

**TECNOLOGICO NACIONAL DE MEXICO  
INSTITUTO TECNOLÓGICO DE APIZACO**

**TESIS**

**CONTROL DE UN ROBOT INDUSTRIAL IRB 2600 + IRC5  
POR MEDIO DE UN SOFTWARE PROPIO**

**QUE PARA OBTENER EL TÍTULO DE:  
MAESTRA EN SISTEMAS COMPUTACIONALES**

**PRESENTA:  
LIC. YAJAIRA GUADALUPE LÁZARO ARVIZU**

**ASESOR DE TESIS:  
DR. ROBERTO MORALES CAPORAL**

## **AGRADECIMIENTOS**

*Antes y primero que nada quiero agradecer a mi familia, porque sin su apoyo y comprensión no habría concluido esta etapa de mi educación profesional de manera satisfactoria.*

*A mis padres Carmen Arvizu Herrera y Bernabé Lázaro Castellanos por siempre creer en mí y brindarme su apoyo incondicional, por alentarme a siempre ser una mejor persona y perseverar para cumplir con mis más grandes deseos.*

*A mi director de tesis el Dr. Roberto Morales Caporal por brindarme su conocimiento durante mis estudios de maestría, para que de esta manera pudiera prepararme y crecer como profesional, pero sobre todo por creer en mi capacidad.*

## **RESUMEN**

Hoy en día los robots han llegado a ser una herramienta importante por la precisión que necesita la industria moderna al realizar sus actividades productivas, es por ello que en la actualidad la programación de robots se puede hacer mediante el uso de herramientas de simulación que reproducen la dinámica del robot, tanto para capacitar al personal que lo utiliza previo a su operación, así como para eliminar movimientos erróneos antes de su implementación en el robot real.

En esta tesis se presenta una herramienta computacional basada en un Toolbox de MATLAB®/SIMULINK®, capaz de simular y después implementar en un robot real los movimientos requeridos. La herramienta computacional ha sido desarrollada con fines didácticos, y permite diseñar y simular las trayectorias del robot industrial (KUKA) IRB-2600 en un entorno visual, previo a su implementación, de modo que los estudiantes aprenden a diseñar y controlar los movimientos repetitivos del robot y después observarlos en un entorno gráfico 3D. Con esta herramienta se evita implementar movimientos peligrosos, puede utilizarse para capacitar a personal o enseñar a los alumnos (laboratorio virtual) sin necesidad de tener el robot real.

## **ABSTRAC**

Today, robots have been becoming an important tool for the precision that the modern industry need to make its production activities, Currently the robot programming can be done by using simulation tools that reproduce the robot dynamics, and it can be used either to train personnel who uses it prior to the real operation and to check and eliminate erroneous movements before implementation in the real robot.

In this thesis, a computational tool based on MATLAB® Toolbox / Simulink®, which is able to simulate and then implement in a real robot movements is introduced. The computational tool has been developed for teaching purposes, and allows the design and simulations of the trajectories for an industrial robot (KUKA) IRB-2600 by using a visual environment, prior to implementation, so that students learn to design and control repetitive movements of the robot and then can observe them in a 3D graphical environment. With this the proposed computational tool prevents dangerous movements before implementation, and it can be used to train staff or students (virtual laboratory) without the need of having the real robot.

## INDICE

<b>CAPÍTULO I: INTRODUCCIÓN</b> .....	<b>10</b>
1.1 Introducción .....	10
1.2 Planteamiento del problema .....	11
1.3 Justificación.....	11
1.4 Hipótesis .....	12
1.5 Objetivo general.....	12
1.5.1 Objetivos específicos.....	13
1.6 Organización de la tesis.....	13
1.7 Estado del arte .....	14
<b>CAPÍTULO II: FUNDAMENTOS DE ROBÓTICA</b> .....	<b>23</b>
2.1 Parámetros para clasificar a los robots .....	23
2.4.3 Configuración de coordenadas cartesianas.....	31
2.4.4 Configuración angular .....	31
2.5 Orientación del efector final .....	32
2.5.1 Posición y orientación en el espacio .....	32
2.6 Posición y orientación en el espacio 3d .....	35
2.6.1 Posición en 3d.....	35
2.6.2 Orientación.....	35
2.7 Cambios de base .....	37
2.8 Análisis de transformaciones.....	38
2.9 Sistemas no lineales e incrementales en el tiempo. ....	39
2.10 Tipos de sensores de un robot.....	40
2.10.1 Sensores de posición .....	41
2.10.1.1 Codificadores angulares de posición (encoders).....	41
2.10.1.2 Captadores angulares de posición (sincro-resolvers) .....	42
2.10.1.3 Sensores lineales de posición (LVDT e Inductosyn).....	44
2.10.2 Sensores de velocidad .....	44

2.10.3 Sensor de presencia y proximidad. ....	45
2.11 Estructura de robots manipuladores .....	46
2.12 Cinemática de los manipuladores .....	46
2.12.1 Cinemática directa.....	48
2.12.2 Resolución del problema cinemático directo mediante matrices de transformación homogénea .....	48
2.12.3 Algoritmo de Denavit- Hartenberg para la obtención del modelo cinemático directo. ....	51
2.13 Cinemática inversa. ....	56
<b>CAPÍTULO III: BRAZO ROBOT INDUSTRIAL ANTROPOMÓRFICO KUKA KR 16</b> .....	<b>58</b>
3.1 Introducción. ....	58
3.2 Arquitectura general del sistema robótico .....	58
3.2.1 Estructura mecánica.....	61
3.2.1.1 Muñeca .....	63
3.2.1.2 Brazo .....	64
3.2.1.3 Columna giratoria .....	65
3.3 Elementos terminales .....	67
3.4 Operación .....	68
3.4.1 Sistema de control.....	68
3.4.1.1 Cerradura del armario.....	72
3.4.1.2 Arranque de la unidad de control .....	72
3.4.1.3 Parada de la unidad de control .....	74
3.5 Unidad manual de programación kuka smartpad .....	75
3.5.1. Lado frontal .....	75
3.6 Interfaz de usuario kuka smarhmi .....	78
3.6.1 Barra de estado .....	81
3.7 Puesta en operación .....	82
3.8 Sistemas de coordenadas.....	84
3.8.1 Word .....	84

3.8.2 Robroot.....	85
3.8.3 Base .....	85
3.8.4 Tool .....	85
<b>CAPITULO IV. DESARROLLO DE LA APLICACIÓN.....</b>	<b>86</b>
<b>4.1 Simulación digital de robots industriales .....</b>	<b>86</b>
4.1.1 Kuka.sim pro.....	87
4.1.2 Kuka.officelite .....	87
4.1.3 ABB de robot studio .....	88
4.1.2 Entornos de simulación basados en Matlab .....	91
<b>4.2 Robotic toolbox de matlab .....</b>	<b>93</b>
<b>4.3 Descripción de la aplicación .....</b>	<b>94</b>
<b>4.4 Sistemas robotizados en matlab® / simulink® .....</b>	<b>96</b>
<b>4.5 Comunicación de la aplicación propuesta con el robot real .....</b>	<b>97</b>
<b>4.6 Descripción de las aplicaciones.....</b>	<b>99</b>
4.6.1. Aplicación grafica de simulación.....	99
4.6.2. Aplicación grafica (GUIDE®) para el movimiento del robot.....	103
<b>4.7 Primeras pruebas para la aplicación de soldadura .....</b>	<b>104</b>
<b>CAPITULO V. CONCLUSIONES, RECOMENDACIONES Y TRABAJOS FUTUROS.</b>	<b>107</b>
.....	107
<b>5.1 CONCLUSIONES .....</b>	<b>107</b>
5.1.1 Capitulo 1 .....	107
5.1.2 Capítulo 2 .....	107
5.1.3 Capítulo 3 .....	108
5.1.4 Capítulo 4 .....	108
<b>5.2 RECOMENDACIONES Y TRABAJOS FUTUROS.....</b>	<b>109</b>
<b>5.3 ANEXOS .....</b>	<b>110</b>
<b>ANEXO A).-PRÁCTICAS DE LABORATORIO.....</b>	<b>110</b>
<b>ANEXO B) ARTÍCULOS PUBLICADOS.....</b>	<b>135</b>
<b>ANEXO C) RECONOCIMIENTOS.....</b>	<b>157</b>
<b>REFERENCIAS BIBLIOGRAFICAS .....</b>	<b>158</b>

## INDICE DE FIGURAS

Fig. 1.1 Primer autómeta .....	14
Fig. 1.2 Pájaro mecánico de vapor .....	16
Fig. 1.3 Telar mecánico .....	16
Fig. 1.4 Robots en la industria .....	20
Fig. 1.5 Cyborg.....	22
Fig. 2.1 Ejemplo de robot play-back. ....	25
Fig. 2.2 Ejemplo de robot controlado por sensores. ....	25
Fig. 2.3 Ejemplo de robot controlado por visión. ....	26
Fig. 2.4 Ejemplo de robot controlado adaptablemente. ....	26
Fig. 2.5 Ejemplo de robot con inteligencia artificial. ....	27
Fig. 2.6 Ejemplo de robot utilizando en la medicina. ....	27
Fig. 2.7 Ejemplo de robot androide. ....	27
Fig. 2.8 Ejemplo de robot móvil. ....	28
Fig. 2.9 Área de trabajo de la configuración polar. ....	30
Fig. 2.10 Área de trabajo de la configuración cilíndrica. ....	30
Fig. 2.11 Área de trabajo de la configuración cartesiana. ....	31
Fig. 2.12 Sistema de vectores primarios. ....	32
Fig. 2.13 Sistema de coordenadas polares. ....	33
Fig. 2.14 Vectores de dirección coincidentes. ....	34
Fig. 2.15 Vector de posición en un sistema {A}. ....	36
Fig. 2.16 Coordenadas cilíndricas. ....	37
Fig. 2.11 Disposición de un codificador óptico (encoder) incremental ....	42
Fig. 2.12 Esquema de funcionamiento de un resolvers ....	42
Fig. 2.13 Esquema de funcionamiento de un LVDT ....	44
Fig. 2.14 Cadena cinemática abierta ....	46
Fig. 2.15 Sistema de coordenadas de referencia fija ....	47
Fig. 2.16 parámetros DH para un eslabón giratorio ....	53
Fig. 2.17 Robot cilíndrico ....	55
Fig. 3.1 Arquitectura básica de un sistema robótico ....	60
Fig. 3.2 Grados de libertad de IRB2600 ....	61
Fig. 3.3 Medidas principales y campo de trabajo KR 16-2S (referidas al software) ....	63
Fig. 3.3 Muñeca central del robot ....	64
Fig. 3.4 Brazo del robot ....	65
Fig. 3.5 Columna giratoria con rango de giro ....	66



Fig. 3.6 Tipos de elementos terminales .....	67
Fig. 3.7 Sistema de control del robot .....	69
Fig. 3.8 CPU integrado al armario de control .....	70
Fig. 3.9 Elementos de seguridad y comunicación .....	70
Fig. 3.10 Controladores de grados de libertad .....	71
Fig. 3.11 Vista frontal del armario de control .....	72
Fig. 3.12 Interruptor principal del armario de control .....	73
Fig. 3.13 Arranque del software .....	73
Fig. 3.14 Superficie de usuario .....	74
Fig. 3.15 Interruptor del armario de control .....	74
Fig. 3.16 KUKA smartPAD, lado delantero .....	75
Fig. 3.17 KUKA smartPAD, parte trasera .....	77
Fig. 3.18 Interfaz de usuario kuka smarthmi .....	79
Fig. 3.19 Barra de estado .....	81
Fig. 3.20 KUKA smartPAD, sistema de coordenadas .....	84
Fig. 4.1 Interfaz de RobotStudio .....	89
Fig. 4.2 Robot IRB-2600 en la plataforma de RobotStudio	91
Fig. 4.3 Entorno de simulación de “ARTE” de Matlab .....	93
Fig. 4.5 Robot simulado en 3D .....	100
Fig. 4.6 Posición inicial del robot simulado .....	100
Fig. 4.7 Posición inicial de robot real .....	101
Fig. 4.8 Posición final del robot simulado .....	102
Fig. 4.9 Posición final del robot real .....	102
Fig. 4.10 Pantalla inicial de la aplicación grafica .....	104

## INDICE DE TABLAS

Tabla 2.1. Parámetros DH para el robot .....	55
Tabla 3.1. Especificaciones mecánicas del robot. ....	62
Tabla 3.2. Descripción de las funciones del lado frontal del SmartPAD. ....	77
Tabla 3.3. Descripción de las funciones de parte trasera del smartPAD .....	78
Tabla 3.5. Descripción de las funciones de la interfaz de usuario KUKA Smarthmi. ....	81
Tabla 3.6. Descripción de las funciones de la barra de estado. ....	82
Tabla 3.7. Descripción de modo de servicio. ....	84

# CAPÍTULO I: INTRODUCCIÓN

---

## 1.1 Introducción

La robótica es la rama de la tecnología que se dedica al diseño, construcción, operación, disposición estructural, manufactura y aplicación de los robots.

Los robots son máquinas en las que se integran componentes mecánicos, eléctricos, electrónicos, de comunicación y dotados de un sistema informático (software) para su control en tiempo real, percepción del entorno, toma de decisiones y control.

Hoy en día pueden encontrarse todo tipo de robots en cualquier lugar y en campos de lo más variado, desde los robots industriales y militares, hasta robots agrícolas, médicos o de exploración submarina.

Los robots industriales están concentrados en operaciones de producción por ser adaptables al entorno, son relativamente de fácil manejo y con gran capacidad de realizar tareas repetitivas muy rápidamente y con una alta precisión. Estas características han llevado a la robótica industrial a su continuo crecimiento.

Un brazo robótico es un tipo de brazo mecánico (manipulador), normalmente programable, con funciones parecidas a las de un brazo humano; este puede ser la suma total del mecanismo o puede ser parte de un robot más complejo. Las partes de estos manipuladores o brazos son interconectadas a través de articulaciones que permiten, tanto un movimiento rotacional (tales como los de un robot articulado), como un movimiento translacional o desplazamiento lineal.

La creación de un software propio para este manipulador industrial básicamente es con la finalidad de poder programar y simular los movimientos del robot antes de que los pueda realizar por si mismo de esta manera se evitaran movimientos involuntarios, y así mismo se podrán realizar diversas prácticas de los movimientos del robot.

También se estudiarán sus aspectos físicos y características eléctricas, electrónicas pero sobre todo se hace especial énfasis en el sistema de programación que estos robots utilizan, todas las herramientas que se necesitan para estructurar sentencias de movimiento y funciones de trayectoria.

## **1.2 Planteamiento del problema**

Actualmente el Instituto Tecnológico de Apizaco cuenta con al menos cinco ingenierías que en sus programas o módulos de especialidad incluyen el estudio de la robótica o automatización (Industrial, Mecatrónica, Electromecánica, Electrónica y Sistemas Automotrices) y dos posgrados en los cuales también se lleva alguna materia de robótica y/o automatización (Maestría en Sistemas Computacionales y Maestría en Ingeniería Mecatrónica). Por lo tanto, se ha adquirido un robot industrial (manipulador), el cual puede ser utilizado en la docencia e investigación.

Para esto, se propone desarrollar e implementar un software propio, el cual sea capaz de modelar los movimientos de dicho manipulador antes de implementarlos, favoreciendo la formación sólida del estudiante sin arriesgar la estructura del robot.

## **1.3 Justificación**

Actualmente en la industria moderna es cada vez más utilizada la tecnología de modo que en las líneas de producción ya no es tan necesaria la mano de obra del ser humano, por tal razón, es recomendable que los estudiantes lleven a cabo prácticas con este tipo de tecnología para que de esta manera adquieran conocimientos y competencias sólidas en el área.

El robot a utilizar es un modelo KUKA IRB 2600, que puede tener diversas aplicaciones industriales, sin embargo, se desea acceder a su hardware y software con fines de docencia e investigación.

Es por ello que en esta tesis se desarrolla y se estudia la implementación de un software propio para el robot industrial IRB 2600, que concretamente es un brazo robótico industrial, que fue adquirido por el Instituto Tecnológico de Apizaco pero que aún no se ha puesto en operación, por lo cual el motivo de la creación de este software primordialmente se centra en la puesta en marcha de este robot, para fines educativos y de investigación.

#### **1.4 Hipótesis**

Es posible programar y simular los movimientos del manipulador robótico para aplicaciones industriales por medio de un software antes de implementarlos, para evitar movimientos imprevistos y verificar movimientos adecuados.

Con el desarrollo, planteamiento e implementación, de un software propio se obtiene la implementación digital de un control para un robot con la finalidad de que este pueda ser operado primero de manera simulada previo a la implementación de los movimientos a través de un software creado para dar solución al aprovechamiento de este robot industrial, por alumnos y docentes investigadores de la institución.

#### **1.5 Objetivo general**

Desarrollar un software propio para el control de los movimientos del robot basado en diversos periféricos orientados a la investigación.

### **1.5.1 Objetivos específicos**

- Instalar el robot.
- Traducción del manual y su estudio, con la finalidad de conocer a fondo el funcionamiento del robot.
- Familiarizarse con el control y operación del robot.
- Programar y verificar por medio de simulación los movimientos del robot.
- Verificar la conexión de la herramienta.
- Implementar los movimientos y el control en el robot utilizando su software y hardware desarrollado.
- Verificar el control del robot por medio de experimentos.

### **1.6 Organización de la tesis**

Esta tesis se divide en cinco capítulos los cuales están divididos de la siguiente manera:

En el capítulo I se habla acerca de la introducción de lo que trata este tema de tesis, la descripción del problema, la hipótesis, así como los objetivos generales y específicos, continuando con el estado del arte.

Capítulo II se hace un estudio de las características principales que deben ser tratadas antes de comenzar un proyecto enfocado a la robótica, como son tipos de robots, grados de libertad, cinemática y dinámica de los robots, tipos de articulaciones, sin mencionar otros de igual importancia.

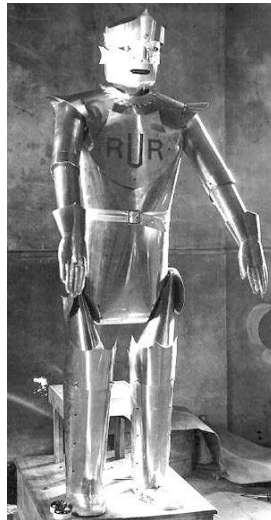
Dentro del capítulo III. Nos centramos en el estudio detallado del robot en uso, así como su gabinete de control y su smartpad los cuales funcionan en conjunto con el manipulador, también se hace un estudio de las principales fases de programación para el buen funcionamiento del robot.

En el capítulo IV se desarrollan las simulaciones de los movimientos del robot utilizando primeramente el software de robot, iniciando con los movimientos principales del robot, hasta poder simular una trayectoria completa, desde un punto inicial, hasta un punto final indicado. De igual manera se verifica la conexión del software con el robot y algún periférico seleccionado y se prueba el software creado del robot con las simulaciones de los movimientos.

En el capítulo V se finaliza con la conclusión y descripción de los resultados obtenidos durante todo el proceso de desarrollo del presente trabajo de tesis y se sugieren los posibles trabajos futuros.

## 1.7 Estado del arte

Robot es un término que proviene del vocablo checo robot, que significa servidumbre o trabajo esclavizador. Fue usado por primera vez por el dramaturgo checoslovaco Karel Capek (1890-1938) en su obra de teatro Rossum's Universal Robots (RUR).



*Fig. 1.1 Primer autómata*

El término robótica es acuñado por Isaac Asimov, definiendo a la ciencia que estudia a los robots. Asimov creó también las Tres Leyes de la Robótica

En 1985, Asimov publicó un relato en la que uno de sus robots se ve obligado a herir a un ser humano por el bien del resto de la humanidad. Surge así una nueva ley, considerada la Ley Definitiva, la llamada Ley Cero, superior a todas las demás: "Un robot no puede lastimar a la humanidad o, por falta de acción, permitir que la humanidad sufra daños". Quedando así modificada la primera ley: "Un robot no debe dañar a un ser humano, o permitir, por inacción, que un ser humano sufra daño, a menos que tal acción viole la Ley Cero".

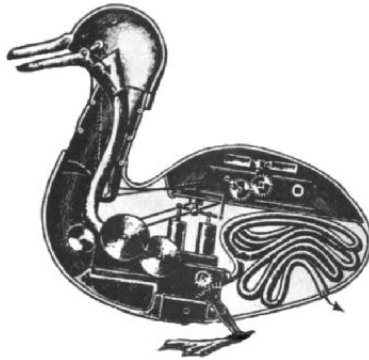
1. Un robot no puede causar daño a un ser humano ni, por omisión, permitir que un ser humano sufra daños.
2. Un robot debe obedecer las órdenes dadas por los seres humanos, salvo cuando tales órdenes entren en conflicto con la Primera Ley.
3. Un robot ha de proteger su existencia, siempre que dicha protección no entre en conflicto con la Primera o la Segunda Ley.

A lo largo de la historia el hombre se ha sentido fascinado por las máquinas y los dispositivos capaces de imitar las funciones y movimientos de los seres vivos. Los griegos tenían una palabra específica para denominar a estas máquinas: autómatas. De esta palabra deriva la actual autómatas: máquina que emita a la figura y movimientos de un ser animado.

Ejemplos de estos autómatas son el hombre de hierro de Alberto Magno (1204-1282), la cabeza parlante de Roger Bacon (1214-1294), el gallo de Estrasburgo (1352) y el león mecánico de Leonardo Da Vinci (1452-1519).

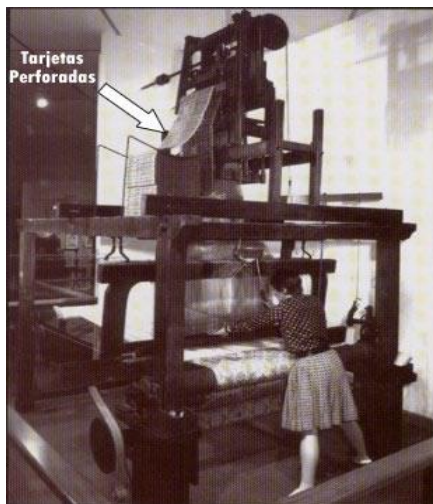
350 a.c: El Matemático griego Arquitas de Tarento construye un pájaro mecánico de Vapor.





*Fig. 1.2 Pájaro mecánico de vapor*

A finales del siglo XVIII y principios del siglo XIX se desarrollaron algunas ingeniosas invenciones mecánicas, utilizadas fundamentalmente en la industria textil, entre las que destacan la hiladora giratoria de Hargreaves (1770), la hiladora mecánica de Crompton (1779), el telar mecánico de Cartwright (1785) y el telar de Jacquard (1801). Este último utilizaba una cinta de papel perforada como un programa para las acciones de la máquina. Es a partir de este momento cuando se empiezan a utilizar dispositivos automáticos en la producción, dando paso a la automatización industrial.



*Fig. 1.3 Telar mecánico*

Durante los siglos XVII y XVIII en Europa fueron contruidos muñecos mecánicos muy ingeniosos que tenían algunas características de robots.

A mediados del siglo XVII J. de Vaucanson construyó varias muñecas mecánicas de tamaño humano que ejecutaban piezas de música.

En 1801 J. Jacquard invento su telar, que era una máquina programable para la urdimbre.

En 1805 H. Maillardet construyó una muñeca mecánica capaz de hacer dibujos.

En 1946 el inventor americano G.C Devol desarrolló un dispositivo controlador que podía registrar señales eléctricas por medio magnéticos y reproducirlas para accionar un máquina mecánica. La patente estadounidense se emitió en 1952

En 1954 se desarrolló el trabajo con tele operadores (manipuladores de control remoto) para manejar materiales radiactivos.

En 1952 una máquina prototipo de control numérico fue objetivo de demostración en el Instituto Tecnológico de Massachusetts después de varios años de desarrollo.

En 1959 se introdujo el primer robot comercial por Planet Corporation. Estaba controlado por interruptores de fin de carrera.

En 1960 el introdujo el primer robot 'Unimate', basada en la transferencia de artic programada de Devol. Utilizan los principios de control numérico para el control de manipulador y era un robot de transmisión hidráulica.

1999: Sony Lanza el Perro mascota Robótico AIBO.

2000: Honda presenta a ASIMO el Robot Humanoide

2001: Cyberknife, un Cirujano robótico, es autorizado para funcionar en los Hospitales Estadounidenses.

En la literatura se encuentran varias definiciones de lo que es un robot, pero la que debería ser definitiva es la dada por la ISO (International Standards Organización) que dice: [1]

*“Un robot industrial es una máquina manipulativa, automáticamente controlada, reprogramable, multi-propósito, con varios ejes reprogramables, los cuales pueden ser fijos o móviles, para uso en aplicaciones de automatización industrial”*

Sin embargo, los robots industriales son una parte del total, digamos que actualmente los no industriales son utilizados fundamentalmente para exploración, para medicina y para entretenimiento, lo cual les otorga una importancia cada vez mayor.

Es por ello que la RIA (Robotics Industries Association o Asociación de Industrias de Robótica) prefiere la siguiente definición: [1]

*“Un robot es una máquina multifuncional, reprogramable, diseñada para manipular materiales, partes, herramientas, o dispositivos especiales, mediante movimientos variables, programados para la realización de una variedad de tareas”*

Las aplicaciones típicas de la industria los robots pueden ser usados para soldadura, montaje, pintura, y la inspección etc., realizando a una alta velocidad y con alta precisión.

Normalmente, el robot está conectado a través de un controlador a una computadora portátil o computadora de escritorio en el que el programación se lleva a cabo. Casi todos los robots industriales vienen con un llamado tech-colgante que se puede utilizar para mover y controlar el robot. [1]

La programación del robot industrial puede dividirse en categorías, en línea y fuera de línea de programación, Desconectado programación significa que se crea el código de robot sin conexión con el robot físico. El programa generado se transfiere posteriormente al robot y ejecutado en

lo real medio ambiente. Normalmente se realiza una verificación del código en algún tipo de software de simulación antes de transferir el programa. La programación en línea incluye programación cuando el software está conectado directamente al robot físico. Esto se puede hacer con unidad de programación utilizado para mover el robot a ciertas posiciones donde están almacenados, y luego se crea una trayectoria entre los puntos almacenados. [4]

La mayoría de los robots industriales tienen programación en línea estructurada, diciéndoles cómo actuar. Se definen puntos en espacio y entonces, cómo se supone que los robots pueden llegar a puntos específicos, normalmente llamados P1, P2, P3 etc. Cuando la programación en KRL, presente una fuente que tiene la extensión (.src) se crea que contiene la declaración de variables, y otros requisitos previos que se leen y declaran automáticamente antes de su ejecución. [4]

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

- Manual: Cuando el operario controla directamente la tarea del manipulador.
  - De secuencia fija: cuando se repite, de forma invariable, el proceso de trabajo preparado previamente.
  - De secuencia variable: Se pueden alterar algunas características de los ciclos de trabajo.
- [1]

Existen muchas operaciones básicas que pueden ser realizadas óptimamente mediante manipuladores, por lo que se debe considerar seriamente el empleo de estos dispositivos, cuando las funciones de trabajo sean sencillas y repetitivas

El software representa toda la parte inmaterial o intangible que hace funcionar a un ordenador para que realice una serie de tareas específicas, coloquialmente conocidos como programas el software engloba a toda la información digital que hace al conjunto de elementos físicos y materiales que componen el computador trabajar de manera inteligente, ya que pueden acelerar la producción y reducir los costos de mano de obra, los robots industriales son comunes en

entornos de producción de alto volumen que requieren acciones repetitivas y fijas como aquellas aplicaciones tipo CNC y de máquina de soldadura. En estas aplicaciones, los movimientos de un robot no cambian; cortan partes de metal o soldadura en los mismos lugares, una y otra vez. [2]



*Fig. 1.4 Robots en la industria.*

Las simulaciones se utilizan en la optimización, experimentos, visualizaciones, desarrollo de las fábricas. Ahorran tiempo y dinero de los fabricantes, protege la propiedad y la salud de los empleados (simulaciones ergonómicas). [3]

Los robots de hoy en día están evolucionando a nuevos tipos de aplicaciones de manejo y colocación de partes que requieren un manejo flexible, preciso y delicado. Los robots industriales han madurado al punto en el que pueden ser programados por expertos para pruebas, investigación y automatización. [3]

Estos nuevos tipos de aplicaciones no se pueden manejar con los movimientos fijos y repetitivos de los entornos de producción a gran escala por las siguientes razones:

Los entornos flexibles de manufactura requieren robots para ajustarse a los nuevos componentes y procedimientos cada vez que la producción cambia a una nueva parte.

El ensamble con precisión requiere robots para ubicar, manejar y colocar componentes pequeños en la ubicación correcta.

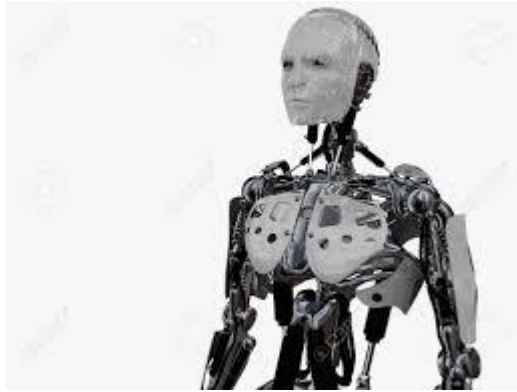
El manejo de partes delicadas requiere robots para ubicar y manejar objetos meticulosamente para que no sean dañados o contaminados. [3]

Un brazo robótico es un tipo de brazo mecánico, normalmente programable, con funciones parecidas a las de un brazo humano; este puede ser la suma total del mecanismo o puede ser parte de un robot más complejo. Las partes de estos manipuladores o brazos son interconectadas a través de articulaciones que permiten, tanto un movimiento rotacional tales como los de un robot articulado, como un movimiento de desplazamiento lineal [4].

La Cibernética es la ciencia que se ocupa de los sistemas de control y de comunicación en las personas y en las máquinas, estudiando y aprovechando todos sus aspectos y mecanismos comunes. El nacimiento de la cibernética se estableció en el año 1942. La unión de diferentes ciencias como la mecánica, electrónica, medicina, física, química y computación, han dado el surgimiento de una nueva doctrina llamada Biónica, La cual busca imitar y curar enfermedades y deficiencias físicas.

A todo esto se une la robótica, la cual se encarga de crear mecanismos de control los cuales funcionen en forma automática.

Todo esto ha conducido al surgimiento de los Cyborg, organismos Bio-mecánicos que buscan imitar la naturaleza humana.



*Fig. 1.5 Cyborg*

## CAPÍTULO II: FUNDAMENTOS DE ROBÓTICA

---

### 2.1 Parámetros para clasificar a los robots

La potencia del software en el controlador determina la utilidad y flexibilidad del robot dentro de las limitantes del diseño mecánico y la capacidad de los sensores. Los robots han sido clasificados de acuerdo a su generación, a su nivel de inteligencia, a su nivel de control, y a su nivel de lenguaje de programación. Estas clasificaciones reflejan la potencia del software en el controlador. [10]

#### 2.1.1 Por generación

La generación de un robot se determina por el orden histórico de desarrollos en la robótica. Cinco generaciones son normalmente asignadas a los robots industriales: [10]

**1º Generación:** El sistema de control está basado en “paradas fijas” mecánicamente. Como ejemplo de esta primera etapa están los mecanismos de relojería que mueven las cajas musicales o los juguetes de cuerda.

Se consideran como dispositivos que actúan como esclavo mecánico de un hombre, quien provee mediante su intervención directa el control de los órganos de movimiento. Esta transmisión tiene lugar mediante servomecanismos actuados por las extremidades superiores del hombre, caso típico manipulación de materiales radiactivos, obtención de muestras submarinas.

**2º Generación:** El movimiento se controla a través de una secuencia numérica almacenada en disco o cinta magnética. Por regla general, este tipo de robots se utiliza en la industria automotriz y son de gran tamaño.

El dispositivo actúa automáticamente sin intervención humana frente a posiciones fijas en las que el trabajo ha sido preparado y ubicado de modo adecuado ejecutando movimientos repetitivos en el tiempo, que obedecen a lógicas combinatorias, secuenciales, programadores



pasó a paso, neumáticos o controladores Lógicos Programables. Un aspecto muy importante está constituido por la facilidad de rápida reprogramación que convierte a estos Robots en unidades versátiles cuyo campo de aplicación no sólo se encuentra en la manipulación de materiales sino en todo los procesos de manufactura, como por ejemplo: en el estampado en frío y en caliente asistiendo a las máquinas-herramientas para la carga y descarga de piezas, en la inyección de termoplásticos y metales no ferrosos, en los procesos de soldadura a punto y continúa en tareas de pintado y reemplazando con ventaja algunas operaciones de máquinas convencionales.[10]

**3º Generación:** Utilizan las computadoras para su control y tienen cierta percepción de su entorno a través del uso de sensores. Con esta generación se inicia la era de los robots inteligentes y aparecen los lenguajes de programación para escribir los programas de control.

Son dispositivos que habiendo sido construidos para alcanzar determinados objetivos serán capaces de elegir la mejor forma de hacerlo teniendo en cuenta el ambiente que los circunda. Para obtener estos resultados es necesario que el robot posea algunas condiciones que permitan su interacción con el ambiente y los objetos. Las mínimas aptitudes requeridas son: capacidad de reconocer un elemento determinado en el espacio y la capacidad de adoptar propias trayectorias para conseguir el objetivo deseado. Los métodos de identificación empleados hacen referencia a la imagen óptica por ser esta el lenguaje humano en la observación de los objetos, sin embargo no puede asegurarse que la que es natural para el hombre, constituye la mejor solución para el robot.

**4º Generación:** Se trata de robots altamente inteligentes con más y mejores extensiones sensoriales, para entender sus acciones y captar el mundo que los rodea. Incorporan conceptos “modélicos” de conducta.

**5º Generación:** Actualmente en desarrollo. Esta nueva generación de robots basará su acción principalmente en modelos conductuales establecidos.

Esta generación será una nueva tecnología que incorporara 100% inteligencia artificial y utilizara métodos como modelos de conducta y una nueva arquitectura de sumisión, además de otras tecnologías actualmente en desarrollo como la nanotecnología. [10]

### 2.1.2 Según su arquitectura generacional

**Robots Play-back**, los cuales regeneran una secuencia de instrucciones grabadas, como un robot utilizado en recubrimiento por spray o soldadura por arco. Estos robots comúnmente tienen un control de lazo abierto. [12]



*Fig. 2.1 Ejemplo de robot play-back.*

**Robots controlados por sensores**, estos tienen un control en lazo cerrado de movimientos manipulados, y hacen decisiones basados en datos obtenidos por sensores. [12]



*Fig. 2.2 Ejemplo de robot controlado por sensores.*

**Robots controlados por visión**, donde los robots pueden manipular un objeto al utilizar información desde un sistema de visión. [12]



*Fig. 2.3 Ejemplo de robot controlado por visión.*

**Robots controlados adaptablemente**, donde los robots pueden automáticamente reprogramar sus acciones sobre la base de los datos obtenidos por los sensores. [12]



*Fig. 2.4 Ejemplo de robot controlado adaptablemente.*

**Robots con Inteligencia Artificial**, donde los robots utilizan las técnicas de inteligencia artificial para hacer sus propias decisiones y resolver problemas. [12]



*Fig. 2.5 Ejemplo de robot con inteligencia artificial.*

**Robots médicos**, fundamentalmente, prótesis para disminuidos físicos que se adaptan al cuerpo y están dotados de potentes sistemas de mando. Con ellos se logra igualar al cuerpo con precisión los movimientos y funciones de los órganos o extremidades que suplen. [12]



*Fig. 2.6 Ejemplo de robot utilizando en la medicina.*

**Androides**, robots que se parecen y actúan como seres humanos. Los robots de hoy en día vienen en todas las formas y tamaños, pero a excepción de los que aparecen en las ferias y espectáculos, no se parecen a las personas y por tanto no son androides. Actualmente, los androides reales sólo existen en la imaginación y en las películas de ficción. [12]



*Fig. 2.7 Ejemplo de robot androide.*

**Robots móviles**, provistos de patas, ruedas u orugas que los capacitan para desplazarse de acuerdo su programación. Elaboran la información que reciben a través de sus propios sistemas de sensores y se emplean en determinado tipo de instalaciones industriales, sobre todo para el transporte de mercancías en cadenas de producción y almacenes. También se utilizan robots de este tipo para la investigación en lugares de difícil acceso o muy distantes, como es el caso de la exploración espacial y las investigaciones o rescates submarinos. [12]



*Fig. 2.8 Ejemplo de robot móvil.*

## **2.2 Manipulador articulado**

El manipulador es un ensamblaje de eslabones y articulaciones que permiten rotación o traslación entre dos de los eslabones. Estos eslabones son sólidos y están sostenidos por una base (horizontal, vertical o suspendida), con una articulación entre la base y el primer eslabón. El movimiento y las articulaciones definen los "grados de libertad" del robot. Una configuración típica de un brazo robot es la de tres grados de libertad, a la que se añaden las posibilidades de movimiento en la muñeca, llegando a un total de cuatro a seis grados de libertad. Algunos robots tienen entre siete y nueve grados de libertad, pero por su complejidad, son menos comunes [8].

La base del manipulador es rígida y está sujeta a una plataforma que la sostiene, generalmente, pero no siempre, del suelo. Cuando se puede mover, comúnmente lo hace a lo largo de un eje y es para sincronizar el movimiento del robot con el de otros equipos. De esta manera el

movimiento de la base sumado al movimiento tridimensional del manipulador proporcionan cuatro grados de libertad [8].

### **2.3 Tipos de articulaciones**

Las articulaciones permiten el movimiento relativo entre los sucesivos eslabones.

Existen diferentes tipos de articulaciones, las más utilizadas en la robótica son las siguientes; la articulación de rotación suministra un grado de libertad consistente en una rotación alrededor del eje de la articulación.

En la articulación prismática el grado de libertad consiste de una translación a lo largo del eje de la articulación. [10]

### **2.4 Configuraciones básicas**

Los robots industriales están disponibles en una amplia gama de tamaños, formas y configuraciones físicas. La gran mayoría de los robots comercialmente disponibles en la actualidad tienen una de estas cuatro configuraciones básicas; estas estructuras tienen diferentes propiedades en cuanto a espacio de trabajo y accesibilidad a posiciones determinadas [10] [9]:

#### **2.4.1 Configuración polar**

La configuración polar utiliza coordenadas polares para especificar cualquier posición en términos de una rotación sobre su base, un ángulo de elevación y una extensión lineal del brazo, con lo que obtiene un medio de trabajo en forma de esfera (figura 2.9) [10] [11] [9]

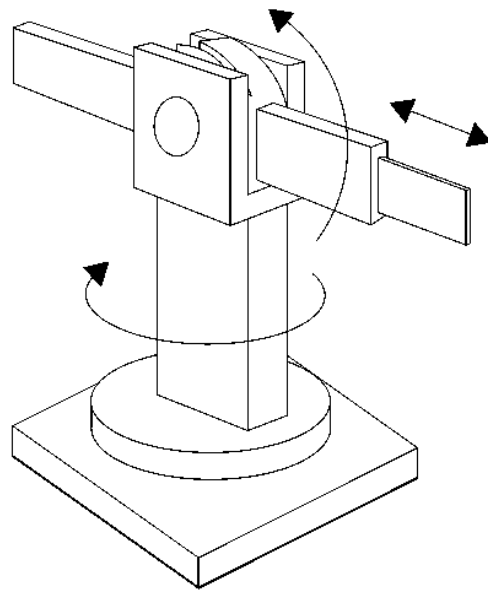


Fig. 2.9 Área de trabajo de la configuración polar.

### 2.4.2 Configuración cilíndrica

La configuración cilíndrica sustituye un movimiento lineal por uno rotacional sobre su base, con los que se obtiene un medio de trabajo en forma de cilindro (figura 2.10).

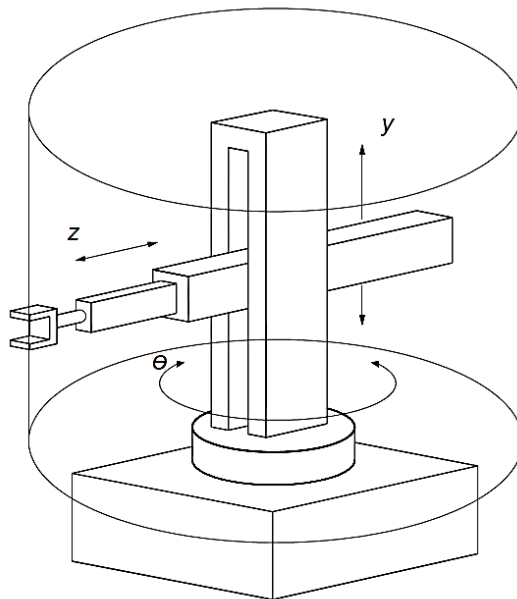
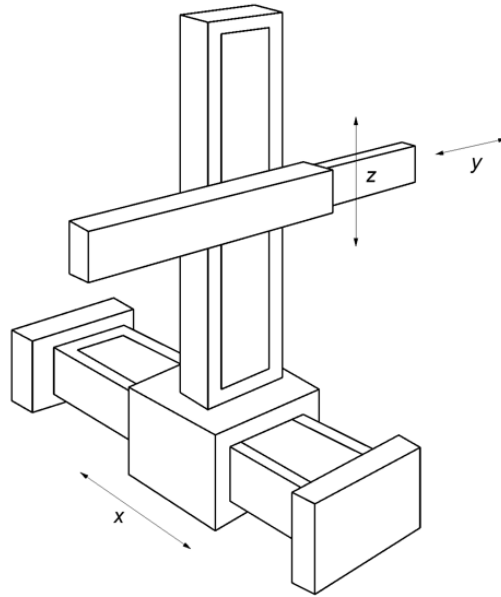


Fig. 2.10 Área de trabajo de la configuración cilíndrica.

### 2.4.3 Configuración de coordenadas cartesianas

La configuración de coordenadas cartesianas, posee tres movimientos lineales, y su nombre proviene de las coordenadas cartesianas, las cuales son más adecuadas para describir la posición y movimiento del brazo. Los robots cartesianos a veces reciben el nombre de X, Y, Z, donde las letras representan a los tres ejes del movimiento (figura 2.11) [10] [11] [9].



*Fig. 2.11 Área de trabajo de la configuración cartesiana.*

### 2.4.4 Configuración angular

Posee una articulación con movimientos rotacionales y dos angulares. El movimiento se basa en interpolación por articulación. El volumen de trabajo es irregular, por lo que suele revisar el plano del robot.

La configuración de brazo articulado, utiliza únicamente articulaciones rotacionales para conseguir cualquier posición y es por esto que es el más versátil. [10] [11] [9].



## 2.5 Orientación del efector final

El movimiento de un brazo robótico provisto por una muñeca con un efector final frecuentemente es tratado en dos pasos. En primer lugar se mueve el brazo para posicionar el extremo del último enlace y posteriormente, se orienta la muñeca para que el efector final tenga la orientación adecuada, existen tareas que pueden requerir el movimiento simultáneo del brazo y la muñeca. [10]

### 2.5.1 Posición y orientación en el espacio

*Posición.*- en robótica es necesario poder describir de forma conveniente las posiciones y orientaciones de los objetos en el espacio.

Primero considere la necesidad de localizar objetos en el espacio plano. Este es un problema frecuente de robots o móviles que navega por terrenos planos. Para este caso se necesitan al menos dos coordenadas y un ángulo de orientación.

Supóngase un sistema de coordenadas de transferencia fijo que se designara como el algoritmo de {A} la posición con respecto a este sistema se representa mediante un vector de posición  ${}^A P$ . tal como se muestra en la figura 2.12 cuyas componentes son las coordenadas de este punto. [10]

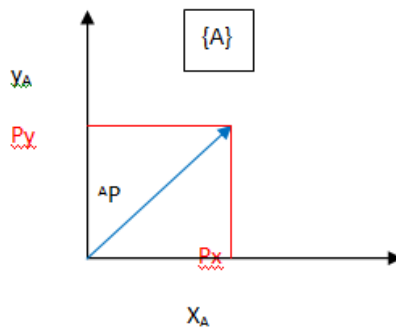


Fig. 2.12 Sistema de vectores primarios.

Otra posible forma de expresar las coordenadas con un sistema de referencia es mediante coordenadas polares tal como se ilustra en la figura 2.13. En este caso las coordenadas son la distancia de origen “r” y  $\theta$  que forma el vector respecto  $\widehat{X}_A$ .

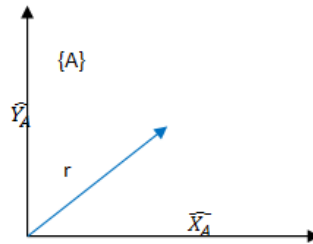


Fig. 2.13 Sistema de coordenadas polares.

A la que se le conoce como matriz de rotación estas matrices juegan un papel muy importante en los modelos empleados en robótica.

Obsérvese que si el ángulo de orientación es  $\theta$  de la figura 2.13 anterior se deduce que:

$${}^B {}^A R = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \quad (2.1)$$

Teniendo en cuenta que los vectores columna son ortogonales puede escribirse también que la inversa es:

$${}^B {}^A R = {}^B {}^A R^{-1} = {}^B {}^A R^T \quad (2.2)$$

Supóngase ahora que el sistema B tiene sus vectores de dirección coincidentes con las de A.

[10]

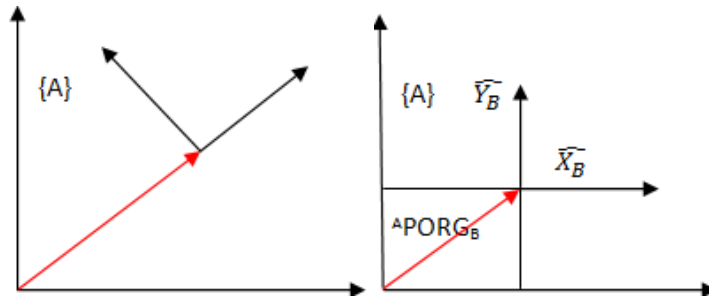


Fig. 2.14 Vectores de dirección coincidentes.

Este problema se presentaría por ejemplo cuando un robot se desplaza sin cambiar de orientación como se muestra en la figura 2.14. [10]

El origen del sistema  $\{B\}$  se localiza con respecto al de  $\{A\}$  mediante el vector

$${}^A P_{ORGB} = \begin{bmatrix} A P_{xORGB} & A P_{yORGB} \end{bmatrix} \quad (2.3)$$

Por consiguiente las coordenadas de un punto cualquiera del plano en los 2 sistemas están relacionadas mediante:

$${}^A P_x = {}^B P_x + {}^A P_{xORG} \quad (2.4)$$

$${}^A P_y = {}^B P_y + {}^A P_{yORG} \quad (2.5)$$

Las cuales son componentes de los vectores resultantes de la suma puesto que están en la misma dirección.

Las ecuaciones anteriores definen la transformación de la traslación de todo sistema de coordenadas. [10]

*Orientación:* Además de la posición es necesario definir la orientación con respecto al sistema de referencia. La orientación en un espacio tridimensional, viene definida por tres grados de libertad, linealmente independientes, normalmente se usan 2 sistemas de referencia.

Si se tiene 2 sistemas de referencia OXY y OUV, con el mismo origen, pero rotado un ángulo. Cada vector del sistema de referencia es y deben ser equivalentes, con una matriz de rotación R se define la orientación de OUV con respecto a OXY.

Sea el sistema de coordenadas OXY y OUV Vectores unitarios:

## **2.6 Posición y orientación en el espacio 3d**

En la robótica resulta indispensable poder representar posiciones y orientaciones en el espacio. Sin embargo la evolución de la robótica tiene cada vez mayor interés en mostrar modelos y movimientos en 3D. [10]

### **2.6.1 Posición en 3d**

La posición en 3D se establece de forma univoca mediante un vector de posición  ${}^A P$  con 3 componentes con respecto a un sistema de referencia  $\{A\}$ , con origen en el sistema y en otro extremo indicando la posición en el espacio. [10]

Puede tener asociado (la posición) un sistema desde ahora el extremo del vector  ${}^A P$  es el origen de un segundo sistema. [10]

### **2.6.2 Orientación**

La orientación en 3D de un cuerpo en el sistema  $\{B\}$  con respecto al de  $\{A\}$  vendrá dado por la orientación relativa de los ejes del  $\{B\}$  cuyo origen es la posición asignada por el vector  ${}^A P$ .

Una forma de indicar la orientación de un sistema por ejemplo  $\{B\}$  respecto a otro por ejemplo  $\{A\}$  es hacerlo mediante las coordenadas en el  $\{A\}$  de los vectores unitarios en la dirección de los ejes del  $\{B\}$ . Sea  ${}^A P$  el vector de posición en un sistema  $\{A\}$  tal como se muestra en la figura 2.15 para un vehículo autónomo. [10]

$${}^A P = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \quad (2.6)$$

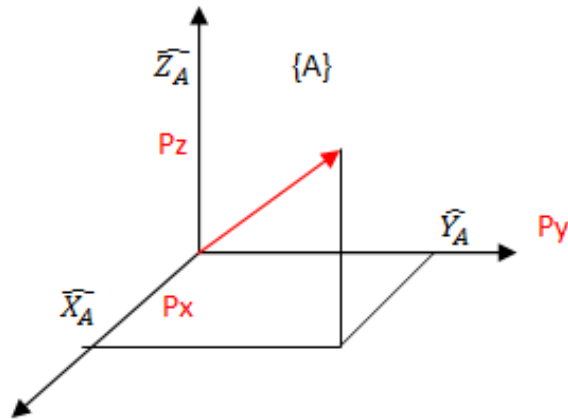


Fig. 2.15 Vector de posición en un sistema  $\{A\}$ .

La posición de un punto en el espacio también puede expresarse mediante coordenadas cilíndricas tal como se ilustra en la figura 2.16. En este caso las coordenadas son la distancia “r” el ángulo  $\theta$  y la distancia Z entre el punto que marca la posición y su proyección sobre el plano  $\widehat{X}_A, \widehat{Y}_A$ , las 2 primeras tienen el mismo significado que las coordenadas polares. [10]

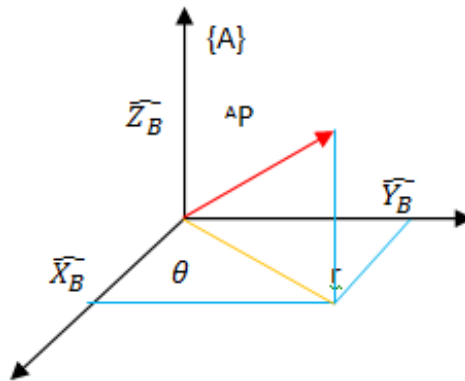


Fig. 2.16 Coordenadas cilíndricas.

Si los vectores  $\begin{bmatrix} \hat{X}_B & \hat{Y}_B & \hat{Z}_B \end{bmatrix}$  unitarios tienen la dirección de los ejes de un sistema de coordenadas  $\{B\}$  y se desea expresar que estos vectores representan el sistema  $\{A\}$  se escribirá  $\begin{bmatrix} A\hat{X}_B & A\hat{Y}_B & A\hat{Z}_B \end{bmatrix}$  denominada matriz de rotación. [10]

$${}^B A R = \begin{bmatrix} A\hat{X}_B & A\hat{Y}_B & A\hat{Z}_B \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad (2.7)$$

Resumiendo lo anterior nos cabe añadir que la localización (posición y orientación) del espacio se determina mediante 4 vectores de 3 elementos formando 3 de ellas la matriz de rotación.

Un cambio de posición manteniendo constante la orientación puede representarse mediante la matriz de rotación unidad y vector con los cambios en las coordenadas del punto. [10]

## 2.7 Cambios de base

En la mayor parte de los problemas que surgen en robótica es necesario realizar cambios entre sistemas de referencia.

Así, los robots manipuladores, los modelos del manipulador involucran cambios entre sistemas asociados a cada una de las articulaciones de una cadena cinemática para describir la posición y orientación principalmente del extremo del manipulador con relación a la base.

Como se ha deducido anteriormente las transformaciones consisten esencialmente en traslaciones y rotaciones. Suponiendo que  $\{A\}$  y  $\{B\}$  tienen la misma orientación, la traslación puede expresarse en la siguiente expresión:

$${}^A P_x = {}^B P + {}^A P O R G_B \quad (2.8)$$

Obsérvese que los 2 vectores de la derecha pueden sumarse debido a que ambos sistemas de referencia tienen la misma orientación.

La rotación puede expresarse mediante la siguiente relación

$${}^B R = {}^A R^{-1} = {}^A R^T \quad (2.9)$$

## 2.8 Análisis de transformaciones

Es una matriz T de 4 x 4 que representa la transformación de un vector de un sistema de coordenadas a otro. [10]

Esta matriz está compuesta por 4 submatrices:

$R_{3 \times 3}$  Submatriz de Rotación

$P_{3 \times 1}$  Submatriz de Traslación

$F_{1 \times 3}$  Submatriz de Perspectiva

$E_{1 \times 1}$  Submatriz de Escalado Global

$$T = \begin{bmatrix} R_{3 \times 3} & P_{3 \times 1} \\ F_{1 \times 3} & E_{1 \times 1} \end{bmatrix} \quad (2.10)$$

En robótica, generalmente se considera la submatriz de perspectiva como nula y la submatriz de escalado global como uno. [10]

## 2.9 Sistemas no lineales e incrementales en el tiempo.

Para efectos prácticos en la robótica se recurre a la linealización de los sistemas ya que estos (eslabón-articulación) no son lineales para llevar a cabo tal tarea se realizan varias consideraciones dependiendo del tipo de control y su exactitud deseada como el caso de: [10]

Analizar cambios de movimiento o velocidad pequeños, depreciación de fuerzas de fricción, coreolis, rozamientos, etc.

O bien despreciando cambios de parámetros con la temperatura bajo efectos magnéticos o bajo variaciones de fuentes de alimentación. [10]

El caso de variación respecto a tiempo de parámetros tales como masa, inercia, temperatura, fricción, resistencia, se opta para linealizar el modelo por considerar que estos parámetros son constantes. Sin embargo en determinados sistemas de control si es prescindible conocer o considerar los valores exactos los cuales son obtenidos ya sea mediante observadores o con ayuda de sensores, ya que muchas veces los sistemas de control dependerán de estos parámetros. [10]



## 2.10 Tipos de sensores de un robot

Para conseguir que un robot realice su tarea con la adecuada precisión, velocidad e inteligencia, es preciso que tenga conocimiento tanto de su estado, como del estado de su entorno. La información relacionada con su estado, es fundamentalmente la posición de sus articulaciones, la consigue con los denominados sensores internos, mientras que la que se refiere al estado de su entorno, se adquiere con los sensores externos. [13]

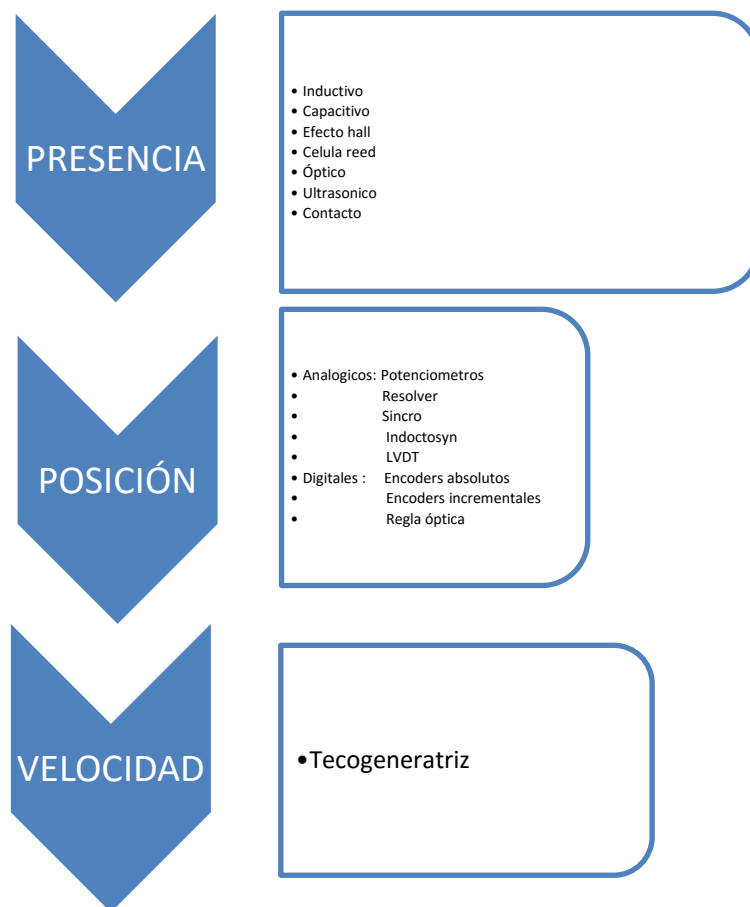


Diagrama 2.1. Tipos de sensores internos del Robot

## **2.10.1 Sensores de posición**

Para el control de la posición angular se emplean fundamentalmente los denominados encoders y resolvers.

### **2.10.1.1 Codificadores angulares de posición (encoders)**

Los codificadores ópticos o encoders incrementales constan, en su forma más simple, de un disco transparente con una serie de marcas opacas colocadas radialmente y equidistantes entre sí, de un sistema de iluminación en el que la luz es colimada de forma correcta, y de un elemento fotorreceptor. El eje cuya posición se quiere medir va acoplado al disco transparente, a medida que el eje gire se irán generando pulsos en el receptor cada vez que la luz atraviese cada marca y llevando una cuenta de estos pulsos es posible conocer la posición del eje. [13]

Normalmente los sensores de posición se acoplan al eje del motor, considerando que en la mayor parte de los casos entre el eje del motor y el de la articulación se sitúa un reductor de relación  $N$ , cada movimiento de la articulación se verá multiplicado por  $N$ , al ser medido por el sensor, este aumentará así su resolución multiplicándola por  $N$ . [13]

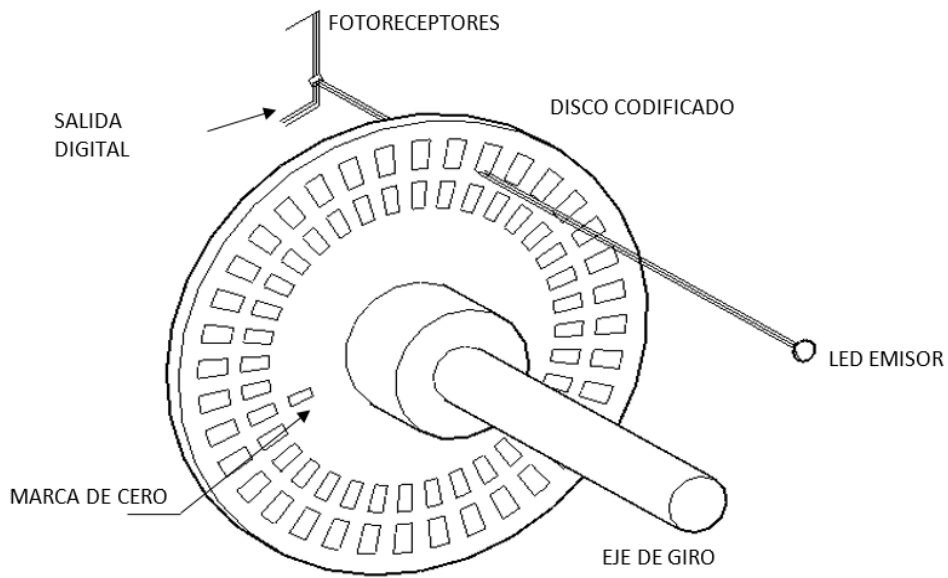


Fig. 2.11 Disposición de un codificador óptico (encoder) incremental

### 2.10.1.2 Captadores angulares de posición (sincro-resolvers)

Se trata de captadores analógicos con resolución teóricamente infinita. El funcionamiento de los resolvers se basa en la utilización de una bobina solidaria al eje excitada por una portadora, y por dos bobinas fijas situadas a su alrededor, (figura 2.12)

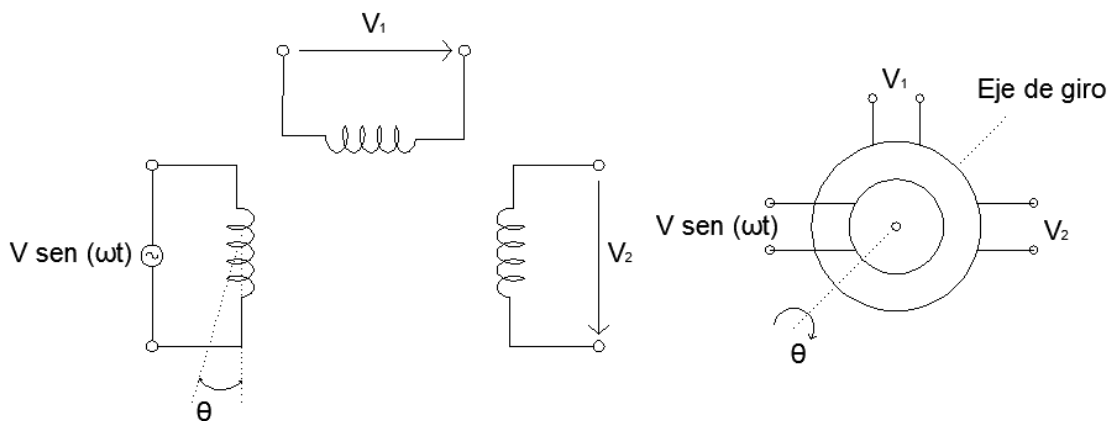


Fig. 2.12 Esquema de funcionamiento de un resolver

El giro de la bobina móvil hace que el acoplamiento con las bobinas fijas varíe, consiguiendo que la señal resultante en estas dependa del seno del ángulo de giro. La bobina móvil excitada con tensión  $V \text{sen}(\omega t)$  y girada un ángulo  $\theta$  induce en las bobinas fijas situadas en cuadratura las siguientes tensiones. [13]

$$V_1 = V \text{sen}(\omega t) \text{sen} \theta \quad (2.11)$$

$$V_2 = V \text{sen}(\omega t) \cos \theta \quad (2.12)$$

El funcionamiento de los sincros es análogo al de los resolvers, excepto que las bobinas fijas forman un sistema trifásico en estrella. Para un giro  $\theta$  de la bobina móvil excitada con tensión  $V \text{sen}(\omega t)$  admitiendo que los acoplamientos y los desfases son los mismos para todos los desvenados, se obtienen las siguientes tensiones entre las fases del estator. [13]

$$\begin{aligned} V_{13} &= \sqrt{3}V \cos(\omega t) \text{sen} \theta \\ V_{32} &= \sqrt{3}V \cos(\omega t) \text{sen}(\theta + 120^\circ) \\ V_{21} &= \sqrt{3}V \cos(\omega t) \text{sen}(\theta + 240^\circ) \end{aligned} \quad (2.13)$$

Que es la llamada representación del ángulo  $\theta$  en formato sincro.

El cambio de formato sincro a formato resolver o viceversa es inmediato, ya que se puede pasar de uno a otro a través la llamada red de Scott o transformador de Scott de funcionamiento bidireccional. [13]

Para tratar con el sistema de control la información generada por los resolver y los sincros es necesario convertir las señales analógicas en digitales, para ello se utilizan los convertidores resolver/digital (R/D), que se basan en dos tipos de estructuras distintas, seguimiento y muestreo. [13]

### 2.10.1.3 Sensores lineales de posición (LVDT e Inductosyn)

Es un transformador diferencial de variación lineal, que consta de un núcleo de material ferromagnético unido al eje, que se mueve linealmente entre un devanado primario y dos secundarios haciendo que varíe la inductancia entre ellos. Figura (2.13) [13]

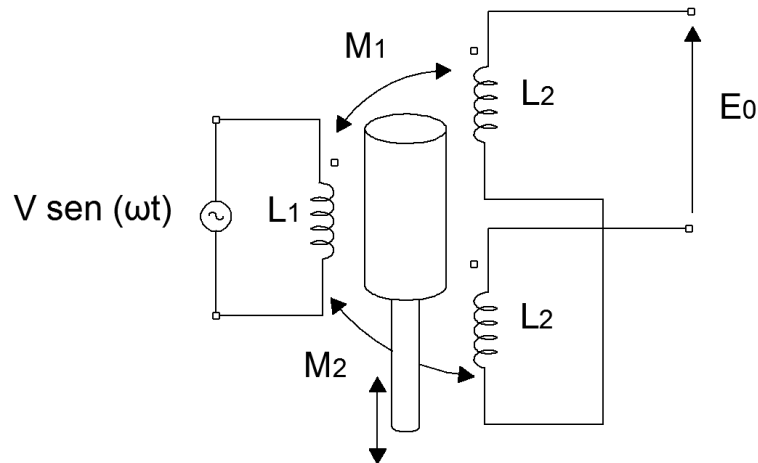


Fig. 2.13 Esquema de funcionamiento de un LVDT

Como resultado de un desplazamiento que se quiere medir, el núcleo magnético es desplazado de manera que una de las bobinas secundarias no recubra totalmente el núcleo la corriente inducida en un secundario será mayor que la inducida en el otro. De la diferencia de las tensiones medidas en los dos secundarios se obtiene el desplazamiento realizado por el núcleo. [13]

Además de las ventajas que presenta LVDT presenta una alta resolución, gran sensibilidad, y una respuesta dinámica elevada los inconvenientes es que sólo puede aplicarse a medición de pequeños desplazamientos. [13]

### 2.10.2 Sensores de velocidad

La captación de la velocidad se hace necesaria para mejorar el comportamiento dinámico de los actuadores del robot. La información de la velocidad de movimiento de cada actuador, se realimenta normalmente a un bucle de control analógico implementando en el propio

accionador del elemento motor, la velocidad de giro de cada actuador es llevada hasta la unidad de control del robot.

Normalmente el bucle de control de velocidad es analógico, el captador usado es una tacogeneratriz que proporciona una tensión proporcional a la velocidad de giro de su eje. [13]

### **2.10.3 Sensor de presencia y proximidad.**

Los más simples sensores detectan la presencia de un objeto los conocidos como finales de carrera en este caso el objeto cierra un contacto mecánico, el cual genera una señal eléctrica.

Sin embargo se emplean otros sensores sin necesidad de un contacto mecánico, si el objeto es metálico utilizan sensores inductivos, si el material es no ferroso pueden emplearse sensores capacitivos.

La presencia de objetos puede ser por sensores visibles o invisibles, de modo que al interponer un objeto en la línea del emisor y el receptor detectan una obstrucción y bien mandan o dejan de enviar una señal. [10]

### **2.10.4 Sensores de fuerza.**

En robótica suelen utilizarse galgas extensiométricas las cuales detectan la aplicación de fuerza o de un peso conforme estas se van deformando debido a la presión envían una señal de voltaje proporcional a su deformación y el principio de operación en el efecto piezo resistivo. [10]

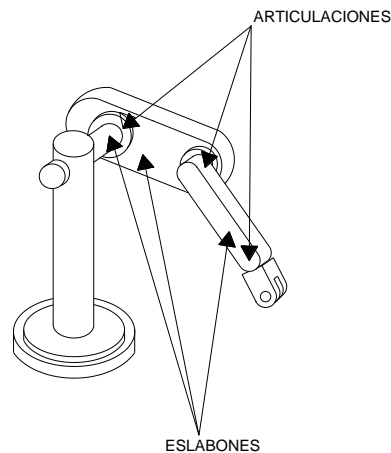
Los materiales que se utilizan tienen determinada sensibilidad de variación de resistencia, el caso del manganeso es de  $K=0.5$ , cromo  $K=2.5$ , platino  $K=4.8$ .

Existen también galgas extensiométricas con hojas metálicas de otro material las cuales se diseñan para minimizar la sensibilidad transversal.

Otro tipo de galga son las galgas extensiométricas con semiconductor las cuales contienen silicio o germanio. Estas galgas se utilizan en las muñecas de los robots (cuando es necesario debido a su alta sensibilidad ya que estas pueden detectar que si el peso que se pretende levantar o manipular se encuentra dentro del rango de diseño estas galgas también se emplean como sensor de par (taquímetro). [10]

## 2.11 Estructura de robots manipuladores

Un manipulador industrial convencional es una cadena cinemática abierta formada por un conjunto de eslabones o elementos de la cadena interrelacionados mediante articulaciones o pares cinemáticos tal como se muestra en la figura 2.14. [10].



*Fig. 2.14 Cadena cinemática abierta*

## 2.12 Cinemática de los manipuladores

La cinemática es la ciencia que trata el movimiento sin considerar las fuerzas que lo ocasionan. Dentro de la cinemática se estudian la posición, velocidad, aceleración, y todas las derivadas

de mayor orden en las variables de posición (respecto al tiempo o a cualquier otra variable), el estudio de la cinemática de los manipuladores se refiere a todas las propiedades del movimiento, las geométricas y las basadas en tiempo.[12]

La cinemática del robot estudia el movimiento del mismo con respecto a un sistema de referencia. 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. [10]

Existen dos problemas fundamentales para resolver en la cinemática del robot, el primero de ellos se conoce como el problema cinemático directo, y consiste en determinar cuál es la posición y orientación del extremo final del robot, con respecto a un sistema de coordenadas que se toma como referencia, conocidos los valores de las articulaciones y los parámetros geométricos de los elementos del robot, tal como se muestra en la figura 2.15 el segundo denominado problema cinemático inverso resuelve la configuración que debe adoptar el robot para una posición y orientación del extremo conocidas.[10]

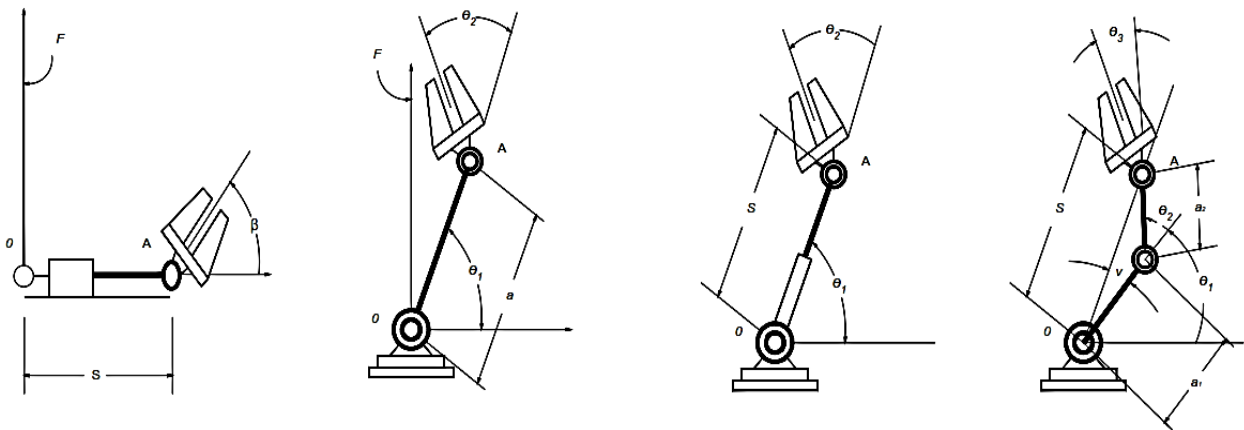


Fig. 2.15 Sistema de coordenadas de referencia fija



El movimiento relativo en las articulaciones resulta en el movimiento de los elementos que posicionan el brazo en una orientación deseada. En la mayoría de las aplicaciones de robótica, se está interesado en la descripción espacial del efector final del manipulador con respecto a un sistema de coordenadas de referencia fija. [10]

La cinemática del brazo del robot trata con el estudio analítico de la geometría del movimiento de un robot con respecto a un sistema de coordenadas de referencia fijo como una función del tiempo sin considerar las fuerzas-momentos que originan dicho movimiento. Así pues, trata con la descripción analítica del desplazamiento espacial del robot como función del tiempo, en particular las relaciones entre variables espaciales de tipo de articulación y la posición y orientación del efector final del robot. [10]

### **2.12.1 Cinemática directa**

Se utiliza generalmente el álgebra vectorial y matricial para representar y describe la localización de un objeto en el espacio tridimensional con respecto a un sistema de referencia fijo.

De esta forma el problema cinemático directo se reduce a encontrar una matriz homogénea de transformación  $T$  que relacione la posición y orientación del extremo del robot respecto del sistema de referencia fijo situado en la base del mismo. [13]

### **2.12.2 Resolución del problema cinemático directo mediante matrices de transformación homogénea**

La resolución del problema cinemático directo consiste en encontrar las relaciones que permiten conocer la localización espacial del extremo del robot a partir de los valores de sus coordenadas articulares. [13]

Así, se han escogido coordenadas cartesianas y ángulos de Euler para representar la posición y orientación del extremo de un robot de seis grados de libertad, la solución del problema cinemático directo vendrá dada por las relaciones:

$$\begin{aligned}
 x &= f_x(q_1, q_2, q_3, q_4, q_5, q_6) \\
 y &= f_y(q_1, q_2, q_3, q_4, q_5, q_6) \\
 z &= f_z(q_1, q_2, q_3, q_4, q_5, q_6) \\
 \alpha &= f_\alpha(q_1, q_2, q_3, q_4, q_5, q_6) \\
 \beta &= f_\beta(q_1, q_2, q_3, q_4, q_5, q_6) \\
 \gamma &= f_\gamma(q_1, q_2, q_3, q_4, q_5, q_6)
 \end{aligned}
 \tag{2.14}$$

La obtención de estas relaciones no es en general complicada, siendo incluso en ciertos casos (robots con pocos GDL) fácil de encontrar mediante simples consideraciones geométricas. [13]

En general un robot de  $n$  grados de libertad está formado por  $n$  eslabones unidos por  $n$  articulaciones, en forma que cada par articulación-eslabón constituye un grado de libertad. A cada eslabón se le puede asociar un sistema de referencia solidario a él, y utilizando las transformaciones homogéneas, es posible representar las rotaciones y translaciones relativas entre los distintos eslabones que compone el robot. La matriz de transformación homogénea que representa la posición y orientación relativa entre los sistemas asociados a dos eslabones consecutivos del robot se suele denominar matriz  ${}^{i-1}A_i$ , así pues  ${}^0A_1$ , describe la posición y orientación del sistema de referencia solidario al primer eslabón con respecto al sistema de referencia solidario a la base.  ${}^1A_2$  Describe la posición y orientación del segundo eslabón respecto al primero, etc. Por ejemplo la posición y orientación del sistema solidario con el segundo eslabón del robot con respecto al sistema de coordenadas de la base se puede expresar mediante la matriz  ${}^0A_2$  que es igual a:

$${}^0A_2 = {}^0A_1 {}^1A_2
 \tag{2.15}$$

Cuando se consideran todos los grados de libertad, a la matriz  ${}^0A_n$  se le suele denominar T. Así dado un robot de seis grados de libertad, se tiene que la posición y orientación del eslabón final vendrá dada por la matriz T:

$$T = {}^0A_6 = {}^0A_1 {}^1A_2 {}^2A_3 {}^3A_4 {}^4A_5 {}^5A_6 \quad (2.16)$$

Aunque para describir la relación que existe entre dos elementos contiguos se puede hacer uso de cualquier sistema de referencia ligado a cada elemento, la forma habitual que se suele utilizar en robótica es la representación de Denavit-Hatenberg (D-H). Denavit y Hatenberg propusieron en 1955, un método matricial que permite establecer de manera sistemática un sistema de coordenadas  $\{S_i\}$  ligado a cada eslabón  $i$  de una cadena articulada. [13]

Según la representación de D-H escogiendo adecuadamente los sistemas de coordenadas asociadas a cada eslabón, será posible pasar de uno al siguiente mediante 4 transformaciones básicas que dependen exclusivamente de las características geométricas del eslabón. [13]

Estas transformaciones básicas consisten en una sucesión de rotaciones y traslaciones que permiten relacionar el sistema de referencia del elemento  $i$  con el sistema del elemento  $i-1$ . las transformaciones en cuestión son las siguientes:

- 1.- Rotación alrededor del eje  $Z_{i-1}$  un ángulo  $\theta_i$ .
- 2.- Traslación a lo largo de  $Z_{i-1}$  una distancia  $d_i$ ; vector  $d_i (0,0, d_i)$ .
- 3.- Traslación a lo largo de  $Z_{i-1}$  una distancia  $a_i$ ; vector  $a_i (0,0, a_i)$ .
- 4.- Rotación alrededor del eje  $x_i$  un ángulo  $\alpha_i$ .

Dado que el producto de matrices no es conmutativo, las transformaciones se han de realizar en el orden indicado. De este modo se tiene que:

$${}^{i-1}A_i = T(z, \theta_i) T(0, 0, d_i) T(a_i, 0, 0) T(x, \alpha_i) \quad (2.17)$$

Y realizando el producto entre matrices:

$$\begin{aligned}
 {}^{i-1}A_i &= \begin{bmatrix} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha_i & -S\alpha_i & 0 \\ 0 & S\alpha_i & C\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
 &= \begin{bmatrix} C\theta_i & -C\alpha_i S\theta_i & S\alpha_i S\theta_i & a_i C\theta_i \\ S\theta_i & C\alpha_i C\theta_i & -S\alpha_i C\theta_i & a_i S\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.18)
 \end{aligned}$$

Donde  $\theta_i, a_i, d_i, \alpha_i$  son los parámetros D-H del eslabón i. De este modo basta con identificar los parámetros  $\theta_i, a_i, d_i, \alpha_i$  para obtener las matrices A y relacionar así todos y cada uno de los eslabones del robot. [13]

Como se ha indicado para la matriz  ${}^{i-1}A_i$  definida anteriormente relacione los sistemas  $\{S_i\}$  y  $\{S_{i-1}\}$ , es necesario que los sistemas se hayan escogido de acuerdo a unas determinadas normas. Estas, junto con la definición de los cuatro parámetros de Denavit Hartenberg, conforme al siguiente algoritmo para la resolución del problema cinemático directo. [13]

### 2.12.3 Algoritmo de Denavit- Hartenberg para la obtención del modelo cinemático directo.

**DH1.** Numerar los eslabones comenzando con 1 (primer eslabón móvil de la cadena) y acabando con n (último eslabón móvil). Se numerara como eslabón 0 a 2.11 la base fija del robot. [13]

**DH2.**Numerar cada articulación comenzando por 1 (la correspondiente al primer grado de libertad y acabando en n).

**DH3.**Localizar el eje de cada articulación. Si esta es rotativa, el eje será su propio eje de giro. Si es prismática, será el eje a lo largo del cual se produce el desplazamiento.

**DH4.**Para  $i$  de 0 a  $n-1$ , situar el eje  $Z_i$ , sobre el eje de la articulación  $i+1$ .

**DH5.**Situar el origen del sistema de la base  $\{S_0\}$  en cualquier punto del eje  $Z_0$ . Los ejes  $X_0$  e  $Y_0$  se situaran de modo que formen un sistema dextrógiro con  $Z_0$ .

**DH6.**Para  $i$  de 1 a  $n-1$ , situar el sistema  $\{S_i\}$  (solidario al eslabón  $i$ ) en la intersección del eje  $Z_i$  con la línea normal común a  $Z_{i-1}$  y  $Z_i$ . Si ambos ejes se cortasen se situaría  $\{S_i\}$  en el punto de corte. Si fuesen paralelos  $\{S_i\}$  se situaría en la articulación  $i+1$ .

**DH7.**Situar  $X_i$  en la línea normal común a  $Z_{i-1}$  y  $Z_i$ .

**DH8.**Situar  $y_i$  de modo que forme un sistema dextrógiro con  $X_i$  y  $Z_i$ .

**DH9.**Situar el sistema  $\{S_n\}$  en el extremo del robot de modo que  $Z_n$  coincida con la dirección de  $z_{n-1}$  y  $X_n$  sea normal a  $Z_{n-1}$  y  $Z_n$ .

**DH10.**Obtener  $\theta_i$  como el ángulo que hay que girar en torno a para que  $X_{i-1}$  y  $X_i$  queden paralelos.

**DH11.**Obtener  $d_i$  como la distancia, medida a lo largo de  $Z_{i-1}$ , que habría que desplazar  $\{S_{i-1}\}$  para que  $X_i$  y  $X_{i-1}$  quedasen alineados.

**DH12.** Obtener  $a_i$  como la distancia medida a lo largo de  $X_i$  (que ahora coincidiría con  $X_{i-1}$ ) que habría que desplazar el nuevo  $\{S_{i-1}\}$  para que su origen coincidiese con  $\{S_i\}$ .

**DH13.** Obtener  $\alpha_i$  como el ángulo que habría que girar en torno a  $X_i$  (que ahora coincidiría con  $X_{i-1}$ ), para que el nuevo  $\{S_{i-1}\}$  coincidiese totalmente con  $\{S_i\}$ .

**DH14.** Obtener las matrices de transformación  ${}^{i-1}A_i$ .

**DH15.** Obtener la matriz de transformación que relaciona el sistema de la base con el del extremo del robot  $T = {}^0A_1, {}^1A_2 \dots {}^{n-1}A_n$ .

**DH16.** La matriz T define la orientación (sub-matriz de rotación) y posición (sub-matriz de traslación) del extremo referido a la base en función de la n coordenadas articulares. [13]

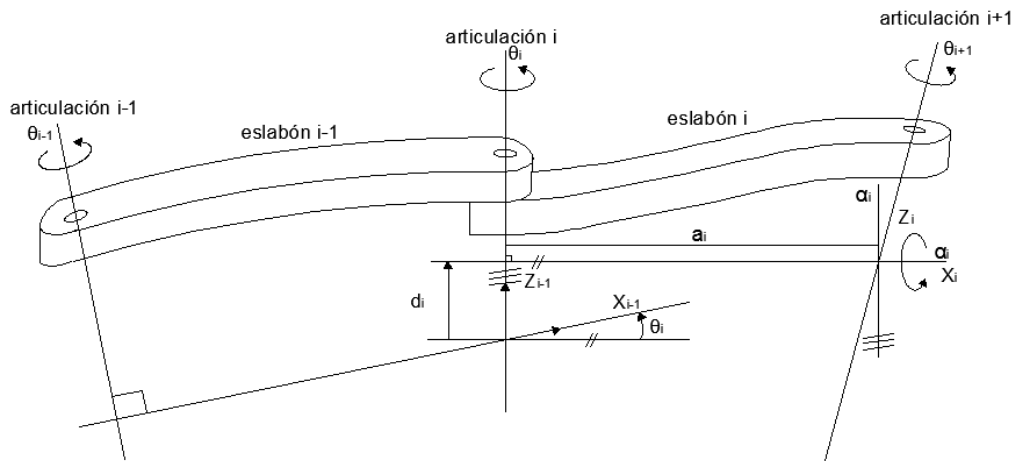


Fig. 2.16 parámetros DH para un eslabón giratorio

Los cuatro parámetros de DH ( $\theta_i, a_i, d_i, \alpha_i$ ) dependen únicamente de las características geométricas de cada eslabón y de las articulaciones que le unen con el anterior y siguiente.

$\theta_i$  Es el ángulo que forman los ejes  $X_{i-1}$  y  $X_i$  medido en un plano perpendicular al eje  $Z_{i-1}$ , utilizando la regla de la mano derecha. Se trata de un parámetro variable en articulaciones giratorias.

$d_i$  Es la distancia a lo largo del eje  $Z_{i-1}$  desde el origen del sistema de coordenadas (i-1)-ésimo hasta la intersección del eje  $Z_{i-1}$  con el eje  $X_i$ . Se trata de un parámetro variable en articulaciones prismáticas.

$a_i$  Es a la distancia a lo largo del eje  $X_i$  que va desde la intersección del eje  $Z_{i-1}$  con el eje  $X_i$  hasta el origen del sistema i-ésimo, en el caso de articulaciones giratorias. En el caso de articulaciones prismáticas, se calcula como la distancia más corta entre los ejes  $Z_{i-1}$  y  $Z_i$ .

$\alpha_i$  Es el ángulo de separación del eje  $Z_{i-1}$  y el eje  $Z_i$ , medido en un plano perpendicular al eje  $X_i$ , utilizando la regla de la mano derecha.

Una vez obtenidos los parámetros DH, el cálculo de las relaciones entre los eslabones consecutivos del robot es inmediato, ya que vienen dadas por las matrices A, que se calcula según la expresión general.

Las relaciones entre eslabones no consecutivos vienen dadas por las matrices T que se obtienen como producto de un conjunto de matrices A. [13]

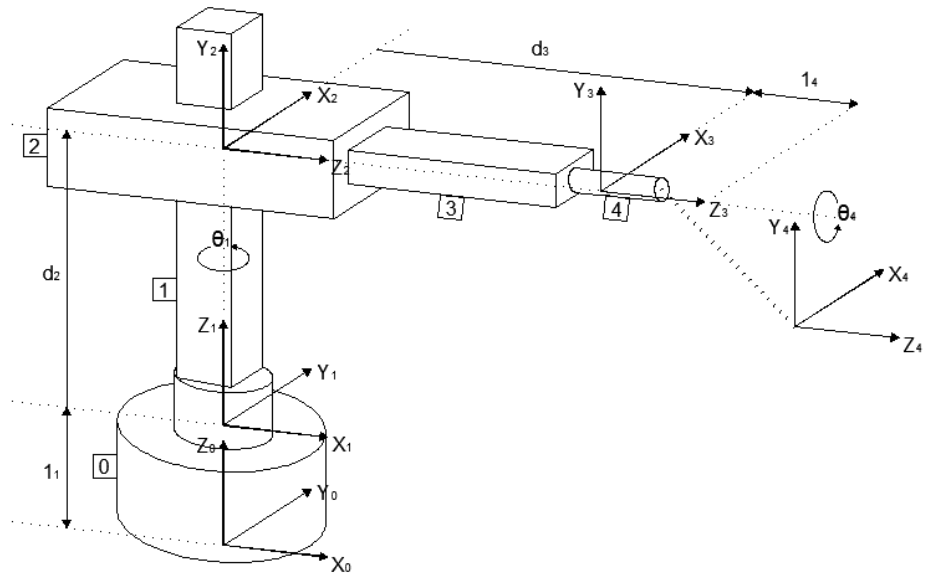


Fig. 2.17 Robot cilíndrico

Obtenida la matriz T, esta expresara la orientación (sub-matriz (3x3) de rotación) y posición (sub-matriz (3x1) de traslación) del extremo del robot en función de sus coordenadas articulares, con lo que quedara resuelto el problema cinemático directo. [13]

Parámetros DH para el robot.				
Articulación	$\theta$	D	A	$\theta$
1	$q_1$	$l_1$	0	0
2	$90^\circ$	$d_2$	0	$90^\circ$
3	0	$d_3$	0	0
4	$q_4$	$l_4$	0	0

Tabla 2.1 Parámetros DH para el robot



Una vez calculados los parámetros de cada eslabón se calculan las matrices A.

$$\begin{aligned}
 A_1 &= \begin{bmatrix} C_1 & -S_1 & 0 & 0 \\ S_1 & C_1 & 0 & 0 \\ 0 & 0 & 1 & I_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} & {}^1A_2 &= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
 {}^2A_3 &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} & {}^3A_4 &= \begin{bmatrix} C_4 & -S_4 & 0 & 0 \\ S_4 & C_4 & 0 & 0 \\ 0 & 0 & 1 & I_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}
 \end{aligned} \tag{2.19}$$

### 2.13 Cinemática inversa.

El objetivo del problema cinemática inverso consiste en encontrar los valores que deban adoptar las coordenadas articulares del robot. [13]

$$q = [q_1, q_2 \dots \dots \dots q_n]^T \tag{2.20}$$

Para que su extremo se posicione y oriente según una determinada localización especial así como se aborda el problema cinemática directo de una manera sistemática a partir de la utilización de un algoritmo y utilizando matrices de transformación el cual puede aplicarse independiente mente de la configuración de un robot, esto no se puede aplicar con el problema cinemática inverso siendo común mente el procedimiento de obtención de ecuaciones fuertemente dependiente de la configuración del robot a diferencia del problema cinemática

directo donde y existen algoritmo y software especializado para determinar las matrices de transformación incluso a través de la figura en el caso de la cinemática inversa por la regular se usan métodos numéricos interactivos, y a velocidad de convergencia e incluso su convergencia en si no siempre está garantizada.[10]

# CAPÍTULO III: BRAZO ROBOT INDUSTRIAL ANTROPOMÓRFICO KUKA KR 16

---

## 3.1 Introducción.

En este capítulo se describe el robot con el que se está trabajando, KUKA Robot Group [6], es un grupo que se encuentra en los lugares principales en el mercado de provisión de robótica industrial, y es además una de las empresas pioneras del mismo. KUKA ofrece una gama de robots industriales y sistemas de robots que cubre las clases de carga y los tipos de robots más habituales.

En 1996 KUKA se convirtió en la primera compañía en lanzar al mercado un robot industrial con unidad de control basada en PC, ofreciendo una trabajada interacción entre software, sistema de control y mecánica, una auténtica aplicación mecatrónica. Ésta es una de las principales características de sus soluciones hasta el día de hoy. Los estándares de KUKA empleados en esta tecnología han convencido a usuarios de robots de todo el mundo, por lo que la compañía es actualmente el líder del mercado de robots industriales con unidades de control de base PC a nivel mundial.

Todos sus robots presentan una plataforma de control de base PC, llamada KR C, siendo posible por tanto aprovechar las ventajas de las tecnologías PC: diagnóstico a distancia, interfaz Windows, bus de campo, etc.

## 3.2 Arquitectura general del sistema robótico

Un sistema robótico estándar industrial, posee sin distinción de un modelo de manipulador específico los mismos parámetros constituyentes. Estos se consideran necesarios como mínimo para el correcto desarrollo de un proyecto completo o aplicación con los robots, a pesar que

pudieran existir subsistemas adicionales que complementen el accionar del robot o del sistema en general.

Es necesario decir que existen ciertas variaciones de la arquitectura general, considerando las diferentes compañías que manufacturan los robots, además de las aplicaciones específicas para las cuales algunos robots son creados. [13]

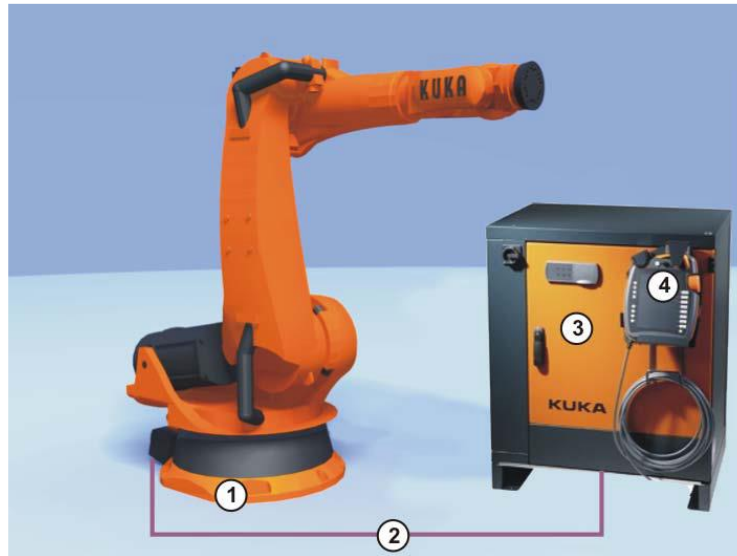
KUKA IRB-2600 pertenece a los robots industriales de la serie KR 16-2S, está ubicado dentro de los robots de tamaño mediano, su carga nominal es de 16 KG, este es un robot antropomórfico de 6 grados de libertad, tienen seis ejes para poder ser montados sobre el piso o contra el techo. Son apropiados para todas las tareas con control de puntos o de trayectoria. Los principales campos de aplicación son:

- Manipulación
- Montaje
- Aplicación de adhesivos, sellantes y medios de conservación
- Procesamiento
- Soldadura MIG/MAG
- Soldadura YAG láser.

El robot industrial consta de los siguientes componentes:

- Manipulador
- Unidad de control del robot
- Unidad manual de programación
- Cables de unión
- Software

En esta figura se observan los elementos:



- |                   |                                 |
|-------------------|---------------------------------|
| 1 Manipulador     | 3 Unidad de control del robot   |
| 2 Cables de unión | 4 Unidad manual de programación |

*Fig. 3.1 Arquitectura básica de un sistema robótico*

Como se especificó, en un sistema general, el modelo o tipo del robot es independiente del sistema como tal, las características del robot como se verá más adelante varían en función de parámetros tales como velocidad, alcance y carga nominal, las cuales son variables propias de la estructura mecánica como tal.

Los cables de unión son los encargados de transmitir las señales (normalmente eléctricas) de control desde el robot hacia el controlador y viceversa.

La unidad de control del robot, alberga los circuitos eléctricos y electrónicos de control, potencia, seguridad y comunicaciones.

El último parámetro constituye un sistema genérico, es la unidad de programación, este componente normalmente está ligado a la versión y características del software y conectividad de la unidad de control, debido a que la unidad de programación es la interfaz de interacción directa con el usuario, espacio en donde se realiza la programación y el manejo del robot.

### 3.2.1 Estructura mecánica

Mecánicamente, un brazo robot antropomórfico industrial está formado por eslabones unidos por medio de articulaciones rotacionales, las mismas que permiten un movimiento relativo entre cada par de eslabones consecutivos [16]. Este tipo específico de robot industrial tiene sus características en base a la estructura de un brazo humano y al igual que la mayoría de los manipuladores industriales actuales, los brazos robots antropomórficos KUKA tienen seis grados de libertad, permitiendo mover cada articulación de forma independiente.

El número de articulaciones está basado en el hecho que es necesario un número mínimo de grados de libertad para posicionar y orientar un cuerpo de cualquier manera en el espacio, tres al menos que definen la posición y al menos dos para orientación de la extremidad dependiendo de la resolución que se desee el robot tenga, en la cual normalmente el robot cuenta con una herramienta específica para realizar un trabajo determinado.

Como se menciona anteriormente los brazos robot antropomórfico industriales KUKA poseen 6 grados de libertad. La figura 3.2 muestra la distribución física del brazo robot y sus grados de libertad. Los grados de libertad están distribuidos de manera que el robot solventa el posicionamiento espacial en cualquier punto de su área de trabajo.



Fig. 3.2 Grados de libertad de IRB2600

Otro parámetro que está relacionado con su estructura física, es su área de trabajo, este importante parámetro proporciona información sobre el alcance y movilidad que tiene el robot. Conceptualmente el área de trabajo o también conocido como volumen de trabajo es el espacio de puntos definidos a los cuales el robot puede llegar, este espacio está definido en base al tamaño y disposición de los eslabones por lo cual sería diferente para cada robot.

El volumen de trabajo normalmente se expresa por medio de dos gráficos descriptivos de una vista superior y una vista lateral del robot dibujando el volumen al cual el robot puede acceder, ambas vistas están debidamente definidas por medidas estándar en milímetros (mm) mostrando el volumen completo, además de una tabla especificando el alcance máximo, carga nominal y carga extra dependiendo de la herramienta a utilizar, repetitividad, unidades de control, etc.

Área de trabajo KUKA KR 16-2S.

<b>ITEM</b>	<b>DESCRIPCIÓN.</b>	<b>VALOR.</b>
Cargas.	Carga	16 Kg
	Carga Adicional	10 Kg
Zona de trabajo.	Max. Alcance	1610 mm
Otros datos.	Numero de ejes	6
	Repetitibilidad	±0.05 mm
	Peso	235 Kg.
	Posiciones de montaje	Suelo, techo
	Unidades de control	KR C2

*Tabla 3.1. Especificaciones mecánicas del robot.*

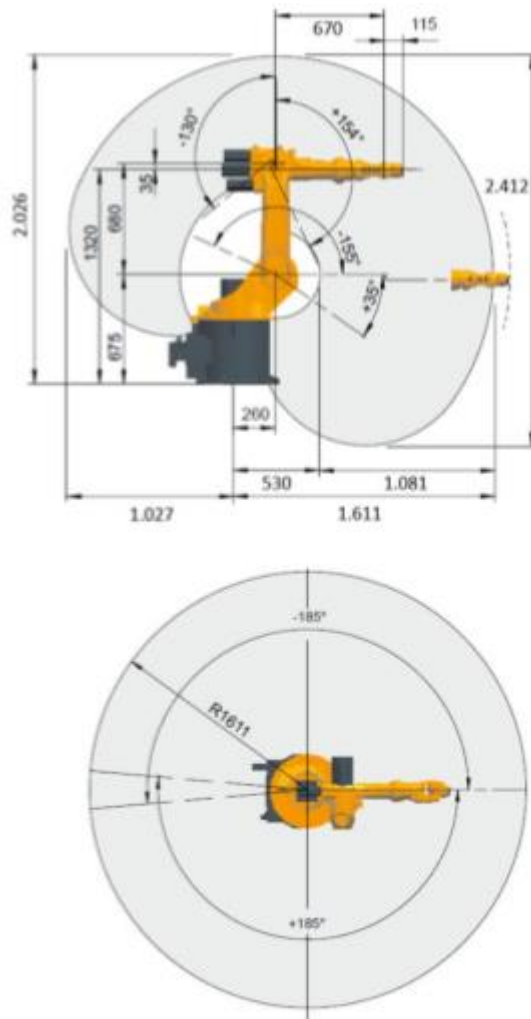
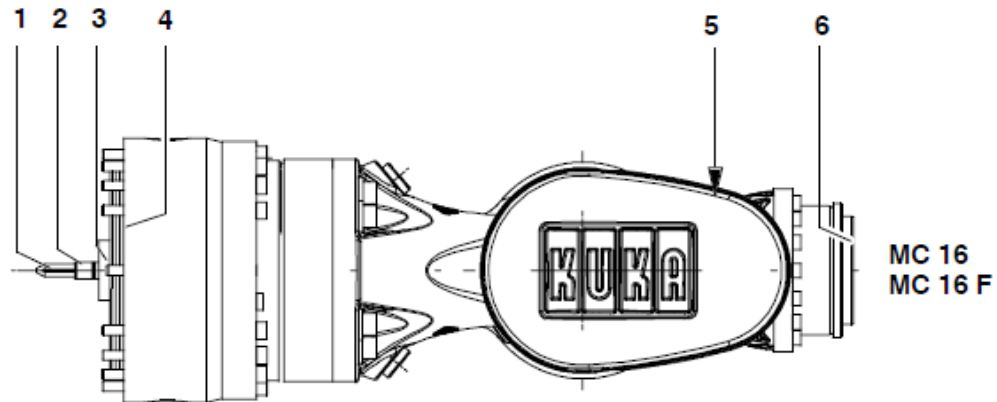


Fig. 3.3 Medidas principales y campo de trabajo KR 16-2S (referidas al software)

### 3.2.1.1 Muñeca

La mecánica del robot va equipada con una muñeca central de tres ejes (Fig. 3.3) para 16 kg de carga útil. La muñeca va fijada con una brida (4) al brazo con uniones atornilladas. El accionamiento de los ejes de la muñeca A4, A5 y A6 se efectúa con los árboles (1, 2, 3). A la brida de acople (6) del eje 6 se montan herramientas. A cada eje le corresponde una instalación de medición (5), a través de la cual, por medio de un comparador electrónico (accesorio), puede comprobarse la posición mecánica cero de cada eje, y ser memorizada en la unidad de control.





- |               |                           |
|---------------|---------------------------|
| 1 Árbol eje 6 | 4 Brida (muñeca--brazo)   |
| 2 Árbol eje 5 | 5 Instalación de medición |
| 3 Árbol eje 4 | 6 Brida de acople         |

*Fig. 3.3 Muñeca central del robot*

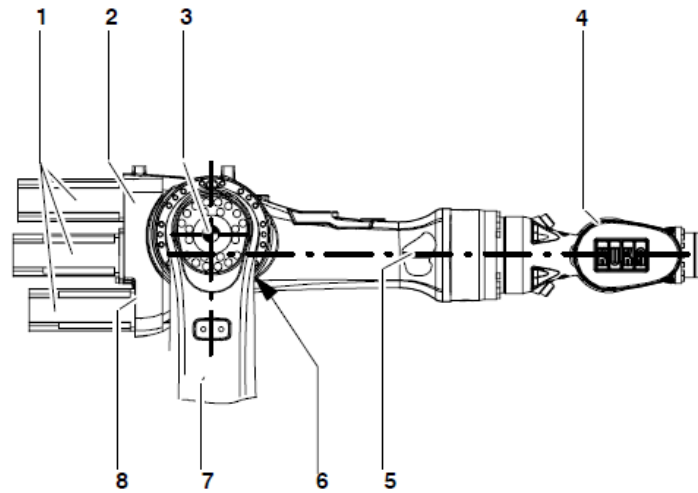
### 3.2.1.2 Brazo

El grupo constructivo del brazo (Fig. 3.4) representa el elemento accionado del eje 3 de la mecánica del robot. El brazo va abridado lateralmente al brazo de oscilación (7) mediante un engranaje con alojamiento integrado y es impulsado por el accionamiento del eje principal A3 (6). El eje de giro (3) del brazo ha sido seleccionado de modo tal que, hasta una carga útil nominal, pueda prescindirse de un contrapeso sobre el brazo como equilibrio de masa.

El rango de giro software útil es de  $+154^\circ$  hasta  $-130^\circ$  para esta mecánica del robot, referido a la posición cero eléctrica del eje 3, que se tiene cuando los ejes longitudinales del brazo y del brazo de oscilación son paralelos. El rango de giro se limita, además de con los interruptores de final de carrera de software, mediante topes mecánicos con efecto amortiguador.

Sobre la parte trasera de la carcasa del brazo (8), van montados los accionamientos para los ejes de la muñeca 4 hasta 6. La carcasa del brazo está construida de fundición de material liviano.

Sobre el lado delantero del brazo se encuentra montada, sobre una brida estandarizada, la muñeca central (4), accionada por las unidades de accionamiento (1) a través de árboles enchufables (5) dispuestos en el interior del brazo. El brazo se encuentra equipado con cuatro taladros roscados en su parte superior a efectos de soportar cargas adicionales.



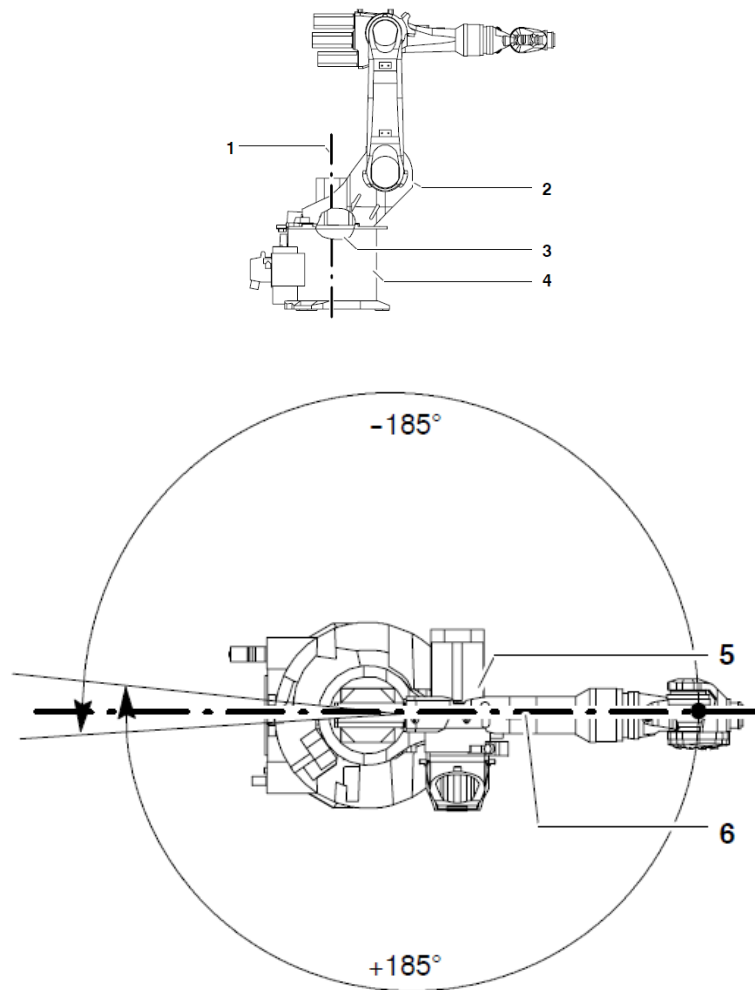
- |  |                                   |
|--|-----------------------------------|
| 1 Unidades de accionamiento para ejes de la muñeca | 6 Accionamiento del eje principal |
| A3   |                                   |
| 2 Brazo  | 7 Brazo de oscilación             |
| 3 Eje de giro A3                                   | 8 Carcasa del brazo               |
| 4 Muñeca central                                   |                                   |
| 5 Árbol  |                                   |

*Fig. 3.4 Brazo del robot*

### 3.2.1.3 Columna giratoria

La columna giratoria (Fig. 3.5) es el grupo constructivo situado entre el brazo de oscilación y la base. Puede girar debido a un reductor (3) atornillado a la base del robot (4). Ejecuta los giros alrededor del eje 1 (1). El rango de giro software útil, medido desde la posición mecánica cero (6), en dirección (+) y (-), es de 185°.

Este se limita, además de con los interruptores de final de carrera de software, mediante topes mecánicos con efecto amortiguador. Este sistema de topes trabaja con un tope de arrastre de acción hacia ambos lados, guiado dentro de la base, que limita mecánicamente el rango de giro grande en dos veces 185°.



- 1 Eje de giro 1
- 2 Columna giratoria
- 3 Reductor A1
- 4 Base
- 5 Accionamiento del eje principal A2
- 6 Posición cero A1

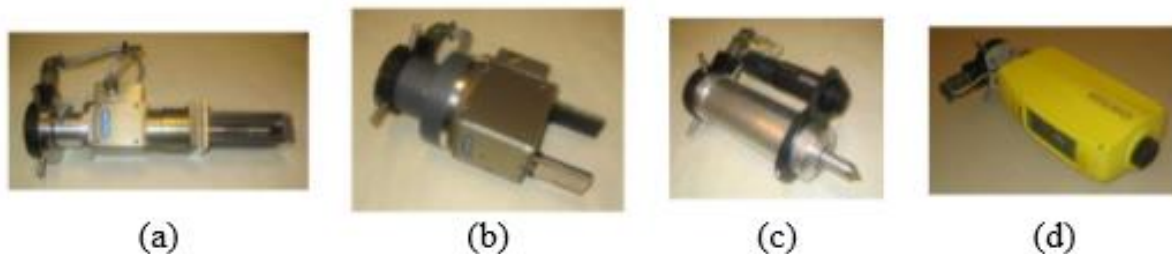
*Fig.3.5 Columna giratoria con rango de giro*

El robot industrial se ha construido de conformidad con el nivel tecnológico actual y con las normas técnicas reconocidas en materia de seguridad. No obstante, en caso de uso indebido puede haber riesgo de lesiones, incluso peligro de muerte, así como riesgo de daños materiales en el robot industrial o en otros.

El robot industrial debe ser utilizado sólo en perfecto estado técnico y para los fines previstos, respetando las normas de seguridad y a sabiendas de los peligros que puedan emanar. Cualquier fallo que pueda afectar a la seguridad deberá subsanarse de inmediato. [3]

### 3.3 Elementos terminales

Una de las principales características de los robots industriales de la empresa KUKA, es su gran versatilidad y sencillas formas de cambio de sus herramientas, también conocidas extremos efectores o elementos terminales. Estos elementos que varían según la aplicación para a cual se va a utilizar el robot, normalmente poseen ciertos protocolos de comunicación y software dedicado para su uso sobre la plataforma de programación debido a esto se les considera como sistemas completos e independientes que pueden ser adquiridos e instalados por separado en la función de la necesidad del usuario.



*Fig.3.6 Tipos de elementos terminales*

Las imágenes muestran algunos ejemplos de extremos efectores que la empresa KUKA dispone para ser adaptados a los dispositivos robóticos, la mayoría de estos sistemas como se mencionó anteriormente basan su control y accionar en programas dedicados cuya programación de alguna manera se fusiona con la programación regular.

El robot sobre el que se realiza el estudio, cuenta con una garra de sujeción controlado por una electro-válvula neumática, la misma que es accionada por medio de pulsos de control enviados desde la tarjeta de adquisición de datos y controlada por un programa de KUKA dedicado a esta, denominado, “KUKA GRIP-TECH”, la fuerza y el nivel de control que esta herramienta posee. La convierte en una de las herramientas más versátiles y muy útiles para aplicaciones de sujeción y traslado.

### **3.4 Operación**

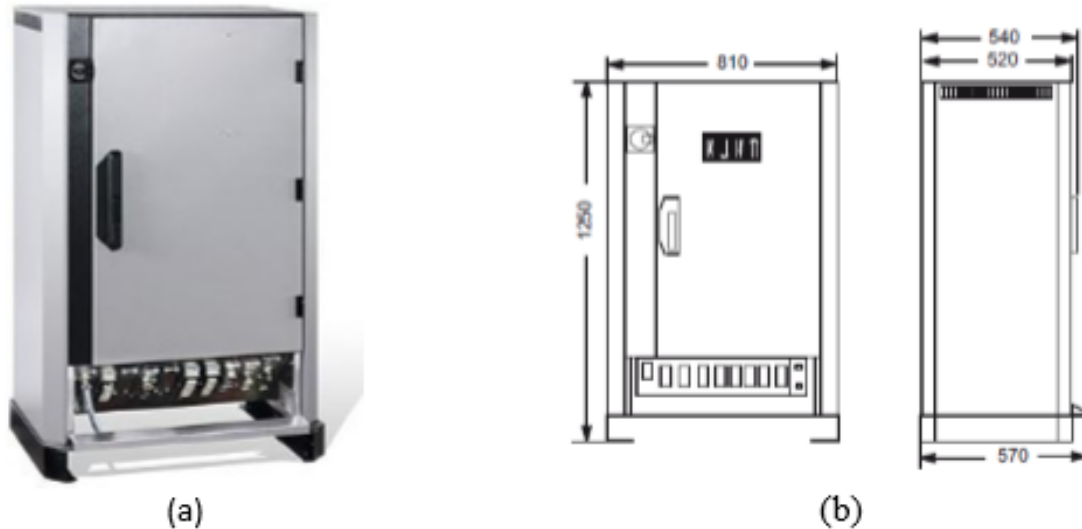
#### **3.4.1 Sistema de control**

Un sistema de control como su nombre lo indica, es el encargado de controlar los procesos que rigen el funcionamiento de todo el robot, desde adquirir las señales de ingreso de la tarjeta de adquisición de datos o sensores internos, hasta interpretar los programas descritos por el usuario de forma textual o guiada; el controlador procesa la información y en base a algoritmos previamente establecidos envía las señales necesarias hacia los actuadores para que el sistema robótico en su totalidad realice una acción determinada como alcanzar un punto, o seguir una trayectoria dictaminada por el usuario, para una aplicación cualquiera.

Las unidades de control desarrolladas por la empresa KUKA poseen ciertas características que los determinan, como nomenclaturas específicas y además tienen todos los subsistemas constituyentes de un sistema de control estándar, siendo estos:

- Dispositivo de almacenamiento.
- Interfaz de usuario.
- Unidad central de procesamiento.
- Unidades de memoria.
- Sistemas de comunicación.
- Controladores de grados de libertad.

Cada uno de estos subsistemas tienen un objetivo para que el sistema de control, y todos están dispuestos a un solo modulo, que será definido en base al robot estudiado. El controlador KR C5 (KUKA Robot Control Version5), es un modelo diseñado en base a las necesidades individuales, que utiliza componentes estándar de procesamiento garantizado fiabilidad y fácil integración de nuevos sistemas.



a) Vista Normal b) Dimensiones y cotas

*Fig. 3.7 Sistema de control del robot*

Físicamente el controlador es un armario de dimensiones estándar dentro del cual se encuentran diferentes componentes que lo constituyen y de los cuales se da una breve descripción a continuación.

Los dispositivos de almacenamiento o memoria se encuentran todos englobados dentro de una única computadora adaptada al armario, la misma que como ordenador principal del sistema solventa todas las características de estos subsistemas.



(a)



(b)

a) Lado frontal de la PC b) Entradas y puertos

*Fig. 3.8 CPU integrado al armario de control*

Los sistemas de comunicación estándar que la computadora posee, están basados en protocolos estándar, tales como RS232, para comunicación serial, USB, TCP para comunicación Ethernet, puertos VGA para la visualización del ambiente de programación e interacción con el usuario y protocolos industriales para la comunicación con las unidades de control para cada grado de libertad.



*Fig. 3.9 Elementos de seguridad y comunicación*

Como se puede ver en la figura 3.9 dentro del armario se encuentran ciertos elementos de seguridad y de comunicación adicionales a los antes definidos, tales como baterías de respaldo y la tarjeta de adquisición de datos.

La memoria interna al igual que la unidad central de procesamiento (CPU), están concebidas dentro de la computadora como un sistema integrado completo, cumpliendo así con las características que una unidad de control para manipuladores robóticos debe tener,

El último parámetro constituyente del controlador y el de mayor importancia son los controladores de grados de libertad, estos controladores en el caso de los robots KUKA son dedicados para cada uno de los grados de libertad y específicamente rigen sus algoritmos de control al posicionamiento y control de velocidad de los servomotores para mover cada articulación, en este caso el controlador posee seis dispositivos dedicados para cada uno de los seis grados de libertad del controlador, como se muestra en la siguiente imagen.



*Fig. 3.10 Accionamientos electrónicos para cada servomotor*



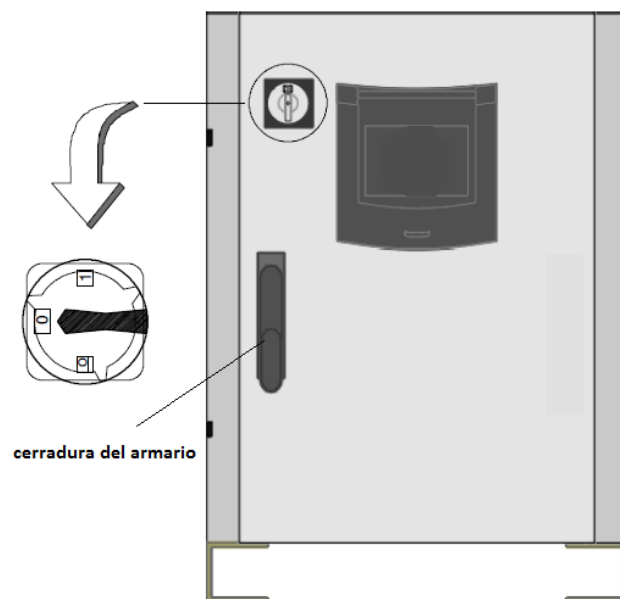
Con el interruptor principal se conectan o desconectan el sistema de robot y el armario de control.

Para impedir de forma segura una reconexión (por ej. durante los trabajos de mantenimiento en el sistema de robot), se puede colocar un candado en el interruptor principal.

En las figura 5 se muestra las funciones del interruptor principal.

### 3.4.1.1 Cerradura del armario

La cerradura del armario es protegida por una tapa que al mismo tiempo sirve de maneta.

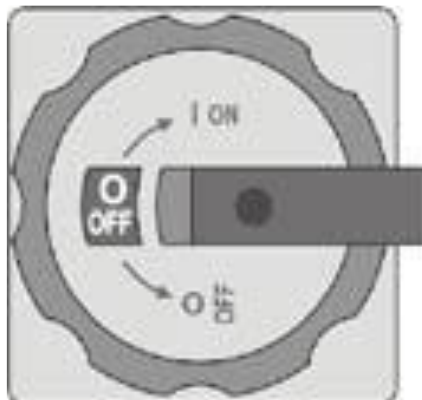


*Fig. 3.11 Vista frontal del armario de control*

### 3.4.1.2 Arranque de la unidad de control

Después de conectar el sistema de robot con el interruptor principal en el armario de control, el ordenador comienza a poner en marcha (cargar) el sistema operativo y el software de control.

En un arranque en frío se visualiza la pantalla de arranque



*Fig. 3.12 Interruptor principal del armario de control*



*Fig. 3.13 Arranque del software*

Después, el display muestra el menú básico que permite crear, seleccionar y ejecutar programas.

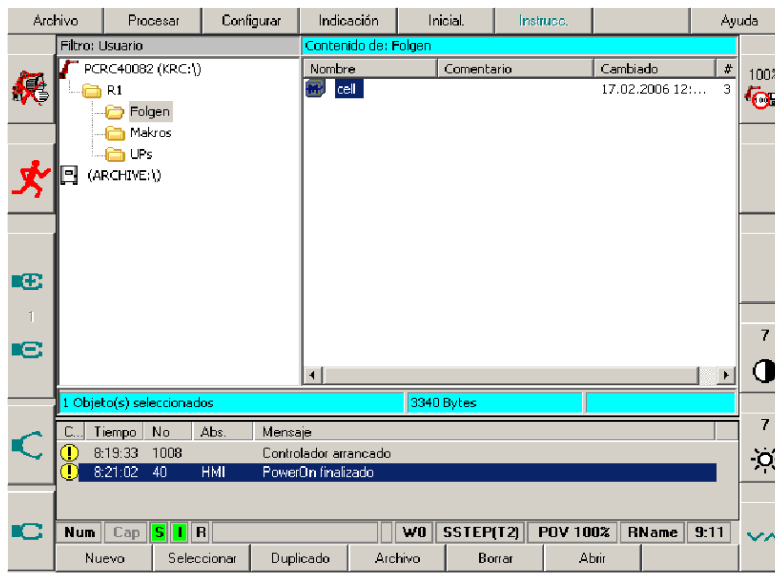


Fig.3.14 Superficie de usuario

### 3.4.1.3 Parada de la unidad de control

Después de desconectar el sistema de robot con el interruptor principal en el armario de control, la unidad de control cierra su propio software y desactiva el sistema operativo.

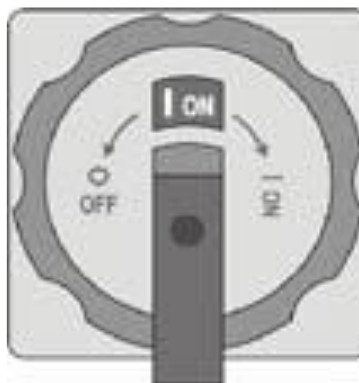


Fig. 3.15 Interruptor del armario de control

### 3.5 Unidad manual de programación kuka smartpad

El smartPAD es la unidad manual de programación del robot industrial. El smartPAD contiene todas las funciones de control e indicación necesarias para el manejo y la programación del robot industrial.

#### 3.5.1. Lado frontal

El smartPAD dispone de una pantalla táctil, a la cual se le nombra, smartHMI se puede manejar con el dedo o un lápiz. No es necesario utilizar un ratón o un teclado externo.

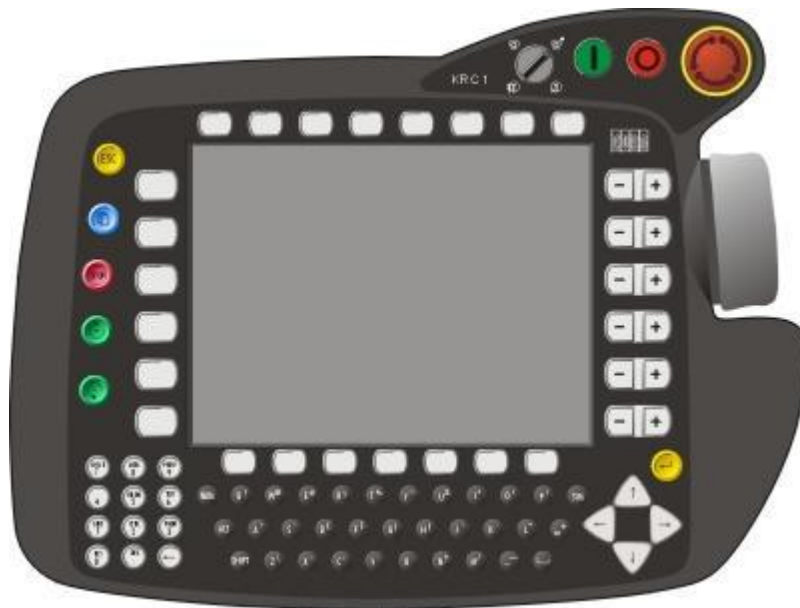


Fig.3.16 KUKA smartPAD, lado delantero

POSICIÓN	DESCRIPCIÓN

1	Botón para desenchufar el smartPAD
2	Interruptor de llave para acceder al gestor de conexiones. El conmutador únicamente se puede cambiar cuando está insertada la llave. El gestor de conexiones permite cambiar el modo de servicio.
3	Dispositivo de PARADA DE EMERGENCIA. Para detener el robot en situaciones de peligro. El dispositivo de PARADA DE EMERGENCIA se bloquea cuando se acciona.
4	Space Mouse: Para el desplazamiento manual del robot.
5	Teclas de desplazamiento: Para el desplazamiento manual del robot.
6	Tecla para ajustar el override de programa.
7	Tecla para ajustar el override manual
8	Tecla de menú principal: Muestra las opciones de menú en el smartHMI.
9	Teclas de estado. Las teclas de estado sirven principalmente para ajustar los parámetros de paquetes tecnológicos. Su función exacta depende del paquete tecnológico instalado.
10	Tecla de arranque: Con la tecla de arranque se inicia un programa.
11	Tecla de arranque hacia atrás: Con la tecla de arranque hacia atrás se inicia un programa en sentido inverso. El programa se ejecuta paso a paso.

12	Tecla STOP: Con la tecla de STOP se detiene un programa en ejecución.
13	Tecla del teclado Muestra el teclado. Generalmente no es necesario mostrar el teclado porque el smartHMI detecta cuando es necesario introducir datos con el teclado y lo abre automáticamente.

Tabla 3.2. Descripción de las funciones del lado frontal del SmartPAD.

### 3.5.2 Parte trasera



- |                             |                             |
|-----------------------------|-----------------------------|
| 1 Pulsador de hombro muerto | 4 Conexión USB              |
| 2 Tecla de arranque (verde) | 5 Pulsador de hombro muerto |
| 3 Pulsador de hombro muerto | 6 Placa de características  |

Fig. 3.17 KUKA smartPAD, parte trasera

ELEMENTO	DESCRIPCIÓN
<p><b>Pulsador de hombre muerto</b></p>	<p>El pulsador de validación tiene 3 posiciones:</p> <ul style="list-style-type: none"> <li>*No pulsado</li> <li>*Posición intermedia</li> <li>*Pulsado a fondo</li> </ul> <p>En los modos de servicio T1 y T2, el interruptor de seguridad debe mantenerse en la posición intermedia para poder efectuar movimientos con el manipulador.</p> <p>En los modos de servicio Automático y Automático Externo, el pulsador de validación carece de función.</p>
<p><b>Tecla de arranque</b></p>	<p>Con la tecla de arranque se inicia un programa.</p>
<p><b>Conexión USB</b></p>	<p>La conexión USB se utiliza, por ejemplo, para el archivado/ la restauración.</p> <p>Únicamente para memorias USB con formato FAT32.</p>

*Tabla 3.3. Descripción de las funciones de parte trasera del smartPAD*

### 3.6 Interfaz de usuario kuka smarthmi

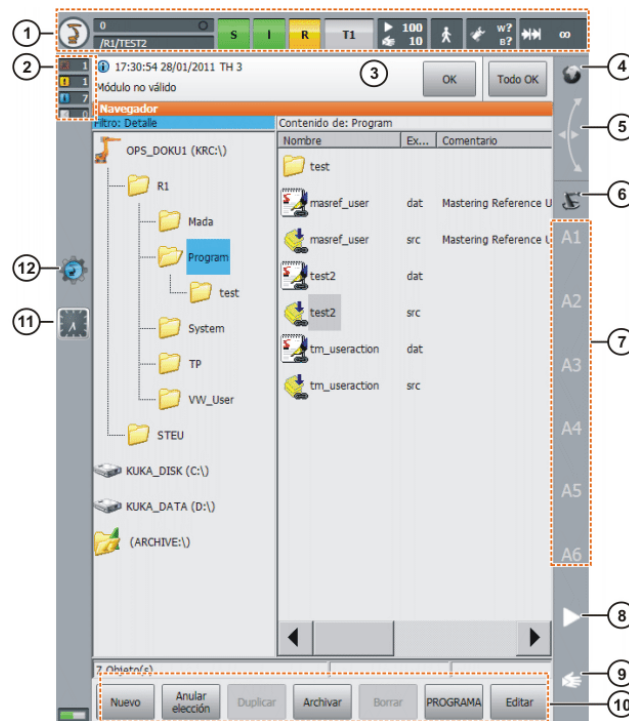


Fig. 3.18 Interfaz de usuario kuka smarthmi

POSICIÓN	DESCRIPCIÓN
1	Barra de estado
2	Contador de mensajes El contador de mensajes muestra el número de mensajes que existe de cada tipo de mensaje. Al tocar el contador de mensajes se ampliará la visualización.
3	Ventana de mensajes Por defecto solo se visualiza el último mensaje. Al tocar la ventana de mensajes se expande y muestra todos los mensajes que existen. Con OK puede confirmarse un mensaje (que pueda confirmarse). Con Todo OK pueden confirmarse todos los mensajes (que puedan confirmarse).
4	Indicador de estado Space Mouse



	<p>Esta visualización muestra el sistema de coordenadas actual para el desplazamiento manual con el Space Mouse. Al tocar la visualización se muestran todos los sistemas de coordenadas y se puede seleccionar otro.</p>
5	<p>Visualización Orientación Space Mouse</p> <p>Al tocar esta indicación, se abre una ventana en la que se muestra y puede modificarse la orientación actual del Space Mouse.</p>
6	<p>Indicador de estado Teclas de desplazamiento</p> <p>Esta visualización muestra el sistema de coordenadas actual para el desplazamiento manual con las teclas de desplazamiento. Al tocar la visualización se muestran todos los sistemas de coordenadas y se puede seleccionar otro.</p>
7	<p>Rotulación de las teclas de desplazamiento</p> <p>Si se ha seleccionado el desplazamiento específico del eje, se mostrarán aquí los números de los ejes (A1, A2, etc.). Si se ha seleccionado el desplazamiento cartesiano, se mostrarán aquí las direcciones del sistema de coordenadas (X, Y, Z, A, B, C).</p> <p>Al tocar la rotulación se muestra el grupo de cinemática que está seleccionado.</p>
8	<p>Override de programa</p>
9	<p>Override manual</p>
10	<p>Barra de botones. Los botones cambian de forma dinámica y se refieren siempre a la ventana que está activa en ese momento en smartHMI.</p> <p>El botón Editar está totalmente a la derecha. Con este botón se pueden activar numerosas instrucciones que van referidas al navegador.</p>

11	<p>Comparador</p> <p>El reloj muestra la hora del sistema. Al tocar el reloj se visualiza la hora del sistema en formato digital y la fecha actual.</p>
12	<p>Símbolo de WorkVisual</p> <p>Si no se puede abrir ningún proyecto, el símbolo tiene una X roja en la parte inferior derecha. Este es el caso, p. ej., si faltan archivos pertenecientes al proyecto. En este caso, el sistema solo está disponible de forma limitada.</p>

Tabla3. 5. Descripción de las funciones de la interfaz de usuario KUKA Smarthmi.

### 3.6.1 Barra de estado

La barra de estado muestra el estado de determinados ajustes centrales del robot industrial. En la mayoría de las visualizaciones se abre una ventana al tocarlas, en la que se pueden modificar los ajustes.

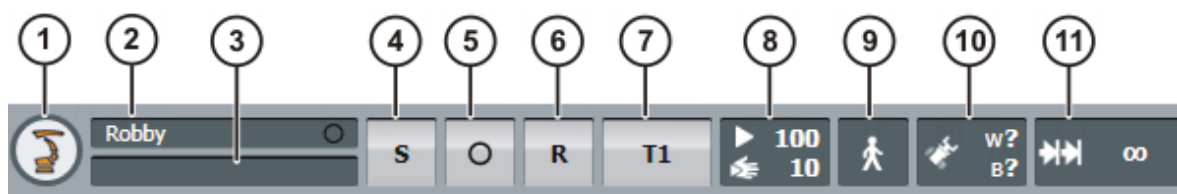


Fig.3.19 Barra de estado

OPCIONES	DESCRIPCIÓN
1	Tecla del menú principal. Muestra las opciones de menú en el smartHMI.

2	Nombre del robot. El nombre del robot puede ser modificado.
3	Cuando está seleccionado un programa, el nombre se visualiza Aquí
4	Indicador de estado Interpretador Submit
5	Indicador de estado Accionamientos Aquí se pueden conectar o desconectar los accionamientos.
6	Indicador de estado Interpretador del robot. Aquí se pueden restaurar o cancelar los programas.
7	Modo de servicio actual
8	Indicador de estado POV/HOV. Visualiza el override del programa actual y el override manual actual.
9	Indicador de estado Modo de flujo de programa. Muestra el modo de flujo de programa actual.
10	Indicador de estado Herramienta/Base. Muestra la herramienta actual y la base actual.
11	Indicador de estado Movimiento manual incremental

*Tabla 3. 6. Descripción de las funciones de la barra de estado.*

### **3.7 Puesta en operación**

Antes de poner en operación el robot es necesario verificar el modo de servicio, para ello el procedimiento es el siguiente.

1. Mover el interruptor del smartPAD para el gestor de conexiones. Se visualiza el gestor de conexiones.
2. Seleccionar el modo de servicio.

3. Volver a colocar el interruptor para el gestor de conexiones en su posición original.

El modo de servicio seleccionado se muestra en la barra de estado del smartPAD, y las cuales se describen en la siguiente tabla.

Modo de servicio	Uso	Velocidades
T1	Para el modo de test, programación y programación por aprendizaje	<ul style="list-style-type: none"> <li>• Verificación del programa:</li> <li>• Velocidad programada, máximo 250 mm/s</li> <li>• Modo manual:</li> <li>• Velocidad de desplazamiento manual, máximo 250 mm/s</li> </ul>
T2	Para servicio de test	<ul style="list-style-type: none"> <li>• Verificación del programa: Velocidad programada</li> <li>• Modo manual: No posible</li> </ul>
AUT	Para robots industriales sin unidad de control superior	<ul style="list-style-type: none"> <li>• Servicio con programa: Velocidad programada</li> <li>• Modo manual: No posible</li> </ul>

AUT EXT	Para robots industriales con unidad de control superior, p. ej. Un PLC	<ul style="list-style-type: none"> <li>• Servicio con programa: Velocidad programada</li> <li>• Modo manual: No posible</li> </ul>
---------	--	--

Tabla 3. 7. Descripción de modo de servicio.

### 3.8 Sistemas de coordenadas

En la unidad de control del robot se encuentran definidos los siguientes sistemas de coordenadas cartesianos:

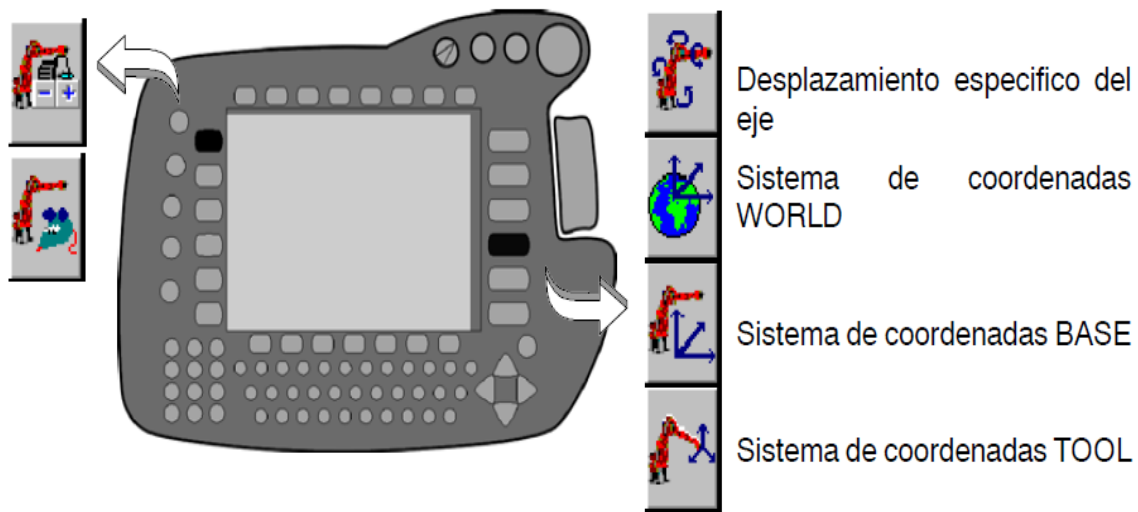


Fig. 3.20 KUKA smartPAD, sistema de coordenadas

#### 3.8.1 World

El sistema de coordenadas world (sistema de coordenadas universales) es un sistema de coordenadas cartesianas de definición fija. Es el sistema genérico de coordenadas para los sistemas de coordenadas base y robot.

Por defecto, el sistema de coordenadas world se encuentra en el pie del robot.

### **3.8.2 Robroot**

El sistema de coordenadas robroot es un sistema de coordenadas cartesianas que siempre se encuentra en el pie del robot. Describe la posición del robot en relación al sistema de coordenadas world.

Por defecto, el sistema de coordenadas robroot se cubre con el sistema de coordenadas world. Con robroot puede definirse un corrimiento del robot respecto al sistema de coordenadas world.

### **3.8.3 Base**

El sistema de coordenadas base es un sistema de coordenadas cartesianas que describe la posición de la pieza de trabajo. Hace referencia al sistema de coordenadas world.

Por defecto, el sistema de coordenadas base se cubre con el sistema de coordenadas world. Es desplazado por el usuario hacia la pieza de trabajo.

### **3.8.4 Tool**

El sistema de coordenadas tool es un sistema de coordenadas cartesianas cuyo punto de trabajo se encuentra en la herramienta.

Por defecto, el origen del sistema de coordenadas tool se encuentra en el centro de la brida. (Se denomina entonces sistema de coordenadas flange)

El sistema de coordenadas tool es desplazado por el usuario en el punto de trabajo de la herramienta.

## CAPITULO IV. DESARROLLO DE LA APLICACIÓN

---

### 4.1 Simulación digital de robots industriales

La simulación digital es una herramienta que presenta varias ventajas aplicada a todo tipo de procesos y situaciones de la vida industrial y científica. Aplicada a la robótica, la posibilidad de simular estaciones en un ordenador nos ofrece una serie de ventajas muy provechosas, como pueden ser:

- Programar estaciones off-line, mientras la producción sigue su curso y sin perder, por tanto, el tiempo de producción que se necesitaría parar.
- Realizar comprobaciones, durante la programación off-line, de los problemas más comunes presentes en éste tipo de soluciones, como pueden ser colisiones, límites de alcance, tiempos de ciclo, etc., permitiendo así anticiparse a ellos antes de la implantación el robot real, y ahorrándonos posteriores revisiones.
- Desarrollar presentaciones útiles y claras con una simulación en 3D a la hora de presentar el producto o servicio a un cliente potencial. Se mejoraría así la percepción que se puede tener en un dibujo CAD en 2D, que a menudo son difíciles de ser percibidos adecuadamente cuando se trata de líneas de ensamblaje y robots.

Estas y otras características hacen de la simulación una técnica útil y provechosa para todo tipo de industria. Este conjunto de ventajas justifican y respaldan el desarrollo y los objetivos del presente proyecto.

Dentro de los software de simulación utilizados se describen a continuación:

#### **4.1.1 Kuka.sim pro**

En cuanto a simulación de sus aplicaciones, la compañía KUKA dispone del programa KUKA.Sim, para ofrecer a sus clientes, antes de iniciar la puesta en servicio, la posibilidad de comprobar el proceso y, si procede, optimizarlo y validarlo.

Éste es el software para programar off-line robots de KUKA. Como una de sus características más destacables, se encuentra la posibilidad de conectarse en tiempo real con la aplicación KUKA.OfficeLite para realizar el control virtual, análisis de ciclos y generación de programas de una estación.

KUKA.Sim Pro permite elaborar y simular garras, pinzas de soldadura y otras estructuras cinemáticas.

Las líneas transportadoras del tipo "push-pull" (empuje-tracción) y las líneas transportadoras con velocidad constante pueden ser simuladas en KUKA.Sim Pro.

La comunicación entre KUKA.Sim Pro está conectado en tiempo real con KUKA.OfficeLite, la unidad virtual de control de KUKA. Este programa corresponde en un 99% al software que opera en las unidades de control "reales" de KUKA. Eso permite pronosticar la duración de los ciclos con extraordinaria exactitud.

#### **4.1.2 Kuka.officelite**

KUKA.OfficeLite es la unidad virtual de control de KUKA, y permite crear y optimizar programas para robots KUKA en cualquier PC.

KUKA.OfficeLite es casi idéntico al software estándar KR C. Gracias al uso de la Interfaz de usuario original KUKA y de la sintaxis KRL, el manejo y la programación fuera de línea coinciden exactamente con los del robot. Los programas se pueden transmitir desde el sistema de programación KUKA.OfficeLite al robot por medio de una red o de un disquete.

Dispone de todo el repertorio de funciones de las respectivas ediciones del software de sistema.



Presenta comprobación de sintaxis mediante el compilador y el interpretador disponibles.

Presenta control completo de la ejecución de un programa de aplicación de robot. Ello permite optimizar la duración de los ciclos aunque éstos no se puedan determinar con exactitud debido a la ausencia de tiempo real preciso.

Las entradas originales se pueden simular.

KUKA.OfficeLite no se puede utilizar para controlar un robot.

Otras de las características generales de estas aplicaciones son:

- Modelado
- Uso de filtros incorporados para importar diseños CAD de otros sistemas o posibilidad de elaborar geometrías con las herramientas CAD de KUKA.Sim Pro.

#### **4.1.3 ABB DE ROBOT STUDIO**

RobotStudio es el software para simulación y programación off-line que la empresa ABB desarrolla para sus robots industriales. Esta empresa es líder en suministro de dichos robots, apuesta por proporcionar soluciones completas a sus clientes, y ya ha instalado más de 150.000 robots en todo el mundo.

RobotStudio forma parte de una familia de productos de software que la empresa ofrece a sus clientes para mejorar su productividad y reducir costes, y que sirven de apoyo a la gestión del ciclo de vida de sus soluciones robóticas.

Con esta herramienta, ABB ofrece las ventajas generales de la programación off-line, entre las que se pueden destacar el permitir programar los robots en PCs sin parar la producción o la posibilidad de preparar los programas con antelación, para mejorar la productividad. [28]

RobotStudio, incorpora herramientas que repercuten en la mejora de la rentabilidad de los sistemas, permitiendo llevar a cabo planes de formación, programación y optimización de sistemas sin parar los robots.

Al instalar RobotStudio, se instala con él el controlador virtual (VC) que es una copia exacta del software real que usan los robots en la producción.

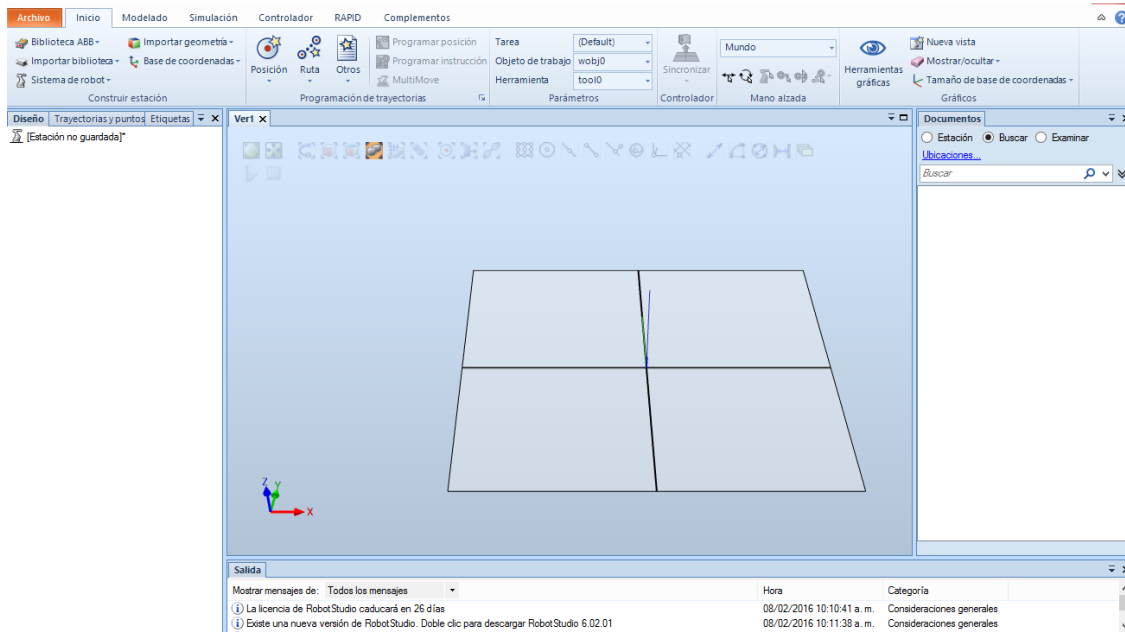


Fig. 4.1 Interfaz de RobotStudio

Así se llevarán a cabo simulaciones muy realistas y se facilita el proceso de carga/descarga de programas al robot hacia el PC.

Algunas características del programa se exponen a continuación:

- Importar CAD: Permite importar una amplia variedad de archivos CAD
- Generación de caminos: Se pueden generar caminos de puntos a partir de una superficie o curva.

- Editor de Programas: Incorpora el software ProgramMaker, que permite realizar y modificar off-line programas RAPID de manera rápida y sencilla (en entorno Windows), repercutiendo en ahorros de tiempo de diseño.
- Permite comprobar si los puntos incluidos en el sistema son alcanzables o no con la configuración actual.
- Teach Pendant (unidad de programación) virtual: El programa dispone de la representación gráfica de un Teach Pendant virtual que permite realizar todas aquellas operaciones que se pueden realizar con el real.
- Tabla de eventos: Utilidad para gestionar eventos como colisiones o simulación de señales I/O. Las señales I/O se pueden visualizar.
- Detección de colisiones: El programa permite detectar colisiones de los diferentes juegos de objetos seleccionados.
- VBA, Visual Basic para Aplicaciones: RobotStudio permite usar VBA, que es una herramienta muy común y potente en aplicaciones informáticas.
- PowerPac's: Existen packs creados por ABB y sus socios, en lenguaje VBA, que extienden el uso del software.
- Auténtica carga y descarga: Todo el programa RAPID puede ser transferido del robot al software de simulación y viceversa sin necesidad de ninguna interfaz o traductor. Es una cualidad única sólo suministrada por ABB gracias a su tecnología VirtualRobot technology.

El simulador RobotStudio funciona sobre RobotWare (software tipo “driver” que se instala previamente), que contiene los programas funciones, configuraciones y datos necesarios para el control del sistema robótico virtual y real. Haciendo posible, la programación y simulación cinemática de las estaciones virtuales.

La configuración del robot la realiza en base:

- Objetivos: puntos (posición, orientación y forma) a los que tiene que acceder el TCP del robot.
- Trayectorias: secuencia de instrucciones de movimiento hacia los objetivos.

- Sistema de coordenadas: diferentes en función de las necesidades de la tarea.
- Configuración ejes del robot: identifica la forma en que el robot accede a un objetivo.

Las instrucciones del robot se producen en el lenguaje RAPID (Robotics Application Programming Interactive Dialogue), es un lenguaje de programación estructurado de alto nivel, abierto, similar al Basic, Pascal o C.

RobotStudio incorpora un editor de código Rapid, de forma que se puede manipular, importar o exportar, las instrucciones que lleva implícitas.

Al igual que el controlador real, el simulador posee un panel de control para activar/ desactivar el sistema robótico, así como teach-pedant virtual.

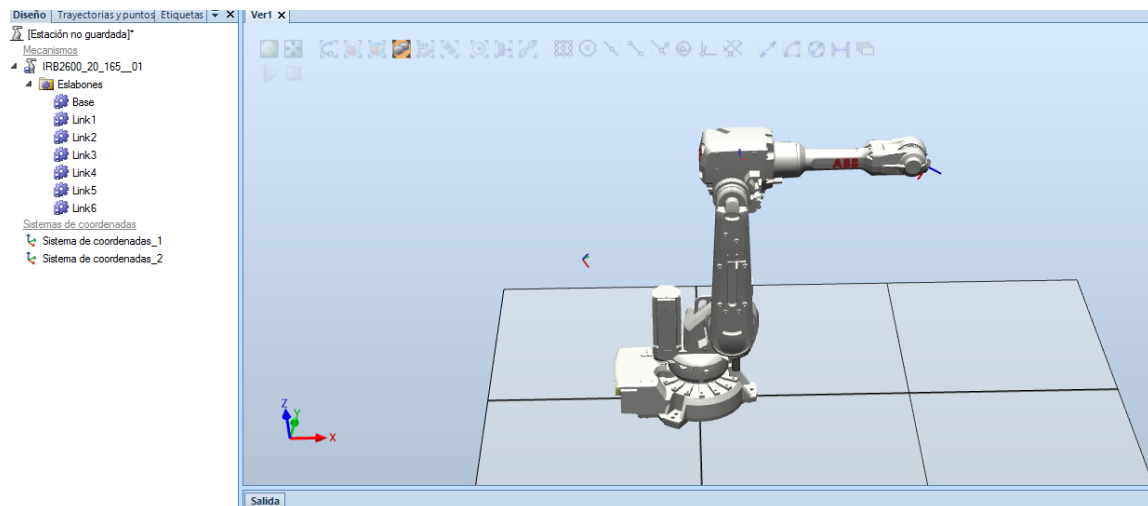


Fig. 4.2 Robot IRB-2600 en la plataforma de RobotStudio

#### 4.1.2 Entornos de simulación basados en Matlab

**MATLAB:** es un producto de la firma norteamericana MathWorks, líder en desarrollo de software de cálculo matemático, para ingenieros y científicos, fundada en 1984. Es una potente

herramienta matemática para el tratamiento de matrices, y por lo tanto ideal para el estudio de la robótica, ya que las matrices son la base para el estudio de los movimientos espaciales de los robots en 3D.

Las aplicaciones que se han encontrado en la red sobre Matlab, proceden de entornos académicos, estos trabajan sobre robots manipuladores industriales y todas ellas son gratuitas, en este proyecto se está trabajando con “ARTE” el cual se describe a continuación.

**ARTE:** “*A Robotic Toolbox for Education*” es una aplicación desarrollada en la Universidad Miguel Hernández de Elche (Alicante), por Arturo Gil. Su departamento de robótica (ARVC) ofrece en la red, una excelente plataforma de estudio de la robótica, desde sus principios teóricos, hasta su implementación en Matlab, RobotStudio y Rapid. Aporta además una excelente documentación, y una extensa colección de videos propios.

Pero sobre todo ARTE, es mucho más que un entorno gráfico, el interfaz que suministra es similar al de ABB, en cuanto a comprensión cinemática del robot, así como variedad de robots que contiene.

Contiene además un editor capaz de procesar el código Rapid, del robot real, para hacer mover el robot virtual, incluso incorpora un traductor de código Matlab a Rapid, una gran ventaja para la interpretación entre diferentes lenguajes de programación de robots.

Todo esto está desarrollado sobre el entorno de Matlab, desde el cual se han desarrollado, funciones, programas y ficheros que generan el control de la simulación de los robots manipuladores.

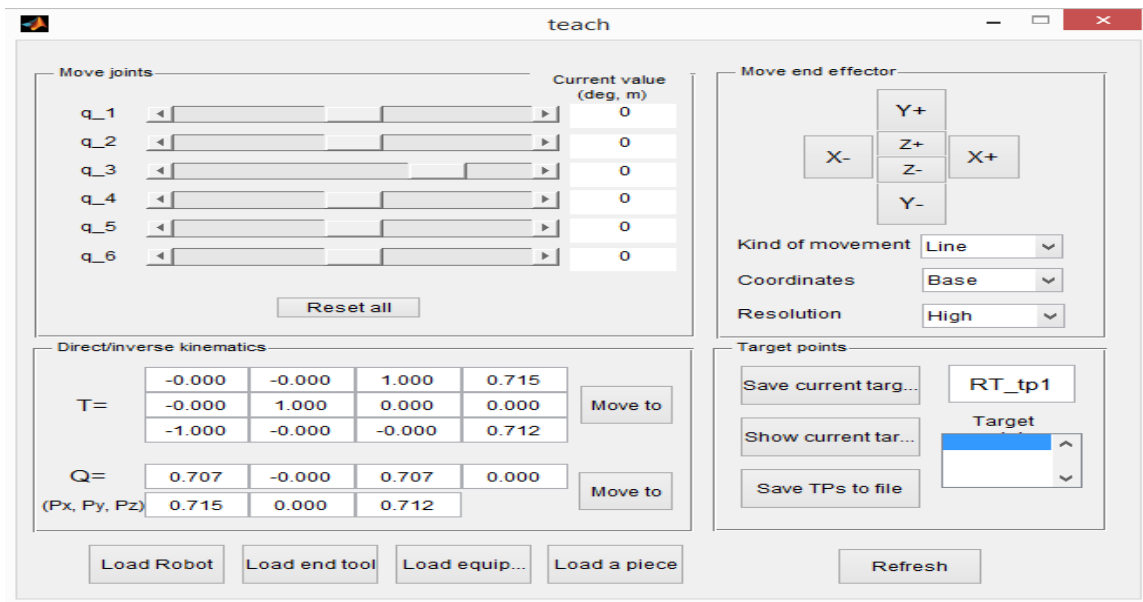
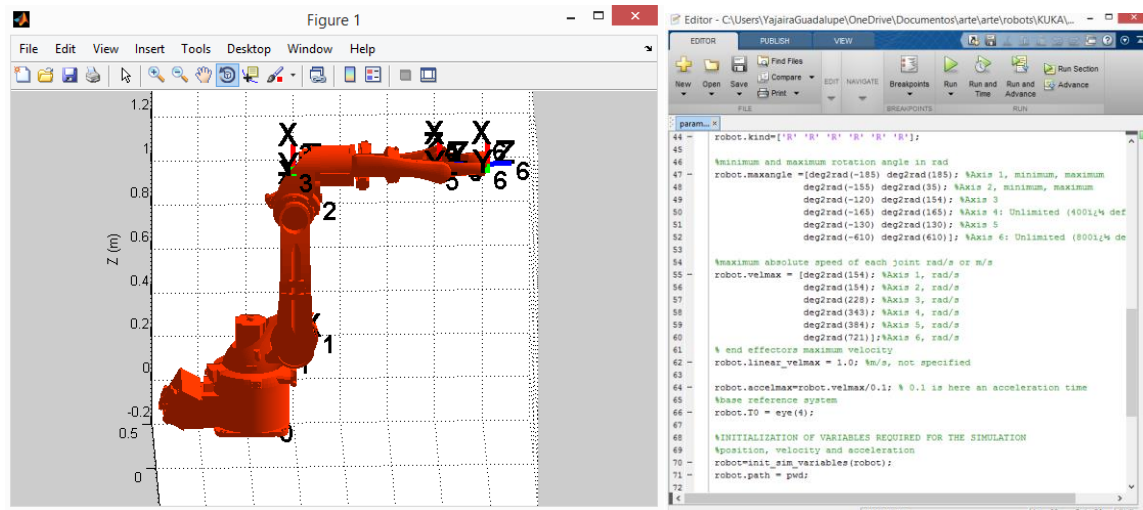


Fig. 4.3 Entorno de simulación de "ARTE" de Matlab

## 4.2 Robotic toolbox de matlab

Otro de los softwares utilizados fue Robotic Toolbox de Matlab, se escogió este software porque es una herramienta de estudio de la robótica, el cual nos ofrece las siguientes ventajas:

Provee gran cantidad de funciones, datos y ficheros, como principios propios de la robótica, fácilmente accesibles.

Facilita la manipulación de datos en forma de vectores, transformaciones homogéneas (necesarios para representar la posición y orientación en el espacio).

Se pueden aplicar a cualquier tipo de robot sin más que conocer sus parámetros cinemáticos y dinámicos.

Posibilita la representación gráfica de la configuración del robot en un entorno 3D, sencillo y muy intuitivo.

Permite la iteración con el entorno gráfico (robot), mediante una botonera (teach), mostrando sus múltiples posiciones y describiendo diversas trayectorias.

El toolbox está en continuo proceso de mejora. Su autor Peter Corke, lo actualiza permanentemente.

Matlab además incorpora otras librerías capaces de mejorar y ampliar otros aspectos del estudio (dinámica, sistema de control, gráficos 3D, etc.).

### **4.3 Descripción de la aplicación**

En este capítulo se describe la unión de dos softwares, con la finalidad de que el alumno aprenda a programar un robot en su entorno real, con sus particularidades en sus lenguajes. Se desarrolló un proyecto con la finalidad de alcanzar los objetivos con el uso de algunas aplicaciones de MATLAB®/ SIMULINK® y otras desarrolladas para robótica, donde de una forma práctica se puede aplicar y visualizar la teoría de robots simulados y reales.

MATLAB® tiene el “Robotic Toolbox®” donde se pueden desarrollar algoritmos teóricos para el modelado de robots, estudio de su cinemática, dinámica y diseño de trayectorias sobre un robot genérico [14] [15].

La aplicación “ARTE®” (“A Robotic Toolbox for Education”) [16] añade a la anterior aplicación el modelo en tres dimensiones (3D) de un gran número de robot industriales, con funciones para obtener sus cinemáticas inversas, de forma que los conceptos teóricos pueden ser aplicados a robots industriales.

Las aplicaciones desarrolladas conseguirán que el alumno pueda comprender la teoría con la herramienta de trabajo que luego va a emplear. Se han realizado pantallas para la visualización del robot en 3D, una aplicación grafica basada en GUIDE<sup>®</sup> de MATLAB<sup>®</sup> [17], y simulaciones del robot en 3D basadas en SimMechanic<sup>®</sup> [18, 19, 20].

La aplicación propuesta está basada en el servidor OPC que dispone ABB [21] y el OPC cliente de MATLAB<sup>®</sup> [22].

Un servidor OPC es el método de conectividad de datos basado en los estándares más populares del mundo. Es utilizado para responder a uno de los mayores retos de la industria de la automatización: cómo comunicar dispositivos, controladores y/o aplicaciones sin caer en los problemas habituales de las conexiones basadas en protocolos propietarios.

Dicha aplicación permite, para un periodo de muestreo dado, mostrar y medir las variables de salida y permanentes del robot desde una aplicación OPC cliente.

Se consideran seis variables de salida en la estación robotizada, una para cada eje del robot. Estas variables serán leídas por el OPC servidor y modificadas desde MATLAB<sup>®</sup> por una aplicación OPC cliente. En la función de visualización en 3D del robot se ha añadido la opción de modificar de forma simultánea el valor de estas variables para definir la posición de los ejes. El procedimiento que el robot ejecuta en su propio lenguaje RAPID lee de forma ininterrumpida las seis variables de los ejes, y se mueve a la posición correspondiente. De esta forma se consigue que la trayectoria del robot sea controlada por la aplicación de MATLAB<sup>®</sup>, y que el robot y su modelo en 3D sigan la misma trayectoria de forma simultánea.

El alumno podrá aplicar esta herramienta tanto en el robot virtual representado con RobotStudio<sup>®</sup> [23] como el real, ya que el servidor OPC es válido para ambos casos. El alumno puede analizar y validar su programa de control en el robot virtual, y una vez verificado su funcionamiento ejecutarlo y verificarlo en el robot real. En cualquier caso, la aplicación es válida para poder ser aplicada con el robot en modo manual en condiciones de máxima seguridad, ya que si se suelta la tecla de hombre muerto, el robot se para automáticamente



#### 4.4 Sistemas robotizados en matlab® / simulink®

MATLAB® / SIMULINK® son dos entornos de cálculo numérico y visualización de datos de la compañía Matworks® [24], con grandes posibilidades para el diseño de sistemas dinámicos y su simulación. Dentro de las aplicaciones comerciales de ambos entornos no existe uno específico para el diseño y simulación de sistemas robotizados, pero diversos autores han desarrollado aplicaciones de software libre usando MATLAB® / SIMULINK® para sistemas robotizados. [25, 26, 27] La aplicación de robótica más conocida para MATLAB® / SIMULINK® es "Robotic Toolbox®" realizada por Peter Corke [14] que desarrolla los contenidos del libro del propio autor [15]. Dicha aplicación dispone de software para robótica fija y móvil. En la parte de robótica fija propone algoritmos para modelar un robot usando el método de Denavit-Hartenberg, con los que se puede obtener la cinemática directa del robot. La aplicación propone un método general de cinemática inversa para un robot con muñeca esférica, que puede ser aplicada a robots PUMA. Los modelos gráficos de los robots son muy simples, pero genéricos, ya que se representan únicamente los ejes y las uniones entre ellos. En la última versión se ha desarrollado un modelo 3D para el robot PUMA. La dinámica de los robots se representa mediante bloques convencionales de Simulink®.

La mayoría de los fabricantes de robots industriales han desarrollado modelos gráficos en 3D utilizando programas de diseño asistido por ordenador (CAD), y estos modelos en muchos casos son libres y se pueden obtener de internet. Del archivo del robot en los principales formatos de CAD se puede pasar a una estructura compuesta por un fichero "\*.xml" con los datos del robot y ficheros "\*.stl" con la información gráfica de cada modelo de brazo del robot. Existen aplicaciones de MATLAB® para leer la información de los ficheros "\*.stl" y obtener graficas en 3D de los brazos del robot en MATLAB®.

La aplicación "ARTE®" ("A Robotic Toolbox for Education") [14] aprovecha estos ficheros para generar modelos de los principales brazos robóticos del mercado.

Esta aplicación dispone de la cinemática directa, basada en Denavit-Hartenberg, e inversa de un conjunto de robots. También dispone de datos dinámicos de alguno de estos robots. Con ello se

puede estudiar los movimientos y trayectorias de un robot industrial viendo su evolución sobre modelos gráficos de 3D. Además, esta aplicación dispone de algoritmos similares a las instrucciones de movimiento del lenguaje RAPID, para comprender la forma en la que se diseñan estas funciones y su aplicación a robot industriales.

La aplicación grafica ROKISIM (Robot Kinematic Simulation) [27] muestra un modelo grafico en 3D de muchos robots industriales y simula sus movimientos básicos.

La aplicación comercial SimMechanic<sup>®</sup> [18] para SIMULINK<sup>®</sup> dispone de una herramienta especial, SimMechanic<sup>®</sup> Link [19], que permite importar modelos de robot realizados con programas de CAD donde la información de su estructura esta en ficheros "\*.xml" y la información sobre sus brazos en ficheros "\*.stl". A partir de esta información, la aplicación desarrolla un modelo SIMULINK<sup>®</sup> (SimMechanic) con la estructura en serie articulación-brazo del robot y un modelo grafico en 3D con la figura del robot. Sobre el diagrama de SIMULINK<sup>®</sup> se pueden hacer cambios para simular los lazos de control de cada brazo del robot y de esta forma, simular la dinámica del mismo.

Se concluye indicando que existen muchas aplicaciones para modelar y simular un robot fijo usando MATLAB<sup>®</sup> / SIMULINK<sup>®</sup>. Algunas son genéricas para cualquier tipo de robot definido con el método de Denavit-Hartenberg y otras aprovechan los modelos de CAD de robot industriales para estudiar la cinemática y dinámica de robots concretos.

#### **4.5 Comunicación de la aplicación propuesta con el robot real**

La comunicación entre la aplicación de MATLAB<sup>®</sup> / SIMULINK<sup>®</sup> y el robot se podrá haber realizado usando múltiples protocolos. Las ventajas e inconvenientes de dos de ellos se resumen a continuación:

Comunicación por OPC: ABB dispone de un servidor OPC que puede ser ejecutado desde una computadora. Esta aplicación detecta los robots conectados a la computadora, ya sean reales o virtuales, y extrae de ellos, en cada período de muestreo fijado, la información del sistema, sus

variables de entradas y salidas, y las variables persistentes del programa Rapid del robot. Desde una aplicación OPC<sup>®</sup> cliente se puede leer y modificar dichas variables. No es preciso que la lectura y escritura estén sincronizadas, cuando se modifique una variable desde el exterior, dicha variable queda modificada en la estación del robot.

La comunicación entre la aplicación de Matlab y el robot de ABB ha sido realizada haciendo uso del servidor OPC proporcionado por ABB [20], y del cliente OPC proporcionado por la aplicación "OPC Toolbox<sup>®</sup>" de MATLAB<sup>®</sup> [21]. La comunicación se ha realizado usando variables de salida analógicas virtuales por una razón de seguridad.

La escritura en variables de salida en RAPID es posible en estado manual y automático, mientras que la escritura en variables persistente solo es posible en automático. Con fines educativos y por seguridad, el robot IRB-2600 se suele usar en modo manual, ya que en este modo se puede parar el robot ante cualquier eventualidad dejando de pulsar el botón de hombre muerto. Se han detenido seis variables de salida virtuales normalizadas entre [-100; 100] para transmitir la información de los seis ejes del robot.

La forma de comunicarse entre MATLAB<sup>®</sup> y el robot IRB-2600 es la siguiente:

MATLAB<sup>®</sup>: Cuando la aplicación de MATLAB<sup>®</sup> quiera cambiar la posición del robot debe modificar el valor de las variables del servidor OPC correspondientes a los ejes del robot.

Robot IRB-2600 (RAPID): El programa en Rapid consiste en un bucle infinito que lee las salidas analógicas correspondientes a los ejes y desplaza al robot a la posición definida por los valores de las variables.

El protocolo que se han empleado proporciona independencia frente al tiempo de lectura y escritura.

Si se envía una trayectoria definida por muchas posiciones de ejes con un período de muestreo muy bajo, puede que el programa Rapid no sea capaz de leer algún punto pero llegara al destino sin quedarse bloqueado. La sincronización entre MATLAB<sup>®</sup> y Rapid depende de la velocidad con que envía los datos a las variables de salida.

## 4.6 Descripción de las aplicaciones.

En este trabajo se han desarrollado tres aplicaciones para modelar y simular el robot IRB-2600 y sus movimientos respectivamente con MATLAB® y después poder mover el robot real, o el virtual si no se cuenta con el robot real.

Las dos primeras aplicaciones hacen uso de las funciones de “Robotic Toolbox®” [13] y “ARTE®” [14]. La tercera aplicación, hace uso de las herramientas de SimMechanic® [18, 19].

### 4.6.1. Aplicación grafica de simulación

Primero se modela el robot IRB-2600, con su cinemática directa e inversa y se programan una función `plot irb2600()` para visualizar las posiciones del robot en 3D. La cinemática directa y las trayectorias del robot se han diseñado utilizando la aplicación de [15], mientras que la cinemática inversa ha sido obtenida de la aplicación de [16].

A partir de los ficheros “\*.stl” del robot, se ha construido una función `plot irb2600()` con las siguientes objetivos. La fig. 1 muestra el robot simulado.

- Dibujar el robot en 3D con la posición de ejes deseada. Se puede dibujar una secuencia de posiciones si la entrada es una matriz de posiciones de ejes. Se puede definir un tiempo de parada entre posiciones que va a ser útil para sincronizar el movimiento de la gráfica y el del robot real.

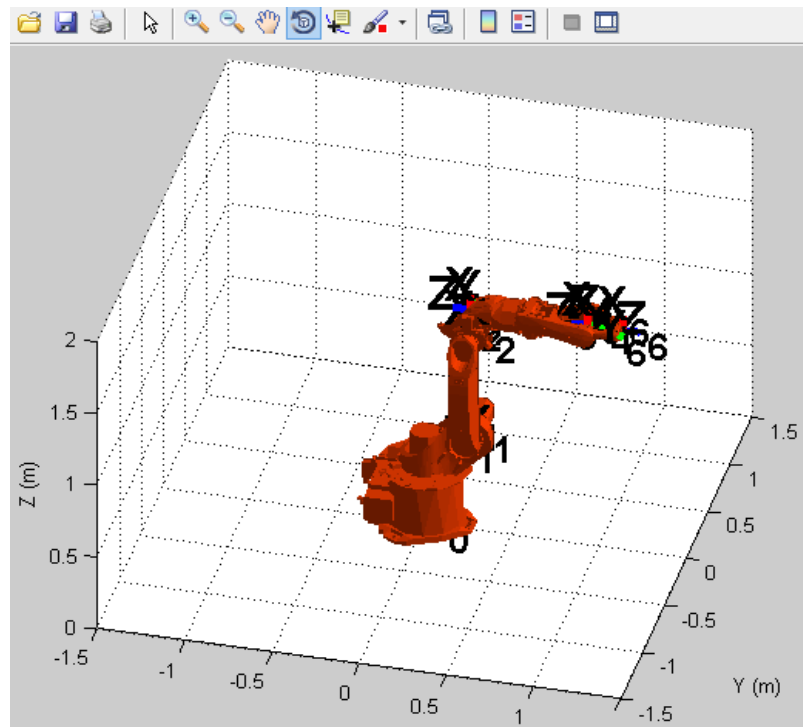


Fig. 4.5 Robot simulado en 3D

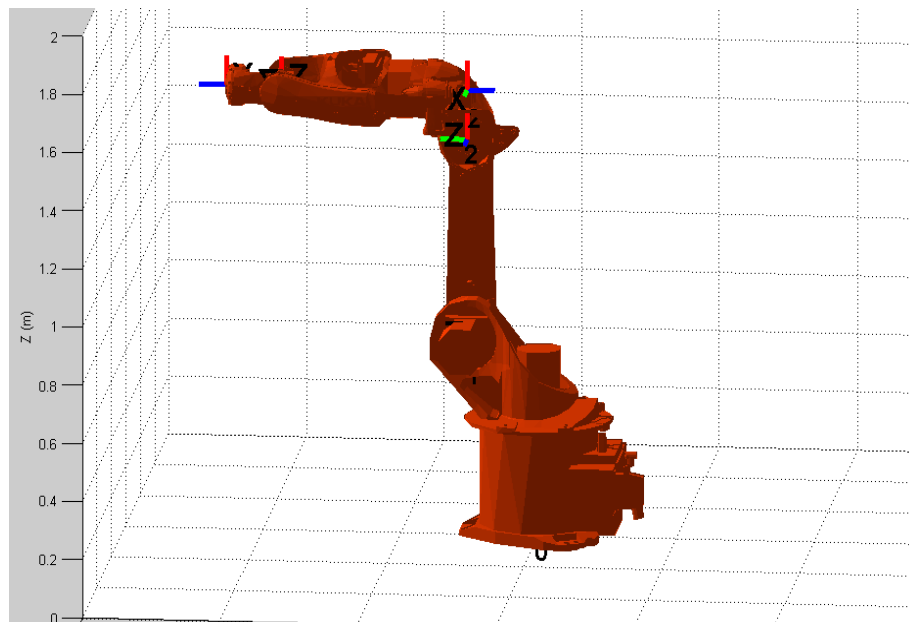


Fig. 4.6 Posición inicial del robot simulado

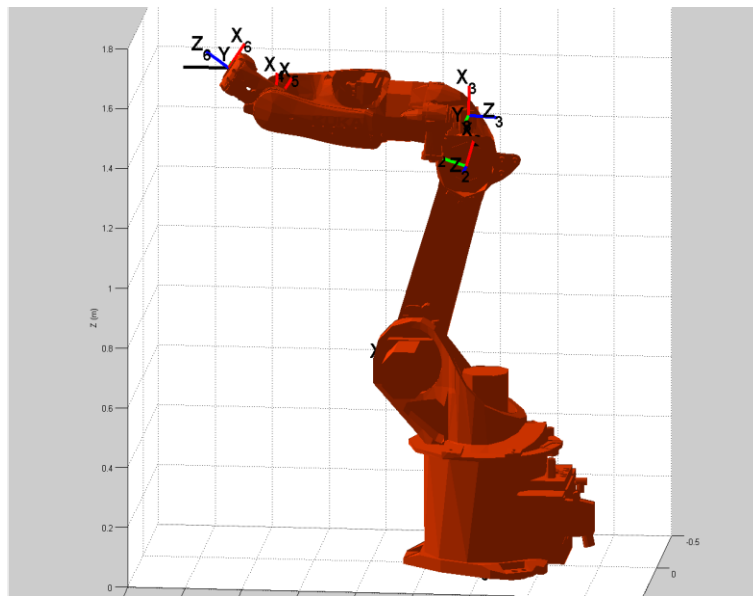


*Fig. 4.7 Posición inicial de robot real*

- El usuario puede programar la posición en las variables de entrada del robot correspondiente, usando el servidor OPC de ABB.
- Cuando el programa de Rapid se activa, el robot se moverá a las mismas posiciones que el robot simulado. El tiempo de espera entre posiciones es un parámetro de sintonización entre el robot y la simulación.

Las siguientes imágenes muestran las pruebas que se hicieron en la fig. 4.6 y 4.7 muestran la posición inicial del robot real y el robot simulado.

Las posiciones finales se muestran en la fig. 4.8 y 4.9.



*Fig. 4.8 Posición final del robot simulado*



*Fig.4.9 Posición final del robot real*

#### **4.6.2. Aplicación grafica (GUIDE®) para el movimiento del robot**

A partir de la función `plot irb2600()` se ha realizado una aplicación grafica en GUIDE® de MATLAB® que permite al usuario realizar de forma segura los movimientos de ejes por ángulos y posiciones cartesianas. Dichos movimientos se visualizan con la función `plot irb2600()` y podrán ser enviadas a las variables de los ejes del robot a través del servidor OPC. Con ello, el usuario podrá mover el robot con una aplicación grafica de MATLAB®, de forma similar a como se mueve con la aplicación FlexPendant de ABB [30].

Las principales características de esta aplicación son las siguientes:

- Movimiento a una posición en coordenadas articulares. Se puede grabar dicha posición para simular una trayectoria.
- Movimiento a una posición del punto de trabajo.
- Se puede grabar dicha posición para simular una trayectoria.
- Movimiento del robot entre las posiciones grabadas anteriormente, ya sean posiciones articulares o del punto de trabajo.
- Movimientos de demostración del robot.
- Conexión con OPC para comunicar datos.



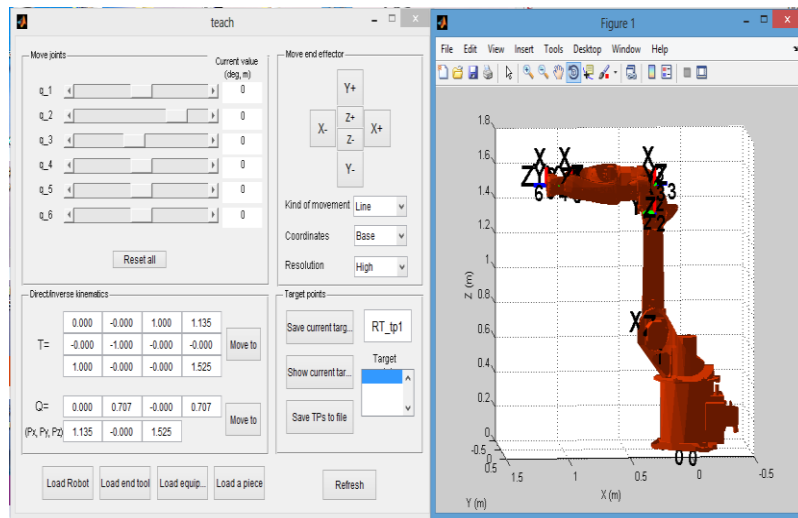


Fig. 4.10 Pantalla inicial de la aplicación gráfica

La fig. 4.10 muestra la pantalla inicial de la aplicación gráfica. Se aprecian los diferentes botones para mover el robot mediante coordenadas articulares, coordenadas de punto de trabajo, y resto de elementos mencionados.

## 4.7 Primeras pruebas para la aplicación de soldadura

Una de las últimas pruebas que se realizaron, fue la programación de la trayectoria para una aplicación de soldadura, la cual primero se probó pintando la trayectoria que debe seguir el robot en una hoja de papel, esto para verificar la distancia de cada punto a soldar entre las dos placas de metal, así como la velocidad que debe seguir el robot.

Las figs 4.11 y 4.12 muestran la secuencia de un video, que se obtuvo durante la programación de un movimiento continuo en zig-zag, el cual es normalmente utilizado en procesos industriales como: pintura y soldadura.

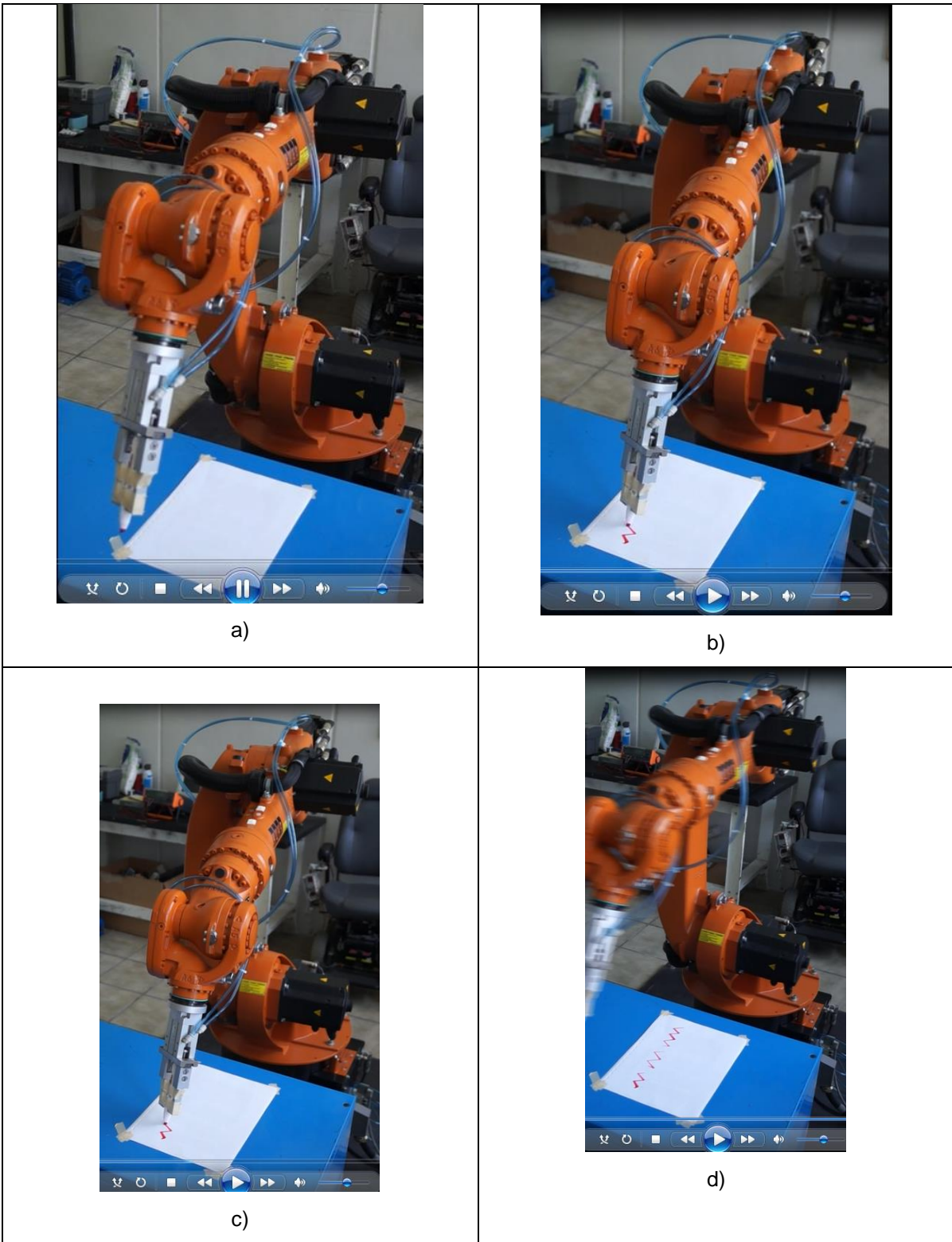


Fig. 4.11 Movimiento continuo en zig-zag.

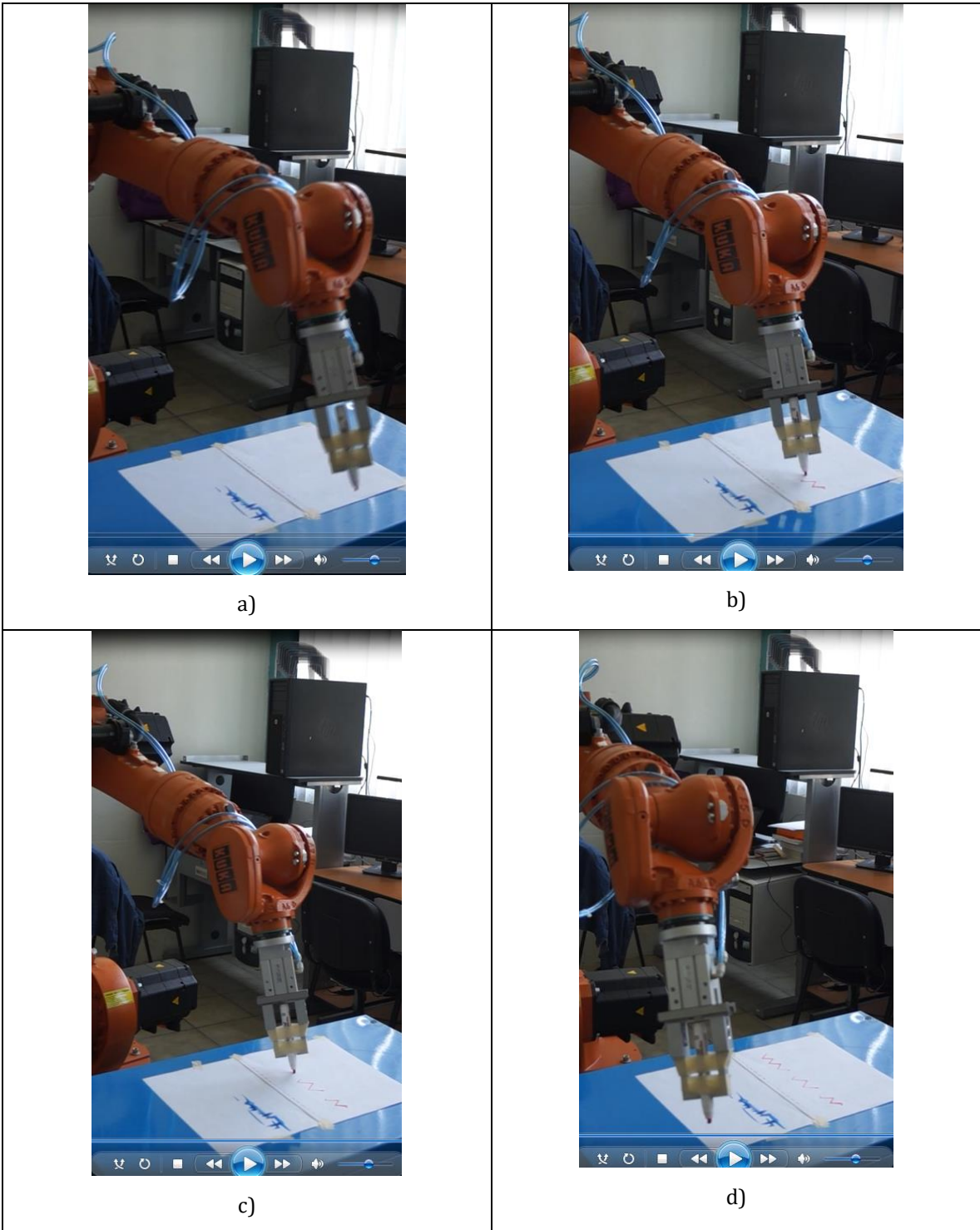


Fig. 4.12 Movimiento continuo en zig-zag, desde otra perspectiva.

# **CAPITULO V. CONCLUSIONES, RECOMENDACIONES Y TRABAJOS FUTUROS.**

---

## **5.1 CONCLUSIONES**

En este capítulo se exponen las principales ideas desarrolladas durante la monografía a manera de resumen permitiendo una adecuada elaboración de conclusiones del proyecto y recomendaciones sobre su uso y futura implementación. Este resumen enfoca los diversos capítulos tratados de forma específica dando una mejor percepción de los temas tratados en cada uno.

### **5.1.1 Capítulo 1**

Durante el desarrollo del primer capítulo se recopiló información conceptual sobre el origen de la robótica, haciendo énfasis en las nociones necesarias para el proyecto. Se realizó y delimitó el planteamiento del problema, se establecieron los objetivos de la tesis, se describió el estado del arte actual en este tema; lo que permitió establecer criterios y dimensionar el proyecto. Las conclusiones de este capítulo son:

El estudio preliminar sobre la actividad actual tanto en el campo de aplicación específica orientado a la educación, fue muy importante debido a que permitió dimensionar el proyecto correctamente, además de establecer importantes parámetros de programación que fueron utilizados para realizar de manera correcta la aplicación, durante el uso de herramientas de comunicación, simulación y control de movimientos.

### **5.1.2 Capítulo 2**

En este capítulo, se realiza el análisis teórico para la generación de trayectoria del robot. Se presentó la cinemática directa e inversa de una cadena de eslabones. Posteriormente se

presentaron los cálculos utilizados para el posicionamiento físico del robot, estos cálculos además fueron utilizados como base para la implementación de las funciones de trayectoria que posteriormente se usaron durante la programación de trayectorias.

### **5.1.3 Capítulo 3**

El tercer capítulo se centró en la descripción de todos los componentes que constituyen el brazo robot antropomórfico bajo estudio, el tipo de robot sobre el cual se desarrolló el proyecto, además de los principales parámetros de su actuar.

Posteriormente se realizó un estudio preliminar sobre la programación del robot industrial, teniendo como principal objetivo, realizar simulaciones digitales de trayectorias que posteriormente podrían ser ejecutadas en tiempo real por el robot físico.

Después se realizó el análisis respectivo sobre la arquitectura de programación y comunicación, se pudo especificar de mejor manera el objetivo alusivo a la apertura que la aplicación pretende dar a los estudiantes.

Las conclusiones de este capítulo se exponen a continuación:

Se conoció a detalle cada uno de los componentes de robot así como el funcionamiento de cada uno de ellos y en su conjunto. El análisis correspondiente de algunos lenguajes para su programación.

También se realizaron prácticas de laboratorio, las cuales se incluyen en los anexos.

### **5.1.4 Capítulo 4**

En este capítulo se explica la propuesta, la cual consiste en unir varias aplicaciones usando el "Robotic Toolbox®" de MATLAB® / SIMULINK® y su comunicación con el software de ABB para controlar, simular, diseñar e implementar los movimiento de un robot industrial. Esto es posible gracias a que la herramienta de comunicación entre el robot y las aplicaciones de MATLAB® la cual se lleva a cabo por medio de un intercambio de información usando el

servidor OPC de ABB y el cliente OPC de MATLAB®. Con esta herramienta los alumnos pueden realizar en un entorno visual la programación previa de los movimientos de cada eslabón del robot para una aplicación específica.

Una vez verificado el diseño de los movimientos por simulación, es posible implementarlos de manera rápida en el robot real, en caso de contar con él. Eliminando movimientos riesgosos tanto para el usuario como para el equipo.

## **5.2 RECOMENDACIONES Y TRABAJOS FUTUROS**

- Se recomienda conseguir las herramientas y terminales necesarias para diversas aplicaciones con el fin de realizar diferentes pruebas, herramientas tales como: soldadura por puntos, autógena, MIG, etc., pintura, manipulación de objetos, etc.
- Adquirir el kit de visión artificial, con lo cual es posible controlar los movimientos del robot haciendo uso de la visión artificial.
- Realizar esquemas de control en lazo cerrado propios para el movimiento de los servomotores, haciendo uso de plataformas digitales ultra rápidas como el DSP y el FPGA.
- Adquirir un segundo robot, al menos, para implementar una estación de trabajo industrial, para la docencia e investigación.
- Se recomienda implementar un sistema de seguridad tipo encapsulado para el usuario, y dentro del área de trabajo del robot, ya que actualmente no cuenta con ningún nivel de seguridad dentro de su entorno de operación.

## 5.3 ANEXOS

### ANEXO A).-PRÁCTICAS DE LABORATORIO.



# TECNOLÓGICO NACIONAL DE MÉXICO

## INSTITUTO TECNOLÓGICO DE APIZACO

### DEPARTAMENTO DE INGENIERIA ELECTRICA

#### PRACTICA N°1 Conocimiento del robot y las partes que lo integran.

Numero de equipo	Integrantes del equipo	Fecha de entrega	Calificación

#### OBJETIVO

Conocer el funcionamiento de cada una de las partes del robot y los elementos que lo integran.

#### OBJETIVOS ESPECIFICOS

- ❖ Identificar claramente cada uno de los eslabones manipulador robótico.
- ❖ Reconocer cada una de las funciones de los elementos más importantes del smartpad.
- ❖ Reconocer cada una de las funciones de los elementos más importantes del gabinete de control.

## ALCANCE

Las habilidades que se requieren transmitir o enseñar al lector serán las bases para un funcionamiento correcto con una adecuada manipulación del robot.

## MEDIDAS DE SEGURIDAD

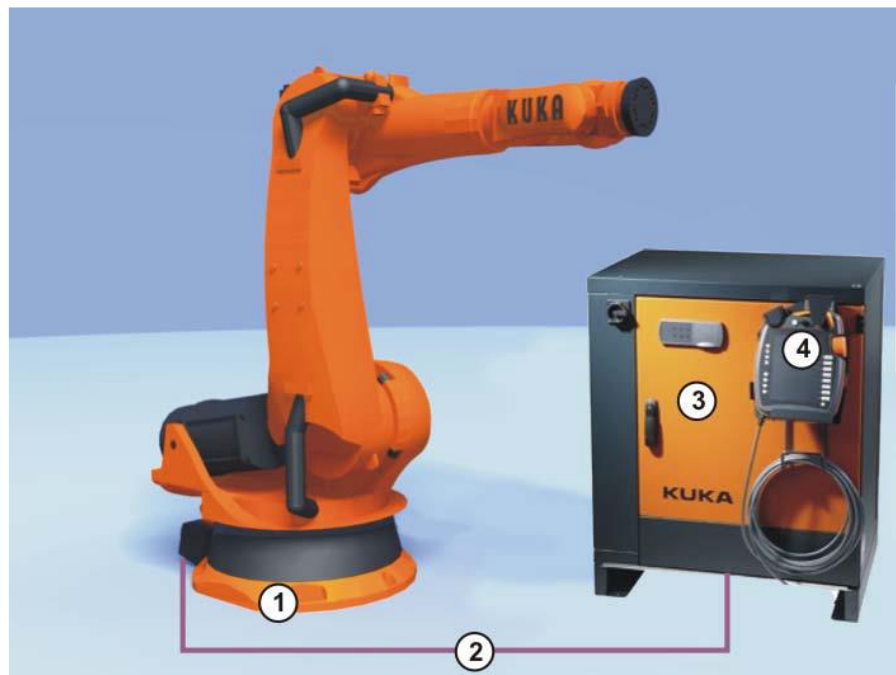
Antes de empezar a manejar el robot es necesario hacer lo siguiente:

- ❖ Observar que no haya ninguna anomalía
- ❖ No tocar ninguna parte móvil del robot durante el funcionamiento
- ❖ No dejar objetos dentro del área de operación del robot
- ❖ Reportar cualquier anomalía al instructor.

## MARCO TEORICO

El robot industrial consta de los siguientes componentes:

- Manipulador
- Unidad de control del robot
- Unidad manual de programación
- Cables de unión
- Software
- Opciones, accesorios



1 Manipulador      3 Unidad de control del robot  
2 Cables de unión    4 Unidad manual de programación



A continuación se describen cada una de las partes del brazo robótico, para un mejor entendimiento del funcionamiento del mismo.

## Muñeca

La mecánica del robot va equipada con una muñeca central de tres ejes (Fig. 1) para 16 kg de carga útil. La muñeca va fijada con una brida (4) al brazo con uniones atornilladas. El accionamiento de los ejes de la muñeca A4, A5 y A6 se efectúa con los árboles (1, 2, 3). A la brida de acople (6) del eje 6 se montan herramientas. A cada eje le corresponde una instalación de medición (5), a través de la cual, por medio de un comparador electrónico (accesorio), puede comprobarse la posición mecánica cero de cada eje, y ser memorizada en la unidad de control.

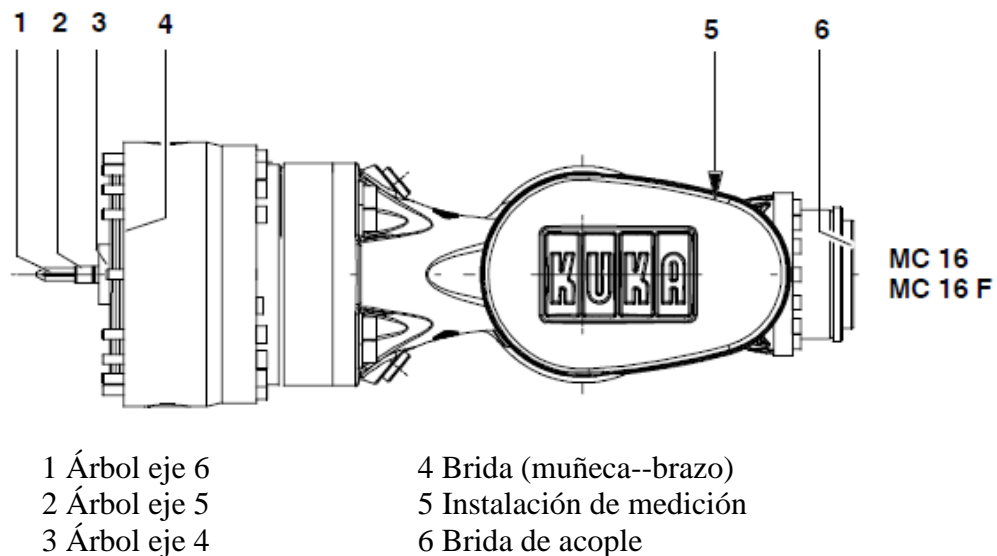


Fig. 1. Muñeca central del robot

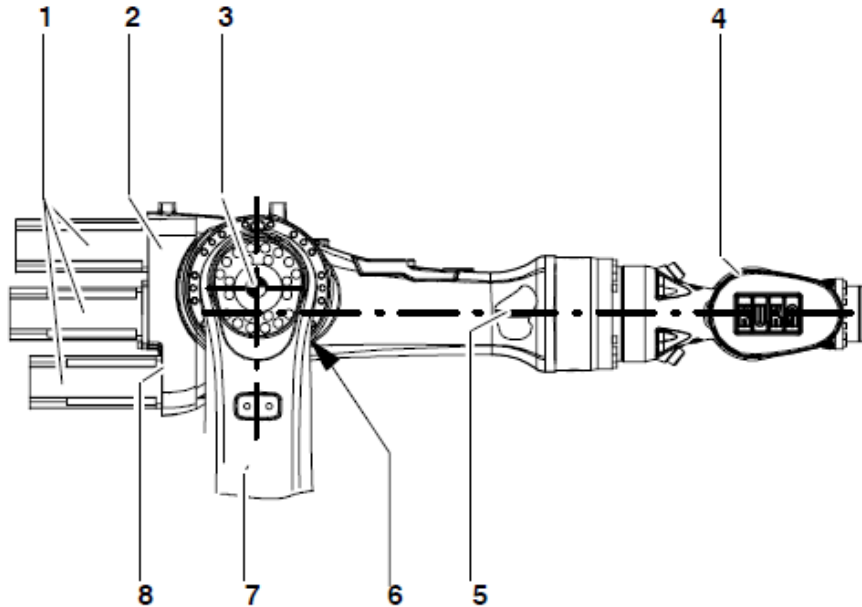
## Brazo

El grupo constructivo del brazo (Fig. 2) representa el elemento accionado del eje 3 de la mecánica del robot. El brazo va abridado lateralmente al brazo de oscilación (7) mediante un engranaje con alojamiento integrado y es impulsado por el accionamiento del eje principal A3 (6). El eje de giro (3) del brazo ha sido seleccionado de modo tal que, hasta una carga útil nominal, pueda prescindirse de un contrapeso sobre el brazo como equilibrio de masa.

El rango de giro software útil es de  $+154^{\circ}$  hasta  $-130^{\circ}$  para esta mecánica del robot, referido a la posición cero eléctrica del eje 3, que se tiene cuando los ejes longitudinales del brazo y del brazo de oscilación son paralelos. El rango de giro se limita, además de con los interruptores de final de carrera de software, mediante topes mecánicos con efecto amortiguador.

Sobre la parte trasera de la carcasa del brazo (8), van montados los accionamientos para los ejes de la muñeca 4 hasta 6. La carcasa del brazo está construida de fundición de material liviano.

Sobre el lado delantero del brazo se encuentra montada, sobre una brida estandarizada, la muñeca central (4), accionada por las unidades de accionamiento (1) a través de árboles enchufables (5) dispuestos en el interior del brazo. El brazo se encuentra equipado con cuatro taladros roscados en su parte superior a efectos de soportar cargas adicionales.



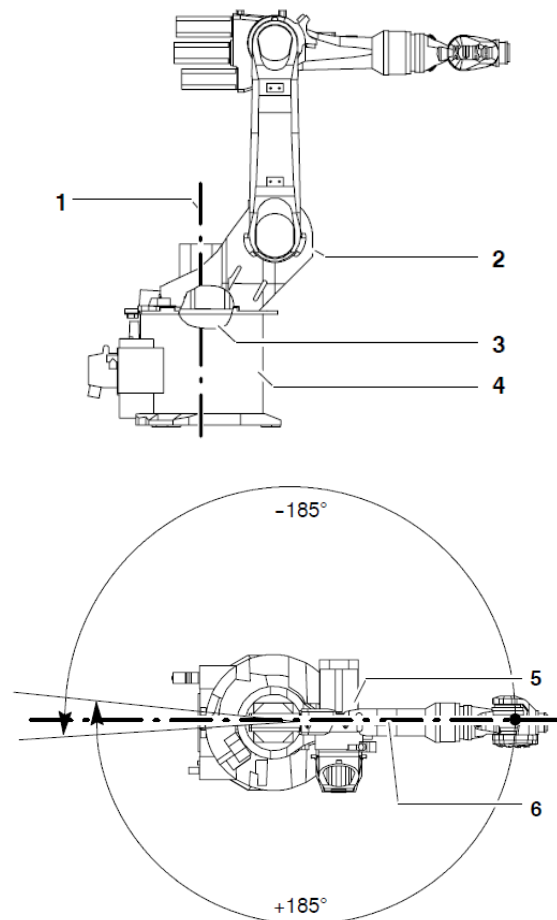
- |  |                                      |
|--|--------------------------------------|
| 1 Unidades de accionamiento para ejes de la muñeca | 6 Accionamiento del eje principal A3 |
| 2 Brazo  | 7 Brazo de oscilación                |
| 3 Eje de giro A3                                   | 8 Carcasa del brazo                  |
| 4 Muñeca central                                   |                                      |
| 5 Arbol  |                                      |

Fig. 2. Brazo del robot

### Columna giratoria

La columna giratoria (Fig. 3) es el grupo constructivo situado entre el brazo de oscilación y la base. Puede girar debido a un reductor (3) atornillado a la base del robot (4). Ejecuta los giros alrededor del eje 1 (1). El rango de giro software útil, medido desde la posición mecánica cero (6), en dirección (+) y (--), es de 185°.

Este se limita, además de con los interruptores de final de carrera de software, mediante topes mecánicos con efecto amortiguador. Este sistema de topes trabaja con un tope de arrastre de acción hacia ambos lados, guiado dentro de la base, que limita mecánicamente el rango de giro grande en dos veces 185°.



- |                                      |                     |
|--------------------------------------|---------------------|
| 1 Eje de giro                        | 2 Columna giratoria |
| 3 Reductor A1                        | 4 Base              |
| 5 Accionamiento del eje principal A2 | 6 Posición cero A1  |

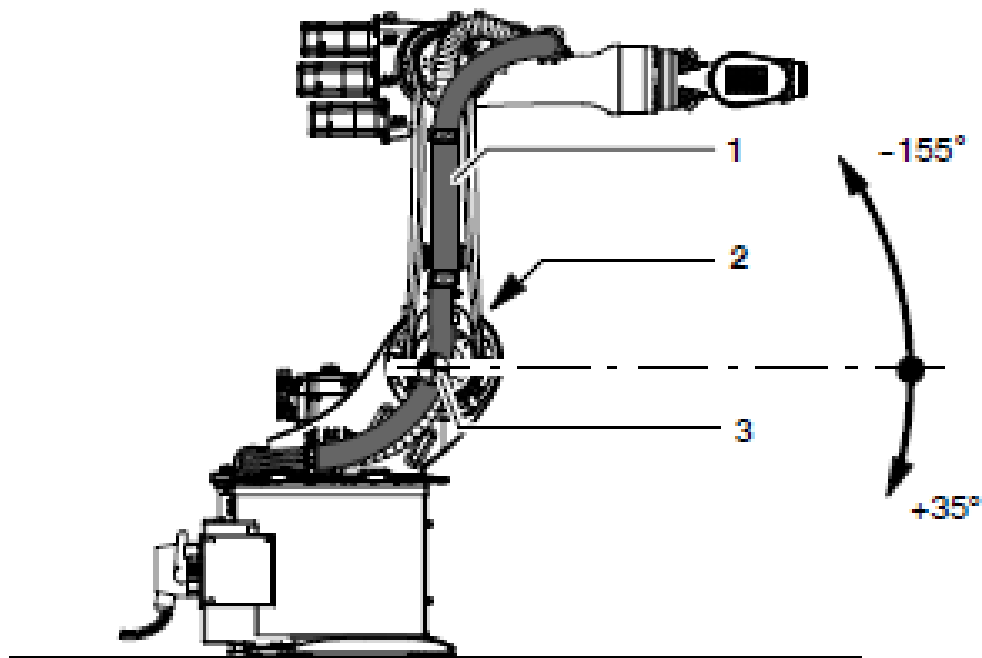
**Fig. 3 Columna giratoria con rango de giro**

El robot industrial se ha construido de conformidad con el nivel tecnológico actual y con las normas técnicas reconocidas en materia de seguridad. No obstante, en caso de uso indebido puede haber riesgo de lesiones, incluso peligro de muerte, así como riesgo de daños materiales en el robot industrial o en otros.

### **Brazo de oscilación**

El brazo de oscilación (Fig. 4) es el elemento accionado del eje 2. Gira en un ángulo software útil de  $+35^\circ$  hasta  $-155^\circ$  – referido a la posición cero eléctrica del eje 2, que corresponde a la posición horizontal del brazo de oscilación en Fig. 4 alrededor del eje de giro 2 (3). El rango de

giro software útil se limita, adicionalmente a los límites de carrera software, por medio de topes mecánicos con efecto amortiguador.



- 1 Brazo de oscilación
- 2 Accionamiento del eje principal A2
- 3 Eje de giro 2

**Fig. 4 Brazo de oscilación con rango de giro**

#### **ARMARIO DE CONTROL**

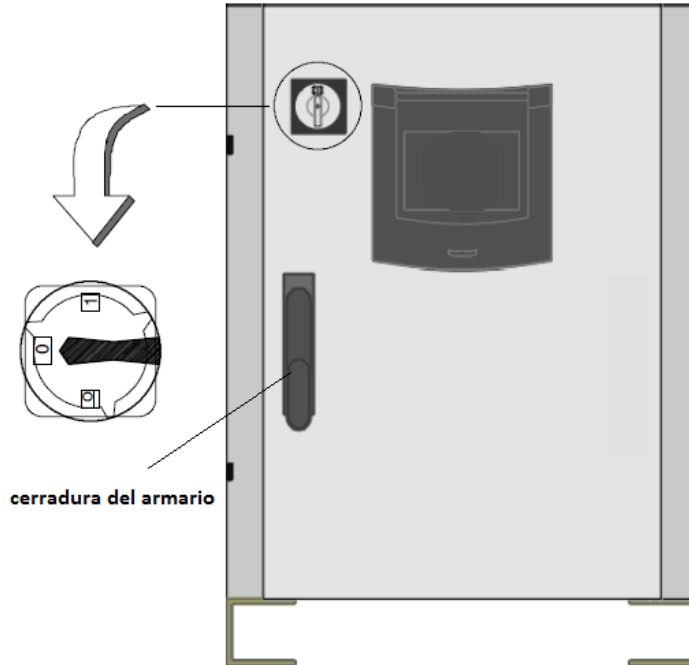
Con el interruptor principal se conectan o desconectan el sistema de robot y el armario de control.

Para impedir de forma segura una reconexión (por ej. durante los trabajos de mantenimiento en el sistema de robot), se puede colocar un candado en el interruptor principal.

En las figura 5 se muestra las funciones del interruptor principal.

## Cerradura del armario

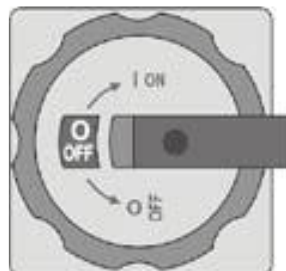
La cerradura del armario es protegida por una tapa que al mismo tiempo sirve de maneta.



**Fig. 5** Vista frontal armario de control "KR C1"

## Arranque de la unidad de control

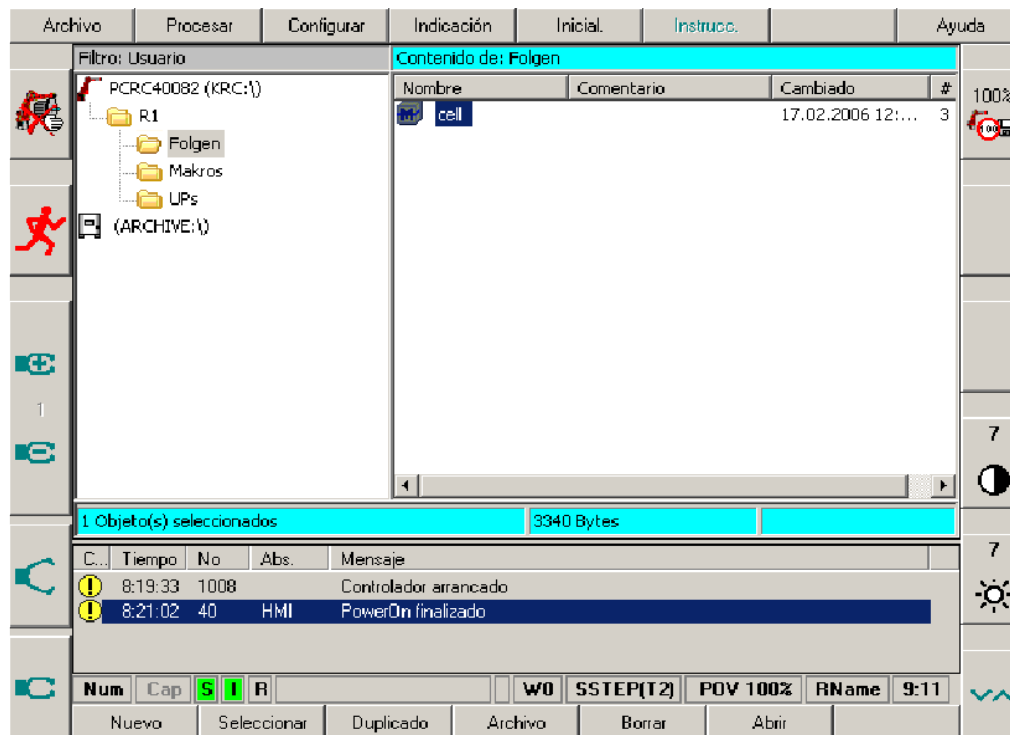
Después de conectar el sistema de robot con el interruptor principal en el armario de control, el ordenador comienza a poner en marcha (cargar) el sistema operativo y el software de control. En un arranque en frío se visualiza la pantalla de arranque





**Fig. 6** Arranque del software VKRC

Después, el display muestra el menú básico que permite crear, seleccionar y ejecutar programas.



**Fig. 7** Superficie de usuario

## Parada de la unidad de control

Después de desconectar el sistema de robot con el interruptor principal en el armario de control, la unidad de control cierra su propio software y desactiva el sistema operativo.



## PROCEDIMIENTO

1. Realizar el procedimiento de encendido del brazo robot.
  - ✓ **Procedimiento:** Colocar el interruptor principal de la unidad de control del robot en ON.
  - ✓ El sistema operativo y el KSS arrancan automáticamente.
  - ✓ Si el KSS no se inicia automáticamente, p. ej. porque está deshabilitado el arranque automático, iniciar el programa StartKRC.exe en el directorio C:\KRC\BIN.
  - ✓ Si la unidad de control debe integrarse en la red, el inicio puede durar más tiempo.
2. Se realizara una explicación que permita al estudiante identificar claramente cada una de las partes que constituye el sistema robótico.

## RECURSOS

- ❖ Robot manipulador KUKA IRB2600
- ❖ Unidad manual de programación SmarPAD
- ❖ Armario de control KUKA.

## EVALUACION

- ❖ El estudiante deberá rendir un informe escrito, una vez realizada la práctica, informe que deberá contener los siguientes aspectos:
- ❖ Describir a través de un esquema todas las partes constitutivas del brazo robot, en el que se describan las funciones de cada una de ellas.
- ❖ Un escrito sobre las características de movimiento de cada uno de los ejes de desplazamiento con los que cuenta la máquina.
- ❖ Conclusiones y recomendaciones sobre lo que se realizó en la práctica.



# TECNOLÓGICO NACIONAL DE MÉXICO

## INSTITUTO TECNOLÓGICO DE APIZACO

### PRACTICA N°2 Unidad manual de programación kuka smartpad como manipulador primario

Numero de equipo	Integrantes del equipo	Fecha de entrega	Calificación

#### OBJETIVO

Conocer el funcionamiento de cada una de las teclas y los elementos más importantes de la unidad manual de programación smartpad, para un correcto funcionamiento del robot.

#### OBJETIVOS ESPECIFICOS

- ❖ Identificar claramente el smartpad como herramienta fundamental para facilitar la manipulación del manipulador robótico.
- ❖ Reconocer cada una de las teclas y elementos más importantes del smartpad.

#### ALCANCE

Las habilidades que se requieren transmitir o enseñar al lector serán las bases para un funcionamiento correcto con una adecuada manipulación del robot.



## MEDIDAS DE SEGURIDAD

Antes de empezar a manejar el robot es necesario hacer lo siguiente:

- ❖ Observar que no haya ninguna anomalía
- ❖ No tocar ninguna parte móvil del robot durante el funcionamiento
- ❖ No dejar objetos dentro del área de operación del robot
- ❖ Reportar cualquier anomalía al instructor.

## MARCO TEORICO

Según el procedimiento de la práctica, es de importancia manejar los recursos que nos da la unidad manual de programación, pues con este se graban las posiciones principales de las rutinas que se van a ejecutar futuramente, no solamente se pueden grabar posiciones si no los programas completos.

El smartPAD es la unidad manual de programación del robot industrial. El smartPAD contiene todas las funciones de control e indicaciones necesarias para el manejo y la programación del robot industrial, a continuación se detallan la explicación de cada una de las teclas del smartpad.

### LADO FRONTAL

El smartPAD dispone de una pantalla táctil, a la cual se le nombra, smartHMI se puede manejar con el dedo o un lápiz. No es necesario utilizar un ratón o un teclado externo.

### Funciones Importantes del teclado



Una acción que se ha iniciado se puede interrumpir en cualquier momento con la tecla ESC.



Es posible cambiar entre las ventanas de los programas, de estado y de mensajes utilizando la ventana Tecla de selección. La ventana activa se indica mediante un fondo azul



Al pulsar la tecla Stop detiene un programa que se ejecuta en modo automático.



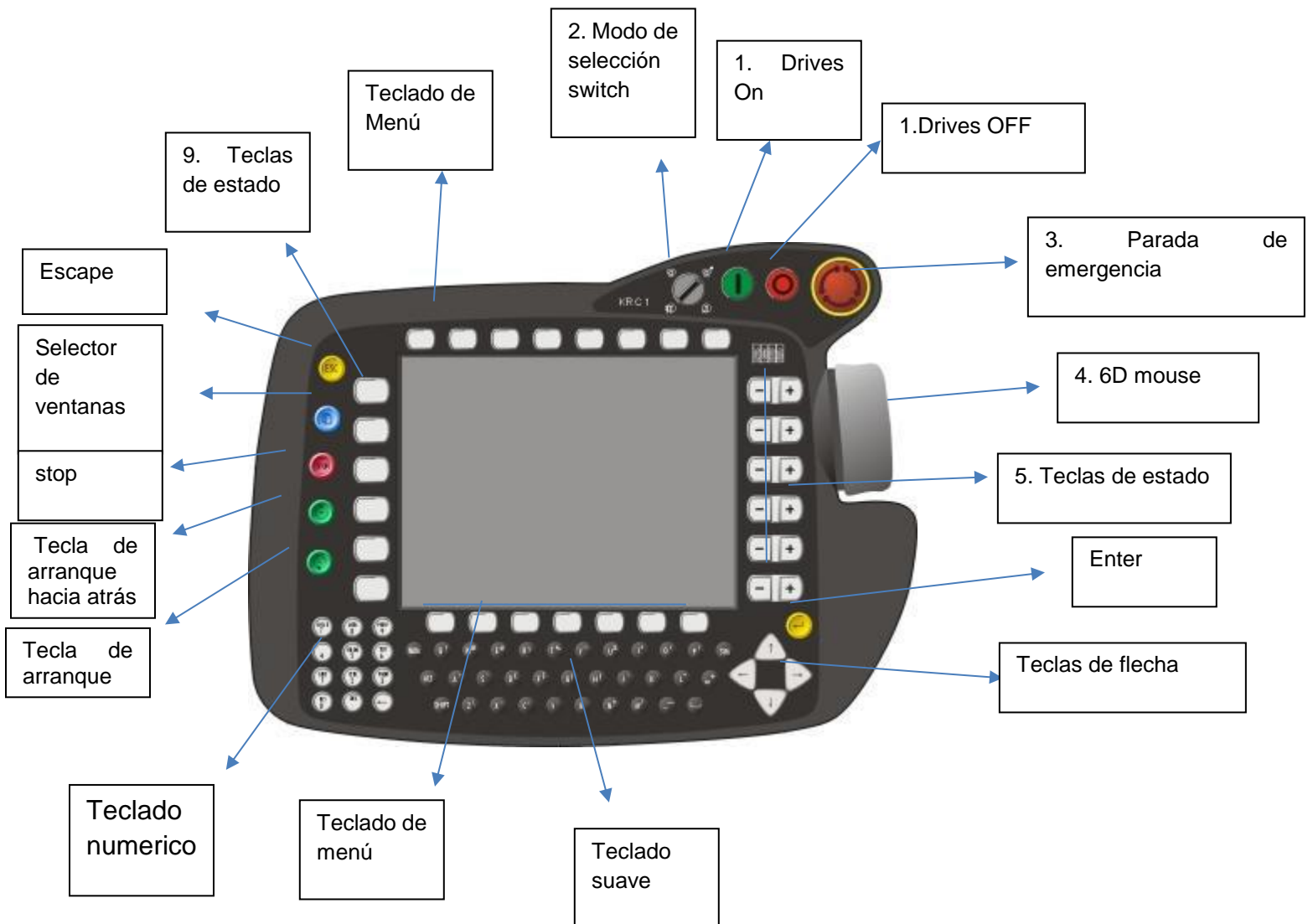
Al presionar el Programa START Inicia delanteros clave de un programa que ha sido seleccionado.



Si el programa de inicio hacia atrás se pulsa la tecla, los bloques de movimiento se ejecutan paso a paso hacia el principio del programa.



Se utiliza la tecla Intro, por ejemplo, para completar los comandos o para confirmar las entradas en formularios, etc.



POSICIÓN	DESCRIPCIÓN
1	Botón para desenchufar el smartPAD
2	Interruptor de llave para acceder al gestor de conexiones. El conmutador únicamente se puede cambiar cuando está insertada la llave. El gestor de conexiones permite cambiar el modo de servicio.
3	Dispositivo de PARADA DE EMERGENCIA. Para detener el robot en situaciones de peligro. El dispositivo de PARADA DE EMERGENCIA se bloquea cuando se acciona.
4	Space Mouse: Para el desplazamiento manual del robot.
5	Teclas de desplazamiento: Para el desplazamiento manual del robot.
6	Tecla para ajustar el override de programa.
7	Tecla para ajustar el override manual
8	Tecla de menú principal: Muestra las opciones de menú en el smartHMI.
9	Teclas de estado. Las teclas de estado sirven principalmente para ajustar los parámetros de paquetes tecnológicos. Su función exacta depende del paquete tecnológico instalado.
10	Tecla de arranque: Con la tecla de arranque se inicia un programa.
11	Tecla de arranque hacia atrás: Con la tecla de arranque hacia atrás se inicia un programa en sentido inverso. El programa se ejecuta paso a paso.
12	Tecla STOP: Con la tecla de STOP se detiene un programa en ejecución.
13	Tecla del teclado Muestra el teclado. Generalmente no es necesario mostrar el teclado porque el smartHMI detecta cuando es necesario introducir datos con el teclado y lo abre automáticamente.

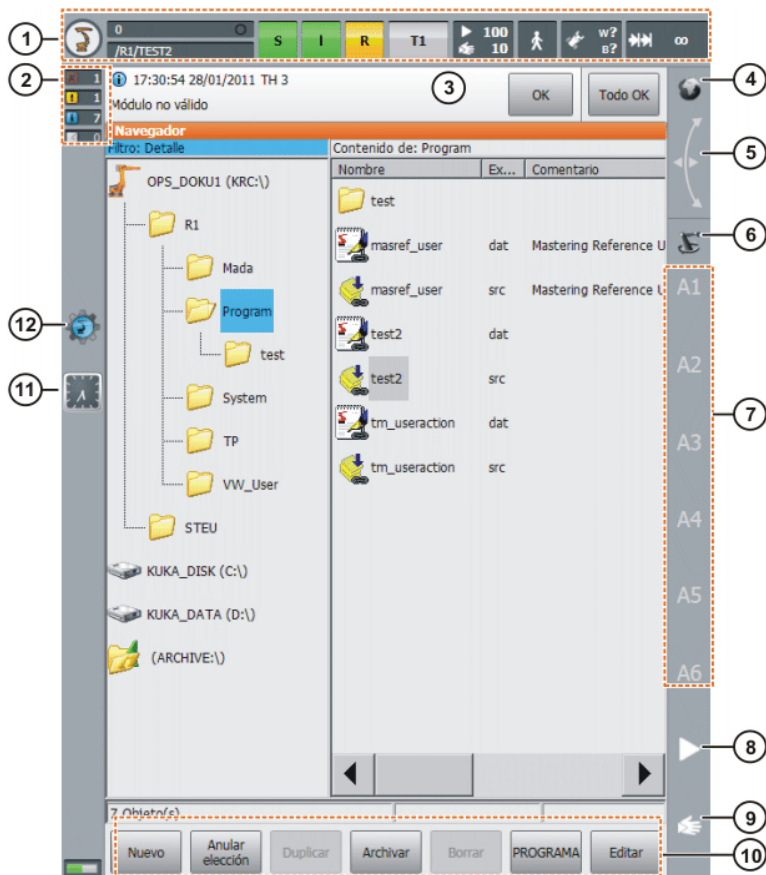
Parte trasera



- |                             |                             |
|-----------------------------|-----------------------------|
| 1 Pulsador de hombre muerto | 4 Tecla de arranque (verde) |
| 2 Pulsador de hombre muerto |                             |
| 3 Pulsador de hombre muerto | 5 Placa de características  |

ELEMENTO	DESCRIPCIÓN
<b>Pulsador de hombre muerto</b>	<p>El pulsador de validación tiene 3 posiciones:</p> <ul style="list-style-type: none"> <li>*No pulsado</li> <li>*Posición intermedia</li> <li>*Pulsado a fondo</li> </ul> <p>En los modos de servicio T1 y T2, el interruptor de seguridad debe mantenerse en la posición intermedia para poder efectuar movimientos con el manipulador.</p> <p>En los modos de servicio Automático y Automático Externo, el pulsador de validación carece de función.</p>

## Interfaz de usuario kuka smarthmi



Interfaz de usuario kuka smarthmi

POSICIÓN	DESCRIPCIÓN
1	Barra de estado
2	Contador de mensajes El contador de mensajes muestra el número de mensajes que existe de cada tipo de mensaje. Al tocar el contador de mensajes se ampliará la visualización.
3	Ventana de mensajes Por defecto solo se visualiza el último mensaje. Al tocar la ventana de mensajes se expandesajes y muestra todos los mensajes que existen. Con OK puede confirmarse un mensaje (que pueda confirmarse).

	Con Todo OK pueden confirmarse todos los mensajes (que puedan confirmarse).
4	Indicador de estado Space Mouse Esta visualización muestra el sistema de coordenadas actual para el desplazamiento manual con el Space Mouse. Al tocar la visualización se muestran todos los sistemas de coordenadas y se puede seleccionar otro.
5	Visualización Orientación Space Mouse Al tocar esta indicación, se abre una ventana en la que se muestra y puede modificarse la orientación actual del Space Mouse.
6	Indicador de estado Teclas de desplazamiento Esta visualización muestra el sistema de coordenadas actual para el desplazamiento manual con las teclas de desplazamiento. Al tocar la visualización se muestran todos los sistemas de coordenadas y se puede seleccionar otro.
7	Rotulación de las teclas de desplazamiento Si se ha seleccionado el desplazamiento específico del eje, se mostrarán aquí los números de los ejes (A1, A2, etc.). Si se ha seleccionado el desplazamiento cartesiano, se mostrarán aquí las direcciones del sistema de coordenadas (X, Y, Z, A, B, C). Al tocar la rotulación se muestra el grupo de cinemática que está seleccionado.
8	Override de programa
9	Override manual
10	Barra de botones. Los botones cambian de forma dinámica y se refieren siempre a la ventana que está activa en ese momento en smartHMI. El botón Editar está totalmente a la derecha. Con este botón se pueden activar numerosas instrucciones que van referidas al navegador.
11	Comparador El reloj muestra la hora del sistema. Al tocar el reloj se visualiza la hora del sistema en formato digital y la fecha actual.
12	Símbolo de WorkVisual Si no se puede abrir ningún proyecto, el símbolo tiene una X roja en la parte inferior derecha. Este es el caso, p. ej., si faltan archivos pertenecientes al proyecto. En este caso, el sistema solo está disponible de forma limitada.

## PROCEDIMIENTO

3. Realizar el procedimiento de encendido del brazo robot.
  - ✓ **Procedimiento:** Colocar el interruptor principal de la unidad de control del robot en ON.
  - ✓ El sistema operativo y el KSS arrancan automáticamente.
  - ✓ Si el KSS no se inicia automáticamente, p. ej. porque está deshabilitado el arranque automático, iniciar el programa StartKRC.exe en el directorio C:\KRC\BIN.
  - ✓ Si la unidad de control debe integrarse en la red, el inicio puede durar más tiempo.
4. Se realizara una explicación que permita al estudiante identificar claramente cada una de las partes que constituye el sistema robótico.

## RECURSOS

- ❖ Robot manipulador KUKA IRB2600
- ❖ Unidad manual de programación SmarPAD

## EVALUACIÓN

- ❖ El estudiante deberá rendir un informe escrito, una vez realizada la práctica, informe que deberá contener los siguientes aspectos:
- ❖ Describir a través de un esquema todas las partes constitutivas del brazo robot, en el que se describan las funciones de cada una de ellas.
- ❖ Un escrito sobre las características de movimiento de cada uno de los ejes de desplazamiento con los que cuenta la máquina.
- ❖ Conclusiones y recomendaciones sobre lo que se realizó en la práctica.



# TECNOLÓGICO NACIONAL DE MÉXICO

## INSTITUTO TECNOLÓGICO DE APIZACO

### PRACTICA N°3: Programación de movimientos a través del kuka smartpad.

Numero de equipo	Integrantes del equipo	Fecha de entrega	Calificación

#### OBJETIVO:

Conocer los movimientos del robot que se pueden programar a partir del Smartpad, y programar un ejemplo de cada uno de ellos.

#### OBJETIVOS ESPECÍFICOS:

- Identificar claramente el smartpad como herramienta fundamental para facilitar la operación del brazo robot.
- Reconocer cada una de las teclas y elementos más importantes del Smartpad para especificar los movimientos del robot.
- Conocer los procedimientos para grabar las rutinas o las posiciones principales de un programa.



## ALCANCE

Tener el conocimiento requerido para transmitir al lector las bases para el buen funcionamiento de rutinas programadas.

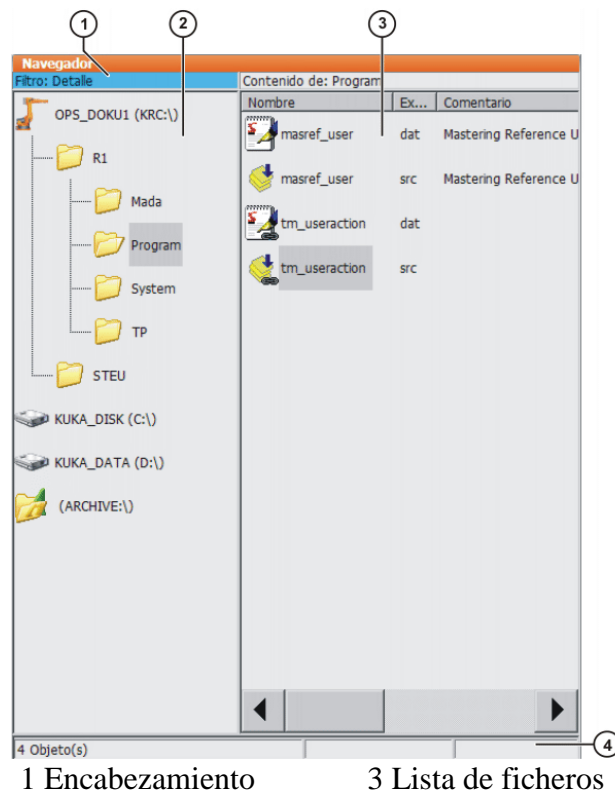
## MEDIDAS DE SEGURIDAD

Antes de empezar a manejar el robot es necesario hacer lo siguiente:

- ❖ Leer atentamente el procedimiento antes de comenzar la practica
- ❖ Observar que no haya ninguna anomalía
- ❖ No tocar ninguna parte móvil del robot durante el funcionamiento
- ❖ No dejar objetos dentro del área de operación del robot
- ❖ Reportar cualquier anomalía al instructor.

## MARCO TEÓRICO

En el navegador el usuario administra programas y todos los ficheros específicos del sistema. La siguiente imagen muestra lo que contiene la pantalla del administrador de programas, y la cual se detalla a continuación.



### **Encabezamiento**

Zona izquierda: Se visualiza el filtro seleccionado.

Zona derecha: Se visualiza el directorio o la unidad de discos marcado en la estructura de directorios.

### **Estructura de directorios**

Vista general sobre directorios y unidades de discos. Qué directorios y unidades de discos se muestran en pantalla depende del grupo de usuarios y de la configuración.

### **Lista de ficheros**

Se visualiza el contenido del directorio o la unidad de discos marcados en la estructura de directorio. De qué forma se muestran los programas depende del filtro seleccionado.

### **Línea de estado**

La línea de estados puede ofrecer la información siguiente:

- Objetos marcados
- Acciones en ejecución
- Diálogos del usuario
- Requerimientos de entradas para el usuario
- Preguntas de seguridad

## **TIPOS DE MOVIMIENTOS**

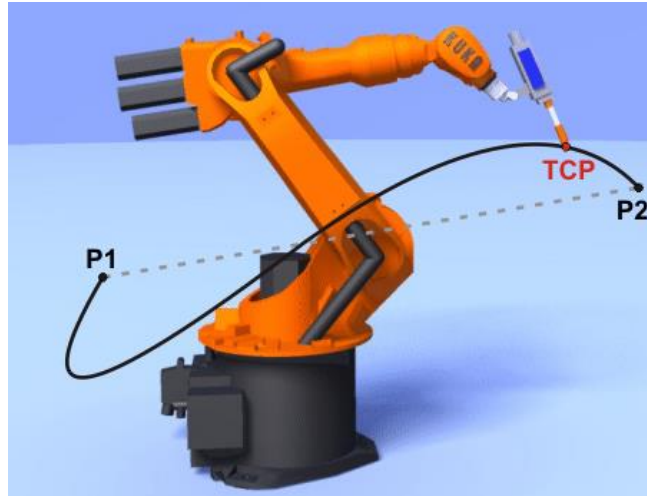
Pueden programarse los siguientes tipos de movimiento:

- Movimiento Point to Point (PTP)
- Movimiento lineal (LIN)
- Movimiento circular (CIRC)
- Movimiento spline

### **Tipo de movimiento PTP**

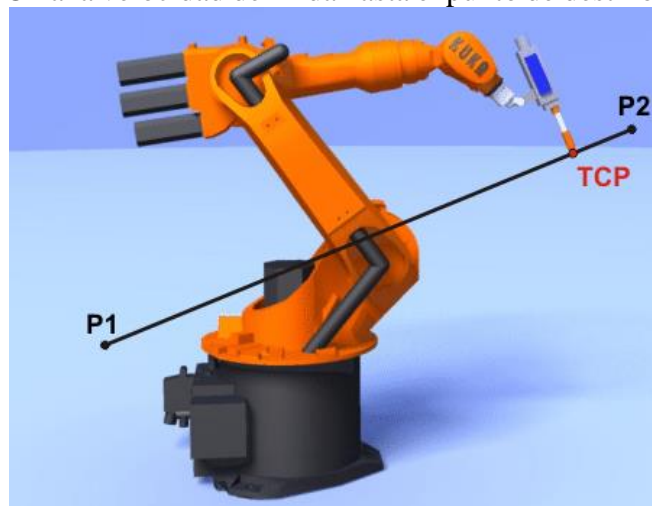
El robot desplaza el TCP al punto de destino a lo largo de la trayectoria más rápida. La trayectoria más rápida no es, en regla general, la trayectoria más corta y por ello no es una recta.

Dado que los ejes del robot se mueven de forma rotacional, trayectorias curvas pueden ser ejecutadas de forma más rápida que las rectas.  
No puede predecirse la trayectoria exacta



### Tipo de movimiento LIN

El robot conduce el TCP a la velocidad definida hasta el punto de destino a lo largo de una recta.



## ESTRUCTURA DE UN PROGRAMA

```
1 DEF my_program( )
2 INI
3
4 PTP HOME Vel= 100 % DEFAULT
...
8 LIN point_5 CONT Vel= 2 m/s CPDAT1 Tool[3] Base[4]
...
14 PTP point_1 CONT Vel= 100 % PDAT1 Tool[3] Base[4]
...
20 PTP HOME Vel= 100 % DEFAULT
21
22 END
```

Línea	Descripción
1	La línea DEF muestra el nombre del programa. Si el programa es una función, la línea DEF comienza con "DEFFCT" y contiene además otras indicaciones. La línea DEF puede mostrarse u ocultarse.
2	La línea INI contiene inicializaciones para variables internas y parámetros.
4	Posición HOME
8	Movimiento LIN
14	Movimiento PTP
20	Posición HOME
22	La línea END es la última línea de cada programa. Si el programa es una función, la línea END se escribe "ENDFCT". La línea END no se debe borrar.

## PROCEDIMIENTO

### 1. Crear nueva carpeta

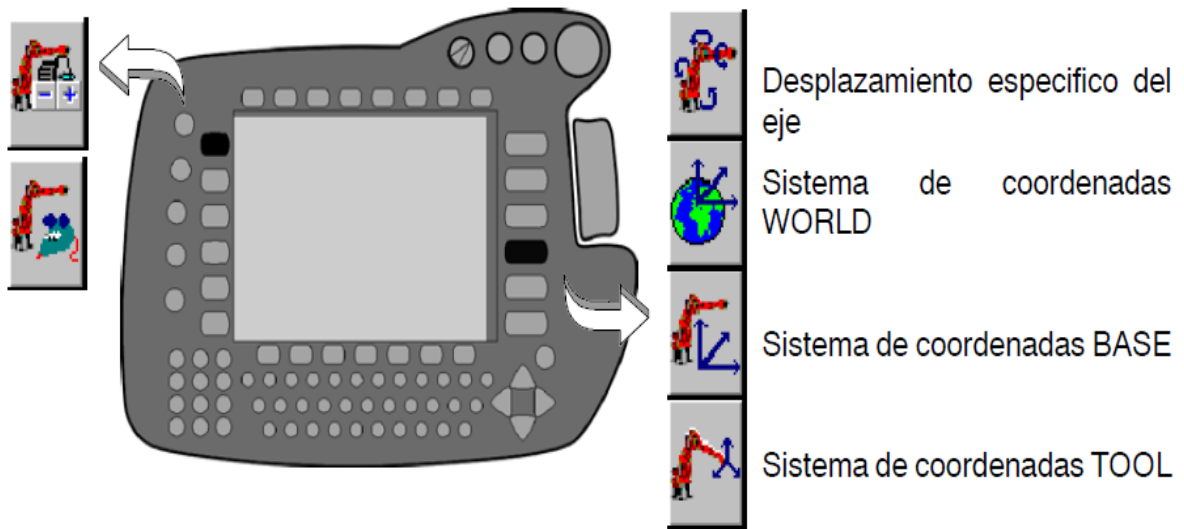
1. En la estructura de directorios, marcar la carpeta en que debe crearse el programa, p. ej. carpeta **R1**. (No se pueden crear carpetas en todas las carpetas). En los grupos de usuario "Usuario" y "Operador" sólo pueden crearse nuevas carpetas en la carpeta **R1**.

2. Pulsar **Nuevo**.
3. Introducir un nombre para la carpeta y confirmar con **OK**.

## 2. Crear nuevo programa

1. En la estructura de directorios, marcar la carpeta en que debe crearse el programa, p. ej. Carpeta **Programa**. (No se pueden crear programas en todas las carpetas.)
2. Pulsar **Nuevo**.
3. Sólo en el grupo de usuario "Experto": Se abre la ventana **Seleccionar plantilla**. Marcar la plantilla deseada y confirmar con **OK**.
4. Introducir un nombre para el programa y confirmar con **OK**.

3 **Cambiar el modo de empleo a Sistemas de coordenadas**, durante el desplazamiento manual, el robot se mueve dependiendo del sistema de coordenadas seleccionado, como se muestra en la figura siguiente:

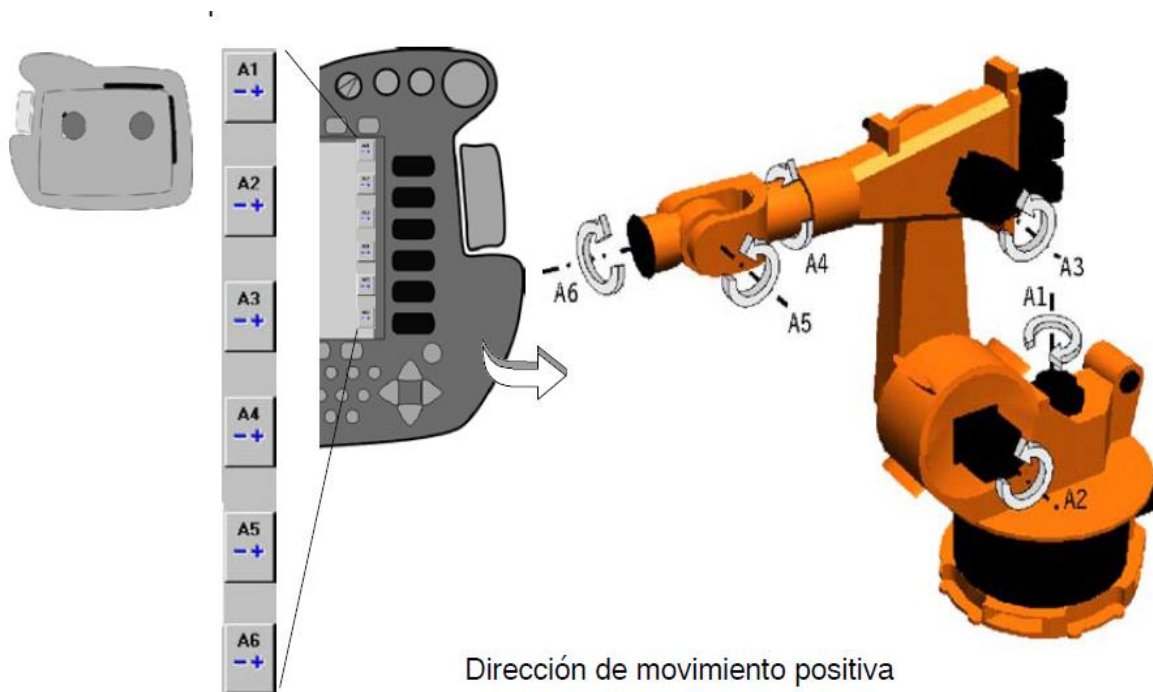


Al elegir cada una de los sistemas de los sistemas de coordenadas el movimiento del brazo será como se especifica a continuación.

## Sistema de coordenadas específico de los ejes

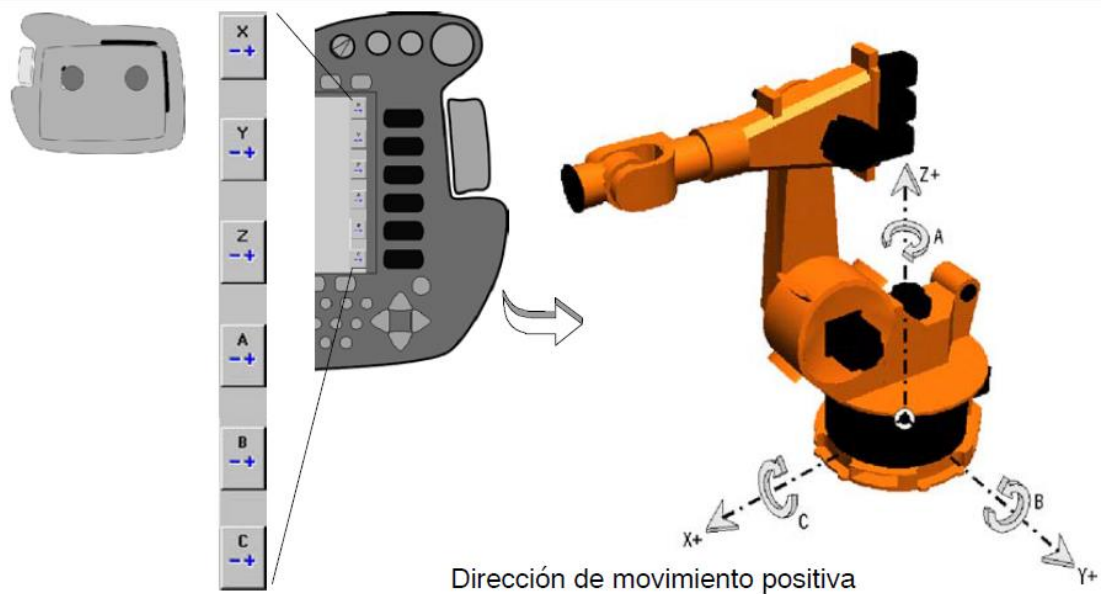


Accionando una de las teclas de hombre muerto (sobre el lado posterior del VKCP), en la barra de teclas de estado de la derecha se visualizan los ejes principales y los ejes de la muñeca A 1... A 6. Oprimiendo una tecla de estado +/- desplaza el eje correspondiente.



## Sistemas de coordenadas TOOL, BASE, WORLD

Accionando una de las teclas de hombre muerto (sobre el lado posterior del VKCP), en la barra de teclas de estado de derecha se visualizan los ejes principales X, Y y Z así como los ejes de la muñeca A, B y C. Premendo una tecla de estado +/- normalmente se desplazan varios ejes. En el **sistema de coordenadas universales (WORLD)** las teclas de desplazamiento tienen asignadas las denominaciones de eje que aparecen en la siguiente figura. También en este caso, las flechas indican las direcciones axiales positivas.



## RECURSOS

- ❖ Robot manipulador KUKA IRB2600
- ❖ Unidad manual de programación SmarPAD

## EVALUACIÓN

- ❖ El estudiante deberá rendir un informe escrito, una vez realizada la práctica, informe que deberá contener los siguientes aspectos:
- ❖ Un escrito sobre las características de movimiento de cada uno de los ejes de desplazamiento con los que cuenta el robot.
- ❖ Elaborar un esquema de cada uno de los movimientos del sistema de coordenadas que puede ejecutar el robot.
- ❖ Conclusiones y recomendaciones sobre lo que se realizó en la práctica.

**ANEXO B) ARTÍCULOS PUBLICADOS**

ISSN 2410-3993

Volumen 2, Número 5 – Octubre – Diciembre -2015

**Revista de  
Tecnología e  
Innovación**

**ECORFAN<sup>®</sup>**



**Bases de datos**

**Google scholar.**



### **Consejo Editorial**

GALICIA-PALACIOS, Alexander, PhD.  
*Instituto Politécnico Nacional, México*

NAVARRO-FRÓMETA, Enrique, PhD.  
*Instituto Azerbaidzhan de Petróleo y  
Química Azizbekov, Rusia*

BARDEY, David, PhD.  
*University of Besançon, Francia.*

IBARRA-ZAVALA, Darío PhD.  
New School for Social Research, U.S.

COBOS-CAMPOS, Amalia, PhD.  
*Universidad de Salamanca, España*

ALVAREZ-ECHEVERRÍA, Francisco,  
PhD.  
*University José Matías Delgado, El  
Salvador.*

BELTRÁN-MORALES, Luis Felipe,  
PhD.  
*Universidad de Concepción, Chile,  
Chile.*

BELTRÁN-MIRANDA, Claudia, PhD.  
*Universidad Industrial de Santander-  
Colombia, Colombia*

### **Consejo Arbitral**

ROMERO-RAMIREZ, Salvador, MsC.  
*Universidad de Londres, México*

ZAVALA, Manuel, MsC.  
*Universidad de Londres, México*

BLANCO-COCOM, Luis, MsC.  
*Universidad Autónoma de Yucatán,  
México.*

CHAN-CHI, Noe, Mtro.  
*Universidad Autónoma de Yucatán,  
México.*

TUTOR-SÁNCHEZ, Joaquín, PhD.  
*Universidad de la Habana*

VERDEGAY-GALDEANO, José, PhD.  
*Universidad de Granada*

OROZCO-GUILLÉN, Eber, PhD.  
*Instituto Nacional de Astrofísica Óptica y  
Electrónica*

QUIROZ-MUÑOZ, Enriqueta, PhD.  
*El Colegio de México*

## Presentación

ECORFAN, es una revista de investigación que publica artículos en las áreas de: Revista de Tecnología e Innovación

En Pro de la Investigación, Docencia, y Formación de los recursos humanos comprometidos con la Ciencia. El contenido de los artículos y opiniones que aparecen en cada número son de los autores y no necesariamente la opinión de la Editora en Jefe.

Como primer artículo esta *Sistema Recomendador Orientado a la Educación Basado en la Distancia entre Likes de Facebook y Conceptos* por MORALES, Alejandro, LÓPEZ-CHAU, Asdrúbal y REYES, Luis, como siguiente artículo esta *El algoritmo de agrupamiento K-Modas: Un caso de estudio* por RENDÓN, Eréndira, ZEPEDA, Ricardo, BARRUETA, Elizabeth y ITZEL-MARÍA, Abundez con adscripción Departamento de Sistema y Computación, Instituto Tecnológico de Toluca, como siguiente artículo esta *Una versión modificada del algoritmo de agrupamiento Isodata* por RENDÓN, Eréndira, MENDOZA, Marcos, CISNIEGA, Roció y CARBAJAL, Guillermo, como siguiente artículo esta *Desarrollo de un software para la simulación y control de un robot industrial* por LAZARO-ARVIZU, Y, MORALES-CAPORAL, R, ORDOÑEZ-FLORES, R, QUINTERO-FLORES, P y LEAL-LÓPEZ, M, como siguiente artículo esta *Adaptación del MMPI Mediante un Sistema Experto en Base a Probabilidades para el Diagnóstico de Desviaciones Psicopáticas en el Instituto Tecnológico de Pachuca* por RAMÍREZ-MEJIA J., MAGGI-NATALE C., ARRIETA-ZUÑIGA J., HERNANDEZ-RAMÍREZ A. y GONZÁLEZ-MARRON D. con adscripción Instituto Tecnológico de Pachuca, como siguiente artículo esta *Metodologías actuales de desarrollo de software* por RIVAS, Carlos Ignacio, CORONA, Verónica Paola, GUTIÉRREZ, José Fructuoso y HERNÁNDEZ, Lizeth, como siguiente artículo esta *Publicación en Internet del inventario de infraestructura física del I.T.P mediante Bases de Datos Geoespaciales y Sistema de Información Geográfica* por HERNÁNDEZ, Javier, ARRIAGA, Sergio y RERGIS, Raúl, como siguiente artículo esta *Sistema de monitoreo del LOBOBUS* por REYES, Cecilia, BARRETO, Aldrin y BAUTISTA, Verónica Edith con adscripción Instituto Tecnológico de Pachuca, Benemérita Universidad Autónoma de Puebla.

<b>Contenido</b>	<b>Artículo</b>	<b>Pag</b>
<b>Sistema Recomendador Orientado a la Educación Basado en la Distancia entre Likes de Facebook y Conceptos</b> MORALES, Alejandro, LÓPEZ-CHAU, Asdrúbal y REYES, Luis		921-928
<b>El algoritmo de agrupamiento K-Modas: Un caso de estudio</b> RENDÓN, Eréndira, ZEPEDA, Ricardo, BARRUETA, Elizabeth y ITZEL-MARÍA, Abundez		929-941
<b>Una versión modificada del algoritmo de agrupamiento Isodata</b> RENDON, Eréndira, MENDOZA, Marcos, CISNIEGA, Roció y CARBAJAL, Guillermo		942-957
<b>Desarrollo de un software para la simulación y control de un robot industrial</b> LAZARO-ARVIZU, Y, MORALES-CAPORAL, R, ORDOÑEZ-FLORES, R, QUINTERO-FLORES, P y LEAL-LÓPEZ, M		958-967
<b>Adaptación del MMPI Mediante un Sistema Experto en Base a Probabilidades para el Diagnóstico de Desviaciones Psicopáticas en el Instituto Tecnológico de Pachuca</b> RAMÍREZ-MEJIA J., MAGGI-NATALE C., ARRIETA-ZUÑIGA J., HERNANDEZ-RAMÍREZ A. y GONZÁLEZ-MARRON D.		968-979
<b>Metodologías actuales de desarrollo de software</b> RIVAS, Carlos Ignacio, CORONA, Verónica Paola, GUTIÉRREZ, José Fructuoso y HERNÁNDEZ, Lizeth		980-986
<b>Publicación en Internet del inventario de infraestructura física del I.T.P mediante Bases de Datos Geoespaciales y Sistema de Información Geográfica</b> HERNÁNDEZ, Javier, ARRIAGA, Sergio y RERGIS, Raúl		987-997
<b>Sistema de monitoreo del LOBOBUS</b> REYES, Cecilia, BARRETO, Aldrin y BAUTISTA, Verónica Edith		998-1006
<i>Instrucciones para Autor</i>		
<i>Formato de Originalidad</i>		
<i>Formato de Autorización</i>		

## Desarrollo de un software para la simulación y control de un robot industrial

LAZARO-ARVIZU, Y\*†, MORALES-CAPORAL, R, ORDOÑEZ-FLORES, R, QUINTERO-FLORES, P y LEAL-LÓPEZ, M

Recibido 5 de Julio, 2015; Aceptado 24 de Noviembre, 2015

### Resumen

Hoy en día la programación de robots hace uso de herramientas de simulación que reproducen la dinámica del robot, tanto para capacitar al personal que lo utiliza previo a su operación, así como para eliminar movimientos erróneos antes de su implementación en el robot real. En algunas aplicaciones académicas, MATLAB® que es un software de desarrollo integrado (IDE) permite cálculos y visualizaciones (gráficas) de alta dimensión y excelente resolución, por lo que lo hace una herramienta muy poderosa para poder desarrollar aplicaciones orientadas a la robótica, como es el caso del Robotic Toolbox® con el cual se pueden simular los movimientos de un robot en un entorno gráfico. En este artículo se presenta un conjunto de aplicaciones utilizando esta herramienta con fines didácticos, para diseñar y controlar las trayectorias de un robot KUKA IRB-2600 previo a su implementación y que se visualiza en tres dimensiones. Las aplicaciones propuestas permiten a los alumnos realizar prácticas, para aprender el manejo de un robot y visualizar los movimientos sin necesidad de tener el robot real. Para comprobar el software propuesto, se utilizan comunicaciones OPC (OLE for Process Control) entre el servidor de ABB y el cliente de MATLAB® desarrollado para mover un robot real, en este caso un KUKA IRB-2600. El software desarrollado dispone de un entorno gráfico que facilita la interacción con el usuario.

**Robótica, Simulación, Programación, Docencia.**

### Abstract

Today robot programming makes use of simulation tools that reproduce the dynamics of the robot, both to train personnel using prior to operation and to eliminate erroneous movements prior to implementation in real robot. In some academic applications, MATLAB software is Integrated Development Environment (IDE) enables calculations and visualizations (graphic) High dimension and excellent resolution, making it a very powerful tool to develop robotics applications-oriented, such as Robotic is the case with Toolbox® which can simulate the movements of a robot in a graphical environment. In this article a set of applications is presented using this tool for teaching purposes, to design and control the trajectories of a KUKA robot IRB-2600 prior to implementation and displayed in three dimensions. The proposed applications allow students to do internships to learn how to use a robot and visualize movements without having the real robot. To test the proposed software, communications OPC (OLE for Process Control) between the server and client ABB developed MATLAB to move a real robot, in this case a KUKA IRB-2600 are used. The developed software has a graphical interface that facilitates user interaction.

**Robotics, Simulation, Programming, Teaching.**

**Citación:** LAZARO-ARVIZU, Y, MORALES-CAPORAL, R, ORDOÑEZ-FLORES, R, QUINTERO-FLORES, P y LEAL-LÓPEZ, M. Desarrollo de un software para la simulación y control de un robot industrial. Revista de Tecnología e Innovación 2015, 2-5: 958-967

\* Correspondencia al Autor (Correo Electrónico)

† Investigador contribuyendo como primer autor.

## Introducción

Durante los estudios de algunas ingenierías, las asignaturas relacionadas con robótica tienen dos objetivos que no siempre son fáciles de combinar. Por un lado, se pretende proporcionar al alumno las bases teóricas de la robótica, entre ellas se explican los sistemas de coordenadas, las trayectorias, la cinemática y dinámica del robot y por otro el control del mismo. [1, 2, 3, 4].

Además, de ser posible se desea que el alumno aprenda a programar un robot en su entorno real, con sus particularidades en sus lenguajes. Una solución aquí propuesta para observar ambos objetivos es el uso de algunas aplicaciones de MATLAB®/ SIMULINK® y otras desarrolladas para robótica, donde de una forma práctica se puede aplicar y visualizar la teoría de robots simulados y reales.

MATLAB® tiene el "Robotic Toolbox®" donde se pueden desarrollar algoritmos teóricos para el modelado de robots, estudio de su cinemática, dinámica y diseño de trayectorias sobre un robot genérico [5, 6].

La aplicación "ARTE®" ("A Robotic Toolbox for Education") [7] añade a la anterior aplicación el modelo en tres dimensiones (3D) de un gran número de robot industriales, con funciones para obtener sus cinemáticas inversas, de forma que los conceptos teóricos pueden ser aplicados a robots industriales.

En el Instituto Tecnológico de Apizaco (ITA) se dispone de dos robots industriales KUKA [8], uno de ellos es un antropomórfico de seis grados de libertad con una capacidad de 16 kg, ubicado dentro del tamaño de los de serie mediano, empleado principalmente en la industria de la automatización para tareas de soldadura, recubrimientos, manipulación de objetos, etc. Y en este caso para fines de investigación y docencia.

Las aplicaciones desarrolladas conseguirán que el alumno pueda comprender la teoría con la herramienta de trabajo que luego va a emplear. Se han realizado pantallas para la visualización del robot en 3D, una aplicación gráfica basada en GUIDE® de MATLAB® [9], y simulaciones del robot en 3D basadas en SimMechanic® [10, 11, 12].

La aplicación propuesta está basada en el servidor OPC que dispone ABB [13] y el OPC cliente de MATLAB® [14].

Un servidor OPC es el método de conectividad de datos basado en los estándares más populares del mundo. Es utilizado para responder a uno de los mayores retos de la industria de la automatización: cómo comunicar dispositivos, controladores y/o aplicaciones sin caer en los problemas habituales de las conexiones basadas en protocolos propietarios.

Dicha aplicación permite, para un período de muestreo dado, mostrar y medir las variables de salida y permanentes del robot desde una aplicación OPC cliente.

Se consideran seis variables de salida en la estación robotizada, una para cada eje del robot. Estas variables serán leídas por el OPC servidor y modificadas desde MATLAB® por una aplicación OPC cliente. En la función de visualización en 3D del robot se ha añadido la opción de modificar de forma simultánea el valor de estas variables para definir la posición de los ejes. El procedimiento que el robot ejecuta en su propio lenguaje RAPID lee de forma ininterrumpida las seis variables de los ejes, y se mueve a la posición correspondiente. De esta forma se consigue que la trayectoria del robot sea controlada por la aplicación de MATLAB®, y que el robot y su modelo en 3D sigan la misma trayectoria de forma simultánea.

El alumno podrá aplicar esta herramienta tanto en el robot virtual representado con RobotStudio® [15] como el real, ya que el servidor OPC es válido para ambos casos. El alumno puede analizar y validar su programa de control en el robot virtual, y una vez verificado su funcionamiento ejecutarlo y verificarlo en el robot real. En cualquier caso, la aplicación es válida para poder ser aplicada con el robot en modo manual en condiciones de máxima seguridad, ya que si se suelta la tecla de hombre muerto, el robot se para automáticamente.

El resto del artículo tiene la siguiente estructura, en la sección 2 se analizan las aplicaciones en MATLAB® / SIMULINK® relacionadas con robótica, además de la cinemática del robot.

En la sección 3, se analiza las posibles comunicaciones entre un robot y una computadora. En la sección 4, se detallan las aplicaciones realizadas en MATLAB® / SIMULINK®, y en la sección 5 se citan las conclusiones del artículo.

#### **Sistemas robotizados en MATLAB® / SIMULINK®**

MATLAB® / SIMULINK® son dos entornos de cálculo numérico y visualización de datos de la compañía Matworks® [10, 16], con grandes posibilidades para el diseño de sistemas dinámicos y su simulación. Dentro de las aplicaciones comerciales de ambos entornos no existe uno específico para el diseño y simulación de sistemas robotizados, pero diversos autores han desarrollado aplicaciones de software libre usando MATLAB® / SIMULINK® para sistemas robotizados. [17, 18, 19] La aplicación de robótica más conocida para MATLAB® / SIMULINK® es "Robotic Toolbox®" realizada por Peter Corke [6] que desarrolla los contenidos del libro del propio autor [5]. Dicha aplicación dispone de software para robótica fija y móvil.

ISSN-2410-3993  
ECORFAN® Todos los derechos reservados

En la parte de robótica fija propone algoritmos para modelar un robot usando el método de Denavit-Hartenberg, con los que se puede obtener la cinemática directa del robot. La aplicación propone un método general de cinemática inversa para un robot con muñeca esférica, que puede ser aplicada a robots PUMA. Los modelos gráficos de los robots son muy simples, pero genéricos, ya que se representan únicamente los ejes y las uniones entre ellos. En la última versión se ha desarrollado un modelo 3D para el robot PUMA. La dinámica de los robots se representa mediante bloques convencionales de Simulink®.

La mayoría de los fabricantes de robots industriales han desarrollado modelos gráficos en 3D utilizando programas de diseño asistido por ordenador (CAD), y estos modelos en muchos casos son libres y se pueden obtener de internet. Del archivo del robot en los principales formatos de CAD se puede pasar a una estructura compuesta por un fichero "\*.xml" con los datos del robot y ficheros "\*.stl" con la información gráfica de cada modelo de brazo del robot. Existen aplicaciones de MATLAB® para leer la información de los ficheros "\*.stl" y obtener graficas en 3D de los brazos del robot en MATLAB®.

La aplicación "ARTE®" ("A Robotic Toolbox for Education") [7] aprovecha estos ficheros para generar modelos de los principales brazos robóticos del mercado.

Esta aplicación dispone de la cinemática directa, basada en Denavit-Hartenberg, e inversa de un conjunto de robots. También dispone de datos dinámicos de alguno de estos robots. Con ello se puede estudiar los movimientos y trayectorias de un robot industrial viendo su evolución sobre modelos gráficos de 3D.

LAZARO-ARVIZU, Y, MORALES-CAPORAL, R, ORDOÑEZ-FLORES, R, QUINTERO-FLORES, P y LEAL-LÓPEZ, M. Desarrollo de un software para la simulación y control de un robot industrial. Revista de Tecnología e Innovación 2015



Además, esta aplicación dispone de algoritmos similares a las instrucciones de movimiento del lenguaje RAPID, para comprender la forma en la que se diseñan estas funciones y su aplicación a robot industriales.

La aplicación gráfica ROKISIM (Robot Kinematic Simulation) [20] muestra un modelo gráfico en 3D de muchos robots industriales y simula sus movimientos básicos.

La aplicación comercial SimMechanic® [11] para SIMULINK® dispone de una herramienta especial, SimMechanic® Link [12], que permite importar modelos de robot realizados con programas de CAD donde la información de su estructura esta en ficheros "\*.xml" y la información sobre sus brazos en ficheros "\*.stl". A partir de esta información, la aplicación desarrolla un modelo SIMULINK® (SimMechanic) con la estructura en serie articulación-brazo del robot y un modelo gráfico en 3D con la figura del robot. Sobre el diagrama de SIMULINK® se pueden hacer cambios para simular los lazos de control de cada brazo del robot y de esta forma, simular la dinámica del mismo.

Se concluye indicando que existen muchas aplicaciones para modelar y simular un robot fijo usando MATLAB® / SIMULINK®. Algunas son genéricas para cualquier tipo de robot definido con el método de Denavit-Hartenberg y otras aprovechan los modelos de CAD de robot industriales para estudiar la cinemática y dinámica de robots concretos.

#### Cinemática del robot

La cinemática del robot estudia el movimiento del mismo con respecto a un sistema de referencia. Así la cinemática se interesa por la descripción analítica del movimiento espacial del robot como una función del tiempo.

Pero primordialmente por la orientación y posición del extremo final del robot con los valores que toman sus coordenadas articulares.

La cinemática estudia dos problemas particulares en el robot. El primero de ellos, se conoce como el problema cinemático directo, y consiste en determinar, los valores conocidos de las articulaciones, cual es la posición del extremo final del robot con respecto al sistema de coordenadas que se toman como referencia. [1]

El segundo es el denominado problema cinemático inverso, este resuelve la configuración que debe adoptar el robot para que su extremo alcance una posición y orientación conocida. [1]

#### Cinemática directa del robot IRB-2600

Un robot manipulador puede considerarse una cadena cinemática formada por objetos rígidos o eslabones unidos entre sí mediante articulaciones.

Se puede establecer un sistema de referencia fijo situado en la base del robot y describir la localización de cada uno de los eslabones con respecto a dicho sistema de referencia.

Las herramientas matemáticas que se han utilizado para describir la configuración del robot son las matrices de transformación homogénea y el algoritmo de Denavit-Hartenberg [1]. En la tabla I se muestran los parámetros del algoritmo de Denavit-Hartenberg correspondientes a la cinemática directa del robot IRB-2600.

i	$\theta_i$	$a_i$	$d_i$	$\alpha_i$
1	$\theta_1 - 90^\circ$	445	150	-90
2	$\theta_2$	0	700	0
3	$\theta_3$	0	0	90
4	$\theta_4$	-795	0	-90
5	$\theta_5$	0	0	90
6	$\theta_6$	0	-85	180

**Tabla 1** Valores de Denavit-Hartenberg correspondiente al robot IRB-2600.

### Comunicación de la aplicación propuesta con el robot real

La comunicación entre la aplicación de MATLAB® / SIMULINK® y el robot se podrá haber realizado usando múltiples protocolos. Las ventajas e inconvenientes de dos de ellos se resumen a continuación:

**Comunicación por OPC:** ABB dispone de un servidor OPC que puede ser ejecutado desde una computadora. Esta aplicación detecta los robots conectados a la computadora, ya sean reales o virtuales, y extrae de ellos, en cada periodo de muestreo fijado, la información del sistema, sus variables de entradas y salidas, y las variables persistentes del programa Rapid del robot. Desde una aplicación OPC® cliente se puede leer y modificar dichas variables. No es preciso que la lectura y escritura estén sincronizadas, cuando se modifique una variable desde el exterior, dicha variable queda modificada en la estación del robot.

La comunicación entre la aplicación de Matlab y el robot de ABB ha sido realizada haciendo uso del servidor OPC proporcionado por ABB [13], y del cliente OPC proporcionado por la aplicación "OPC Toolbox®" de MATLAB® [14]. La comunicación se ha realizado usando variables de salida analógicas virtuales por una razón de seguridad.

La escritura en variables de salida en RAPID es posible en estado manual y automático, mientras que la escritura en variables persistente solo es posible en automático. Con fines educativos y por seguridad, el robot IRB-2600 se suele usar en modo manual, ya que en este modo se puede parar el robot ante cualquier eventualidad dejando de pulsar el botón de hombre muerto. Se han detenido seis variables de salida virtuales normalizadas entre [-100; 100] para transmitir la información de los seis ejes del robot.

La forma de comunicarse entre MATLAB® y el robot IRB-2600 es la siguiente:

**MATLAB®:** Cuando la aplicación de MATLAB® quiera cambiar la posición del robot debe modificar el valor de las variables del servidor OPC correspondientes a los ejes del robot.

**Robot IRB-2600 (RAPID):** El programa en Rapid consiste en un bucle infinito que lee las salidas analógicas correspondientes a los ejes y desplaza al robot a la posición definida por los valores de las variables.

El protocolo que se han empleado proporciona independencia frente al tiempo de lectura y escritura.

Si se envía una trayectoria definida por muchas posiciones de ejes con un periodo de muestreo muy bajo, puede que el programa Rapid no sea capaz de leer algún punto pero llegara al destino sin quedarse bloqueado. La sincronización entre MATLAB® y Rapid depende de la velocidad con que envía los datos a las variables de salida.

### Descripción de las aplicaciones

En este trabajo se han desarrollado tres aplicaciones para modelar y simular el robot IRB-2600 y sus movimientos respectivamente con MATLAB® y después poder mover el robot real, o el virtual si no se cuenta con el robot real.

Las dos primeras aplicaciones hacen uso de las funciones de "Robotic Toolbox®" [6] y "ARTE®" [7]. La tercera aplicación, hace uso de las herramientas de SimMechanic® [11, 12].

### Aplicación grafica de simulación

Primero se modela el robot IRB-2600, con su cinemática directa e inversa y se programan una función `plot irb2600()` para visualizar las posiciones del robot en 3D. La cinemática directa y las trayectorias del robot se han diseñado utilizando la aplicación de [6], mientras que la cinemática inversa ha sido obtenida de la aplicación de [7].

A partir de los ficheros "\*.stl" del robot, se ha construido una función `plot irb2600()` con las siguientes objetivos (Ver anexo). La fig. 1 muestra el robot simulado.

Dibujar el robot en 3D con la posición de ejes deseada. Se puede dibujar una secuencia de posiciones si la entrada es una matriz de posiciones de ejes. Se puede definir un tiempo de parada entre posiciones que va a ser útil para sincronizar el movimiento de la gráfica y el del robot real.

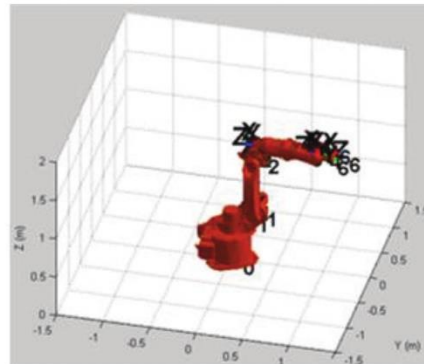


Figura 1 Robot simulado en 3D.

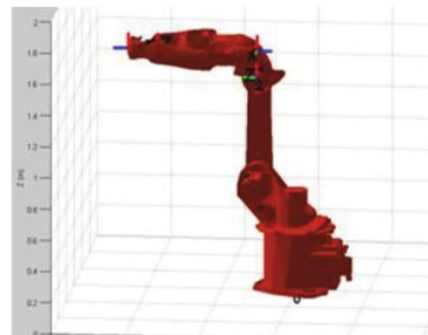


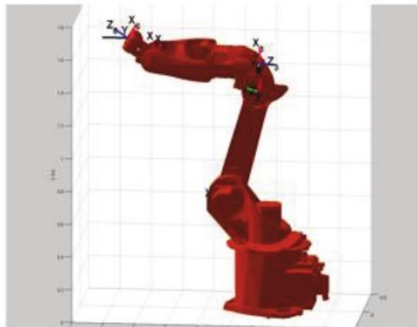
Figura 2 Posición inicial del robot simulado



**Figura 3** Posición inicial del robot real.



**Figura 5** Posición robot real.



**Figura 4** Posición final del robot simulado.

El usuario puede programar la posición en las variables de entrada del robot correspondiente, usando el servidor OPC de ABB.

Cuando el programa de Rapid se activa, el robot se moverá a las mismas posiciones que el robot simulado. El tiempo de espera entre posiciones es un parámetro de sintonización entre el robot y la simulación.

Las siguientes imágenes muestran las pruebas que se hicieron en la fig. 2 y 3 muestran la posición inicial del robot real y el robot simulado.

Las posiciones finales se muestran en la fig. 4 y 5.

### Aplicación grafica (GUIDE®) para el movimiento del robot

A partir de la función `plot irb2600()` se ha realizado una aplicación grafica en GUIDE® de MATLAB® [9] que permite al usuario realizar de forma segura los movimientos de ejes por ángulos y posiciones cartesianas. Dichos movimientos se visualizan con la función `plot irb2600()` y podrán ser enviadas a las variables de los ejes del robot a través del servidor OPC. Con ello, el usuario podrá mover el robot con una aplicación grafica de MATLAB®, de forma similar a como se mueve con la aplicación FlexPendant de ABB [21].

Las principales características de esta aplicación son las siguientes:

- Movimiento a una posición en coordenadas articulares. Se puede grabar dicha posición para simular una trayectoria.
- Movimiento a una posición del punto de trabajo.
- Se puede grabar dicha posición para simular una trayectoria
- Movimiento del robot entre las posiciones grabadas anteriormente, ya sean posiciones articulares o del punto de trabajo.
- Movimientos de demostración del robot.
- Conexión con OPC para comunicar datos.

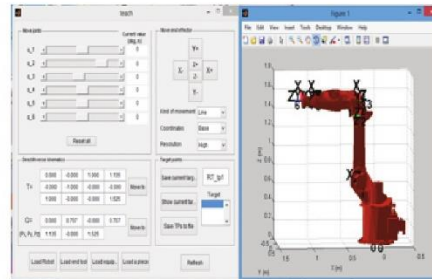


Figura 6 Pantalla inicial de la aplicación gráfica.

La fig. 6 muestra la pantalla inicial de la aplicación gráfica. Se aprecian los diferentes botones para mover el robot mediante coordenadas articulares, coordenadas de punto de trabajo, y resto de elementos mencionados.

### Aplicaciones didácticas y de investigación

El objetivo de las aplicaciones realizadas en este artículo es servir de puente entre el estudio matemático y simulado de los sistemas robotizados y el práctico de movimiento real de un robot comercial. Se desea que el alumno pueda realizar algoritmos semejantes a los implementados en lenguaje RAPID y otros algoritmos nuevos que el alumno desarrolle.

Como ejemplos de ejercicios realizables con estas aplicaciones se pueden mencionar:

Modelar trayectorias interpolando puntos en el espacio de coordenadas articulares.

Modelar trayectorias interpolando puntos en el espacio de coordenadas espaciales del punto de trabajo.

Mover el punto de trabajo del robot para minimizar el error entre la posición de dicho punto y una referencia dada, usando un algoritmo de optimización local. Usar para ello incrementos en las coordenadas articulares.

Mover el punto de trabajo del robot en la dirección medida por un sensor de esfuerzos, que el robot IRB-2600 (que actualmente el robot no tiene).

Mover el punto de trabajo del robot hacia una posición móvil detectada por una cámara de visión (proyecto futuro) asociada a una aplicación de MATLAB®.

Programar las trayectorias punto a punto, adicionando al robot una pistola para soldadura y que este pueda realizar las trayectorias repetidamente (En proceso).

### Conclusiones

Se han presentado las propuestas de aplicaciones usando el "Robotic Toolbox®" de MATLAB® / SIMULINK® y su comunicación con el software de ABB para controlar, simular, diseñar e implementar los movimientos de un robot industrial. Esto es posible gracias a que la herramienta de comunicación entre el robot y las aplicaciones de MATLAB® la cual se lleva a cabo por medio de un intercambio de información usando el servidor OPC de ABB y el cliente OPC de MATLAB®. Con esta herramienta los alumnos pueden realizar en un entorno visual la programación previa de los movimientos de cada eslabón del robot para una aplicación específica. Una vez verificado el diseño de los movimientos por simulación, es posible implementarlos de manera rápida en el robot real, en caso de contar con él. Eliminando movimientos riesgosos tanto para el usuario como para el equipo.

### Referencias

Barrientos A., Peñin L.F., Balaguer C. y Aracil P. (2007) Fundamentos de Robótica (2 edición), MacGraw-Hill.

Ollero A. (2001) Robótica, manipuladores y robots móviles, Editorial Macombó.

Pérez Cisneros M.A., Cuevas Jiménez E.V., Zaldívar Navarro D. (2014) Fundamentos de Robótica y Mecatrónica con MATLAB y SIMULINK. Editorial RA-MA.

Spong M., Hutchinson S y Vidyasagar (2006) M., Robot, Modeling & Control, Wiley.

Corke, P. (2013) Robotics, Vision & Control, Springer.

Peter Corke (2014) Robotics Toolbox. [http://petercorke.com/Robotics Toolbox.html](http://petercorke.com/Robotics%20Toolbox.html).

Gil Aparicio A. (2014) ARTE (A Robotics Toolbox for Education) Universidad Miguel Hernández (Elche, España), [http://arvc.umh.es/arte/index en.html](http://arvc.umh.es/arte/index%20en.html)

[www.kuka-robotics.com/Robotics](http://www.kuka-robotics.com/Robotics).

MathWorks (2014). GUIDE Toolbox User's Guide.

MathWorks (2014). SIMULINK User's Guide.

MathWorks (2014). SimMechanic User's Guide.

MathWorks (2014). SimMechanic Link User's Guide (2014).

ABB robotic. Application manual: IRC5 OPC Server. Doc. ID: 3HAC023113-001.

MathWorks (2014). OPC Toolbox User's Guide.

ABB robotic. Manual del operador: RobotStudio. Doc. ID: 3HAC032104-005.

MathWorks (2014). MATLAB® User's Guide.

Hassine Belhadj, Saber Ben Hassen, Khaled Kaániche and Hassen Mekki, "KUKA Robot control based Kinect image analysis", in IEEE International Conference on Individual, and Collective Behaviors in Robotics, pp. 21-26.

Rene González Rodríguez and Luis Hernández Santana, "Platform to develop real time visual servoing control in kinematics systems", in revista Ingeniería Mecánica, vol. 15, no. 3, pp. 233-241, 2012. ISSN 1815-5944.

Velásquez Lobo, Ramirez Cortés and Rangel Magdaleno, "Modeling a Biped Robot on Matlab/SimMechanics", in IEEE International Conference on Individual, and Collective Behaviors in Robotics, pp. 203-206.

Bonev I. (2013) ROKISIM (Robot Kinematic Simulation), Escuela de Superior de Tecnología (Quebec, Canadá)  
<http://www.parallemic.org/RoKiSim.html>.

ABB robotic. Manual del operador: Introducción a RAPID (RobotWare 5). Doc. ID: 3HAC029364-005

ABB robotic. Manual del operador IRC5 con FlexPendant. Doc. ID: 3HAC16590-5.

## Herramienta Computacional para Manipular un Robot (KUKA) IRB-2600 Basada en un Toolbox de MATLAB®/ SIMULINK®

L.I Yajaira Guadalupe Lázaro Arvizu<sup>1</sup>, Dr. Roberto Morales Caporal<sup>2</sup>,  
Dr. José de Jesús Rangel Magdaleno<sup>3</sup> y M.C Mario Eduardo Leal López<sup>4</sup>

**Resumen**—Hoy en día los robots han llegado a ser una herramienta importante por la precisión que necesita la industria moderna al realizar sus actividades productivas, es por ello que en la actualidad la programación de robots se puede hacer mediante el uso de herramientas de simulación que reproducen la dinámica del robot, tanto para capacitar al personal que lo utiliza previo a su operación, así como para eliminar movimientos erróneos antes de su implementación en el robot real.

En este trabajo se presenta una herramienta computacional basada en un Toolbox de MATLAB®/ SIMULINK®, capaz de simular y después implementar en un robot real los movimientos requeridos. La herramienta computacional ha sido desarrollada con fines didácticos, y permite diseñar y simular las trayectorias del robot industrial (KUKA) IRB-2600 en un entorno visual, previo a su implementación, de modo que los estudiantes aprenden a diseñar y controlar los movimientos repetitivos del robot y después observarlos en un entorno gráfico 3D. Con esta herramienta se evita implementar movimientos peligrosos, puede utilizarse para capacitar a personal o enseñar a los alumnos (laboratorio virtual) sin necesidad de tener el robot real.

**Palabras clave**—Manipulador industrial, Herramienta de Simulación, Laboratorio virtual.

### Introducción

Durante los estudios de algunas ingenierías, las asignaturas relacionadas con robótica tienen dos objetivos que no siempre son fáciles de combinar. Por un lado, se pretende proporcionar al alumno las bases teóricas de la robótica, entre ellas se explican los sistemas de coordenadas, las trayectorias, la cinemática y dinámica del robot (Barrientos A. et al. 2007 y Ollero, 2001 y Pérez, 2014 y Spong 2006).

Al mismo tiempo, de ser posible se desea que el alumno aprenda a programar un robot en su entorno real, con sus particularidades en sus lenguajes. Una solución aquí propuesta para realizar ambos objetivos es la unión de tres aplicaciones previamente desarrolladas por otros trabajos basados en MATLAB®/ SIMULINK®, con la cual de forma práctica se puede aplicar y visualizar la teoría de robots simulados y reales.

En "Robotic Toolbox" de MATLAB® se pueden desarrollar algoritmos teóricos para el modelado de robots, estudio de su cinemática, dinámica y diseño de trayectorias sobre un robot genérico (Corke Peter 2014).

La aplicación "ARTE" ("A Robotic Toolbox for Education") añade a la anterior aplicación el modelo en tres dimensiones (3D) de un gran número de robots industriales, con funciones para obtener sus cinemáticas inversas, de forma que los conceptos teóricos pueden ser aplicados a robots industriales (Gil Aparicio 2014).

En el Instituto Tecnológico de Apizaco (ITA) se dispone de dos robots industriales KUKA, ambos antropomórfico de seis grados de libertad y uno de ellos con una capacidad de 16 kg, ubicado dentro del tamaño de los de serie mediano, empleado principalmente en la industria de la automatización para tareas de soldadura, recubrimientos, manipulación de objetos, etc. Y en este caso para fines de investigación y docencia.

La conjunción de las aplicaciones desarrolladas conseguirán que el alumno pueda comprender la teoría con la herramienta de trabajo que luego va a emplear. Se han realizado pantallas para la visualización del robot en 3D, una aplicación gráfica basada en GUIDE® de MATLAB® (MathWorks 2014), y simulaciones del robot en 3D basadas en SimMechanic®. La aplicación propuesta está basada en el servidor OPC que dispone ABB y el OPC cliente de MATLAB®. Un servidor OPC es el método de conectividad de datos basado en los estándares más populares del mundo. Es utilizado para responder a uno de los mayores retos de la industria de la automatización: cómo comunicar dispositivos, controladores y/o aplicaciones sin caer en los problemas habituales de las conexiones basadas en protocolos propietarios.

<sup>1</sup> Yajaira Guadalupe Lázaro Arvizu es Estudiante de la Maestría en Sistemas Computacionales en el Instituto Tecnológico de Apizaco, Apizaco, Tlaxcala. [yajairala7@gmail.com](mailto:yajairala7@gmail.com) (**autor correspondiente**)

<sup>2</sup> El Dr. Roberto Morales Caporal es Profesor-Investigador Titular en el Instituto Tecnológico de Apizaco, Apizaco, Tlaxcala. [rmcaporal@ieee.org.mx](mailto:rmcaporal@ieee.org.mx)

<sup>3</sup> El Dr. José de Jesús Rangel Magdaleno es Profesor-Investigador Titular en el Instituto Nacional de Astrofísica, Óptica y Electrónica, Tonantzintla, Puebla. [jose.rangel.magdaleno@gmail.com](mailto:jose.rangel.magdaleno@gmail.com)

<sup>4</sup> El M.C. Mario Eduardo Leal López es Profesor en el Instituto Tecnológico de Apizaco, Apizaco, Tlaxcala. [colt\\_mario@hotmail.com](mailto:colt_mario@hotmail.com)



Se consideran seis variables de salida en la estación robotizada, una para cada eje del robot. Estas variables serán leídas por el OPC servidor y modificadas desde MATLAB® por una aplicación OPC cliente. En la función de visualización en 3D del robot se ha añadido la opción de modificar de forma simultánea el valor de estas variables para definir la posición de los ejes. El procedimiento que el robot ejecuta en su propio lenguaje RAPID lee de forma ininterrumpida las seis variables de los ejes, y se mueve a la posición correspondiente. De esta forma se consigue que la trayectoria del robot sea controlada por la aplicación de MATLAB®, y que el robot y su modelo en 3D sigan la misma trayectoria de forma simultánea.

El alumno podrá aplicar esta herramienta tanto en el robot virtual representado con RobotStudio® como el real, ya que el servidor OPC es válido para ambos casos. El alumno puede analizar y validar su programa de control en el robot virtual, y una vez verificado su funcionamiento ejecutarlo y verificarlo en el robot real.

El resto del artículo se estructura de la siguiente manera, en la sección 2 se analizan las aplicaciones en MATLAB® / SIMULINK® relacionadas con robótica, además de la cinemática del robot. En la sección 3, se analiza las posibles comunicaciones entre un robot y una computadora. En la sección 4, se detallan las aplicaciones realizadas en MATLAB® / SIMULINK®, finalmente en la sección 5 se mencionan las conclusiones del artículo.

#### **Sistemas robotizados en matlab® / simulink®**

MATLAB® / SIMULINK® son dos entornos de cálculo numérico y visualización de datos de la compañía Matworks® con grandes posibilidades para el diseño de sistemas dinámicos y su simulación. Dentro de las aplicaciones comerciales de ambos entornos no existe uno específico para el diseño y simulación de sistemas robotizados, pero diversos autores han desarrollado aplicaciones de software libre usando MATLAB® / SIMULINK® para sistemas robotizados (Hassine A. et al. 2013 y González, 2012 y Velázquez, 2014). La aplicación de robótica más conocida para MATLAB® / SIMULINK® es "Robotic Toolbox"™ realizada por Peter Corke que desarrolla los contenidos del libro del propio autor (Corke Peter 2014). Este Toolbox se basa en un método muy general de lo que representa la cinemática y la dinámica de la serie de manipuladores. Dicha aplicación dispone de software para robótica fija y móvil. En la parte de robótica fija propone algoritmos para modelar un robot usando el método de Denavit-Hartenberg, con los que se puede obtener la cinemática directa del robot. Los modelos gráficos de los robots son muy simples, pero genéricos, ya que se representan únicamente los ejes y las uniones entre ellos. La dinámica de los robots se representa mediante bloques convencionales de MATLAB® / SIMULINK®. (Corke Peter 2014). La mayoría de los fabricantes de robots industriales han desarrollado modelos gráficos en 3D utilizando programas de diseño asistido por ordenador (CAD), y estos modelos en muchos casos son libres y se pueden obtener de internet del archivo del robot en los principales formatos de CAD se puede pasar a una estructura compuesta por un fichero "\*.xml" con los datos del robot y ficheros "\*.stl" con la información gráfica de cada modelo de brazo del robot. Existen aplicaciones de MATLAB® para leer la información de los ficheros "\*.stl" y obtener gráficas en 3D de los brazos del robot en MATLAB®.

La aplicación "ARTE"™ ("A Robotic Toolbox for Education") (Aparicio 2014). aprovecha estos ficheros para generar modelos de los principales brazos robóticos del mercado.

La aplicación gráfica ROKISIM (Robot Kinematic Simulation) (Boney I. 2013) muestra un modelo gráfico en 3D de muchos robots industriales y simula sus movimientos básicos.

La aplicación comercial SimMechanic® para SIMULINK® dispone de una herramienta especial que permite importar modelos de robot realizados con programas de CAD donde la información de su estructura está en ficheros "\*.xml" y la información sobre sus brazos en ficheros "\*.stl" (MathWorks 2014). A partir de esta información, la aplicación desarrolla un modelo SIMULINK® (SimMechanic) con la estructura en serie articulación-brazo del robot y un modelo gráfico en 3D con la figura del robot. Sobre el diagrama de SIMULINK® se pueden hacer cambios para simular los lazos de control de cada brazo del robot y de esta forma, simular la dinámica del mismo.

Se concluye indicando que existen muchas aplicaciones para modelar y simular un robot fijo usando MATLAB® / SIMULINK®. Algunas son genéricas para cualquier tipo de robot definido con el método de Denavit-Hartenberg y otras aprovechan los modelos de CAD de robot industriales para estudiar la cinemática y dinámica de robots concretos.

#### *Cinemática del robot*

La cinemática del robot estudia el movimiento del mismo con respecto a un sistema de referencia. La cinemática estudia dos problemas particulares en el robot.

El primero de ellos, se conoce como el problema cinemático directo, y consiste en determinar, los valores conocidos de las articulaciones, cual es la posición del extremo final del robot con respecto al sistema de coordenadas que se toman como referencia.

El segundo es el denominado problema cinemático inverso, este resuelve la configuración que debe adoptar el robot para que su extremo alcance una posición y orientación conocida. (Barrientos A. 2007).

#### *Cinemática directa del robot IRB-2600*

Un robot manipulador puede considerarse una cadena cinemática formada por objetos rígidos o eslabones unidos entre sí mediante articulaciones. Se puede establecer un sistema de referencia fijo situado en la base del robot y describir la localización de cada uno de los eslabones con respecto a dicho sistema de referencia.

Las herramientas matemáticas que se han utilizado para describir la configuración del robot son las matrices de transformación homogénea y el algoritmo de Denavit-Hartenberg. (Barrientos A. 2007). En el cuadro 1 se muestran los parámetros del algoritmo de Denavit-Hartenberg correspondientes a la cinemática directa del robot IRB-2600.

$i$	$\theta_i$	$d_i$	$a_i$	$\alpha_i$
1	$\theta_1 - 90^\circ$	445	150	-90
2	$\theta_2$	0	700	0
3	$\theta_3$	0	0	90
4	$\theta_4$	-795	0	-90
5	$\theta_5$	0	0	90
6	$\theta_6$	0	-85	180

Cuadro 1: Valores de Denavit-Hartenberg correspondiente al robot IRB-2600.

#### Comunicación de la aplicación propuesta con el robot real

La comunicación entre la aplicación de MATLAB® / SIMULINK® y el robot se podrá haber realizado usando múltiples protocolos. Las ventajas e inconvenientes de dos de ellos se resumen a continuación:

Comunicación por OPC: ABB dispone de un servidor OPC que puede ser ejecutado desde una computadora. Esta aplicación detecta los robots conectados a la computadora, ya sean reales o virtuales, y extrae de ellos, en cada período de muestreo fijado, la información del sistema, sus variables de entradas y salidas, y las variables persistentes del programa Rapid del robot. Desde una aplicación OPC® cliente se puede leer y modificar dichas variables. No es preciso que la lectura y escritura estén sincronizadas, cuando se modifique una variable desde el exterior, dicha variable queda modificada en la estación del robot.

La comunicación entre la aplicación de Matlab y el robot de ABB ha sido realizada haciendo uso del servidor OPC proporcionado por ABB, y del cliente OPC proporcionado por la aplicación "OPC Toolbox®" de MATLAB®. (MathWorks 2014). La comunicación se ha realizado usando variables de salida analógicas virtuales por una razón de seguridad.

La escritura en variables de salida en RAPID es posible en estado manual y automático, mientras que la escritura en variables persistente solo es posible en automático. Con fines educativos y por seguridad, el robot IRB-2600 se suele usar en modo manual, ya que en este modo se puede parar el robot ante cualquier eventualidad dejando de pulsar el botón de hombre muerto. Se han detenido seis variables de salida virtuales normalizadas entre [-100; 100] para transmitir la información de los seis ejes del robot.

La forma de comunicarse entre MATLAB® y el robot IRB-2600 es la siguiente:

- MATLAB®: Cuando la aplicación de MATLAB® quiera cambiar la posición del robot debe modificar el valor de las variables del servidor OPC correspondientes a los ejes del robot.
- RAPID: El programa en RAPID consiste en un bucle infinito que lee las salidas analógicas correspondientes a los ejes y desplaza al robot a la posición definida por los valores de las variables.

#### Descripción de las aplicaciones.

En este trabajo se han desarrollado tres aplicaciones para modelar y simular el robot IRB-2600 y sus movimientos respectivamente con MATLAB® y después poder mover el robot real, o el virtual si no se cuenta con el robot real. Las dos primeras aplicaciones hacen uso de las funciones de "Robotic Toolbox®" (Corke 2013) y "ARTE®". (Aparicio 2014). La tercera aplicación, hace uso de las herramientas de SimMechanic®.

#### Aplicación grafica de simulación

Primero se modela el robot IRB-2600, con su cinemática directa e inversa y se programan una función plot irb2600() para visualizar las posiciones del robot en 3D. La cinemática directa y las trayectorias del robot se han diseñado utilizando la aplicación de (Corke 2014), mientras que la cinemática inversa ha sido obtenida de la aplicación de (Aparicio 2014).

A partir de los ficheros "\*.stl" del robot, se ha construido una función plot irb2600() con las siguientes objetivos. La figura 1 muestra el robot simulado similar al robot real que existe en el laboratorio.

- Dibujar el robot en 3D con la posición de ejes deseada. Se puede dibujar una secuencia de posiciones si la entrada es una matriz de posiciones de ejes. Se puede definir un tiempo de parada entre posiciones que va a ser útil para sincronizar el movimiento de la gráfica y el del robot real.
- Cuando el programa de Rapid se activa, el robot se moverá a las mismas posiciones que el robot simulado. El tiempo de espera entre posiciones es un parámetro de sintonización entre el robot y la simulación.

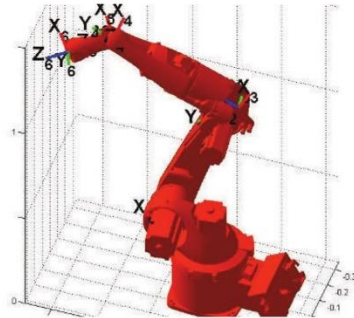


Figura 1. Robot simulado en 3D.

Las siguientes imágenes muestran las pruebas que se hicieron en las figuras 2 y 3 muestran la posición inicial del robot simulado y el robot real respectivamente. Las posiciones finales que el robot simulado y el real tienen una vez transferido el software al controlador, se muestran en las figuras 4 y 5 respectivamente.

*Aplicación grafica (GUIDE®) para el movimiento del robot*

A partir de la función plot irb2600() se ha realizado una aplicación grafica en GUIDE® de MATLAB® que permite al usuario realizar de forma segura los movimientos de ejes por ángulos y posiciones cartesianas. Dichos movimientos se visualizan con la función plot irb2600() y podrán ser enviadas a las variables de los ejes del robot a través del servidor OPC. Con ello, el usuario podrá mover el robot con una aplicación grafica de MATLAB®.

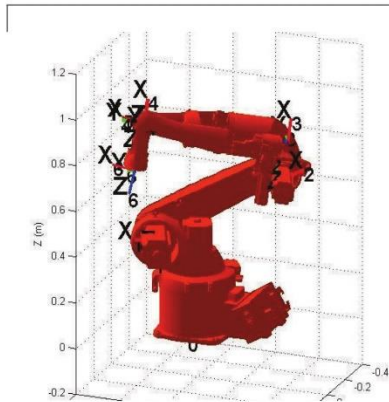


Figura 2. Posición inicial del robot simulado.



Figura 3. Posición inicial del robot real.

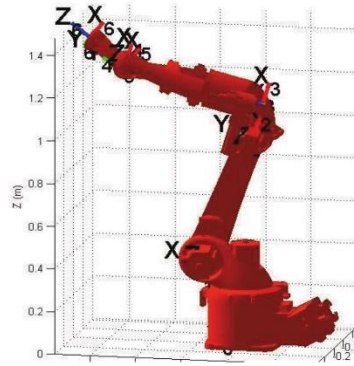


Figura. 4 Posición final del robot simulado.



Figura. 5 Posición final del robot real.

Las principales características de esta aplicación son las siguientes:

- Movimiento a una posición en coordenadas articulares. Se puede grabar dicha posición para simular una trayectoria.
- Movimiento a una posición del punto de trabajo.
- Se puede grabar dicha posición para simular una trayectoria.
- Movimiento del robot entre las posiciones grabadas anteriormente, ya sean posiciones articulares o del punto de trabajo.
- Movimientos de demostración del robot.
- Conexión con OPC para comunicar datos.

La figura 6 muestra la pantalla inicial de la aplicación gráfica. Se aprecian los diferentes botones para mover el robot mediante coordenadas articulares, coordenadas de punto de trabajo, y resto de elementos mencionados.

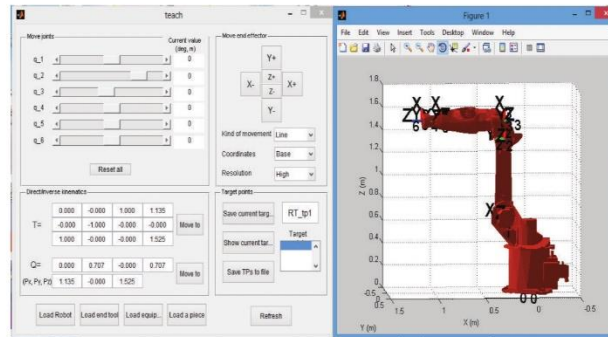


Figura 6. Pantalla inicial de la aplicación gráfica.

#### Aplicaciones didácticas y de investigación.

El objetivo de las aplicaciones realizadas en este artículo es servir de puente entre el estudio matemático y simulado de los sistemas robotizados y el práctico de movimiento real de un robot comercial. Se desea que el alumno pueda realizar algoritmos semejantes a los implementados en lenguaje RAPID y otros algoritmos nuevos que el alumno desarrolle.

Como ejemplos de ejercicios realizables con estas aplicaciones se pueden mencionar:

- Modelar trayectorias interpolando puntos en el espacio de coordenadas auriculares.
- Modelar trayectorias interpolando puntos en el espacio de coordenadas espaciales del punto de trabajo.
- Mover el punto de trabajo del robot para minimizar el error entre la posición de dicho punto y una referencia dada, usando un algoritmo de optimización local. Usar para ello incrementos en las coordenadas auriculares.
- Mover el punto de trabajo del robot en la dirección medida por un sensor de esfuerzos, que el robot IRB-2600 (que actualmente el robot no tiene).
- Mover el punto de trabajo del robot hacia una posición móvil detectada por una cámara de visión (proyecto futuro) asociada a una aplicación de MATLAB®.
- Programar las trayectorias punto a punto, adicionando al robot una pistola para soldadura y que este pueda realizar las trayectorias repetidamente (En proceso).

#### Conclusiones

Se han presentado la propuesta de unir varias aplicaciones usando el "Robotic Toolbox®" de MATLAB® / SIMULINK® y su comunicación con el software de ABB para controlar, simular, diseñar e implementar los movimientos de un robot industrial. Esto es posible gracias a que la herramienta de comunicación entre el robot y las aplicaciones de MATLAB® la cual se lleva a cabo por medio de un intercambio de información usando el servidor OPC de ABB y el cliente OPC de MATLAB®. Con esta herramienta los alumnos pueden realizar en un entorno visual la programación previa de los movimientos de cada eslabón del robot para una aplicación específica. Una vez verificado el diseño de los movimientos por simulación, es posible implementarlos de manera rápida en el robot real, en caso de contar con él. Eliminando movimientos riesgosos tanto para el usuario como para el equipo.

#### Referencias

- Aparicio A. (2014) ARTE (A Robotics Toolbox for Education) Universidad Miguel Hernández (Elche, España), [http://arvc.umh.es/arte/index\\_en.html](http://arvc.umh.es/arte/index_en.html)
- Barrientos A., Peñín L.F., Balaguer C. y Aracil P. (2007) Fundamentos de Robótica (2 edición), MacGraw-Hill.
- Boney I. (2013) ROKISIM (Robot Kinematic Simulation), Escuela de Superior de Tecnología (Quebec, Canadá) <http://www.parallemic.org/RoKiSim.html>.
- Corke, P. (2013) Robotics, Vision & Control, Springer
- Corke Peter (2014) Robotics Toolbox. [http://petercorke.com/Robotics\\_Toolbox.html](http://petercorke.com/Robotics_Toolbox.html).
- González Rodríguez and Luis Hernández Santana, "Platform to develop real time visual servoing control in kinematics systems", in revista Ingeniería Mecánica, vol. 15, no. 3, pp. 233-241, 2012. ISSN 1815-5944.
- Hassine Belhadj, Saber Ben Hassen, Khaled Kaániche and Hassen Mekki, "KUKA Robot control based Kinect image analysis", in IEEE 2013 International Conference on Individual, and Collective Behaviors in Robotics, pp. 21-26. MathWorks (2014). GUIDE Toolbox User's Guide.
- MathWorks (2014). SIMULINK User's Guide.
- MathWorks (2014). SimMechanic User's Guide.
- MathWorks (2014). SimMechanic Link User's Guide.
- MathWorks (2014). OPC Toolbox User's Guide.
- MathWorks (2014). MATLAB® User's Guide.
- MathWorks (2014). OPC Toolbox User's Guide
- Ollero A. (2001) Robótica, manipuladores y robots móviles, Editorial Macombo.
- Pérez Cisneros M.A., Cuevas Jiménez E.V., Zaldívar Navarro D. (2014) Fundamentos de Robótica y Mecatrónica con MATLAB y SIMULINK. Editorial RA-MA.
- Spong M., Hutchinson S y Vidyasagar (2006) M., Robot, Modeling & Control, Wiley.
- Velásquez Lobo, Ramirez Cortés and Rangel Magdaleno, "Modeling a Biped Robot on Matlab/SimMechanics", in IEEE International Conference on Individual, and Collective Behaviors in Robotics, pp. 203-206.

## ANEXO C) RECONOCIMIENTOS



## REFERENCIAS BIBLIOGRAFICAS

- [1] [www.um.edu.ar/catedras/claroline/backends/download.php?url](http://www.um.edu.ar/catedras/claroline/backends/download.php?url).
- [2] [www.quees.info/que-es-software.htm](http://www.quees.info/que-es-software.htm)
- [3] Benjamin Keiser (September 2012) “Torque Control of a KUKA youBot Arm”, master thesis, Robotics and Perception Group University of Zurich
- [4] [www.kuka-robotics.com/Robotics](http://www.kuka-robotics.com/Robotics)
- [5] <http://proton.ucting.udg.mx/materias/robotica/r166/r63/r63.htm>
- [6] Mr Manigandan M Senior Lecturer, Department of ECE, Velammal Engineering College Chennai, INDIA
- [7] Olivares-Mendez 2014 International Conference on Unmanned Aircraft Systems (ICUAS) May 27-30, 2014. Orlando, FL, USA Miguel A
- [8] <http://www2.ing.puc.cl/icm cursos/procesos/apuntes/cap4/42/421/>
- [9] [http://webs.demasiado.com/ing\\_industrial/ingenieria/mecanismos/](http://webs.demasiado.com/ing_industrial/ingenieria/mecanismos/)
- [10] Robótica: Manipuladores y robots móviles Aníbal Ollero Baturone Alfaomega marcombo, 2001 TJ 211. O43 2001
- [11] Robot dynamics and control Mark W. Spong, M. Vidyasagar John Wiley & Sons, 1989 TJ 211. 4566 1989.
- [12] Robotica Craig, John J. Pearson Educación, 2006 ISBN: 970-26-0772-8
- [13] Fundamentos de robótica Antonio Barrientos, Luis Felipe Peñin, Carlos Balaguer, Rafael Aracil, Mc Graw Hill, 1997 TJ 211. B781.
- [14] Corke, P. (2013) Robotics, Vision & Control, Springer.

- [15] Peter Corke (2014) Robotics Toolbox. [http://petercorke.com/Robotics Toolbox.html](http://petercorke.com/Robotics%20Toolbox.html).
- [16] Gil Aparicio A. (2014) ARTE (A Robotics Toolbox for Education) Universidad Miguel Hernández (Elche, España), [http://arvc.umh.es/arte/index en.html](http://arvc.umh.es/arte/index%20en.html)
- [17] MathWorks (2014). GUIDE Toolbox User's Guide.
- [18] MathWorks (2014). SIMULINK User's Guide.
- [19] MathWorks (2014). SimMechanic User's Guide.
- [20] MathWorks (2014). SimMechanic Link User's Guide (2014).
- [21] ABB robotic. Application manual: IRC5 OPC Server. Doc. ID: 3HAC023113-001.
- [22] MathWorks (2014). OPC Toolbox User's Guide.
- [23] ABB robotic. Manual del operador: RobotStudio. Doc. ID: 3HAC032104-005.
- [24] MathWorks (2014). MATLAB<sup>®</sup> User's Guide.
- [25] Hassine Belhadj, Saber Ben Hassen, Khaled Kaániche and Hassen Mekki, “KUKA Robot control based Kinect image analysis”, in IEEE International Conference on Individual, and Collective Behaviors in Robotics, pp. 21-26.
- [26] Rene González Rodríguez and Luis Hernández Santana, “Platform to develop real time visual servoing control in kinematics systems”, in revista Ingeniería Mecánica, vol. 15, no. 3, pp. 233-241, 2012. ISSN 1815-5944.
- [27] Velásquez Lobo, Ramirez Cortés and Rangel Magdaleno, “Modeling a Biped Robot on Matlab/SimMechanics”, in IEEE International Conference on Individual, and Collective Behaviors in Robotics, pp. 203-206.
- [28] Bonev I. (2013) ROKISIM (Robot Kinematic Simulation), Escuela de Superior de Tecnología (Quebec, Canadá) <http://www.parallemic.org/RoKiSim.html>.
- [29] ABB robotic. Manual del operador: Introducción a RAPID (RobotWare 5). Doc. ID: 3HAC029364-005
- [30] ABB robotic. Manual del operador IRC5 con FlexPendant. Doc. ID: 3HAC16590-5.