro de la cinemática se estudian la posición, velocidad, aceleración y todas las derivadas de mayor orden de las variables de posición (respecto al tiempo o a cualquier otra variable).

En muchas ocasiones es necesario conocer en cuál posición se encontrará el efector final del robot dadas ciertas posiciones de las articulaciones y dicho problema es abordado por la cinemática directa. En otras ocasiones es necesario conocer cuál debe ser la posición de las articulaciones para que el efecto sea ubicado en una posición deseada. El último problema es abordado por la cinemática inversa. La figura 2.7 muestra la relación entre las variables para cada uno de los tipos de cinemática.



Fig.2.7. – Cinemática directa e inversa.

El laboratorio virtual de robótica que se propone cuenta con recursos que permitirán al estudiante realizar experimentos sencillos que le ayudarán a comprender las características de la cinemática directa e inversa.

A partir del modelo de un robot, construido utilizando el formato URDF, es posible obtener toda la información necesaria de la estructura de este utilizando la librería KDL. La información obtenida al utilizar el KDL en combinación con la información suministrada por el usuario (por ejemplo, los ángulos de los joints) mediante ciertos widgets de la GUI permiten manipular y/o determinar la posición del robot.

La librería KDL también puede ser utilizada para determinar la matriz de transformación homogénea la cual es utilizada para determinar la posición y rotación del efector final del robot. Los datos obtenidos serán visualizados mediante widgets del GUI desarrollados para la representación de datos.

* + - 1. **Programación y simulación d****el robot**

En la robótica, en general, uno de los más aspectos de mayor importancia es el relacionado con la programación de los robots. Durante la ejecución de un programa se interacciona con la memoria del sistema, leyendo y actualizando el contenido de las variables utilizadas en el programa; con el sistema de control cinemático y dinámico del robot, encargados de dar la señal de mando a los accionamientos del robot a partir de las especificaciones del movimiento que se les proporciona; y con las entradas-salidas del sistema, consiguiéndose así la sincronización del robot con el resto de las máquinas y elementos que componen su entorno.

El sistema de programación es, por tanto, la herramienta con que cuenta el usuario para acceder a las diversas prestaciones del robot, existiendo una relación directa entre las características y posibilidades del sistema de programación y las del robot en sí mismo.

La programación puede ser guiada o textual. Es importante destacar que en la actualidad es muy frecuente que los sistemas de programación de robots tiendan a combinar los dos modos básicos (guiado y textual), permitiéndose desarrollar el programa mediante la escritura de las instrucciones, y utilizando la posibilidad de guiado en línea en aquellos momentos en que sea necesario. Sistemas como RAPID de ABB, VAL II de Staübli y V+ de Adept Technology son ejemplos de esta ambivalencia.

Uno de los problemas relacionados con la programación de los robots es que no existe un estándar y el programador debe aprender el lenguaje específico para los robots de diferentes fabricantes. ROS brinda la posibilidad de programar robot de diferentes fabricantes desde el mismo entorno.

Supongamos que es necesario que dos robots de diferentes fabricantes, pero la misma morfología, deben seguir la misma trayectoria. En el ambiente de ROS, es posible escribir mensajes de ROS (que contiene la posición, velocidad, aceleración, etc., de cada articulación del robot), escritos en C++ o Python. El driver de cada robot, una vez recibido el mensaje, generará los comandos para que su robot siga la trayectoria indicada.

El laboratorio virtual de robótica permite al usuario escribir un programa (script, programación textual) que adoptará ciertas convenciones, utilizadas en el software de programación de robots de ABB Rapid, para que uno de los robots (modelos disponibles en el LVR) realice una tarea sencilla.

Una forma mediante la cual podemos verificar si el robot realiza la tarea indicada es la simulación. En la actualidad existen muchos simuladores, sencillos y complejos, para robots. También podemos clasificarlos en comerciales y de código libre.

Entre los comerciales destaca V-Rep el cual es un simulador dinámico con un entorno de desarrollo integrado que permite desde su interfaz lenguajes de programación tales como C, C++, Phyton, entre otros.

**V-Rep** es un simulador con entorno de desarrollo integrado donde cada objeto/modelo se puede controlar individualmente a través de un script de programación que puede ser escrito usando lenguajes de programación como C/C++, Python, Java, Lua, Matlab or Octave, V-REP se utiliza para el desarrollo rápido de algoritmos, simulaciones de automatización de fábricas, prototipado rápido y verificación y educación relacionada con la robótica.

De los simuladores open-source uno de los más utilizados es GAZEBO, el cual provee simulaciones dinámicas donde los robots pueden interactuar con el entorno (pueden coger, empujar, rodar, deslizarse por el suelo), capacidad de manipular las características del entorno tal como la gravedad del mundo virtual.

**Gazebo** es simulador dinámico 3D que tiene la habilidad para simular poblaciones de robots, de forma exacta y eficiente, en ambientes complejos internos y externos. Aunque similar a máquinas de juego, Gazebo ofrece una simulación física a un grado de fidelidad mayor, un conjunto de sensores, e interfaces tanto para programas como para los usuarios. Usos típicos de Gazebo incluyen:

* Prueba de algoritmos robóticos
* Diseño de robots
* Realización de pruebas de regresión con escenarios realísticos.

Algunas características claves de Gazebo:

* Máquinas físicas múltiples
* Una librería de modelos de robots y ambientes.
* Una amplia variedad de sensores

Aunque ambos simuladores tienen la posibilidad de comunicarse utilizando ROS se ha seleccionado GAZEBO como el simulador a incorporar en el LVR debido a que este es open-source. El estudiante podrá observar en el simulador GAZEBO el comportamiento del modelo del robot en respuesta al programa (script) escrito por el usuario. El script incorporará el conjunto de instrucciones las cuales serán procesadas por un nodo el cual enviará a GAZEBO, utilizando el pluging de conexión correspondiente, los mensajes apropiados para lograr el movimiento de las articulaciones del robot en el mundo.

**SDF** es un formato XML utilizado para describir objetos y ambientes para simuladores de robots, visualización, y control. SDF tiene capacidad para describir todos los aspectos de un robot, objetos estáticos y dinámicos, iluminación, terreno y también la física. En el laboratorio virtual el lenguaje SDF se utiliza para modelar los escenarios de trabajo en el ambiente de Gazebo.

Podrán ser utilizados los siguientes parámetros:

* **Escena**: Iluminación ambiental, propiedades del cielo, sombras.
* **Física**: Gravedad, paso del tiempo, motor de la física.
* **Modelos**: Colección de enlaces, objetos de colisión, articulaciones y sensores.
* **Luces**: Punto, espacio y fuentes de luz direccionales.
* **Plugins**: plugins soportado por gazebo, plugins del mundo, del modelo, del sensor y del sistema.
  1. **Uso del laboratorio virtual de robótica industrial**

Los diferentes elementos involucrados en el uso, y posible ampliación del laboratorio virtual, son mostrados en la figura 2.8. El laboratorio virtual incluye un manual de usuario cuyo objetivo es presentar al usuario los recursos disponibles y un par de guías de laboratorio para mostrar cómo utilizarlos. Como se aprecia en la figura 2.7, el docente podrá, partiendo de los temas relacionados con los fundamentos de la robótica industrial que desee desarrollar y de los recursos disponibles en el laboratorio, elaborar las guías que utilizarán los estudiantes para realizar las actividades de estudio o las prácticas de laboratorio correspondientes.

Se pretende, como parte del trabajo monográfico, desarrollar un taller, para los docentes interesados, con el objetivo de brindarles los detalles de diseño y herramientas utilizadas en la implementación del LVR para que tengan una base para la ampliación de los recursos de este.

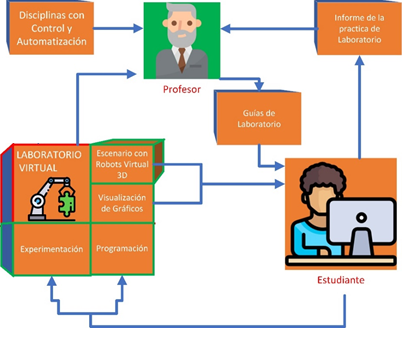


Fig.2.8. – Modelo de utilización del LVR.

**CAPÍTULO II: DISEÑO E IMPLEMENTACIÓN DEL LABORATORIO VIRTUAL DE ROBÓTICA INSDUSTRIAL**

* RVIZ es utilizado en el desarrollo del laboratorio virtual principalmente para visualizar la morfología y estudiar la cinemática de los robots.
* En el laboratorio virtual de robótica RQT se utilizó para graficar la estructura de datos correspondiente a la posición del robot durante su movimiento.
* el documento se presentarán las características de GAZEBO dado que es el simulador que se incorporo en el LVR en la practicas de programación del robot.
* De igual forma, cuenta con las herramientas y la guía para desarrollar sus modelos propios.
* La GUI está diseñada de tal forma que el estudiante podrá usar los recursos de manera transparente sin necesidad de tener que configurar aspectos básicos de ROS tales como su inicialización y el lanzamiento de nodos.

**CAPÍTULO III: CONCLUSIONES Y RECOMENDACIONES**

**CONCLUSIONES**

A continuación, son presentadas conclusiones sobre los objetivos específicos establecidos para lograr el objetivo general.

* .
* .
* .
* .
* .
* .
* .
* .

**RECOMENDACIONES**

A continuación, se presentan algunas recomendaciones que pueden contribuir a mejorar las prestaciones del prototipo desarrollado en este proyecto.

* .
* .
* .

**ANEXOS**