Universidad Autónoma de San Luis Potosí
Facultad de Ingeniería

Centro de Investigación y Estudios de Posgrado

Sistema para el Guiado Activo de Robots Industriales

Tesis
Que para obtener el grado de
Maestro en Ingeniería Mecánica
Opción Mecatrónica y Sistemas Mecánicos

Presenta:

Ing. Reynaldo Soubervielle Montalvo

Asesor:

Dr. Emilio Jorge González Galván

San Luis Potosí, S.L.P., Diciembre 2013

Agradecimientos

A mi Familia que siempre me dio palabras de aliento.

En especial, gracias a mi **Madre y Hermanos: Jorge y Carlos**, por el apoyo, ejemplo y cariño que incondicionalmente brindaron en todo momento.

Al **Dr. Emilio Jorge González Galván**, gracias, por la confianza depositada en mí, por darme todas las herramientas necesarias para realizar este proyecto y por la paciencia masiva que tuvo durante este proceso.

A mis amigos del laboratorio y del posgrado: Kessler, Ares, Quique, Liz, Aarón, Lina, Gaby, Cesar, Isela, Ubaldo, Jorge, gracias, por mostrar siempre un auténtico interés y disposición en los momentos de necesidad.

A mis sinodales: **Dr. Mauro Eduardo Maya Méndez, Dra. Nancy Visairo Cruz, Dr. Hugo Iván Medellín Castillo,** por sus críticas y cometarios, siempre dados con el afán de que mi logro se enalteciera.

Al personal responsable del **Laboratorio de Máquinas y Herramientas: TMH Gerardo Jasso Silva, Magdaleno Ponce Mireles y M.I Carlos Alberto Amaro Betancourt** por su apoyo y consejo en la fabricación de piezas para el proyecto.

Al **Consejo Nacional de Ciencia y Tecnología (CONACyT)** por su apoyo y fomento a la investigación.

¡Gracias a todos!

Resumen

El presente documento de tesis aborda el tema de programación de robots manipuladores de uso industrial. En específico, la implementación de un sistema de programación no convencional denominado de guiado activo, en robots de arquitectura cerrada. Este tipo de sistemas requiere la interacción física humano-robot.

La estrategia aplicada al control de interacción, es el denominado control cinemático de impedancia. Esta estrategia ha demostrado un buen desempeño en tareas que involucran interacción humano-robot en sistemas robóticos de arquitectura cerrada.

El sistema de programación está estructurado por medio de tres interfaces: una es la ya mencionada interacción física humano-robot, con la cual el usuario llevará al robot hacia las posiciones de interés; la segunda es una comunicación por medio de señales generadas con botones, de manera que el usuario pueda seleccionar posiciones y trayectorias, además de factores que facilitan el guiado del robot durante la interacción física como pueden ser la que grados de libertad se operan y a qué velocidad; finalmente, el sistema cuenta con una interfaz visual desplegada en la pantalla de la terminal de enseñanza (comúnmente conocida como: “teach pendant”) la cual da la confirmación al usuario de cada selección realizada.

Este tipo de sistemas de programación hace posible la adaptación de robots industriales no diseñados con esta opción, para su uso y programación por personas con mínimos conocimientos en robótica como son: operadores especializados en procesos de manufactura en pequeñas y medianas empresas y prestadores de servicios en terapias de rehabilitación donde, debido a su desarrollo y disminución en costo, los robots están incursionando cada vez más.

Contenido

Introducción

El uso de robots industriales en pequeñas y medianas empresas y en roles más orientados a servicios como pueden ser tareas de rehabilitación, ha sido más frecuente debido a que estos dispositivos se han hecho considerablemente más poderosos, inteligentes y económicos. Como consecuencia de lo anterior, los robots son usados cada vez más por personas con mínimas habilidades técnicas, lo que hace necesario sistemas de programación más fáciles y flexibles.

Un robot industrial es básicamente un manipulador multifuncional reprogramable diseñado para mover materiales, piezas o herramientas a través de trayectorias programadas con el propósito de realizar casi cualquier tarea, lo cual permite su adaptación de manera rápida y económica a diferentes aplicaciones. La programación de un robot se puede definir como el proceso mediante el cual se indica a éste la secuencia de acciones que deberá llevar a cabo durante la realización de una tarea. Estas acciones consisten generalmente en moverse a puntos predefinidos y manipular objetos del entorno. Dicha programación es una tarea compleja y requiere una considerable habilidad y conocimiento técnico ya que la mayoría de los robots tienen interfaces complejas, que por lo general implican un lenguaje de programación basado en unas pocas abstracciones de alto nivel (programación textual), o en el mejor de los casos el uso de sistemas de guiado muy poco intuitivos que requieren periodos largos de entrenamiento. Debido a que el promedio de usuarios no desea programar su robot en un nivel bajo, es necesario un sistema que proporcione el nivel requerido por el usuario para que éste logre el control sobre las tareas que el robot debe realizar, prescindiendo de una gran inversión de tiempo y dinero en capacitación.

En los robots industriales los procesos de programación fuera de línea no han sido tan exitosos, entre otras razones, debido al que los robots industriales se consideran dispositivos de precisión baja. En contraste, el éxito de la programación en línea recae en la repetitividad, característica en la que destacan los robots industriales. Lo anterior ha hecho que los sistemas más comúnmente utilizados en la industria son los de guiado donde durante la enseñanza el robot es llevado a través de las posiciones que el robot ha de alcanzar durante la ejecución del programa. En este tipo de sistemas es necesario guiar al robot operándolo con una terminal de mando conocida como “teach pendant”, lo cual requiere una etapa de adiestramiento por parte del usuario. Aunque este método ha mostrado ser la opción más exitosa integrada en el grueso de los sistemas robóticos, los proveedores permanecen en la búsqueda de interfaces más amigables, dado que las dificultades y el tiempo requerido para su programación representa la causa principal por la

que plantas manufactureras en vías de crecimiento y otros sectores no dan el paso a tal grado de automatización. El sistema de guiado convencional, muy similar en los fabricantes de robots, es un lenguaje no compilado orientado a procesos, que se basa en un juego de instrucciones disponibles las cuales se ejecutan secuencialmente. Las sentencias de movimiento están asociadas a posiciones que se obtienen directamente de la información de los sensores en cada junta. Es la necesidad de conducir el robot hacia cada posición, además de la gran cantidad de botones y opciones en la terminal de enseñanza, lo que hace lento y difícil el proceso de programación.

Objetivo

El objetivo de este trabajo consiste en implementar un sistema de programación que reduce las carencias mencionadas anteriormente, que el avance tecnológico actual presenta, a partir de las capacidades que los sistemas robóticos ofrecen como son: un lenguaje de programación de bajo nivel, comunicación con sensores y señales digitales o analógicas y un sistema de programación por guiado.

La propuesta de esta tesis es generar un sistema de programación alterno orientado a generación de trayectorias mediante el cual se puedan crear programas básicos de movimiento a través de puntos alcanzables en el espacio de trabajo del robot. El posicionamiento del robot se llevará a cabo vía interacción física humano-robot, mediante un sensor de fuerza/par montado en el efector final del robot, proporcionándole al sistema una de las maneras más intuitivas para indicar movimiento, sin la constante necesidad de estar consciente de la ubicación y orientación de los sistemas de referencia. El usuario indicará al sistema cuándo generar un comando de movimiento mediante señales digitales accionadas con botones instalados en una terminal portable, más sencilla que la proporcionada por el fabricante, pero que deberá compartir con ésta algunas características de seguridad como un paro de emergencia del tipo de hombre muerto.

La plataforma de experimentación será el sistema robótico denominado *intelligent robot* de la marca FANUC que consiste en un brazo robótico de 6 grados de libertad (GdL) LR-Mate 200iC con un sensor de fuerza/torque FS-10iA. Las opciones que presenta para su programación son: un sistema de guiado vía terminal de enseñanza denominado (TPE; Teach Pendant Editor), un lenguaje de alto nivel fuera de línea denominado KAREL, el cual es una de las opciones del paquete ROBOGUIDE. Además cuenta con la opción de crear una celda de manufactura virtual en la cual se pueden editar y probar los programas generados de manera segura utilizando el TPE virtual, con la ventaja de que el software tiene la capacidad de mostrar información extra como son: trayectorias velocidades y puntos de colisión a lo largo de la trayectoria generada.

El resultado de una sesión de programación consistirá en un programa básico del tipo TPE el cual podrá ser modificado en su totalidad usando cualquiera de las opciones con que cuenta el sistema robótico como son: vía terminal de enseñanza, o mediante el simulador fuera de línea. Por otra parte, la información de cada posición quedará registrada en formato de archivo de texto, de manera que ésta pueda ser decodificada y utilizada en un programa de lenguaje KAREL. De esta manera la información generada puede ser modificada en el nivel de lenguaje que corresponde de acuerdo a las necesidades y propósito particular de cada usuario. En este caso es importante hacer notar que los

lenguajes más complicados tienen más capacidades como es el caso de KAREL donde es posible definir un comportamiento mientras que en el lenguaje TPE, que es más simple, sólo es posible definir un proceso.

Motivación

Los robots son cada vez más potentes, con más sensores, más inteligencia, y componentes más baratos. Esto ha hecho que los robots se desempeñen no solo en entornos industriales controlados, sino que también, en entornos de servicios no controlados tales como viviendas, hospitales y lugares de trabajo, donde se realizan tareas que van desde servicios de entrega hasta el propósito del entretenimiento. Es este aumento en la exposición de robots a personas no calificadas, lo que requiere que los mismos sean más fáciles de programar.

En la última década se han presentado diversos desarrollos tecnológicos orientados al apoyo en terapias de rehabilitación de pacientes que han perdido movilidad de una o varias extremidades. Sin embargo, no todas las instituciones de atención a este tipo de problemas son capaces de adquirirlos ya que suelen ser muy costosos. Una alternativa muy viable se basa en la adaptación de robots industriales para su uso en este tipo de tareas ya que cuentan con las habilidades necesarias entre las que se pueden mencionar repetitividad y robustez. En diferentes universidades ya se ha hecho investigación robótica enfocada al área de la medicina. En el laboratorio de Robótica en la Universidad Autónoma de San Luis Potosí se han desarrollado proyectos que involucran robots industriales del tipo antropomórfico para aplicaciones médicas de rehabilitación [67], [60]. De llevarse a la experimentación como en el caso de [67], este tipo de sistema serán programados por profesionistas con vastos conocimientos en los tipos de movimientos que favorecen la recuperación de un paciente ante una lesión, pero escasos en cuanto al uso y funcionamiento de máquinas industriales, por lo que su aplicación exige el desarrollo e implementación de interfaces de programación que permitan generar, de manera rápida y sencilla, la trayectoria deseada. A diferencia del uso de un robot en la industria, donde éste es programado una vez para realizar un mismo proceso durante mucho tiempo en una serie de piezas idénticas entre sí, para su uso en terapias será necesario realizar programas que cubran las necesidades de cada paciente donde las trayectorias dependerán del tipo y grado de la lesión y de la complexión física del mismo.

El sector industrial no está exento de la necesidad de sistemas de programación de robots más simples, ya que aunque cuentan con las capacidades necesarias, siempre son favorables métodos que disminuyan la relación tiempo en programación/tiempo en producción. A nivel PYME, caracterizadas por manufacturar lotes de piezas pequeños, la transición hacia sistemas automatizados por robots se hace difícil, De ahí que el método de programación juega un papel importante ya que la frecuencia con que se realizarían programas es aún mayor.

Diferentes instituciones de investigación han enfocado sus esfuerzos en generar sistemas de programación con características similares a las propuestas para este trabajo de tesis. Sin embargo, la información respecto de los detalles de la implementación no se presenta del todo. Además se pueden encontrar una gran cantidad de marcas robóticas para

las cuales, por alguna razón, sistemas de este tipo no se han implementado. Tal es el caso de una de las marcas líderes en venta mundiales en sistemas robóticos; la marca FANUC.

Contribuciones del trabajo

El control de impedancia ha sido implementado en otros trabajos de tesis utilizando la misma infraestructura [65, 66]. Sin embargo, el desempeño no había alcanzado el grado que se requiere para ser empleado como control de interacción humano–robot en un sistema de programación por lo que una aportación importante consistió en mejorar su desempeño, de manera que pueda ser utilizado en otros proyectos que así lo requieran.

En la obtención del objetivo final de esta tesis se generaron otras contribuciones entre las que se puede mencionar:

- Implementación de una botonera portable en una mano, cuyas señales pueden ser utilizadas por cualquier programa. Ésta cuenta con un botón de paro de emergencia estilo de hombre muerto.
- Se diseñó un aditamento que, unido al robot, permite contar con un plato de montaje para herramienta y otro para el sensor el cual permite desmontar el sensor de manera rápida y sin herramienta.

El trabajo realizado se presentó en formato de artículo en el Congreso Mexicano de Robótica COMRob 2012 con el título de "Sistema para el guiado activo de robots industriales".

Estructura de la tesis

El presente trabajo de tesis se estructura de la siguiente manera:

En el capítulo 1 se revisa el estado del arte de los sistemas de programación de robots industriales.

En el capítulo 2 se hace una revisión de los sistemas de interacción humano robot.

En el capítulo 3 se presenta la metodología y desarrollo que se llevó a cabo para obtener el objetivo planteado en este trabajo de tesis.

En el capítulo 4 se presentan las pruebas realizadas y se discuten los resultados

Capítulo 1

1. Sistemas de programación de robots industriales

Un robot industrial se puede definir, de acuerdo a la norma ISO 8373, como un manipulador multifuncional reprogramable controlado automáticamente en tres o más ejes, el cual puede estar fijo en un lugar móvil para su uso en aplicaciones de automatización industrial. Estos dispositivos son ampliamente utilizados en la industria para todo tipo de procesos. Sin embargo, tal grado de automatización no es accesible en todos los sectores ni en todos los niveles económicos como son sectores de servicio y en pequeñas y medianas empresas respectivamente, aun cuando los precios en los equipos robóticos han tenido una disminución importante en los últimos años. Esto en buena medida es debido a que la inversión de tiempo y dinero en adiestramiento y programación no se justifica cuando se tiene una relación tiempo en programación/tiempo en producción alta.

La programación de un robot se puede definir como el proceso mediante el cual se indica a éste la secuencia de acciones que deberá llevar a cabo durante la realización de una tarea. Estas acciones consisten generalmente en moverse a puntos predefinidos y manipular objetos del entorno. Desde la creación del primer robot industrial en 1959 surgió la necesidad de desarrollar sistemas que hicieran posible la integración de los mismos en los procesos que encajaban dentro de las 3Ds del inglés (dull; tedioso, dirty; sucio, dangerous; peligrosos). En esta sección se analizará el estado de la técnica en sistemas de programación de robots industriales.

1.1. Clasificaciones

Desde el surgimiento del primer robot industrial en 1959 [1] la programación de éste se realizó mediante el esquema de una fase enseñanza y repetición en producción. Durante la enseñanza se guardaban las coordenadas articulares en una bobina magnética, para reproducirse durante la etapa de producción. Debido al éxito de los robots industriales en muy variadas tareas, surgió la necesidad de nuevas maneras de programar las capacidades que el desarrollo tecnológico otorgaba a los robots al paso del tiempo, permaneciendo la búsqueda de métodos sencillos que permitieran su integración para uso industrial, razón por la cual la idea básica de enseñanza y repetición ha permanecido en uso hasta la actualidad con ciertas variantes debido a las capacidades que la tecnología del momento permite. A continuación se presentan algunas de las clasificaciones que han surgido durante el desarrollo de la robótica industrial.

En un inicio el simple esquema enseñanza y reproducción era suficiente por lo que este método permaneció como la única opción comercial, hasta 1979 [2] cuando surgió el primer lenguaje de programación Val de la marca UNIMATION. Aunque para ese entonces ya se habían creado diferentes lenguajes de programación, éstos permanecían en experimentación.

1.1.1. Clasificación de Tómas Lozano

Fue hasta 1982 cuando Tómas Lozano [3] estableció la primera clasificación para los diferentes sistemas de programación de robots industriales. Lozano Pérez dividió los sistemas de programación en tres categorías: sistemas de guiado, concepto que se aborda en la sección 1.3, sistemas de programación a nivel del robot [23, 24] y sistemas de programación a nivel de la tarea [27]. En un sistema de guiado el robot se mueve de forma manual a cada posición deseada mientras que las variables de junta son registradas para su posterior reproducción. En cuanto al sistema de programación a nivel del robot, se refiere al lenguaje de programación de alto nivel que proporciona el fabricante del robot o que se diseña con el propósito de controlarlo, algunos casos se estudiaron en [2]. Por último, en el sistema de programación a nivel de la tarea, se especifica la meta a ser alcanzada, por ejemplo, las posiciones relativas de objetos, por lo que en algunos trabajos a este sistema se le conoce como programación orientada a objetos [5].

1.1.2. Clasificación de Biggs y MacDonald

Geoffrey Biggs y Bruce MacDonald [4] en 2003 dividen el campo de la programación de robots en: programación automática, programación manual y arquitecturas de software [28, 32], como se muestra en la figura 1.1. En los sistemas de programación automática el usuario/programador tiene poco o ningún control directo sobre el código del robot. En la figura 1.1 se resalta programación automática, que es la clasificación en la que corresponde el desarrollo de esta tesis. Los sistemas manuales requieren que el usuario/programador directamente introduzca el comportamiento deseado del robot, por lo general utilizando un lenguaje gráfico o textual. Las arquitecturas del software son importantes para todos los sistemas de programación, ya que proporcionan la infraestructura como es la comunicación con periféricos y sensores, así como el acceso a los robots.

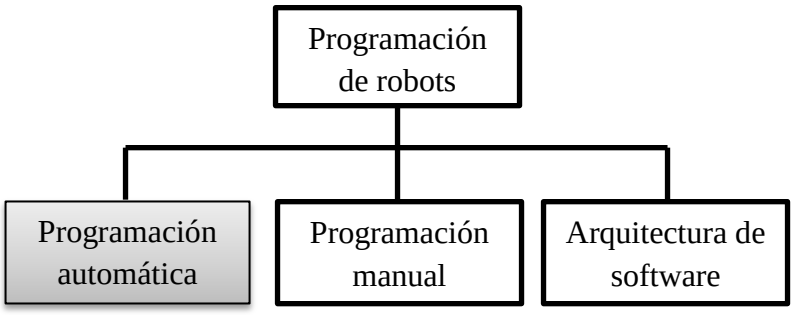


Figura 1.1. Sistemas de programación de robots.

Como se muestra en la figura 1.2, los sistemas manuales de programación pueden dividirse en sistemas de programación textual, que se describen en la sección 1.2, y sistemas de programación gráficos, también conocidos como sistemas basados en iconos [29]. La programación gráfica no se considera programación automática debido a que el usuario debe crear el código del programa del robot manualmente antes de ejecutarlo en el sistema robótico. Existe una correspondencia directa entre los iconos gráficos y las declaraciones del programa. Es pertinente mencionar que los lenguajes basados en comportamiento se refieren a sistemas en los cuales no se especifica procedimiento a

seguir, sino una serie de posibilidades que indican al sistema cómo reaccionar ante un evento, descripción que puede encajar en la clasificación de Lozano para programación a nivel tareas [30, 31]. Mientras que el lenguaje específico del controlador encaja en la clasificación de Lozano para programación a nivel robot. Los sistemas de programación manual también suelen denominarse como sistemas de programación "fuera de línea" ya que se realizan sin la necesidad de la línea de producción.

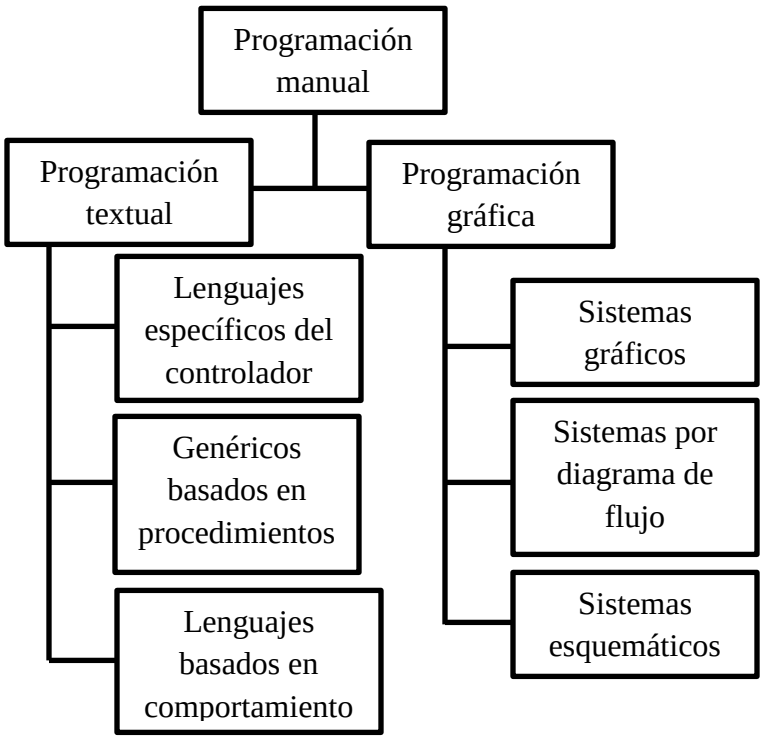


Figura 1.2. Sistemas de programación manual.

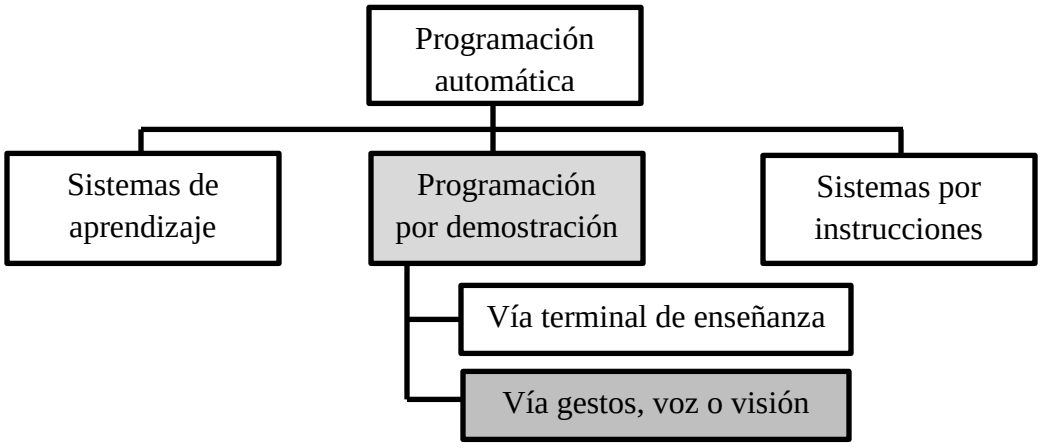


Figura 1.3. Tipos de programación automática.

En cuanto a los sistemas de programación automáticos, el código es generado a partir de información introducida al sistema de manera indirecta en muy variadas formas. A menudo el sistema del robot está en funcionamiento mientras la programación automática se está desarrollando; estos sistemas se han denominado "sistemas en línea". Sin embargo, la programación automática también se realiza en robots virtuales o simulados [8]. La figura 1.3 muestra las tres categorías que pueden ser consideradas dentro de la programación automática: sistemas de aprendizaje [25, 26], programación por demostración (PpD) [16] y sistemas de programación por instrucción. Nótese que la clasificación de Lozano de sistemas de guiado cae dentro de los sistemas de programación por demostración. Los sistemas de aprendizaje crean el programa por inferencia inductiva a partir de ejemplos proporcionados por el usuario y una auto-exploración realizada por el robot. En sistemas instructivos se da una secuencia de instrucciones, por lo general en tiempo real. Esta técnica es más adecuada para el mando de robots que llevan a cabo tareas para las que ya han sido entrenados o programados, este sistema podría ser considerado como el más alto nivel de la programación.

Por lo general, los sistemas PpD trabajan en conjunción con un sistema de reconocimiento de gestos o de reconocimiento de voz. El lenguaje de comunicación basado en voz es el método más natural para los seres humanos para comunicar instrucciones de uno a otro, por lo que es un buen candidato para la instrucción de un robot. Un sistema de lenguaje natural para proporcionar instrucciones a un robot puede ser considerado uno de los sistemas más intuitivos, ya que es de la manera en que el ser humano aprende a interaccionar con su entorno desde etapas tempranas. En algunos sistemas de programación, un lenguaje natural se usa para enseñar al robot cómo moverse a diferentes lugares por rutas específicas.

El sistema de programación por demostración es el más comúnmente usado de los sistemas de programación automáticos. Se muestra resaltado en la figura 1.3 PpD por medio de gestos, es decir interacción física como el área de estudio del presente trabajo de tesis. En éste se pueden usar terminales de enseñanza para la demostración, o bien se pueden utilizar otras maneras más naturales como son gestos, comandos de voz o por visión. Por ejemplo, una tarea de montaje con la terminal de enseñanza. Las posiciones del robot se registran y el resultado se usa para generar un programa en el controlador, que moverá el brazo del robot a través de las mismas posiciones. Por otra parte, el demostrador puede mover el brazo del robot a través de las posiciones requeridas, ya sea físicamente o con el uso de algún sistema de teleoperación. En este caso, las posiciones deseadas se registran y se utilizan para generar el programa de robot. Aunque sencillo, este tipo de sistema ha sido utilizado exitosamente para la creación rápida de programas donde intervienen tareas de montaje.

1.1.3. Concepto actual de PpD

Ya desde los 90's empezaban a surgir desarrollos en los que la información generada durante la enseñanza era procesada para reproducir sólo partes que se consideraban más importantes. La idea tradicional de PpD consistía en simplemente reproducir la tarea enseñada sin variación alguna [4]. La mayoría de los desarrollos actuales asocian cierta inteligencia a los sistemas PpD, que les permita procesar las características de la tarea, para obtener una ejecución mejorada, en vez de sólo imitar la información

almacenada. Por ejemplo, en [14, 15] se presenta un sistema donde, a partir de varias etapas de guiado se generan trayectorias para la misma tarea, que debido a la inconsistencia humana difieren una de otra, pero que tienen la característica de localizarse en zonas libres de colisión. De estas trayectorias se obtiene otra que se encuentra libre de colisiones y que elimina movimientos innecesarios haciéndola más corta y, por tanto, más eficiente.

En [16] una simple trayectoria enseñada, definida por una serie de puntos, se procesa con el algoritmo de Douglas-Peucker para obtener una trayectoria más suave compuesta por menos puntos que la original, finalmente los puntos obtenidos se aproximan a líneas o NURBS. En [17], los datos de la demostración son procesados por un algoritmo que identifica subtareas predefinidas para una tarea de pick-and-place; además, ya que el proceso de guiado es indirecto, establece un trayectoria que es accesible para el espacio de trabajo del robot. En [18], la información de múltiples etapas de guiado generan un volumen a través del cual el movimiento del robot es libre de colisiones, después ese volumen se considera el espacio restringido por donde automáticamente se produce una trayectoria optimizada. En [19] se presenta y categoriza de manera muy completa las técnicas que se involucran en los sistemas PpD, eliminando la división que en [4] se hacía entre sistemas de aprendizaje y PpD.

Se puede concluir que los sistemas PpD, requieren de una etapa, donde, por medio de un sistema de guiado, se le provean ejemplos, para que a partir de éstos, se genere una tarea que bajo ciertas consideraciones tiene las trayectorias más adecuadas o que faciliten sesiones de programación posteriores al sugerir la siguiente acción necesaria, para obtener un programa eficiente [20]. Algunos de los trabajos más recientes en PpD son [33, 34, 35, 37].

1.1.4. Clasificación de la OSHA

La Administración de Seguridad y Salud Ocupacional (Occupational Safety and Health Administration, OSHA) es una agencia del departamento de trabajo de Estados Unidos, que hace cumplir las leyes sobre la seguridad y salud de trabajadores. La OSHA en su manual técnico [6] en la sección de riesgos para seguridad, capítulo robots industriales y seguridad en sistemas robóticos, propone una clasificación en tres tipos de programación de robots industriales: fuera de línea, guiado vía terminal de enseñanza (lead-through) y guiado vía interacción física (walk-through). Este manual es estudiado por los auditores de la OSHA, para tener los conocimientos acerca de los sistemas que están en uso en las plantas industriales y de las medidas de seguridad que requiere cada uno.

A diferencia del de Lozano, el trabajo de Biggs y MacDonald no se limita al análisis de sistemas de programación de robots industriales sino que, considera nuevos sistemas de programación diseñados para robots no industriales, que habían surgido, debido a que entonces había una creciente inserción de los robots en otros sectores. La OSHA por su parte, ya que actualmente en robótica industrial se cuenta con solo dos sistemas provistos comercialmente; que son los de guiado y programación textual, establece una clasificación enfocada a las posibilidades actuales.

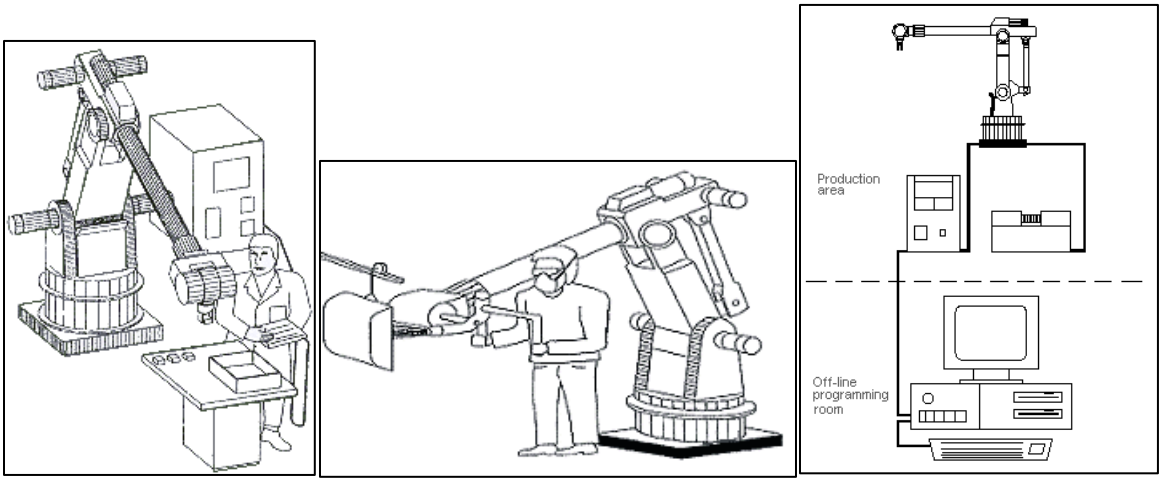


Figura 1.4. Tipos de programación: izquierda, guiado vía terminal de enseñanza (lead-through); centro, guiado vía interacción física (walk-through); derecha, programación fuera de línea.

1.2. Sistemas de programación textual para robots industriales

Se consideran lenguajes de alto nivel compilados cuya sintaxis es similar a PASCAL o C, por medio de los cuales se puede definir un comportamiento en el sistema robótico; dicho comportamiento puede estar definido por información obtenida del entorno a través de señales de sensores u otros dispositivos eléctricos.

1.2.1. Origen

La incorporación de los sistemas de programación textuales a los sistemas robóticos tuvo su origen en las deficiencias que los sistemas de guiado vía terminal de enseñanza presentaban como son:

- En la realización de programas para estibar productos, donde en lugar de enseñar las nuevas posiciones, éstas pueden ser calculadas, por lo que el uso de la terminal de enseñanza. resultaba complicado, al no tener dicha posibilidad.
- Los sistemas de guiado, a diferencia de los lenguajes textuales, requieren el uso de la unidad mecánica, es decir del robot, por lo que durante la programación éste no puede utilizarse en producción.
- Los programas realizados con terminal de enseñanza no podían ser llevados de un robot a otro. En lenguaje textual tal procedimiento es posible si los sistemas robóticos cuentan con la precisión adecuada.
- La integración CAD/CAM además de sensores como pueden ser sensores de fuerza o cámaras, requería de lenguajes de programación de alto nivel.

1.2.2. Primeros desarrollos

En 1984 [2] se hace una evaluación de los lenguajes de programación que se habían desarrollado hasta entonces mediante una comparativa con base en las capacidades que cada uno tenía. Los lenguajes analizados se caracterizaban por haber alcanzado su aplicación en la industria o en laboratorios, aunque nuevos lenguajes se encontraban en

desarrollo e introducción por parte de proveedores de sistemas robóticos de USA, Europa y Japón, de los cuales se esperaba cubrieran las carencias identificadas en los lenguajes desarrollados hasta entonces. Estos nuevos lenguajes, denominados lenguajes específicos del controlador [4] o lenguajes genéricos basados en procedimientos, los cuales consisten en lenguajes de programación provistos por el fabricante del sistema robótico o lenguajes obtenidos de la adaptación de lenguajes genéricos como el caso de C++ [7], donde mediante la incorporación de variables y rutinas se facilita la programación de tareas de manipulación de objetos.

1.2.3. Estado actual

En 2010 en [13] se presenta una recopilación de los lenguajes de programación que existen, además en este trabajo los lenguajes textuales son considerados como una herramienta más en el desarrollo de un tarea de programación fuera de línea (OLP; Offline Programming). El concepto de OLP se extiende para considerar el uso de celdas de manufactura virtual y otras características que los proveedores de sistemas robóticos han desarrollado, además de requerir una etapa de calibración en línea, es decir que aun dada la potencia que han alcanzado los sistemas OLP aun no es posible prescindir de los sistemas de guiado, ya que mediante la herramienta TOUCH UP (que se puede traducir como retocar) en varios fabricantes es posible volver a enseñar cierta posición que así lo requiera.

Tabla 1.1 Paquetes para OLP.

	Software	Compañía
Paquetes robóticos genéricos	Delmia (IGRIP, ENVISION): Kineo, CENIT	Dassault Sistems
	RobCAD (Em-workplace)	Technomatix
	Robomaster	Robomaster
	Robsim	Camelot
	Workspace 5	Wat solution
	Cosimir	Festo
Paquetes robóticos de fabricantes de robots	RobotStudio	ABB
	MotoSim	Motoman
	KUKA-Sim, CAMrob	KUKA
	ROBOGUIDE	Fanuc
	WincapsIII	Denso
	3D STUDIO	Stäubli
	MEFA WORKS	Mitsubishi
	Pc-ROSET	Kawasaki
	AX on Desk	Nachi

En la actualidad la mayoría de proveedores de sistemas robóticos cuentan con su propio sistema de programación textual, es decir un lenguaje propio del controlador, el cual es provisto como una de las herramientas de programación de sus equipos. Además, de manera opcional algunas compañías de robótica ofrecen la posibilidad del uso del elemento principal que caracteriza un sistema de programación OLP [13], la cual es una interfaz que

permite integrar modelos CAD de la celda de manufactura y del equipo robótico para, mediante ésta, agregar algunas capacidad que se han vuelto hasta cierto punto esenciales en la implementación de programas en tareas más complejas como pueden ser procesos de ensamblado de piezas. En la tabla 1.1, se muestran los paquetes con que cuentan los fabricantes líderes en el mercado, además de algunos de algunos desarrollados genéricamente por empresas no especializadas en robótica. Cabe mencionar que existen otros desarrollados por instituciones de investigación los cuales permanecen como software libre.

1.3. Sistemas de guiado

Los sistemas de programación mediante guiado representan la metodología más recurrente en la industria para la generación de trayectorias, para procesos como pueden ser alimentación, pintura, soldadura y pulido de piezas, entre otros. En general, esta metodología consiste en que, mediante alguna interfaz sea posible operar el robot de manera que éste adopte las configuraciones requeridas en la tarea a programar, para que éstas sean grabadas para su posterior reproducción. En la actualidad, la interfaz más común se basa en el uso de una consola de control denominada terminal de enseñanza con la cual, a través de botones se modifica la posición de cada grado de libertad, ya sea cartesiano o en juntas. Aunque esta estrategia ha mostrado ser práctica, funcional y relativamente sencilla de utilizar, los nuevos desarrollos en robótica tienen como objetivo interfaces más intuitivas, independientes de sistemas de referencia, con la expectativa de reducir el tiempo consumido en la programación. En esta sección se presenta el desarrollo que han tenido los sistemas de programación por guiado a partir de su origen hasta el estado actual.

1.3.1. Guiado activo y guiado pasivo

El origen de los sistemas de guiado se dio de la mano con el primer robot industrial usado en producción, pues la descripción que se hace de la metodología para programar éste [1], coincide con el concepto de programación por guiado. Actualmente existen variadas maneras de aplicar este método debido a la capacidad de los sistemas robóticos de comunicarse con sensores. Sin embargo, la idea básica de algunas técnicas actuales se basa en metodologías utilizadas en las primeras variantes de los sistemas de programación por guiado. Inicialmente, la programación se dividía en guiado activo y guiado pasivo [9]. En el primer caso se refiere a sistemas en los que el control de los servomotores de cada junta permanece activo durante la programación, es decir mediante algún dispositivo (como teclado, terminal de enseñanza, joystick, etc), se opera el robot para que éste adopte las posturas necesarias en la tarea, estas posturas son grabadas y reproducidas en producción. En el guiado pasivo, que cae dentro de método “walk-through”, existen dos vertientes: guiado pasivo directo [10] y guiado pasivo indirecto [11 y 12]. En el primer caso, el efector final del robot es llevado por el operador a través de la trayectoria deseada mientras las posturas del robot son capturadas con cierta frecuencia, de acuerdo a la fidelidad requerida entre la trayectoria original y la trayectoria reproducida por el robot. Esta metodología exige que el operador aporte la energía para mover el brazo robótico, por lo que, sólo es apropiada para robots pequeños o ligeros. En la programación por guiado pasivo indirecto, el operador mueve una réplica del robot, el cual tiene sensores de posición que

corresponden con los sensores del robot y una configuración idéntica pero mucho más ligera, por tanto, más fácil de mover.

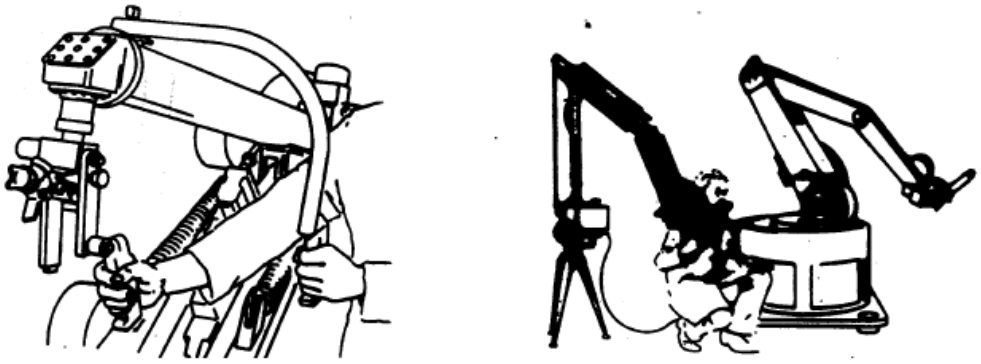


Figura 1.5. Guiado pasivo directo (izquierda) y guiado pasivo indirecto o por maniquí (derecha)

1.3.2. Registro de trayectorias

Existen dos estrategias en la generación de la trayectoria:

- Registro de puntos de paso. El robot es conducido a través del espacio de trabajo y, mediante una interfaz de comunicación, se indica al sistema la selección del punto de interés y su tipo de interpolación. Durante la ejecución del programa, el controlador del robot interpola la trayectoria desde el punto anterior hasta el punto seleccionado. En la actualidad en la mayoría de los fabricantes hay tres posibilidades: interpolación lineal, interpolación circular e interpolación en juntas. En el caso circular se requiere el registro de dos puntos, un punto intermedio denominado "VIA" y el punto final del arco. La interpolación en juntas realiza un movimiento en el cual todas las juntas del robot empiezan su movimiento en el mismo instante y terminan en el mismo instante, en este caso la trayectoria resultante es desconocida y la velocidad dependerá de la junta más lenta, por tanto, esta debe usarse solo para puntos entre los cuales el espacio intermedio esté libre de obstáculos. Puede decirse que las interpolaciones lineal y circular pertenecen al espacio cartesiano y, dado que la mayoría de los robots industriales son de 6 GdL, se requiere la interpolación de la orientación de la herramienta, la cual puede realizarse de la misma manera que la interpolación en juntas pero, para la variación en los ángulos de orientación, por ejemplo, oscilación balanceo y cabeceo. Esta estrategia se utiliza cuando la tarea requiere pocos puntos bien definidos que se planearon previamente, unidos por geometrías sencillas como líneas rectas y arcos de círculo, que el ser humano no puede generar de manera exacta.
- Registro continuo. Se registran las posiciones con cierta frecuencia. Con frecuencias elevadas se obtienen puntos muy próximos, para aplicaciones en las que el movimiento reproducido deba ser muy similar al realizado por el operador. Esta estrategia es utilizada cuando el programador es el experto en el proceso, en el cual la trayectoria está compuesta de curvas irregulares, es decir

con radio de curvatura variable. Respecto de la estrategia anterior, esta estrategia es más rápida, ya que la selección de los puntos es automática.

1.3.3. Guiado básico y extendido

Hasta ahora se ha expuesto lo relativo a generación de trayectorias, que es lo que se conoce como guiado básico pero, en la mayoría de las aplicaciones, se requiere que durante la generación de un programa se asignen atributos a ciertos puntos de la trayectoria, algunos referentes a la velocidad y aceleración con que se aproxima y aleja a dichos puntos, otros referentes al accionamiento de la herramienta montada en el robot, como puede ser: la apertura o cierre de una pinza, el encendido o apagado de arco para soldadura, entre otros. También, con el avance en los sistemas de programación por guiado, se ha logrado incluir la posibilidad de introducir saltos condicionados a estados de sensores o establecer movimientos finos al aproximarse a la pieza de trabajo. A los sistemas de guiado que cuentan con estas capacidades se les conoce como sistemas de guiado extendido, que consiste en la metodología más ampliamente utilizada en la industria, ya que este sistema es provisto por los fabricantes.

1.3.4. Guiado asistido por sensores

La programación de robots industriales se había realizado convencionalmente por operadores especializados por medio de la terminal de enseñanza, donde el operador debe, mediante botones o joystick, conducir la posición y orientación del robot para cada configuración de la herramienta necesaria durante la tarea. Sin embargo, esta metodología no es intuitiva ya que los movimientos se realizan alrededor de varios sistemas de referencia y el operador debe estar consiente en cuál y cómo está definido, como se muestra en la figura 1.6. El uso de interfaces humano-robot más intuitivas basadas en información obtenida de sensores, es una de las prácticas más recurrentes al lidiar con la manipulación de objetos en tres dimensiones. El inicio de estos sistemas se puede considerase el mouse de 6D, el cual al principio de los ochenta integraba un sensor de fuerza/par en 6 ejes con una esfera hueca [21], y evolucionó hasta una versión optoelectrónica. Actualmente, la marca KUKA y Reis han implementado el mouse de 6D en sus terminales de enseñanza, facilitando el método “lead-through”. Estos dispositivos también pueden montarse sobre la herramienta del robot para obtener un sistema “walk-through”, bajo la estrategia de guiado activo, como es el caso de [22], donde el transductor utilizado es un “micro-switch”, lo que hace la implementación del sistema muy económica en comparación con los casos que expone de marcas comerciales como el sistema MOTOMAN-ET (basado en sensor de fuerza) y la ya mencionada terminal de enseñanza de KUKA con mouse 6D, característica que actualmente se puede agregar montada sobre la herramienta de los robots KUKA, como se menciona en [16], donde se presenta un sistema “walk-through” basado en sensor de fuerza.

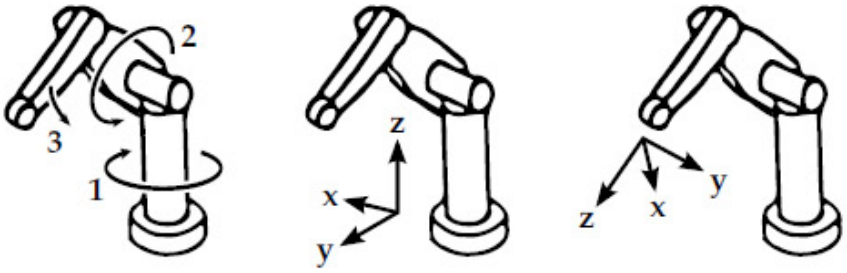


Figura 1.6. Sistemas de referencia típicos: respecto a los ejes del robot (izquierda), respecto a un sistema coordenado fijo también conocido mundial (centro), respecto a un sistema ubicado en el punto central de la herramienta (derecha). Una variante del sistema mundial es el sistema de referencia del usuario el cual es fijo y se define por el usuario.

En [13] se hace una recopilación del uso de sensores en sistemas de programación en línea y se da una breve descripción de las características de cada sistema mencionado. Se puede concluir, con base a la literatura encontrada, que la investigación en sistemas de programación basados en sensores recae en sistemas que usan visión, sensores de fuerza/par [14, 15, 16, 17, 18, 20, 36] y reconocimiento de comandos de voz [38].

1.4. Conclusión del capítulo

En este capítulo se ha realizado una revisión de los avances en el desarrollo de los sistemas de programación, además se presentan las diferentes clasificaciones de dichos sistemas, lo que hace posible identificar el sistema de programación del presente trabajo como un sistema de guiado activo por medio de interacción física, que además se caracteriza por ser un sistema de programación automática y constituye un paso importante en el desarrollo de un sistema de programación por demostración (PpD), uno de los niveles más altos en programación de robots industriales. Los sistemas PpD se caracterizan por permitir que el operador indique al robot las acciones a tomar sin la necesidad de un conocimiento explícito en programación. Esta estrategia de programación ha sido ampliamente abordada; sin embargo, no ha alcanzado la aplicación comercial debido al aumento en costo que representa.

Uno de los tópicos más importantes en este tipo de estrategias es la manera en que la persona interacciona con el robot para guiarlo a través de la posiciones requeridas, ya que esto definirá qué tan intuitivo resulta el sistema. En el siguiente capítulo se abordará lo concerniente a los avances en interacción humano-robot, con especial atención a la interacción física, la cual es puesta en práctica en el presente trabajo de tesis.

Cabe mencionar que, dado el avance en la tecnología robótica industrial y los grandes esfuerzos realizados en investigación, se ha logrado establecer el concepto de sistemas de programación modular [13, 39, 34], los cuales permiten la integración de diferentes desarrollos en programación como son: planeación de trayectorias fuera de línea [40, 41], PpD, realidad aumentada [42], simulación de celdas de manufactura (con los sistemas citados en la tabal 1.1) y lenguajes de alto nivel entre otros, para crear y editar programas en diferentes niveles y a partir de variadas estrategias, para encontrar la aplicación adecuada para resolver el problema de comunicarle al robot la tarea que se desea

realizar. Al haber identificado en la plataforma experimental las capacidades suficientes, es correcto mencionar que el desarrollo del presente trabajo de tesis, constituye una etapa de programación modular, que permita el uso de sistemas robóticos por expertos en la tarea que se desea ejecutar, aun cuando estos no tengan experiencia alguna en programación de robots industriales.

Capítulo 2

2. Control de interacción humano-robot

Como se vio en el capítulo 1, el robot industrial es una herramienta primordial en el desarrollo de productos a gran escala, con calidad uniforme. Hasta ahora la única manera de integrar los robots en los procesos de producción industriales es mediante la programación de los mismos por parte de un experto en tal tarea, lo cual en algunos casos implica la interacción del usuario con el robot a través de diversas interfaces que varían en complejidad de acuerdo a la forma en que el usuario se comunica con el sistema robótico para indicarle la tarea a realizar, mientras el robot se encuentra aislado de la presencia de humanos para asegurar su integridad. En algunos casos, el robot es utilizado cooperativamente para realizar una tarea con la asistencia de un operador, estableciendo una sinergia en la que las capacidades del robot y del humano logran un cometido que, por sí solos, sería imposible. Esta situación crea la necesidad de medidas de seguridad en las áreas de intersección entre robots y humanos, que las medidas convencionales no pueden garantizar.

La interacción humano-robot (IHR) es el campo multidisciplinario de estudio dedicado a entender, diseñar y evaluar los sistemas robóticos para su uso por o con humanos, con contribuciones de robótica, interacción humano-computadora, inteligencia artificial y ciencias sociales, por mencionar algunas. La interacción humano-robot ha sido un tópico en ciencia ficción incluso desde antes de que los robots existieran, como en el caso de la novela de Isaac Asimov, “I, robot”, donde por primera vez se establece de manera discreta la problemática en IHR y, mediante las tres leyes de la robótica expuestas en la novela, se daba la idea de una interacción segura. Mientras más cerca se encuentran el ser humano y el robot y más intrincada se vuelve la relación, mayor el peligro de que el humano resulte dañado. En la realidad actual, hasta apenas el 2012 se reconocen las colaboraciones humano-robot [54] por lo que, de acuerdo a estándares de seguridad, los robots industriales antes debían estar aislados de otras máquinas y cualquier inclusión del ser humano dentro del espacio de trabajo del robot se permitía sólo si el robot estaba apagado. Para el caso de robots de servicio las normas se encuentran en desarrollo.

Aunque hay mucho trabajo que se puede considerar parte de IHR, fue hasta los 90s cuando esta área se estableció como un campo de estudio que empezó a emerger con numerosos eventos, con la principal característica de ser multidisciplinario. Investigadores de robótica, ciencia cognitiva, factores humanos, lenguajes naturales, psicología e interacción humano-computadora (IHC), empezaron a asistir a estos eventos como el International Symposium on Robot & Human Interactive Communication (RoMan) de IEEE y reconocer la importancia de trabajar juntos. De esta manera se ha logrado incentivar la generación de trabajos que integralmente analicen, desde los diferentes puntos de vista de cada ciencia relacionada a IHR, las problemáticas fundamentales. En este capítulo se hace

una revisión de la teoría en IHR, enfocándose en el control de interacción física o táctil con robots del tipo industrial.

El objetivo de esta tesis es crear un sistema de programación con el que, por medio de interacción física, el humano indique al robot la tarea a seguir. Para poder lograr este objetivo se deben analizar los tipos de interacción que existen entre humanos y robots y la información que humanos y robots necesitan uno del otro para lograr la meta.

Hasta hace algunos años la diferencia entre IHR e IHC (Interacción Humano-Computadora) era un poco difusa ya que la interacción con los robots era por plataformas desarrolladas para computadoras o por sistemas cuyo funcionamiento no difería mucho de éstos. En [45] se hace clara la diferencia entre los dos tipos de interacción, notando que en la interacción humano-robot interfiere un sistema dinámico complejo que presenta cierta autonomía y que opera en un entorno cambiante, además de contar con una gran variedad de niveles y tipos de interacción. Se ha de considerar el ambiente donde se utiliza el robot, ya que la interacción podría ocurrir en un ambiente agresivo para los dispositivos además ha de considerarse la movilidad del usuario y tomar en cuenta que los robots tienen sensores que se degradan y pueden fallar alterando la funcionalidad del sistema.

2.1. Categorías principales

IHR en un principio se asociaba con teleoperación de plataformas robóticas industriales. En interacción humano-robot está implícita la comunicación entre humanos y robots, que puede tomar diferentes formas, las cuales dependen en gran medida de la proximidad del ser humano y el robot, por lo que dicha comunicación e interacción puede separase en dos categorías principales:

- Interacción remota – el humano y el robot están en espacios diferentes o incluso en momentos diferentes por ejemplo en [43], se presenta un sistema que permite la teleoperación de un robot vía internet por medio de una interfaz que ofrece la ventaja de conjuntar realidad virtual y realidad aumentada en una PC. Además, permite el uso de sentencias basadas en lenguaje natural para manipular objetos previamente reconocidos por el sistema. Aquí se aplican los conceptos de IHC considerando las dificultades presentes en la manipulación de objetos en 3 dimensiones.
- Interacción próxima – humano y robot se encuentran en el mismo lugar. La persona estará muy cerca o en contacto con el robot, aumentando las probabilidades de un accidente por lo que la seguridad se vuelve un problema muy importante en este caso. En el caso convencional de robots industriales la solución es implementar barreras, marcas en el piso o sensores [53] para indicar al trabajador que está invadiendo el espacio de trabajo del robot, por mencionar algunos. Sin embargo estas estrategias no aplican en el caso de trabajo cooperativo humano-robot [44], donde el contacto directo con el robot es necesario y los desarrollos de ingeniería hacen frente a las problemáticas de seguridad mediante el diseño de actuadores, sistemas de control o ambos, de manera que la colisión con el usuario se evite o para que el sistema se comporte de la manera más dócil posible.

2.2. Factores que influyen la confianza humano-robot

Para que un objetivo se cumpla por medio de colaboración humano-robot, el humano debe confiar en que el robot ha de proteger los intereses y bienestar de los integrantes del equipo. La confianza es importante en el sentido de que ésta afecta directamente en qué medida el humano acepta la información producida por el robot. Sin embargo, los niveles inapropiados de confianza pueden tener consecuencias negativas, como uso inapropiado en el caso de exceso de confianza o abandono en el caso de niveles bajos de confianza. Ambos, desconfianza y exceso de confianza, pueden minar el desempeño del sistema. La confianza está relacionada con la negligencia, es decir, el usuario cede responsabilidad al desempeño del robot, en el momento que el humano dirige su atención hacia otra tarea o mientras la complejidad de su tarea se incrementa. Cuando una persona tiene mucha confianza en el robot no siente la necesidad de supervisarlo, por lo que lo ignora por lapsos de tiempo largos. Mucha desatención puede hacer difícil retomar conciencia del estado del robot y, por otro lado, poca desatención significa que el operador no está concentrado en su tarea lo que causa bajo rendimiento del sistema.

Tabla 2.1. Factores que influyen en la confianza humano-robot. Para una descripción detallada de algunos de estos factores ver [45, 46, 47, 48, 49,50, 51,]

Relacionados al humano
Basados en la habilidad
Capacidad de atención
Experiencia
Competitividad
Carga de trabajo
Conciencia de la situación
Características
Demográficas
Personalidad
Actitud ante los robots
Comodidad con robots
Autoestima
Propensión a confiar

Relacionados al robot
Basados en desempeño
Dependencia
Confiabilidad del robot
Predictibilidad
Nivel de autonomía
Porcentaje de fallas
Falsas alarmas
Comportamiento
Transparencia
Basadas a atributos
Proximidad
Personalidad del robot
Adaptabilidad
Antropomorfismo
Tipo de robot

Relacionados al entorno
Colaboración en equipo
Miembros del grupo
Cultura
Comunicación
Modelos mentales comunes
Basados en la tarea
Tipo de tarea
Complejidad de la tarea
Requerimiento de multitareas
Entorno físico

La confianza está influenciada dinámicamente por ciertos factores que tienen que ver con el sistema robótico, el entorno de operación y la naturaleza y características del humano que opera dicho sistema. En la tabla 2.1 se enumeran algunos factores que se

analizaron en [49]. A continuación se mencionan algunos de los factores revisados en la literatura.

2.2.1. Relacionados al humano

La confianza es un aspecto crucial para mantener relaciones efectivas con los robots. Los estudios en [49] indican que lo que más influencia la confianza humano-robot es la manipulación de los diferentes factores del desempeño del robot, por lo que en el futuro se ha de poner especial atención a estos factores durante el diseño de sistemas de interacción humano-robot. A pesar de que los resultados indican que, de acuerdo a la literatura actual, la influencia en la confianza humano-robot es poca en lo que respecta a los factores relacionados al humano. Ya desde lo observado en [50], donde se justificaba el uso de robots mediante la descripción de sus cualidades y aplicaciones y se exponían los principales problemas a tratar para lograr una implementación adecuada, se consideraban desde los factores de diseño, ambientales y de seguridad y se le daba especial importancia a los factores psicosociales. Es decir, es importante la justificación del uso de robots ya que esto implica reducción de trabajo para los obreros, de tal manera que desde muchos puntos de vista el robot es percibido como amenaza. Dependiendo en la manera en que el robot es introducido en los procesos de producción, el robot puede significar beneficios económicos o problemas de seguridad y de sindicatos, entre otros relacionados con el ser humano.

Conciencia de la situación

Conocimiento de la situación. Está relacionado al conocimiento del entorno considerando que el entorno (robot) es móvil, es decir, conocimiento de lo que está ocurriendo. El conocer el estado del sistema robótico y las acciones correspondientes en cada instante ayuda a entender posibles discrepancias entre el comportamiento deseado y el que ocurre en realidad. Se puede decir que en la conciencia se logran tres niveles:

1. Percepción básica de indicaciones
2. Comprensión o integración de información variada para determinar su relevancia para las metas que el usuario quiere alcanzar.
3. Previsión de eventos futuros en base a la percepción y comprensión de la situación actual.

Capacidad de atención

En [48] se habla de la atención y conciencia humana. La atención humana es un factor importante en la prevención de accidentes causados por errores. A pesar de los programas de entrenamiento, señales y experiencia, la gente tiende a cometer errores que pueden ser causados por distracciones, espacios de trabajo mal diseñados, falta de conocimiento, percepción y conciencia de lo que sucede en el entorno. Por ejemplo, en ocasiones se subestiman las capacidades de velocidad, fuerza y alcance del robot, no se perciben señales de alarma, se ignora el estado actual del robot, etc.

Para entender los mecanismos de la atención y utilizarlos para disminuir los errores del ser humano en ambientes industriales, es necesario el estudio de cognición. La gente tiene variadas habilidades cognitivas para percibir lo que ocurre a su alrededor y procesar la información recibida. Usualmente se recibe la información del entorno visualmente o

auditivamente (percepción) para que, basándose en información almacenada previamente, sea posible interpretar la información recibida y tomar decisiones (sistema cognitivo), proceso que se ilustra en la figura 2.1. Existen varios niveles de percepción que dependen de los estímulos y de la situación en que se encuentra la persona. La forma más básica de percepción es la detección; las más complicadas son identificación y reconocimiento. El acto de percibir un estímulo involucra el uso de la experiencia. En la siguiente figura se muestra el modelo cognitivo humano.

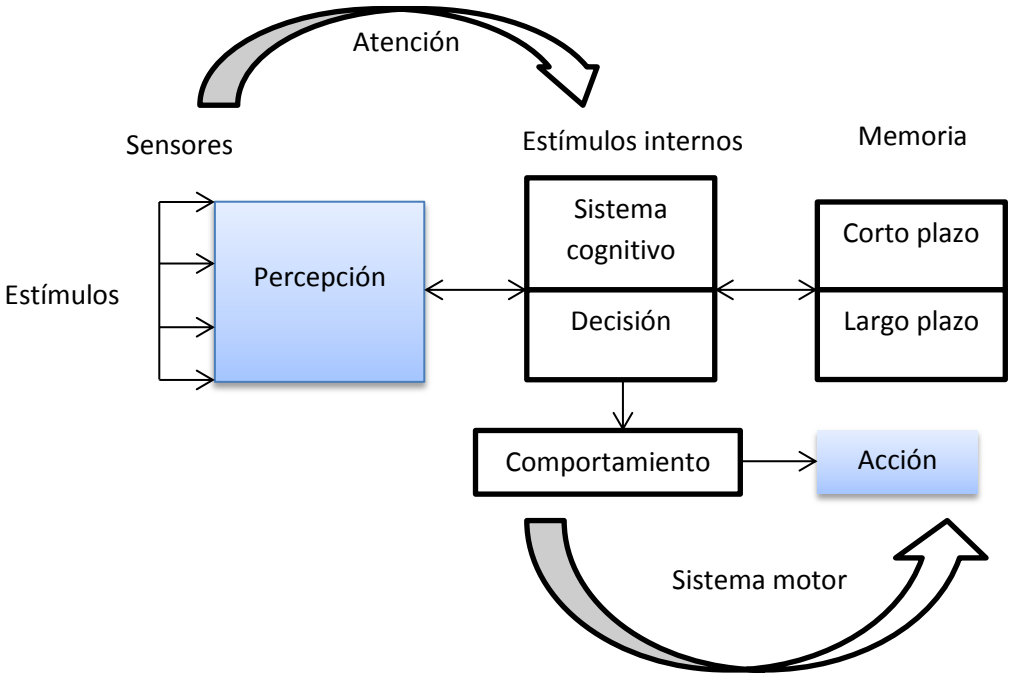


Figura 2.1. Modelo cognitivo humano.

Comúnmente la conciencia involucra la comparación de información percibida por los sentidos humanos con situaciones conocidas, lo que permite entender la situación actual y predecir lo que podría pasar; desafortunadamente se tienen deficiencias cognitivas que son muy sensibles a factores externos. De hecho, la persona es consciente de que el estímulo sucederá dentro de un rango específico de tiempo y en cierta área. Sin embargo, la mayoría del tiempo muy variados estímulos aparecerán simultáneamente, requiriendo diferentes respuestas; mientras más natural sea la relación entre el estímulo y la respuesta, menor será el tiempo de reacción y más precisa será la acción tomada. La expectativa tiene un impacto importante en el resultado, puede relacionarse al estado de funcionamiento del robot, rango de movimiento o a las señales de alarma e información desplegada.

También hay limitaciones en las personas para dividir la atención hacia diferentes estímulos presentes en el ambiente, simplemente entre modalidades como pueden ser visión y auditivas. El ser humano tiende a percibir solo ciertos aspectos del total de características del entorno. Aunque se pretende procesar la totalidad de los estímulos, la percepción humana es más sensible a ciertas características conocidas como factores destacados, sonidos altos, cuerpos grandes, aspectos que tienen la ventaja de llamar la atención de las personas.

Roles de interacción

En [45] se mencionan tres niveles de interacción, de acuerdo al rol que en cada nivel el ser humano desempeña. Más tarde, en [46 y 47] a esta clasificación se agregan 2 niveles para quedar de la siguiente manera:

- Supervisión – En este caso se puede asumir que el usuario interactúa remotamente con el robot ya que tal caso se da cuando el humano necesita supervisar el comportamiento del robot, pero no necesita controlarlo directamente. En su desempeño como supervisor, el humano debe tener acceso a la siguiente información:
  - Una descripción general de la situación.
  - El plan de trabajo.
  - Capacidades de la plataforma robótica y sus posibles desviaciones de lo normal.
  - Interacciones del robot con otros robots o máquinas

  Su función es planear qué tarea hacer y cómo hacerla, programar el robot para que realice la tarea, monitorear su funcionamiento para detectar fallas e intervenir en caso de falla o para designar una nueva tarea.
- Operador – el operador es requerido para modificar software interno o modelos, en caso de que éstos estén causando un comportamiento inaceptable, es decir, deberá verificar que las acciones necesarias para lograr la tarea se estén llevando adecuadamente por lo que ocasionalmente tendrá que teleoperar al robot.
- Cooperativo – para este caso, humano y robot contribuyen con sus habilidades. El control recae en el usuario y la problemática principal es cómo el robot retroalimenta al usuario respecto de su comprensión de la situación y las acciones iniciadas. Otro reto interesante radica en la restricción de distancia entre el robot y el usuario. La información que puede ser necesaria para el usuario es:
  - Qué otras interacciones se encuentran planeadas.
  - Estado actual del robot.
  - Marco de referencia de movimiento del robot.
  - A qué acciones el robot tiene acceso (capacidades: volumen de trabajo, capacidad de carga, velocidad y etc.)
- Mecánica – el mecánico se ha de encargar de modificar el hardware en caso de fallas físicas, por lo que requiere conocimientos similares al operador para poder verificar el desempeño de sus modificaciones. Se puede considerar una interacción al igual remota o directamente en el espacio de trabajo, cuando la cantidad de conocimiento de la situación requerida por el usuario sea reducida. El usuario toma el control del robot, por lo cual deberá tener conocimientos de la arquitectura del robot y habilidades en programación del mismo. El usuario en nivel de mecánico debe conocer:
  - El modelo del robot.
  - El entorno de trabajo.
  - El plan de la tarea.
  - El estatus de los sensores.

- Transeúnte – no controla el robot pero debe tener cierto conocimiento de lo que está haciendo el robot para poder compartir la misma área de trabajo. Sin embargo, con las adecuadas capacidades sensoriales, un robot al detectar su presencia en una zona de peligro podrá detener la acción de movimiento en ese instante.

Hay que notar que los 5 niveles de interacción se pueden presentar en un mismo escenario y que un solo usuario puede adoptar diferentes niveles de acuerdo a las necesidades de interacción. En [47] se presentan otras clasificaciones que ayudarán a identificar modelos de interacción y que influyen tanto en el diseño como en el desarrollo de la interacción.

2.2.2. Relacionados al robot

Interfaces de Información

Establece cuatro categorías que indican de qué manera se muestra la información extraída de sensores para que el usuario tome decisiones: lista de información disponible, información provista, fusiones y pre-procesamientos. La lista de información disponible es una base que permite entender la información provista, las fusiones y su pre-proceso de manera que, en la etapa de diseño, se genere una interfaz que ofrezca la información necesaria al usuario. La información provista es un subconjunto de la lista de información disponible, ya que es posible que no toda la información sea necesaria para que el usuario tome una decisión. El tipo de fusiones de información es una lista de funciones que combinan dos o más conjuntos de datos relacionados a uno o más sensores para generar la información que ha de mostrarse al usuario. Finalmente, la lista de pre-procesamientos indica el procesamiento al que ha sido expuesto cada dato para generar la información adecuada para cada interface. La cantidad de información mostrada en la interfaz y el procesamiento que reciben los datos obtenidos de los sensores permite saber qué tanto apoyo recibe el usuario del sistema para su toma de decisiones. Hay dos dimensiones que determinan la forma en que se intercambia información entre humano y robot: el medio y formato de la comunicación [52]. Como se verá más adelante a detalle, hay tres medios principales de comunicación: auditivos, por tacto y por visión. Estos medios se manifiestan en IHR de la siguiente manera:

- Pantallas, comúnmente muestran interfaces gráficas orientadas al usuario o interfaces de realidad aumentada.
- Gestos, incluyen movimientos de las extremidades o del rostro así como señales basadas en intento de movimiento.
- Expresiones en lenguaje natural, que incluyen tanto expresiones escritas como audibles, con énfasis en diálogo de iniciativa compartida.
- Sonidos no relacionados a vocablos, comúnmente usados como señales de alarma.
- Interacción física y háptica, con frecuencia usada remotamente en realidad aumentada o en teleoperación para evocar la sensación de presencia; también utilizada en proximidad para sistemas cooperativos de manipulación de objetos pesados.

Recientemente, se ha prestado especial atención en el diseño de interfaces multimodales, motivados por la búsqueda de interacciones más naturales y fáciles de aprender. El formato varía ampliamente dentro de cada dominio. Las expresiones en lenguaje natural podrían sustentar el uso de un lenguaje natural completo o lo podrían restringir al uso de un subconjunto de expresiones relacionadas a acciones [55]. La información háptica puede presentarse como vibraciones para dar señales de alarma o como reacciones de fuerza para dar la sensación de contacto en sistemas de telepresencia. En interfaces visuales pueden presentarse desde las clásicas ventanas con información textual, imágenes, gráficas, hasta realidad virtual. También existe la posibilidad de interactuar con robots de manera social por medio de expresiones faciales [55].

Morfología del robot

Se puede decir que es importante en IHR ya que influye socialmente creando expectación. Hay muchas posibilidades morfológicas en los robots, pueden ser antropomórficos, zoomórficos y funcionales (que tiene una forma que favorece cierta función).

Nivel de autonomía

Se define como un porcentaje obtenido del tiempo en operación autónoma contra el tiempo invertido para poner el robot en operación, de tal manera que en un sistema robótico de teleoperación se tiene una autonomía del 0%, mientras que un sistema provisto con inteligencia artificial puede llegar a porcentajes muy cercanos al 100%.

Composición de grupos de robots

Se refiere a grupos de robots designados a una misma tarea. La clasificación se realiza con base en el tipo de robots que componen el grupo. Los grupos pueden ser homogéneos es decir, tienen un solo tipo de robot; como ventaja las interfaces son iguales haciendo más simple la presentación de información. Puede ocurrir la necesidad de diferentes tipos de robots, en tal caso el grupo es heterogéneo donde es más difícil presentar la información de los sensores de robots de diferente tipo en una forma coherente de manera que beneficie la toma de decisiones para cada uno.

2.2.3. Relacionados al entorno

Tipo de tarea

Esta clasificación se debe especificar durante el diseño del sistema ya que, para que se logre la tarea, el sistema tendrá características particulares. Por ejemplo, el tipo de tarea permite conocer el entorno donde se desarrollará la misma.

Importancia de la tarea

Es una medida muy subjetiva de clasificación, pues se refiere a las consecuencias de que la tarea no se logre; hay tres categorías: alta, media y baja. Por ejemplo, si falla un robot de apoyo en cirugía las consecuencias para el paciente pueden ser letales, por lo que su importancia es ALTA. En cambio, en un entorno industrial, si el robot daña una pieza las consecuencias son enmendables por lo que su importancia es MEDIA. Finalmente, en el

caso de un robot de soccer, si falla simplemente perderá el partido por lo que su importancia es BAJA.

La proporción humanos a robots

Esta característica afecta directamente la IHR; se expresa como un número fraccional no simplificado donde el numerador es la cantidad de personas y denominador es el número de robots. Si estos números son variables, se expresan como un rango.

Nivel de interacciones compartidas entre grupos

La proporción de humanos a robots no da la suficiente información necesaria para establecer el control adecuado para cada robot, por lo que surgen numerosas preguntas: ¿quiénes comandan el robot, que prioridad tienen cada comando y en el caso de varios robots cuál atiende qué comando?

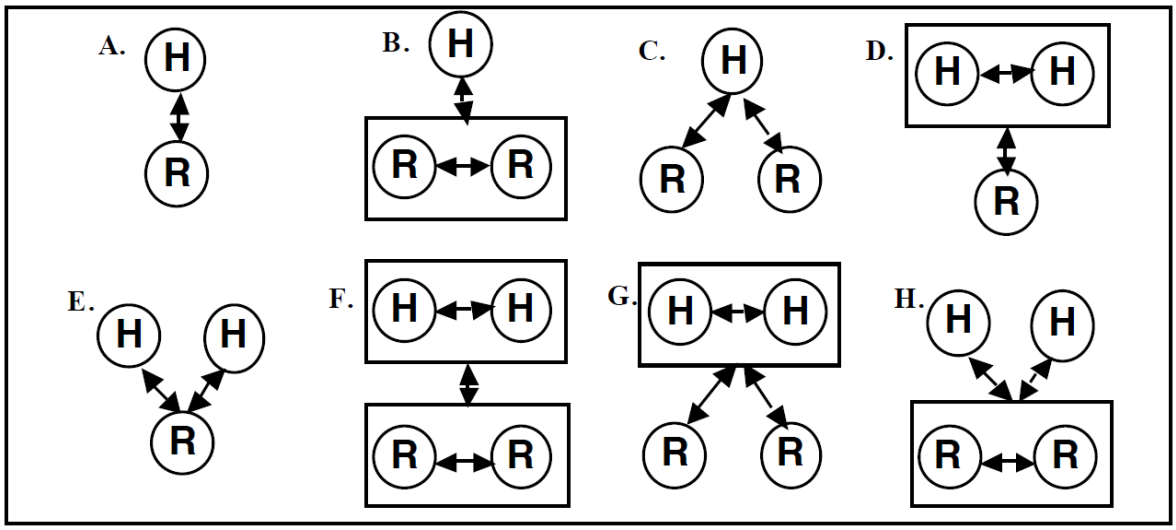


Figura 2.2. Posibles combinaciones con simples o múltiples humanos y robots interactuando como individuos o grupos.

En la figura 2.2 en cada caso una persona es representada por un círculo con una letra “H”, mientras que un robot se representa con un círculo con una letra “R” y las flechas con puntas en ambos extremos indican interacción entre humanos y robots. En el caso más simple de la figura 2.2(A) se muestra una persona dando comandos a un robot mientras el robot regresa información de sensores por medio de cierta interface. En la figura 2.2(B) un humano controla un grupo de dos robots (el mismo concepto aplica para más de dos robots para cada caso en la figura 2.2) y éstos se coordinan para determinar cuál de ellos realiza cada parte de la tarea. En la figura 2.2(C) se muestra un humano controlando dos robots que pueden estar trabajando en áreas separadas o en tareas que no implican interacción robot-robot. Las figuras 2.2(D) y 2.2(E) representan los casos opuestos de 2.2(A) y 2.2(B) respectivamente; en el caso 2.2(D) dos humanos se coordinan para generar los comandos para un robot y en el caso 2.2(E) el robot debe determinar que prioridad tiene cada comando de dos humanos de acuerdo a cierta tarea. En figura 2.2(F) se muestra una situación en la que dos humanos se ponen de acuerdo para generar los comandos que se

proveerán a un grupo de robots para que éstos decidan cual robot realizará qué comando. En figura 2.2(G) un grupo de dos humanos generan comandos para dos robots independientes. Finalmente, en la figura 2.2(H) dos humanos independientemente generan comandos para un grupo de robots que resuelven los conflictos y la prioridad en los comandos, además de decidir cuál robot ha de realizar cuál comando.

En [49] se utilizó un método meta-analítico para evaluar los datos recabados para los tres grupos de factores que influyen en la confianza humano–robot y determinar el patrón en los resultados de la investigación actual en confianza humano-robot. Los estudios meta-analíticos indicaron que había un efecto de moderado a grande entre la confianza humano-robot y la totalidad de los factores. Los estudios que se realizaron entre la confianza y los factores relacionados a los humanos, robot y entorno, individualmente indicaron que había poco efecto relacionado a los factores humanos y de poco a moderado al entorno, mientras que los factores relacionados con el robot mostraron un efecto de moderado a grande. En las subcategorías en base a desempeño y atributos, dentro de los factores relacionados al robot se encontró que los factores relacionados al desempeño estaban fuertemente asociados con la confianza humano-robot, en contraste, los relacionados a atributos tenían poco influencia.

2.3. Medios de la percepción

Cuando se diseñan espacios de trabajo o sistemas de alarma, con el objetivo de favorecer la atracción de la atención de las personas hacia cierto conjunto de estímulos, se deben considerar las capacidades naturales y limitaciones del ser humano. Dentro de las capacidades sensoriales humanas se pueden resaltar tres medios principales:

2.3.1. Visión

El sistema sensorial de la vista está muy desarrollado en los humanos. La estructura de los estímulos visuales es una materia bien comprendida, y la tecnología en equipo de presentación de información visual produce estímulos muy expresivos. Sin embargo, un aspecto que limita su uso es la necesidad de que la persona dirija su atención hacia la fuente del estímulo visual, es decir, que si la persona está mirando hacia otra dirección o si está ocupada con otra tarea que exige su atención visual, puede privarse del estímulo. Usualmente, para un humano es difícil dirigir la atención hacia un área específica, ya que debido a la gran cantidad de fuentes paralelas de información, la visión como canal sensorial puede saturase. En el diseño de interfaces visuales se recomienda que la fuente se encuentre dentro de 30° alrededor de la línea de visión normal y que al menos genere un ángulo visual de 1°. Los estímulos que se presentan en la periferia del campo visual generan respuestas que son entre 15 y 30 ms más lentas que las que se presentan en el área central.

2.3.2. Auditiva

Como la vista, el sentido del oído está muy desarrollado. El humano es sensible a características temporales, espaciales y de forma de onda de señales auditivas. Los estímulos auditivos son omnidireccionales, en el sentido de que quien escucha no requiere orientarse en cierta dirección para percatarse del sonido. Una señal de sonido se compone de ondas de frecuencia y amplitud variable, por lo que son utilizadas en señales de alarma y

como complemento en despliegue de información visual. Sin embargo, las señales de alta intensidad generan un impulso reflejo que puede ser útil sólo si tal respuesta es requerida.

2.3.3. Tacto

El sentido del tacto podría ser el medio más complejo. Parcialmente por la variedad de tipos de sensaciones que es capaz de percibir, como pueden ser temperaturas, vibraciones, presiones y dolor. El sentido del tacto es el único capaz de llevar a cabo el sensado y actuación. La sensibilidad varía dependiendo de múltiples factores como el área, la ubicación y la fuerza del contacto, la temperatura de la piel, la persistencia del estímulo y muchos más intrínsecos de la persona, por mencionar algunos.

Es importante, cuando se diseñan interfaces de advertencia con múltiples estímulos tales como auditivos, visuales y táctiles, tomar en cuenta que estímulos que requieren respuestas individuales han de separarse temporalmente, más de 0.25 seg, para evitar intercalación. Además, es recomendable que las señales sean mínimas y que las señales importantes tengan un nivel de intensidad mayor.

En [48] se presenta un sistema para llamar la atención y la conciencia del usuario de un robot industrial por medio de estímulos táctiles y visuales, en caso de posibles riesgos cuando el usuario se encuentra dentro de la celda de manufactura y el robot se encuentra en operación. Para lograrlo se implementó un dispositivo vibro-táctil que se usa en la muñeca. Vibraciones con diferentes intensidades y luces proveen al usuario de información complementaria relativo a peligros. La respuesta producida son acciones rápidas que ayudan a evitar consecuencias no deseadas. Más adelante se verá a detalle lo referente a control de interacción por medio de tacto.

2.4. Interacción táctil

La presencia de los robots es cada vez más común en áreas de servicio. Es decir, la tecnología robótica está dejando los entornos controlados de la industria y laboratorios de investigación. Al ser el tacto al medio más complicado y crítico, en el sentido que es más probable que un ser humano sufra daños debido a la interacción táctil con un robot que por otro medio. La clave para una introducción exitosa de los robots en sociedad son la seguridad y confiabilidad. Por lo que la motivación en desarrollos para la detección del contacto entre humanos y robots se ha vuelto crucial.

2.4.1. De acuerdo al sensor

En [56] se analiza el estado del arte en tecnología para la detección de eventos de contacto, identificando tres tópicos principales: películas rígidas, películas suaves y alternativas a las dos anteriores, dentro de las que cae el uso de sensores de par en las articulaciones o sensores de 3 ejes fuerza/par en la muñeca del robot. Los sensores pueden consistir en sensores de fuerza/par, resistores sensores de fuerza, sensores de contacto, arreglos de sensores capacitivos, arreglos de sensores resistivos, por mencionar algunos. Las capacidades de medición están relacionadas al tipo de sensor. Entre lo que puede detectar un sensor se encuentra: lugar, área y duración de contacto, magnitud, orientación y momento de la fuerza, relaciones estáticas y dinámicas entre lecturas de sensores, temperatura, vibración, cinestesia y campo eléctrico.

2.4.2. De acuerdo al efecto de la interacción

Se detectaron cinco tópicos principales de acuerdo a cómo es usada la información obtenida del sensor: *respuesta ante una perturbación*, el robot responde ante un contacto que interfiere con el desarrollo de la operación, por ejemplo, reducción de torque en las juntas para no aplastar un objeto atrapado; *clasificación de estados*, se usa la información obtenida para diagnosticar estados internos (robot) o externos (un paciente con deficiencias motrices), contacto humano para interferir en el comportamiento del robot (*contacto de interferencia*), contacto humano que contribuye al comportamiento del robot (*contacto de contribución*) y contacto humano que genera un comportamiento o adaptación (*contacto de desarrollo*). Dentro del contacto de interferencia, el contacto humano se considera una posibilidad dentro de la operación del robot y no solo puede darse el caso de detección de colisión, sino que existe la posibilidad de una reacción para recuperar el objetivo. Dentro de contacto de contribución se encuentran dos posibilidades: una es utilizar el contacto humano para controlar directamente el robot, a través de guiado del robot con comportamiento dócil, la otra consiste en inferir la intención del humano y utilizar estas intenciones para generar un comportamiento. En contacto de desarrollo se usan algoritmos de aprendizaje para generar un comportamiento.

2.4.3. Seguridad en la interacción

La principal preocupación en seguridad en robótica es la posibilidad de colisión entre el robot y el usuario, por lo que es necesario un índice que evalúe el daño. En otros dominios como el automotriz ya existen tales índices sin embargo, en robótica no se han establecido. Para alcanzar los niveles de seguridad requeridos es necesario atender tanto el diseño mecánico, electrónico y de software del robot, sin menoscabo de las capacidades que justifican el uso de robots.

En el 2008 [57], la ANSI/RIA R15.06-1999 (American National Standard for Industrial Robots and Robot Systems – Safety Requirements) era la principal estandarización en seguridad para robots industriales, donde se establecen los requerimientos para la seguridad personal en lugares donde hay robots manipuladores. Sin embargo, esta estandarización no abarcaba el caso donde humano y robot compartían el área de trabajo mientras éste está en operación.

La OSHA de USA [6], dedica la sección 4 del capítulo 4 de su manual a la seguridad con robots y sistemas robóticos. En forma similar, otros países incluyeron este tópico. Como ejemplo están la CAN/CSA Z434-03 de Canadá e internacionales como la ISO 10218. Cada una de las anteriores se centra en temas de seguridad para el caso de humanos compartiendo el espacio de trabajo de los robots. Actualmente, los robots se clasifican en dos grandes categorías: robots industriales y robots de servicio. Aunque las estandarizaciones en robots de servicio son relativamente nuevas, generan mucho interés y están directamente relacionadas con la interacción humano-robot [54]. Las estandarizaciones en robots industriales se han estado llevando principalmente por la ISO desde los 80s. Probablemente la ISO 10218 es la estandarización más usada en robótica, que hasta cierta revisión especificaba solo los requerimientos de seguridad para robots industriales. Previamente, la norma 10218 sólo constaba de una parte, y ahora está dividida en dos partes. La parte 1 revisa lo referente a seguridad con robots industriales, mientras la

parte 2 se enfoca en los aspectos relacionados a sistemas robóticos en general y como deben integrarse donde se han de utilizar.

Debido a lo variado en las maneras de establecer los criterios de estandarización alrededor del mundo, es necesaria la creación de un documento de igual manera en diferentes países, estableciendo criterios generalizados, por ejemplo de manera similar que en la industria automotriz, donde se definieron criterios y mediciones cuantitativas para evaluar lesiones debido a un impacto [57].

2.4.4. Control de la interacción física humano-robot

La mayoría de los robots industriales son controlados con base en posición. Sin embargo, su uso en tareas que requieran interacción con el entorno resulta inadecuado, dado que el control de posición se obtiene exitosamente cuando la tarea se logra de manera precisa, situación que en presencia de restricciones de movimiento puede causar fuerzas de contacto indeseables. De esta manera es necesario agregar la posibilidad del uso de sensores y control de fuerza, que reduzca el riesgo de daño en caso de tareas cooperativas entre humanos y robots.

Control de impedancia

Las estrategias en control de interacción pueden agruparse en dos categorías: control directo e indirecto de fuerza. La diferencia radica en que el control directo limita las fuerzas de contacto a cierta magnitud mediante un lazo de retroalimentación de fuerza, mientras el control indirecto lo obtiene mediante un lazo de control de movimiento. Debido a que el control de interacción del robot con su entorno tiene muchas aplicaciones en la industria, hay diferentes enfoques de control fuerza, entre los que destaca el control de impedancia propuesto en 1985 por Hogan [58], por ser la estrategia de control indirecto con más auge, la cual está basada en el control de la relación entre la fuerza de interacción y los errores de posición, resultantes de esta fuerza. La idea original acerca del control de impedancia propuesta por Hogan, tiene hoy en día diversas interpretaciones, tal es el caso del control de impedancia basado en posición y fuerza [59], aplicable a robots de arquitectura cerrada.

¿Por qué control de impedancia?

En la búsqueda de control de interacción, es imposible generar un control adecuado para posición y fuerza al mismo tiempo. Esto es debido a que durante la interacción fuerza y posición están relacionados. Dado que el espacio en el que un manipulador realiza una tarea de interacción está compuesto en su mayoría de elementos inerciales, es decir sistemas físicos cuya respuesta es un movimiento cuando se les imprime una fuerza. Cuando un manipulador esta acoplado a este tipo de sistemas, conocidos propiamente como admitancias, para lograr compatibilidad física el manipulador debe adoptar un comportamiento de impedancia. Por lo que una estrategia de control a considerar es agregar una perturbación al control de movimiento del manipulador, que tiene la forma de una impedancia, obtenida como respuesta ante la desviación en movimiento debida a la interacción del manipulador con su entorno. Es decir que no se controla ni posición ni fuerza, sino que se controla el canal de intercambio de energía, en este caso la impedancia.

Control de impedancia basado en par

En 1985 Hogan [58], estableció algunas consideraciones que permiten aplicar el concepto de impedancia para generar un nuevo enfoque de control de manipuladores. Hogan describe el control de impedancia como una extensión a la estrategia de control de movimiento convencional, donde se define una trayectoria y una impedancia; dicha impedancia es una relación entre la fuerza de interacción y la desviación que se tiene en la trayectoria. Esta relación puede ser definida en función de características de inercia, amortiguamiento y rigidez de tal manera que, para asegurar que sea dinámicamente viable, la selección de la impedancia que ha de imponerse en el control deberá basarse en el comportamiento dinámico del manipulador.

Control de impedancia basado en posición

En 1987, Lawrence [59] derivó un esquema de control de impedancia basado en posición, el cual posteriormente es analizado en 1988, en una búsqueda de establecer bases para evitar inestabilidad mediante una apropiada selección de los parámetros que definen la impedancia [59]. Dado que este es el enfoque que se pretende implementar en la plataforma asignada a este trabajo de tesis, es conveniente hacer una descripción más detallada de su desarrollo.

En el enfoque basado en posición, las fuerzas y pares son sensados explícitamente vía un sensor de fuerza/par ubicado en la muñeca del manipulador, y los comandos de posición se emplean en el lazo interno del controlador. En particular, se crea un vector de ajuste de posición x_a mediante la relación que hay entre fuerzas de interacción medidas y la posición del manipulador, la cual es:

$$F_e = K_d x_a + B_d \dot{x}_a + M_d \ddot{x}_a \tag{2.1}$$

Donde x_a en el dominio de la frecuencia se expresa de la siguiente manera.

$$x_a(s) = (K_d + B_d \mathrm{s} + M_d s^2)^{-1} F_e(s) \tag{2.2}$$

Nótese que (2.1) es un sistema de ecuaciones diferenciales de segundo orden. La simplificación de que K_d, B_d y M_d que representan a las matrices de rigidez, amortiguamiento e inercia respectivamente, sean diagonales, hace que las ecuaciones sean independientes una de otra. Esto reduce la complejidad de los cálculos, a resolver seis ecuaciones diferenciales de segundo orden independientemente, por lo que cada componente de x_a depende solo de su correspondiente componente de la fuerza F_e. El ajuste x_a es restado a la trayectoria de posición deseada x_d para generar la posición actual x_c:

$$x_c = x_d - x_a \tag{2.3}$$

De (2.2), se puede observar que, cuando no existe contacto con el entorno, $F_e = 0$, entonces $x_c = x_d$, obteniendo control de posición puro.

A continuación se presenta un esquema que describe cómo se establece el control de impedancia a través del conocimiento de las fuerzas de interacción:

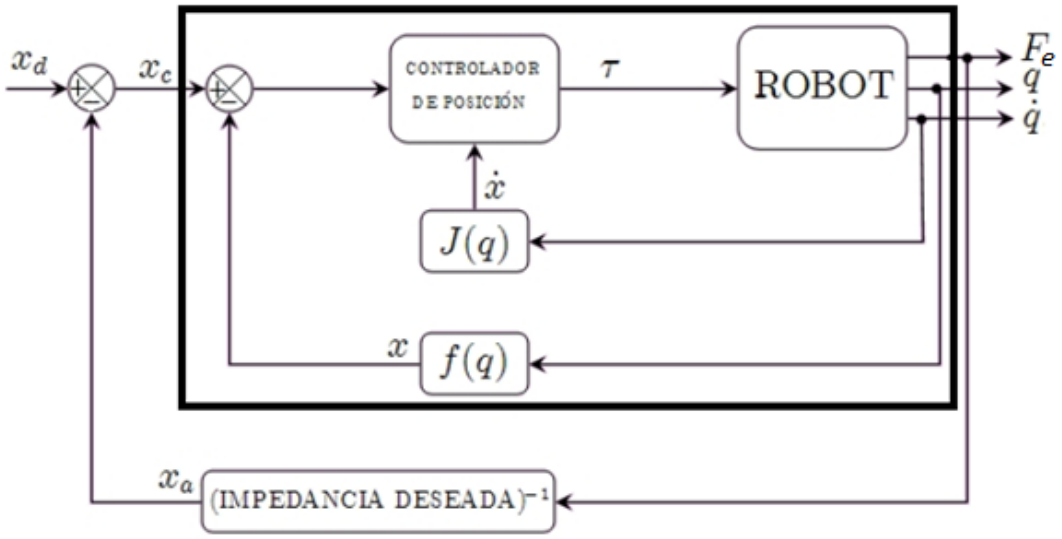


Figura 2.3. Diagrama de bloques del control de impedancia basado en posición.

Esta estrategia permite utilizar el control de posición con que la mayoría de los robots manipuladores industriales cuentan y que muestran una robustez adecuada. Sin embargo, su aplicación en robots de arquitectura cerrada tiene la desventaja de no poder implementar impedancias dóciles, es decir, rigidez y amortiguamiento pequeños [60]. Por lo que las interacciones con alta docilidad podrán resultar peligrosas.

2.5. Conclusión del capítulo

La información presentada en este capítulo, permite entender las implicaciones que existen en el desarrollo de sistemas robóticos, al describir los posibles escenarios y agruparlos de acuerdo a variados enfoques. En la robótica no se puede prescindir del planteamiento de interacción; ya sea con el entorno, otros robots o con humanos. Por tal motivo el uso o implementación de nuevas estrategias para la incorporación de estos equipos a la resolución de cualquier tarea, requiere el análisis de una gran cantidad de factores, que de no guiarse en alguna estructura, se corre el riesgo de pasar por alto consideraciones importantes. La principal problemática, se encuentra en la comunicación, dada las reducidas capacidades de los equipos robóticos para comunicarse y recibir información de los usuarios, en comparación con la comunicación interpersonal. Lo que obliga al usuario a adquirir cierto nivel de conciencia de lo que ocurre o va a ocurrir en determinada situación. Por su parte el desarrollador debe tener pleno conocimiento de las capacidades del equipo y de acuerdo con el uso que se dará del mismo, definir la metodología a seguir para generar una aplicación funcional.

Capítulo 3

3. Desarrollo e implementación del sistema de guiado activo

Como se vio en el capítulo anterior, en el desarrollo de sistemas que requieren interacción humano-robot es necesario el análisis de las necesidades y posibilidades del usuario, tecnología a la mano y la tarea a desempeñar, todo desde el punto de vista de muy variadas disciplinas.

En el presente trabajo de tesis se han definido algunas restricciones como son el modelo del robot y del sensor. Asimismo, se considera el uso de algoritmos de control previamente desarrollados en el laboratorio de robótica para un previo trabajo de tesis en el cual no se consideraba una aplicación tangible, pero se había identificado como una opción para su aplicación en sistemas que involucran IHR, como es el caso del presente trabajo de tesis donde se plantea utilizar el control de impedancia cinemático basado en posición para implementar un sistema de guiado con el cual el usuario sea capaz de generar trayectorias básicas.

En este capítulo se presenta la metodología empleada para obtener los resultados requeridos para una interacción segura en programación de un robot, considerando guiado por medio de contacto físico.

La metodología establecida para el desarrollo del sistema fue el siguiente: Las dos primeras etapas se enfocan en conocer el sistema robótico así como el estado de los algoritmos desarrollados en el laboratorio con el objetivo de modular la fuerza de contacto. La etapa siguiente se centra en la mejora de los algoritmos ya mencionados. En este punto era necesario implementar la comunicación con el robot por medio de botones (etapa 4).En la etapa 5 a partir de las señales asociadas a los botones se desarrollaron algoritmos que dan funcionalidad al sistema de guiado. La etapa final expone lo referente a diseño del portasensor, como se muestra a continuación:

1. Análisis del sistema robótico.
2. Interpretación de los conceptos teóricos
3. Análisis del desempeño del programa previo de control de impedancia.
4. Implementaciones para la interacción.
5. Desarrollo de la interfaz de comunicación (terminal de enseñanza).
6. Desarrollo de programas para la administración de la interacción.
7. Desarrollo de portasensor.

Al final de este capítulo se da una breve descripción del uso adecuado del sistema desarrollado en este trabajo de tesis.

3.1. Análisis del sistema robótico

Es necesario establecer que el sistema robótico a utilizar debe contar con ciertas características necesarias que, por un lado, permitan implementar los conceptos teóricos. Es decir, es necesario que el sistema robótico cuente con un lenguaje de programación de nivel tal que sea posible codificar los conceptos matemáticos que intervienen. Además, el controlador debe contar con un intérprete capaz de realizar las operaciones eficientemente, de manera que no se generen inestabilidades debido a los retrasos en la ejecución.

Por otro lado, la comunicación del sensor con el controlador del manipulador debe ser capaz de proporcionar la información del estado de las fuerzas de interacción, máximo cada 100 milisegundos, para evitar la inestabilidad causada por el operador. En esta sección se revisan las capacidades que tiene el sistema robótico para detectar cuáles proveen la mejor opción para su uso en el desarrollo del sistema de programación por guiado.

Figura 3.1. Sistema robótico Fanuc: robot LR-Mate 200*i*C, controlador R30*i*A Mate y sensor de fuerza/torque FS-10*i*A.

El sistema robótico a utilizar es el denominado *intelligent robot* de la marca FANUC, que consiste en un brazo robótico de 6 grados de libertad LR-Mate 200iC con un sensor de fuerza/torque FS-10iA. Este sistema cuenta con un controlador R30*i*A Mate. En la figura 3.1 el robot se muestra invertido y montado en una estructura en forma de puente. Inicialmente se colocó de esta manera, con la idea que hubiese la menor intersección entre el usuario y el robot y que además se aprovechara el espacio de trabajo. Sin embargo, para su desempeño de tareas de manipulación sobre la mesa generaba singularidades que causaban constantes interrupciones durante las interacciones, razón por la cual se optó por instalarlo de manera lateral, dejando la mayoría del volumen de trabajo del robot en una zona que favorece la interacción, como se ve en la figura 3.2. Se agregó una pieza a la estructura que permite montar el robot de manera lateral. Usando el robot en dicha configuración, el usuario entra en contacto con el robot manteniendo su torso fuera del volumen de trabajo, el codo del robot permanece alejado del brazo del usuario y el volumen de alcance compartido por el brazo del usuario y la muñeca del robot es más amplio, por lo

que fue posible realizar etapas de interacción más extensas en espacio y tiempo, necesarias para evaluar el desempeño de los algoritmos de control de impedancia.

3.1.1. Brazo manipulador

El robot es un brazo manipulador de 6 grados de libertad, antropomórfico, de muñeca esférica, con una capacidad de carga de 5 kg y una repetitividad de ±0.02 mm. El límite de velocidad lo define la capacidad de cada junta; si cierto movimiento requiere que cualquier junta se mueva a más velocidad de su capacidad, la velocidad del movimiento se ajustará para cumplir el límite de dicha junta.

3.1.2. Terminal de enseñanza (“Teach pendant”)

El sistema robótico cuenta con una terminal de enseñanza denominada (en inglés, teach-pendant). Ésta es utilizada para realizar la programación mediante teleoperación, además de establecer las condiciones de operación del robot, además de que representa el único medio para seleccionar el programa a ejecutar. La terminal de enseñanza posee una pantalla LCD la cual permite ver los menús en ventanas múltiples. La información como la posición del robot en coordenadas cartesianas y en juntas, además del estatus de sensores, se presenta en un área amplia denominada USER, mientras que en la parte superior de la pantalla, en pequeños sectores, se presenta información referente al estatus del programa en ciclo, además del porcentaje de velocidad con que se ejecutaran los movimientos.

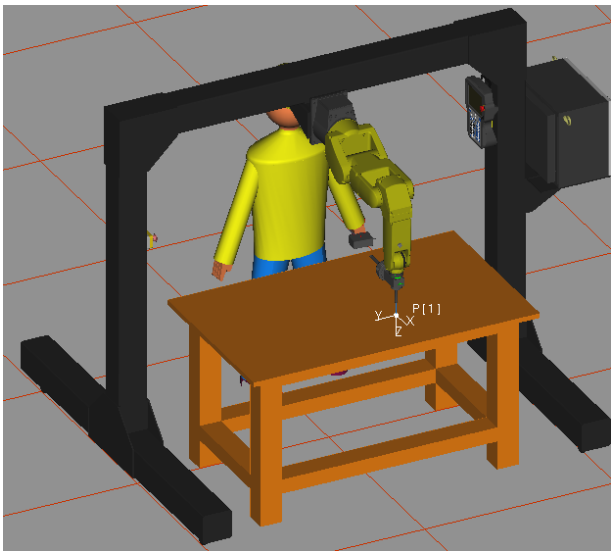


Figura 3.2. Plataforma de experimentación en la que el brazo manipulador se encuentra montado lateralmente.

3.1.3. Controlador

El controlador R30*i*A Mate ofrece la posibilidad de comunicarse con equipos periféricos a través de varias vías como son: Ethernet, puerto serial, y señales digitales, entre otras. Mediante estas posibilidades, presentes en casi todos los robots industriales, se pueden establecer comunicaciones con diferentes tipos de dispositivos periféricos o sensores. Este controlador tiene la posibilidad de ejecutar en paralelo los comandos de

movimiento, ya que estas tareas son ejecutadas por un procesador independiente. Esta característica mantiene las tazas de muestreo en sensores y dispositivos de comunicación independientes del movimiento del robot. El sistema operativo es de la marca FANUC, no basado en PC, lo que lo hace libre de virus y de arranque rápido.

3.1.4. Sensor de fuerza/torque

El sensor de fuerza percibe los estímulos en los 6 grados de libertad del espacio cartesiano de la herramienta, que es un marco de referencia móvil ubicado en un extremo del sensor, que a su vez está montado en el extremo del manipulador. El sensor muestrea aproximadamente cada 4 milisegundos, fuerzas de hasta 10 kilogramos fuerza y torques de hasta 80 kilogramo fuerza centímetro, con una resolución de 0.04 kg y 0.16 kg-cm respectivamente, para cada eje. El sensor se integra al sistema robótico mediante algoritmos basados en tareas preplaneadas provistos por el fabricante; éstos se incluyen en un programa de tipo TPE para generar etapas de control de fuerza. En el desarrollo del sistema los algoritmos del fabricante no se utilizan, sin embargo, las características del sensor se aprovechan para generar un control de fuerza cuyo desempeño depende de la velocidad de procesamiento del intérprete, encargado de realizar las operaciones necesarias para generar la respuesta ante el estímulo de fuerza presente en un instante dado. El intérprete tiene acceso a la información del sensor por medio de una variable del sistema. Las mediciones del sensor tienen una desviación que varía con la temperatura. Se ha observado que ésta nunca llega a estabilizarse en un valor dado, por lo que periódicamente debe ser sujeto a un ajuste.

3.1.5. Lenguaje KAREL

El lenguaje de programación KAREL con el que cuenta la marca FANUC contiene características de lenguajes de alto nivel, como Pascal y PL/1, e incorpora además aplicaciones especialmente desarrolladas para manipulaciones robóticas.

Estructura

El control del flujo de la ejecución del programa o rutina puede llevarse a cabo por medio de diferentes tipos de sentencias como son:

- Alternaciones: sentencias condicionales (IF) y de selección (SELECT).
- Ciclos: de ciclos específicos (FOR), que se repiten siempre que una expresión sea falsa (REPEAT), y ciclos que se ejecutan mientras una expresión sea verdadera (WHILE).
- Saltos incondicionales: (GO TO), permite saltar de un punto hacia una etiqueta en cualquier línea del programa.
- Administrador de condiciones: (CONDITION HANDLERS) por este medio se definen una serie de acciones a ser ejecutadas en el instante en que cierta condición se presente
- Controladores de ejecución: detienen definitivamente la ejecución del programa incluyendo movimientos iniciados (ABORT), suspende la ejecución del programa por un lapso de tiempo definido (DELAY), suspende la ejecución del programa (PAUSE) hasta que otra sentencia lo reinicia (CONTINUE) y

(WAITFOR) que detiene la ejecución hasta que una condición o una lista de ellas se satisface.

Movimiento

Cada sentencia de movimiento (MOVE TO) está relacionada a cierta posición del manipulador. El robot irá hasta tal configuración desde la posición en que se encuentre. Toda posición se puede representar mediante 2 tipos de variables: las relacionadas al espacio cartesiano (POSITION, XYZWPR, XYZWPREXT) y las relacionadas al el espacio de juntas (JOINTPOS). En el primer caso la posición se compone básicamente de 6 parámetros del tipo real que representan la posición y la orientación, en mm y grados sexagesimales, respectivamente, del marco de referencia de la herramienta o centro de la herramienta (TCP; Tool Center Point) respecto de un sistema de referencia cartesiano definible por el usuario. En el segundo caso, la posición se compone de 6 elementos del tipo real que indican el ángulo que cada junta ha de alcanzar al final del movimiento. El ángulo se mide desde una posición de referencia denominada el CERO DEL ROBOT donde cada junta se encuentra en 0°, la cual se muestra en la figura 3.3. El sistema de referencia cartesiano principal (WORLD), a partir del cual se definen los marcos de referencia del usuario, tiene su origen en la intersección de la normal común a los ejes J1 y J2 con el eje J1, la dirección del eje z es en la dirección del eje J1, el eje y está en la dirección del eje J2 y el eje x se define para formar un sistema coordenado de mano derecha.

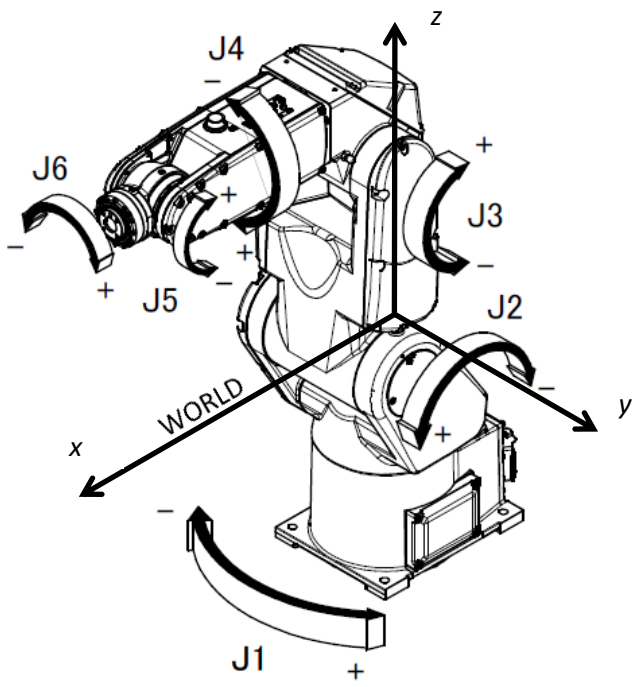


Figura 3.3. Posición CERO DEL ROBOT (todas las juntas de encuentran en 0°) y ubicación de marco de referencia WORLD.

El movimiento del robot está definido por muchos parámetros. Las características de movimiento se establecen desde variables del sistema las cuales pueden ser modificadas en cualquier instante de la ejecución. Las principales características del movimiento son:

- Trayectoria – hay tres opciones LINEAR, CIRCULAR y JOINT. En los dos primeros casos las trayectorias del TCP están bien definidas geométricamente por una línea y un arco de circunferencia respectivamente, mientras el tercer caso la trayectoria se define de manera que todas las juntas inicien y terminen su movimiento sincronizadamente.
- Velocidad – este parámetro se define de acuerdo a la trayectoria que se ha elegido. Con las opciones LINEAR o CIRCULAR la variable del sistema $SPEED establece la velocidad en [mm/seg]. En el caso JOINT, aunque la idea es definir la velocidad en mm/seg, el valor de $SPEED sólo es usado para calcular el porcentaje de la máxima velocidad al que han de moverse las juntas para que la trayectoria del TCP se mueva a una velocidad promedio cercana al valor asignado a $SPEED. También existe una manera indirecta de definir la velocidad, estableciendo la duración en milisegundos del movimiento mediante la variable $SEG_TIME.
- Terminación – esta característica define el criterio para saber cuándo, durante la etapa de desaceleración, se ha terminado la ejecución de una sentencia de movimiento para continuar con la siguiente, influyendo significativamente en la precisión de la trayectoria, ya que es posible que el robot no llegue al punto destino. Hay cinco posibles valores para la variable $TERMTYPE: FINE, COARSE, NOSETTLE, VARDECEL y NODECEL. En la figura 3.4 queda descrita su influencia, además de la influencia que tiene la sentencia NOWAIT, que indica al intérprete continuar con la ejecución del programa mientras el procesador del movimiento ejecuta la sentencia de movimiento previa.

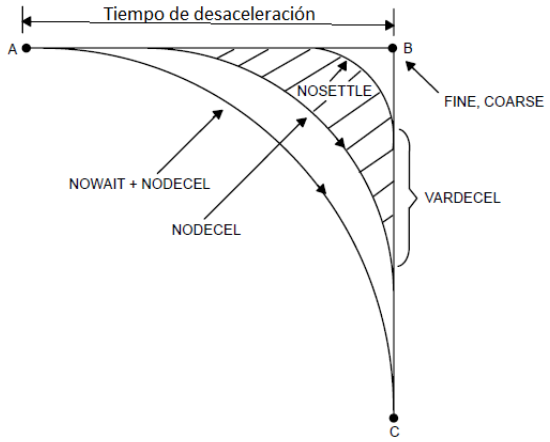


Figura 3.4. Efecto de la variable $TERMTYPE y la sentencia NOWAIT en la trayectoria del robot ante dos sentencias de movimiento, que generan una trayectoria en un ángulo de 90 grados. El robot parte de la posición A, acelera hasta la velocidad programada; cuando llega al punto de desaceleración, empieza a influir el tipo de terminación redondeando la trayectoria debido a que, sin llegar al punto B, el intérprete cambia de objetivo para buscar llegar al punto C, eliminando etapas de desaceleración.

Operadores

El lenguaje KAREL cuenta con los operadores aritméticos (+, -, *, /, DIV, MOD), relacionales (<, >, <=, =, <>, >=) y boléanos (AND, OR, NOT) necesarios para cálculos relacionados al control de impedancia. Además de los anteriores, que se presentan comúnmente en los lenguajes genéricos, se cuenta con operadores especiales (se muestran entre paréntesis) que lidian con variables de posición: (>=<); operador relacional – determina si dos posiciones son aproximadamente iguales de acuerdo a ciertas tolerancias, (:); operador relativo – se usa para transformar una posición de un marco de referencia a otro y vectores: (#); producto cruz de dos vectores, (@); producto interno de dos vectores.

El operador especial más importante es el operador de "relativo a". Mediante este operador, representado por los dos puntos (:), es posible determinar una posición dentro de cualquier marco de referencia como se muestra en la figura 3.5. En esta figura se obtiene la posición de la manija de la puerta de un coche (W_HANDLE) respecto del marco de referencia (WORLD), mediante el conocimiento la posición de la manija (B_HANDLE), respecto de un marco de referencia en la defensa trasera (BUMPER), cuya posición es conocida respecto del marco de referencia (WORLD) mediante la posición de un tornillo (BOLT).

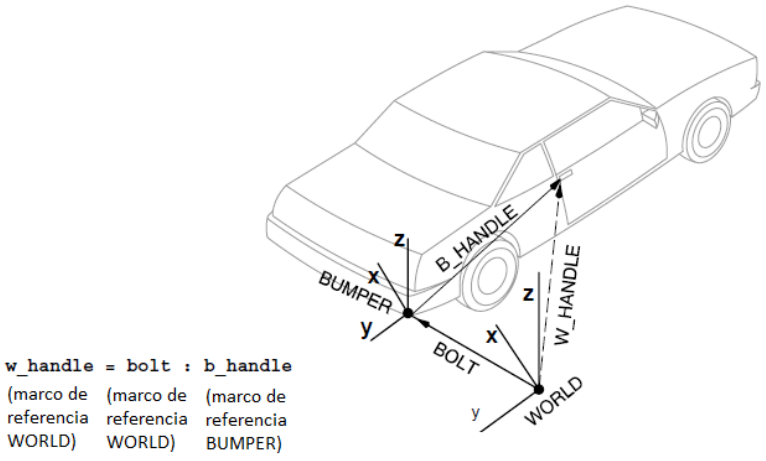


Figura 3.5. Uso del operador relativo a (:) para obtener la posición de W_HANDLE respecto WORLD.

3.1.6. Software de simulación ROBOGUIDE

Es un software de simulación fuera de línea (off line) que permite al usuario simular un proceso robótico en un espacio tridimensional, facilitando la enseñanza y edición de trayectorias sin la necesidad de incurrir en el costo de utilizar la celda de manufactura real, como se ilustra en la figura 3.6.

El software cuenta con un gran catálogo de robots, además de modelos CAD de objetos comúnmente utilizados en celdas de manufactura, de tal manera que es posible crear una copia bastante fiel de la celda de manufactura real y en dicha celda llevar a cabo pruebas de validación de las trayectorias programadas y, con la ayuda del sistema, localizar colisiones antes de implementarlas en el sistema real.

En esta sección se han mencionado las características de la marca FANUC más relevantes para la implementación del sistema de programación por guiado activo vía interacción, de las cuales las más importantes son: contar con un lenguaje de programación de alto nivel y con un sistemas comunicación con dispositivos periféricos por medio de señales analógicas o digitales, han sido identificadas en las marcas líderes en robótica industrial por lo que se puede decir que la metodología descrita en este caso particular es aplicable a la mayoría de los robots industriales del mercado actual.

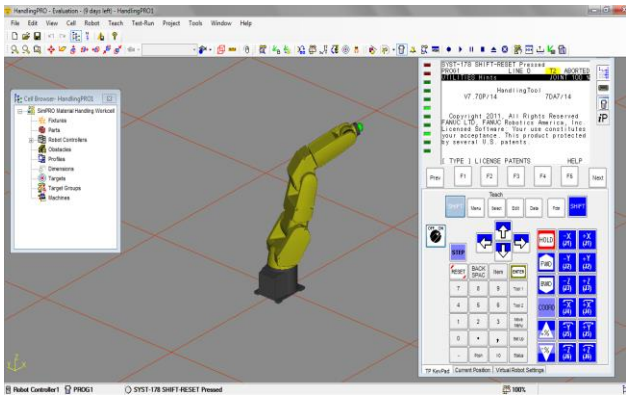


Figura 3.6. Software de simulación 3D. Izquierda; Se muestra la etapa inicial donde solo se ha escogido el modelo del robot, derecha; la terminal de enseñanza simulada, totalmente funcional

3.2. Interpretación de los conceptos teóricos

Antes de realizar cualquier implementación era necesario interpretar los conceptos teóricos que intervienen, con la intención de definir qué es lo correcto al llevar acabo el análisis e implementación de cada una de las partes que conforman los algoritmos.

3.2.1. Control de impedancia

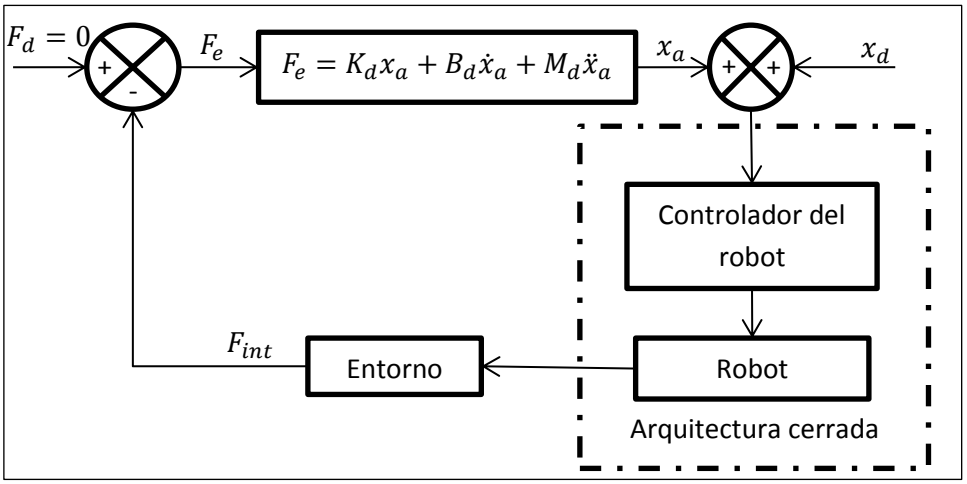


Figura 3.7. Control de impedancia, donde x_d es la posición de referencia y F_{int} es la fuerza de interacción ejercida por el robot sobre el entorno.

En el control de interacción se produce el acoplamiento de dos sistemas físicos donde controlar fuerza y posición independientemente es imposible debido a que en la interacción están relacionados la fuerza y el movimiento. La alternativa más prometedora es controlar el canal que define el comportamiento de ambos sistemas, en el punto donde la interacción se genera, que es independiente de la interacción. La impedancia y la admitancia son canales de comportamiento, el primero aplica para cualquier sistema que tiene como entrada un movimiento y por salida un esfuerzo, asimismo una admitancia aplica para cualquier sistema que tiene por entrada un esfuerzo y por salida un movimiento que concuerda con la descripción de un cuerpo que tiene masa e inercia. Si se busca un sistema que interactúe con admitancias, éste debe tener el comportamiento de su contraparte, la impedancia. Tal es el caso de un manipulador que interactúa en su mayoría con objetos inerciales. Hogan enfatiza que el control de impedancia no se limita a definir el comportamiento de un sistema masa-resorte-amortiguador en el TCP del robot, sin embargo, esta interpretación, aunque sencilla, es la que más se ajusta para su implementación en sistemas robóticos de arquitectura cerrada.

El control de la fuerza de interacción entre el operador y el robot utilizado en este trabajo es el denominado control de impedancia cinemático, basado en posición y velocidad. Este enfoque es aplicable a robots manipuladores de arquitectura cerrada. Esta es una estrategia de control indirecto de fuerza, con fuerza de referencia *"F_d"* igual a cero, como se muestra en la figura 3.7, que se caracteriza por definir el comportamiento del efector final del robot, de acuerdo a una relación dinámica entre las fuerzas de interacción y la desviación en trayectoria que en este se genera. Aunque Hogan ha insistido en aclarar que el control de impedancia no es una versión de control de fuerza, esta interpretación de control de impedancia permite entender el comportamiento del robot al interactuar con entornos rígidos.

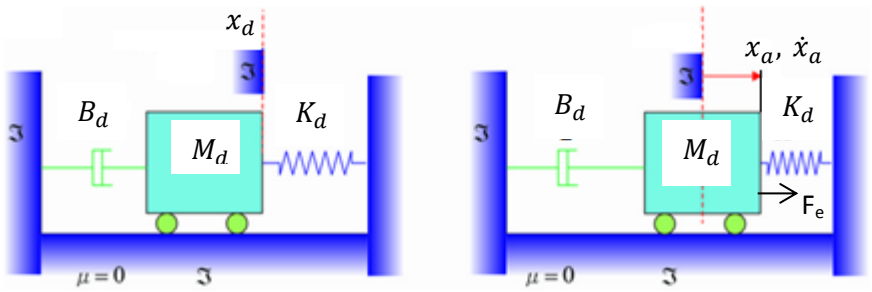


Figura 3.8. Sistema masa-resorte-amortiguador, que definirá el comportamiento dinámico de un grado de libertad cartesiano en traslación del robot manipulador.

La relación dinámica mediante la cual se genera la respuesta para cada grado de libertad cartesiano en traslación, ante la interacción del efector final del robot y su entorno, es la que corresponde a un sistema idealizado masa-resorte-amortiguador como el que se ilustra en la figura 3.8. Para el control de la orientación, la relación dinámica que se debe imponer es la del sistema análogo en rotación como el que se muestra en la figura 3.9. Es decir, un sistema momento de inercia-resorte-amortiguador de manera que el control del

momento ejercido en la herramienta sea congruente dimensionalmente con el estímulo que perciben los grados de libertad en orientación.

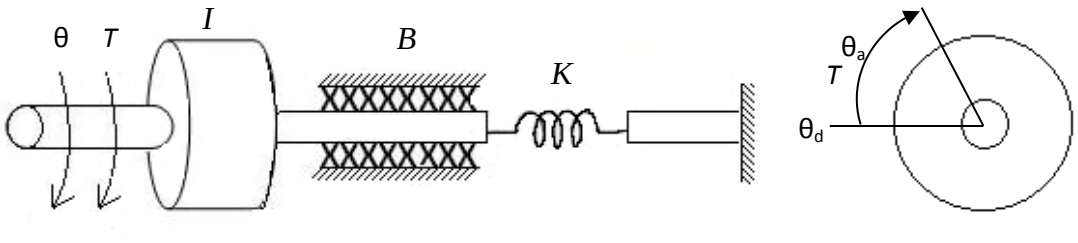


Figura 3.9. Sistema momento de inercia-resorte-amortiguador, que definirá el comportamiento dinámico de un grado de libertad cartesiano en orientación del robot manipulador.

El parámetro "K_d" es la matriz de rigidez, "B_d" es la matriz de amortiguamiento y "M_d" es la matriz de inercia. Se consideran matrices diagonales para fines prácticos, ya que esto permite que cada grado de libertad pueda ser controlado independientemente, obteniendo la respuesta del sistema de control para cada grado de libertad mediante la resolución de una ecuación diferencial de segundo orden (3.1). De igual manera para la orientación, la respuesta se obtiene para cada grado de libertad por medio de la ecuación (3.2). En la tabla 3.1. Se muestra la nomenclatura y unidades correspondientes para los parámetros que intervienen en los cálculos de las configuraciones de control.

Nomenclatura	Unidades
Movimiento de traslación	
Masa, M	kg
Rigidez, K	N/m
Amortiguamiento, B	$N.s/m$
Fuerza externa, F	N
Movimiento de rotación	
Momento de inercia de la masa, I	$kg.m^2$
Rigidez, K	$N.m/rad$
Amortiguamiento, B	$N.m.s/rad$
Momento externo, T	$N.m$

Tabla 3.1. Nomenclatura y unidades de los parámetros de impedancia.

$$F_e = K_d x_a + B_d \dot{x}_a + M_d \ddot{x}_a \tag{3.1}$$

$$T = K\theta_a + B\dot{\theta}_a + I\ddot{\theta}_a \tag{3.2}$$

3.2.2. Algoritmo de Runge-Kutta

Las ecuaciones (3.1 y 3.2) pueden resolverse mediante el método numérico de Runge-Kutta (R-K) de segundo orden, detallado en [65], del cual se obtiene la siguiente solución.

$$x_a = x_0 + \tau\left(\dot{x}_0 + \frac{\tau}{2M_d}(F_e - B_d\dot{x}_0 - K_d x_0)\right) \tag{3.3}$$

$$\dot{x}_a = \dot{x}_0 + \frac{\tau}{M_d}\left(F_e - B_d\left(\dot{x}_0 + \frac{\tau}{2M_d}(F_e - B_d\dot{x}_0 - K_d x_0)\right) - K_d\left(x_0 + \frac{\tau\dot{x}_0}{2}\right)\right) \tag{3.4}$$

Esta solución proporciona el valor de la desviación en posición y velocidad para un grado de libertad cartesiano en traslación, donde K_d, B_d y M_d representan los parámetros de impedancia, rigidez, amortiguamiento e inercia respectivamente. F_e es la fuerza de interacción, mientras que $\dot{x}_0$ y x_0 son las condiciones iniciales para cada iteración, definidos en la dirección de ese grado de libertad. Además, τ es el paso de iteración o lo que es lo mismo, el incremento en la variable independiente entre los puntos sucesivos de la solución de la ecuación diferencial, en nuestro caso, el lapso de tiempo para el cual se está prediciendo la desviación en posición x_a debida a la fuerza de interacción F_e.

3.3. Análisis del desempeño del programa previo de control de impedancia

En el laboratorio de Robótica en la Universidad Autónoma de San Luis Potosí se han desarrollado proyectos que involucran robots industriales del tipo antropomórfico para aplicaciones de interacción [60, 65, 66, 67]. De manera específica con la misma plataforma y coincidencias en la metodología, como es el esquema de control de impedancia y el algoritmo de Runge-Kutta, por [60, 66]. Sin embargo, el funcionamiento del control de impedancia, implementado en la plataforma designada para el presente trabajo de tesis, presentaba deficiencias muy importantes. De tal manera que el uso de este esquema en un sistema de programación por guiado, que pretende disminuir la dificultad al momento de conducir al robot de un punto a otro, no se justificaba. A continuación se describen las deficiencias que se identificaron así como una breve explicación de la causa asociada:

3.3.1. Deficiencias al guiar

- Al guiar el robot en posición, éste respondía de manera congruente ante el estímulo exterior, aunque solo en algunas zonas del espacio de trabajo bien definidas (aquellas más alejadas de singularidades) y para rangos de junta característicos.

- Al guiar el robot en orientación, no se identificó zona alguna del espacio de trabajo donde todas las rotaciones presentaran una respuesta congruente.

Estos dos problemas se atribuían a la matriz que llevaba las fuerzas/torques presentes en el marco de referencia "TOOL" al marco de referencia "WORLD". La cual en el código se generaba por medio de la multiplicación de dos matrices de rotación, de las que no se describió el desarrollo en reportes, por lo que no se sabía su procedencia.

- Al guiar intermitentemente, es decir intentar conducir el robot después de un lapso de tiempo sin interacción, precedido por una etapa de conducción en una dirección diferente. En este caso el primer movimiento que realizaba el robot, se

generaba en la dirección de la etapa anterior de conducción, ignorando la dirección del estímulo actual.

Esto se debía a que en el código se estableció que, cuando no hubiese fuerza de interacción, la rutina que calculaba Runge-Kutta (R-K) no se ejecutara, sin embargo los valores de las variables que intervenían en el proceso no se reiniciaban.

- Movimientos bruscos. Se presentaban cuando el algoritmo generaba como respuesta un desplazamiento muy grande, el cual se reproducía a velocidades relativamente altas. Durante el tiempo que le tomaba al controlador realizar este movimiento, el sistema no hacía caso a nuevas respuestas generadas por el algoritmo, causando inestabilidad en el sistema debido al esfuerzo que el usuario imprimía al tratar de detener dicho movimiento.

Este problema se presentaba debido a que el paso utilizado en el algoritmo de R-K, se obtenía de medir el tiempo que al intérprete le tomaba ejecutar todas las sentencias necesarias para generar la posición del movimiento anterior. Mediante este proceso, se predecía cuanto tiempo le tomaría al intérprete generar la nueva posición.

Dejando de lado la discusión acerca de cómo se relaciona el paso en R-K con el tiempo de procesamiento, está el hecho de que este procedimiento puede arrojar pasos que no garantizan la convergencia en el algoritmo de R-K causando el problema ya mencionado.

- Errores de movimiento. las etapas de interacción resultaban ser muy cortas debido a frecuentes alarmas de error de movimiento. Éstas se presentaban cuando el controlador identificaba la posición comandada como no alcanzable, provocando la interrupción del programa.

Lo anterior podía deberse a que el robot no se encontraba en la mejor posición, causando que el espacio de trabajo disponible fuese muy pequeño en la zona donde se pretendía operar, además de que era probable que el algoritmo de R-K generaba posiciones fuera del límite o singulares cuando había interacción bajo movimientos bruscos.

3.3.2. Deficiencias al sintonizar parámetros de impedancia

- El comportamiento del robot cambiaba drásticamente al modificar los parámetros.

Se supuso que estas dificultades se presentaban debido a que las variables no se utilizaban en las unidades de medición correctas.

- No había manera de sintonizar los parámetros para obtener un comportamiento aceptable, tanto en posición como en orientación.

Esto se debe a que no se pueden aplicar los mismos parámetros de impedancia a ambos casos ya que estos operan bajo conceptos mecánicos diferentes.

3.4. Implementaciones para la interacción

Se implementaron dos opciones para generar el control de la interacción una implica la detección de fuerza, mientras la segunda sólo detecta la existencia de contacto.

3.4.1. Interacción vía control de impedancia

De los análisis realizados, se notó que las deficiencias encontradas obedecían a la lógica en la implementación. Se optó por realizar una nueva implementación, utilizando de manera eficiente las herramientas que el lenguaje KAREL ofrece.

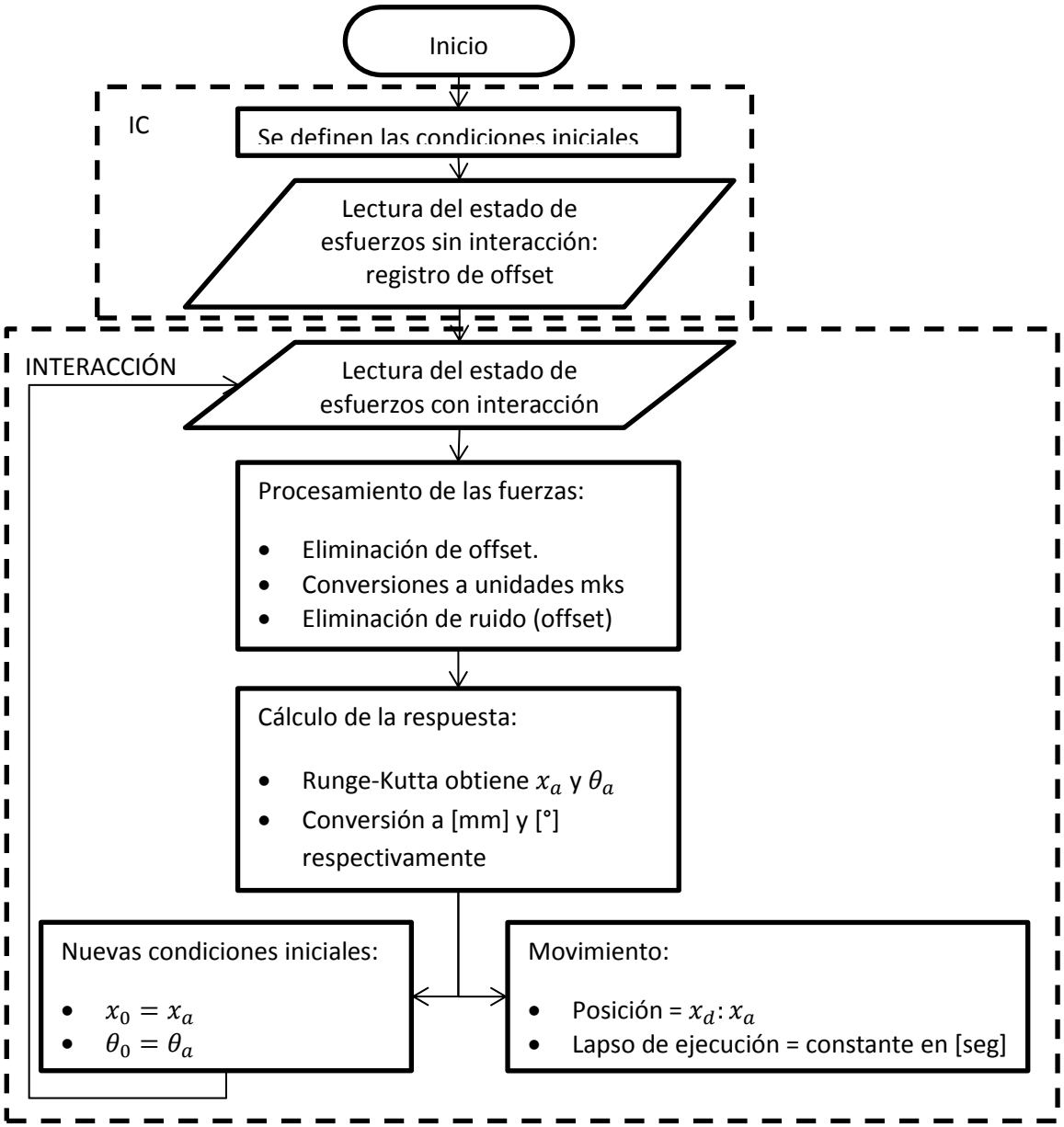


Figura 3.10. Diagrama de flujo para el algoritmo de control de interacción via control de impedancia.

Anteriormente el estado de esfuerzos se proyectaba al marco de referencia WORLD del robot, por medio de una matriz de rotaciones que involucraba los ángulos de rotación WPR de la configuración del TCP del robot. Aunque no se encontraron errores en la formulación e implementación, esta estrategia no daba un resultado exitoso. Se optó por una matriz de rotación que involucra los ángulos de cada junta de la configuración del robot. Aunque hubo mejora la interacción, aun presentaba deficiencias como las descritas en la sección anterior. Dado que la estrategia de proyectar las fuerzas presentes en el TCP hacia WORLD no era exitosa.

Finalmente se optó por calcular la respuesta directamente en el TCP y, de esta manera, obtener cada componente del vector x_a el cual se agrega a la posición x_d que, para fines del sistema de programación, es una posición fija. En la figura 3.10 se muestra un esquema de la implementación, donde la respuesta generada por Runge-Kutta se agrega a la posición actual mediante el operador "relativo a (:)".

Dado que se quiere implementar un sistema de programación por guiado, la posición deseada es fija. Debido a la deviación variante presente en el sensor de fuerza, se requiere establecer un nivel mínimo de magnitud de fuerza; valores debajo de ese mínimo se consideran ruido. En la figura 3.10 se muestra el diagrama de flujo del algoritmo que computa la interacción. Nótese cómo, desde que se obtiene la información del estado de esfuerzo hasta que se realiza el movimiento correspondiente, ha de pasar un lapso de tiempo que dependerá de la velocidad de procesamiento del intérprete. El movimiento se realizará en paralelo con los cálculos necesarios para generar el siguiente movimiento. Existen tres posibilidades, una es que el robot acabe el movimiento antes de que se genere el siguiente comando de movimiento, por lo que el robot detendrá su movimiento cada iteración; la segunda es que el intérprete genere comandos de movimiento más rápido de lo que el robot los ejecuta, por lo que se genera un cola de espera para el intérprete de movimiento. La tercera posibilidad y la más deseable, es que el movimiento acabe justo cuando se ha generado el siguiente movimiento, de esta manera el movimiento será continuo y con el menor retraso respecto del instante en que se generó el estímulo. Pero, ¿cómo se relaciona lo anterior con el paso del algoritmo de R-K?

El uso de este método de solución se selecciona, con la idea de lidiar con la posibilidad de frecuencias de muestreo del sensor variable o bien, el tiempo de procesamiento variable, ya que en éste, el paso se puede definir como variable. Sin embargo, se debe recordar que el paso de este método numérico afecta la convergencia por lo que se ha de poner especial cuidado al definir el valor de la variable τ. A diferencia de la implementación anterior, para este trabajo de tesis el paso se establece constante en un valor pequeño, es decir uno para el cual el método de R-K de segundo orden asegure convergencia y se elige de manera que tanto el tiempo de ejecución de movimiento, el tiempo de procesamiento de la iteración y el paso de R-K sean muy similares. Aquí a la frecuencia de muestreo del sensor se le concede menor importancia, considerando que, dado que la comunicación es provista por el fabricante y el sistema no está basado en PC, la frecuencia debe ser relativamente alta. En el caso de que ésta sea baja sólo está la posibilidad de que el sistema realice varias iteraciones para un mismo valor de fuerza, situación que se hace grave si el sensor muestrea muy lentamente.

3.4.2. Interacción vía detección de contacto.

Como alternativa de control de interacción, se tiene la posibilidad de utilizar el sensor de fuerza para detectar si hay contacto; en este caso el sensor hace las veces de un conjunto de interruptores que se activarían al ser presionados, indicando en que direcciones y sentidos hay contacto para generar movimiento en esas direcciones. Dado que la generación del comando de movimiento requiere menos procesamiento, el tiempo entre el instante en que se da el contacto y el inicio del movimiento es menor, por lo que a pesar de que se ignora la intensidad de la fuerza el desempeño podría resultar mejor.

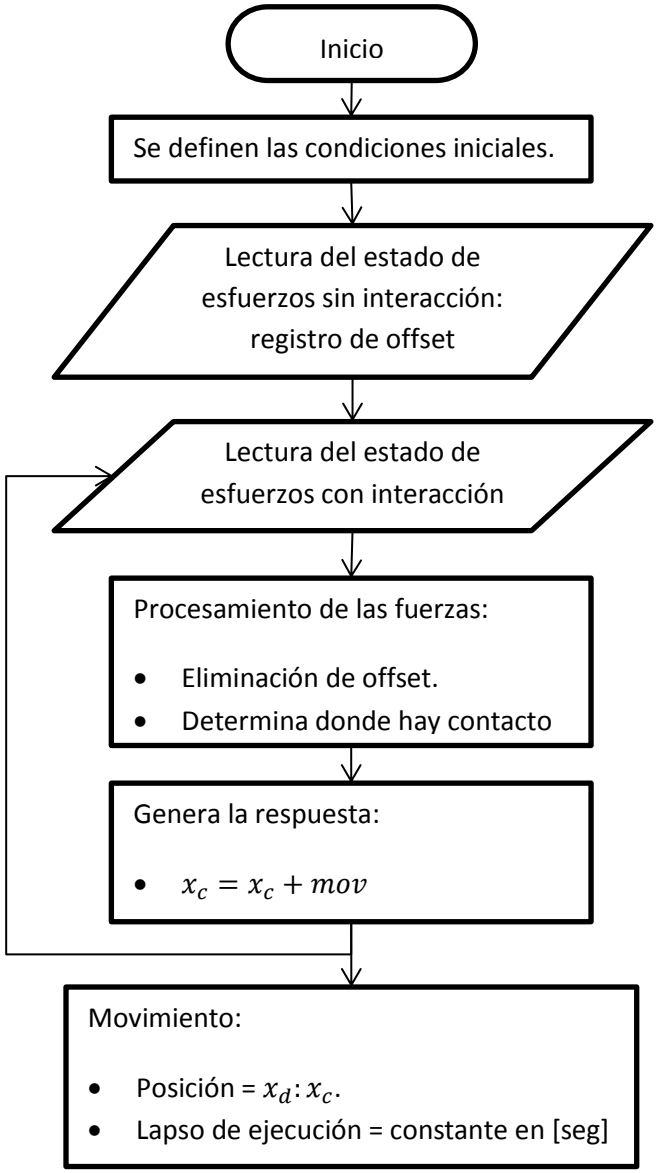


Figura 3.11. Diagrama de flujo para el algoritmo de control de interacción vía detección de contacto.

A partir del algoritmo de control de impedancia se generó un algoritmo que solo omite el procesamiento innecesario para detectar el contacto (figura 3.11) y se estableció una magnitud de lectura del sensor para la cual se considera que existe contacto. También

se establece una cantidad de movimiento (mov) a velocidad constante, para imponerse a cada grado de libertad en el que se haya detectado contacto durante la iteración del algoritmo. De esta manera se genera la posición recursiva x_c referenciada a la posición fija x_d. Como en el caso anterior, el desempeño dependerá de que la ejecución el movimiento sea tan rápida como la generación de cada comando de movimiento, para asegurar el mínimo retraso entre el contacto y su correspondiente movimiento.

La cantidad de movimiento que se ha de ejecutar cada vez que se detecte contacto, deberá ser pequeña así como el tiempo que le tome su ejecución, para asegurar mínimo tiempo de pérdida de control.

3.5. Desarrollo de la interfaz de comunicación

Con el objetivo de indicar al sistema de programación por guiado las órdenes del operador, se diseñó una terminal de enseñanza que permite indicar comandos por medio de botones, además de una señal de paro de emergencia. Este diseño se logró permaneciendo bajo algunos ciertos criterios previamente establecidos:

- Solo botones de control básicos
- Liviano
- Portable en una mano
- Ergonómico
- Debe contar con un paro de emergencia (botón de hombre muerto).

A continuación se detalla la manera en que se logró la comunicación entre el controlador y la terminal de enseñanza.

3.5.1. Señal de paro de emergencia (botón de "hombre muerto")

El controlador del robot manipulador está diseñado con la capacidad para conectarse a un botón de paro de emergencia externo vía "hardware", como se ve en la figura 3.12. Este botón suele ser utilizado para detener el sistema en el momento en que la puerta de una valla de seguridad se abra, evitando la presencia de personas en el área de trabajo cuando el robot se encuentra en ciclo.

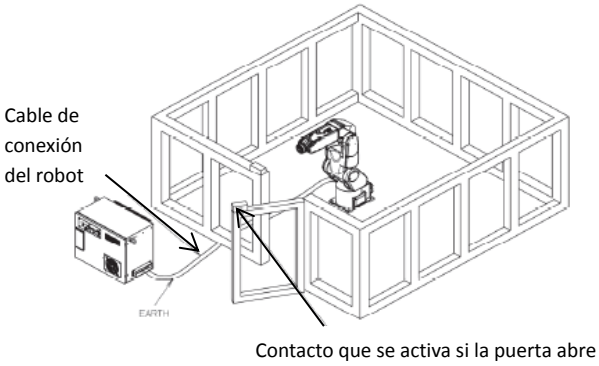


Figura 3.12. Configuración convencional de una celda de manufactura.

A continuación en la figura 3.13 se muestra el diagrama de conexión del botón de paro de emergencia. Como el fabricante lo indica, se deben utilizar botones de paro con contactos redundantes ya que, aunque con una señal el sistema detecta cuando la puerta se ha abierto, si no se agrega la señal redundante el sistema marca un error que indica que hay una conexión anormal.

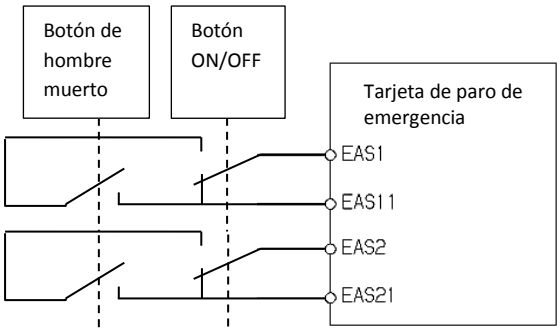


Figura 3.13. Conexión de la señal de paro de emergencia para botón de hombre muerto.

En la tarjeta de paro de emergencia se ofrece la posibilidad de utilizar dos paros de emergencia externos, uno suele ser utilizado para detectar cuando la valla de seguridad se encuentra abierta y el otro detecta cuando una persona presiona un botón instalado en algún lugar estratégicamente seleccionado de la valla de seguridad. El error identificado cuando la valla se abre, ocasiona que el robot desacelere hasta detenerse, mientras que el efecto que tiene el botón de paro de emergencia externo en el sistema, es que el robot se detendrá inmediatamente, por lo que esta señal de seguridad es la adecuada para generar un botón de hombre muerto en la terminal de enseñanza. Se unieron dos interruptores de dos posiciones de regreso con resorte para generar la señal de paro redundante. Los interruptores son muy suaves al tacto y se utilizaron en modo normalmente abierto, en conjunto con un interruptor de dos polos, dos vías, dos posiciones, para nulificar esta señal de paro cuando el sistema de programación no esté en uso, como se ve en la figura 3.13.

3.5.2. Señales de control de periféricos

Las señales de control de periféricos, tanto entradas como salidas, se realizan mediante los conectores CRMA58 y CRMA59 que se encuentran en la tarjeta de conversión como se muestra en la figura 3.14.

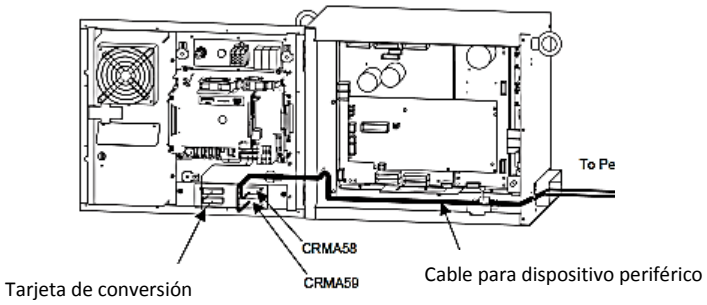
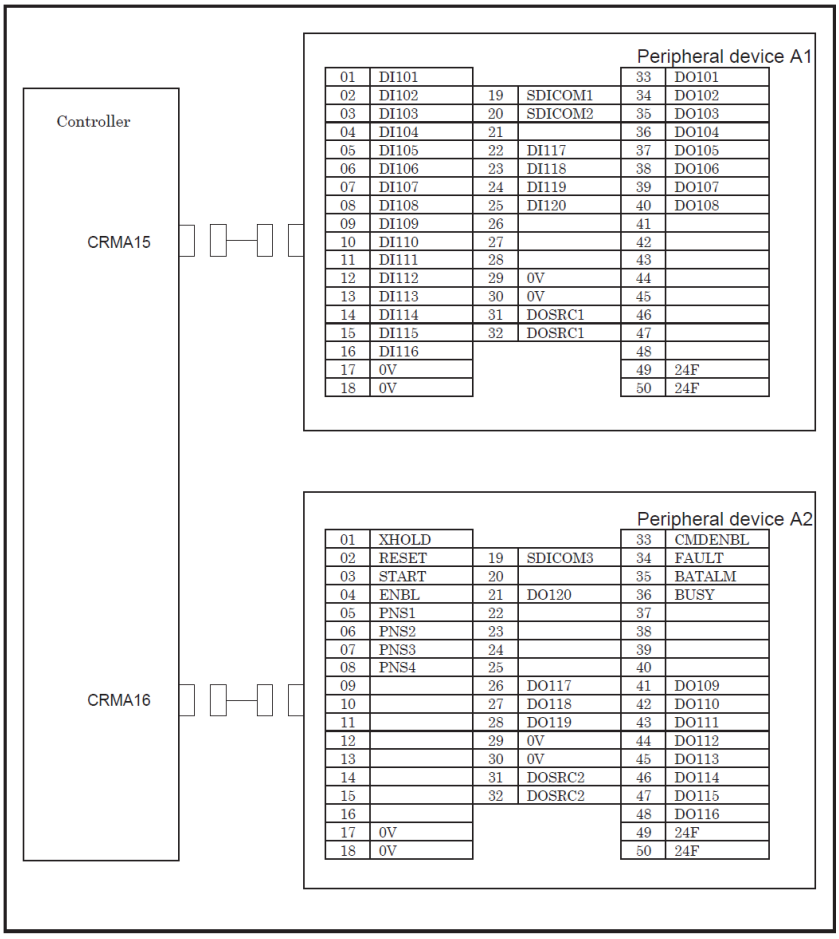


Figura 3.14. Ubicación de los puertos de comunicación en el controlador.



01	DI101			33	DO101
02	DI102	19	SDICOM1	34	DO102
03	DI103	20	SDICOM2	35	DO103
04	DI104	21		36	DO104
05	DI105	22	DI117	37	DO105
06	DI106	23	DI118	38	DO106
07	DI107	24	DI119	39	DO107
08	DI108	25	DI120	40	DO108
09	DI109	26		41	
10	DI110	27		42	
11	DI111	28		43	
12	DI112	29	0V	44	
13	DI113	30	0V	45	
14	DI114	31	DOSRC1	46	
15	DI115	32	DOSRC1	47	
16	DI116			48	
17	0V			49	24F
18	0V			50	24F

01	XHOLD			33	CMDENBL
02	RESET	19	SDICOM3	34	FAULT
03	START	20		35	BATALM
04	ENBL	21	DO120	36	BUSY
05	PNS1	22		37	
06	PNS2	23		38	
07	PNS3	24		39	
08	PNS4	25		40	
09		26	DO117	41	DO109
10		27	DO118	42	DO110
11		28	DO119	43	DO111
12		29	0V	44	DO112
13		30	0V	45	DO113
14		31	DOSRC2	46	DO114
15		32	DOSRC2	47	DO115
16				48	DO116
17	0V			49	24F
18	0V			50	24F

Figura 3.15. Esquema que identifica cada una de los bornes de los conectores CRMA58 y CRMA59, los cuales están conectados a los conectores CRMA15 Y CRMA16 respectivamente.

En la figura 3.15, las señales de entrada se identifican como DI, se cuenta con 20 de ellas en el conector CRMA58. Este conector también cuenta con 8 señales digitales de salida denominadas DO. El conector es del tipo MR-50LW. En la figura 3.16 se muestra la configuración interna del conector CRMA58, así como la manera de conectar los interruptores. Como se aprecia en la figura 3.16. La señal digital se obtiene de los dos estados de un interruptor (ON/OFF), cada una de las entradas cuenta con una resistencia, las cuales se conectan a un punto común, que a su vez se conecta a 0 volts. Para cerrar el circuito, uno de los extremos de los interruptores se conecta a la fuente de 24 volts.

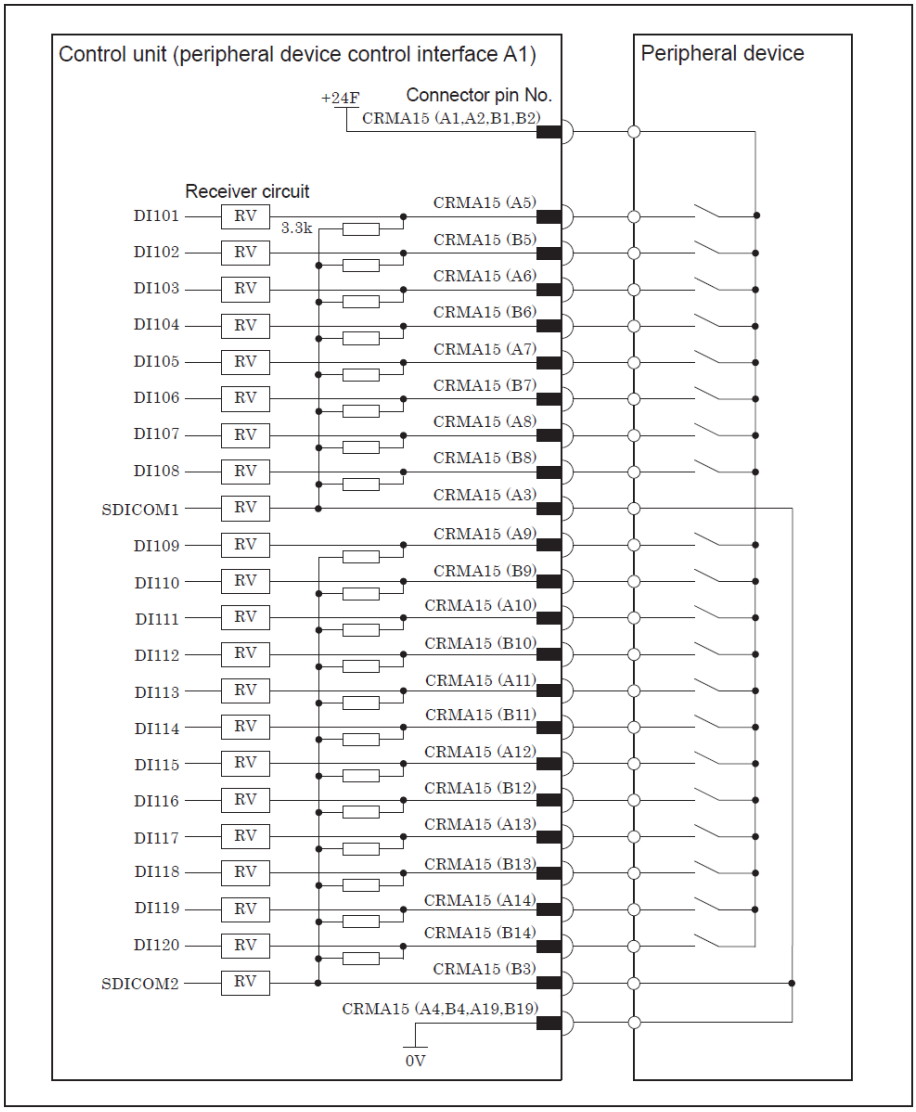


Figura 3.16. Diagrama de conexión de las señales digitales. Se puede apreciar que para generar una señal de entrada no es necesario contar con una fuente de voltaje ni resistencias “pull down”.

Siguiendo los diagramas y criterios de diseño establecidos, se fabricó la terminal de enseñanza mostrada en la figura 3.17, la cual cuenta con un mango portable en una mano donde se puede presionar un botón que hace las veces de un botón de hombre muerto. Además cuenta con 4 botones tipo “push” que se utilizarán como interfaz de comunicación entre el operador y el controlador. La conexión con el controlador se realiza por medio de un cable con 9 líneas, 4 para el paro de emergencia y 5 para los 4 botones.

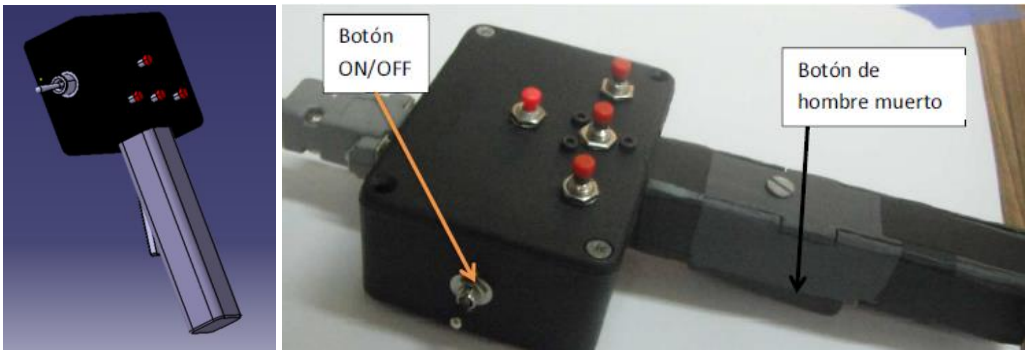


Figura 3.17. Terminal de enseñanza. Izquierda: diseño CAD. Derecha, producto real.

3.6. Desarrollo de programas para la administración de la interacción

La realización de un programa por medio de guiado exige, entre otras, la capacidad del operador para comunicarse con el controlador durante la ejecución del programa que permite la manipulación del robot mediante interacción física. Esto implica la interrupción de dicha ejecución para alternar la ejecución de rutinas que permitan grabar las posiciones de interés, modificar parámetros que definen el comportamiento de la interacción física o modificar parámetros que definen de qué manera se guardarán las posiciones elegidas.

Para esto se utilizó la propiedad del lenguaje Karel denominada "condition handler", la cual permite al programa responder a condiciones externas, como son las señales de la terminal de enseñanza. Mediante las "condition handler", una condición específica se monitorea en paralelo con la ejecución del programa para que, cuando dicha condición ocurra, una acción correspondiente pueda tomarse insertándose en la secuencia de la ejecución. Las "condition handler" constan básicamente de 3 operaciones:

1) Definición – se realiza en la sección ejecutable del programa y requiere de la sentencia WHEN.

```
CONDITION[n]
WHEN condición DO
        Acciones
ENDCONDITION
```

2) Activación – establece cuándo se empieza a monitorear cierta condición mediante la sentencia ENABLE CONDITION[n].
3) Desactivación – establece cuándo se deja de monitorear cierta condición mediante la sentencia DISABLE CONDITION[n]. Las condiciones también se desactivan cada que se detectan, por lo que una práctica común es activarlas nuevamente en sus propias acciones.

Previamente se habían identificado 6 posibles tareas básicas a realizar durante la generación de un programa. Tres están relacionadas con parámetros que definen la interacción física del operador con el robot durante el guiado (acciones del "modo guiando") y tres relacionadas con la generación del programa (acciones del "modo grabando").

3.6.1. Tareas del modo guiando

Con base en la experiencia con el uso de los sistemas convencionales de programación (vía terminal de enseñanza), se notó la necesidad de guiar el robot a diferentes velocidades de acuerdo a la precisión de movimiento que requiere la proximidad con el objetivo. Es decir, mientras más cerca se encuentra el TCP del objetivo, más lento debe ser el movimiento de aproximación. En las terminales de enseñanza la velocidad se modifica de acuerdo a un porcentaje de la máxima capacidad designada para las etapas de enseñanza. En el sistema convencional esto se logra con tres botones, uno para aumentarla, otro para reducirla y otro para modificar el paso. Debido al ahorro en botones impuesto durante la etapa de diseño de la terminal de enseñanza, en el sistema de programación el cambio de velocidad se hará con un solo botón a través de opciones preestablecidas seleccionables por cada pulsación del botón, que representan un porcentaje de la máxima velocidad programada.

La implementación del cambio de velocidad se realiza con una condición que se activa cada que se oprime un botón, cambiando el valor de una variable que multiplica los valores de los parámetros de impedancia, haciendo el sistema más rígido o más dócil. Esto es posible dado que el tiempo de ejecución de cada comando de movimiento es constante, por ende, la velocidad depende solo del tamaño del desplazamiento y a su vez, el desplazamiento depende de los parámetros de impedancia, los cuales pueden modificarse multiplicándolos por un factor común, sin modificar la selección del factor de amortiguamiento. Se definieron cuatro niveles de velocidad seleccionables mediante dos condiciones, una para disminuirla gradualmente hasta cierto nivel que da la precisión requerida y otra para regresarla gradualmente a la condición inicial, que permite manipulación a mayor velocidad.

Otra capacidad útil al momento de guiar en los sistemas convencionales, es la de conseguir manipular posición u orientación de manera independiente, es decir, lograr manipular una manteniendo fija la otra. Para esto se establecieron tres opciones seleccionables por cada pulsación de un botón en la terminal de enseñanza: guiar posición y orientación, guiar solo posición y guiar solo orientación.

Para lidiar con la desviación variante del sensor, se asoció una condición a un botón de la terminal de enseñanza para que cuando sea presionado, se ejecute la parte IC de la figura 3.10. Como consecuencia, después de esta acción, el algoritmo habrá obtenido el nuevo offset del sensor y además las condiciones iniciales para el algoritmo de R-K se hacen cero, para detener el movimiento del robot. Es importante mencionar que el uso de este botón requiere ausencia de contacto con el sensor.

3.6.2. Tareas del modo grabando

El modo grabando, convenientemente, exige a su vez tres tareas. Una permite finalizar el programa y las dos restantes están relacionadas con el tipo de trayectoria seleccionada para la ejecución de los puntos enseñados.

Es necesario, durante la realización de un programa guiado, indicar cuándo se ha terminado. Esto se realiza mediante la pulsación de un botón. El estado de esta condición se usa para detener y finalizar el programa en ejecución.

Una manera de evitar la inconsistencia humana es capturar solo puntos clave de la enseñanza y definir trayectorias básicas entre ellos. Lo que implica la necesidad indicar al sistema cuando se ha conseguido la configuración deseada y la trayectoria que seguirá el robot para llegar hasta ésta. En los sistemas convencionales comúnmente hay tres posibles trayectorias, lineal, circular y en juntas. Dado que la última genera trayectorias indefinidas en el espacio cartesiano, resulta poco útil a pesar de ser la que aprovecha más la máxima velocidad de movimiento del robot. Por tal motivo se definieron dos trayectorias seleccionables mediante un botón, cada que se pulsa el botón cambia el tipo de trayectoria de lineal a circular y viceversa. La captura también se hace mediante la señal de un botón, mediante esta señal se inician una serie de acciones que guardan la información referente al punto enseñado, como es el número de punto, trayectoria y su ubicación. Estos datos se registran en archivos de texto almacenados en la memoria del robot.

3.6.3. Estructura principal

Se ha definido el uso de tres botones en los dos modos, por lo que estos realizan dos tareas, una para cada modo. La estructura principal del programa se generó sobre el algoritmo de la figura 3.10, afectada por las acciones de las condiciones definidas para cada modo de operación, como se muestra en la figura 3.18. La selección del modo se realiza por medio de una condición ejecutada cuando se presiona el botón restante y esta selección afecta el uso de los tres primeros botones y el comportamiento del robot ya que, en el modo grabando, el robot permanecerá fijo e indiferente ante las fuerzas de interacción.

Mediante este método se generó un programa en el que, por medio de cuatro botones, se puede establecer cuándo realizar seis acciones diferentes, las cuales se dividen en acciones del “modo guiando” y acciones del “modo grabando”. Cuando se selecciona el modo grabando el programa se cicla en espera de que alguna de las opciones correspondientes ocurra y se ejecuten las acciones necesarias para cada tarea. Al regresar al modo guiado, el sensor deberá estar sin esfuerzos ya que, inmediatamente después de la selección, se ejecuta IC, para después ejecutar INTERACCIÓN. Dentro de INTERACCIÓN se definen los parámetros de impedancia multiplicados por una variable “c” que modifica su valor y por ende la velocidad de interacción, también se selecciona qué parte de R-K se ha de ejecutar para obtener control de posición, orientación o ambas.

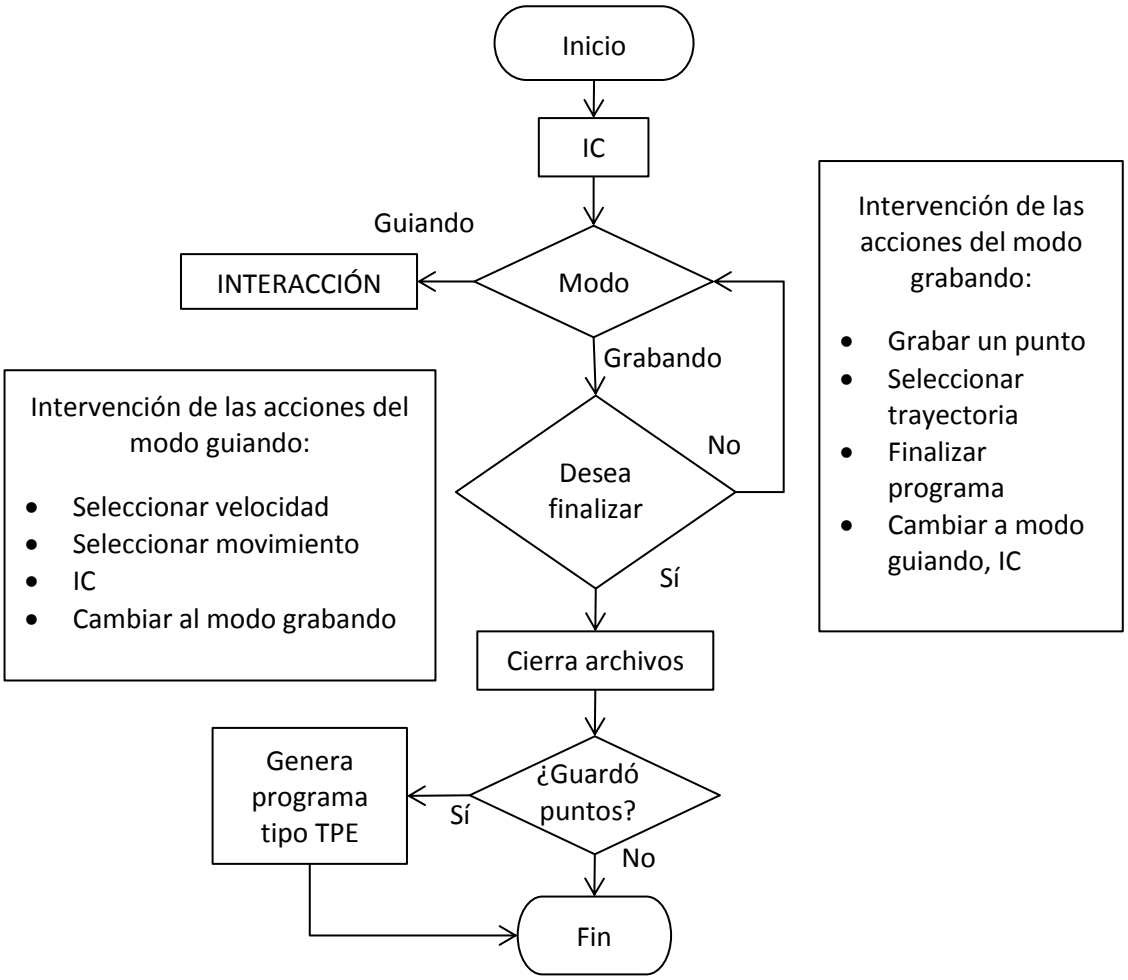


Figura 3.18. Estructura básica del programa para la interacción y generación de trayectorias.

3.6.4. Programa tipo TPE

Después de indicar la finalización del programa, la información generada queda guardada en una serie de archivos de texto. En el caso de que se haya guardado por lo menos una posición, el programa de la figura 31.8 manda llamar otro programa en el que, a partir de los archivos, genera un programa de tipo TPE. Los programas tipo TPE son útiles para poder editar al programa generado mediante la interacción, agregando atributos que le dan funcionalidad al sistema de programación, constituyéndose como una etapa de post-procesamiento.

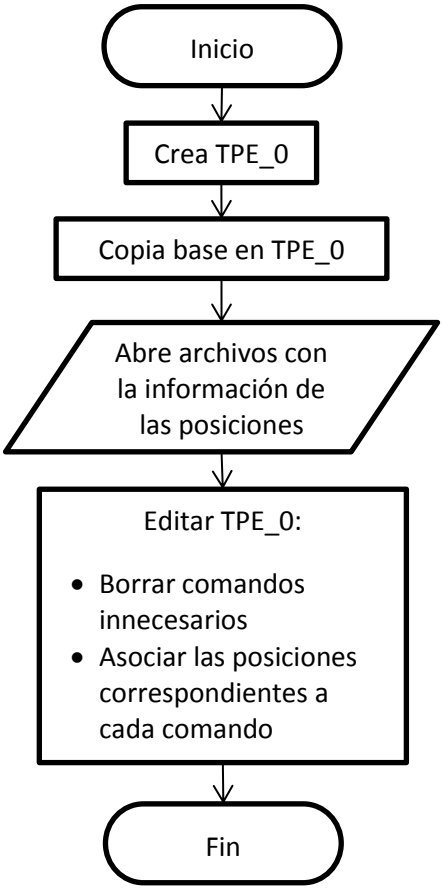


Figura 3.19. Esquema del proceso para crear el programa tipo TPE a partir de la información guardada en los archivos durante la enseñanza.

KAREL cuenta con una serie de rutinas para la edición de un programa del tipo TPE, sin embargo, la capacidad de agregar instrucciones no es una posibilidad. Por lo que fue necesario crear un programa TPE previo, el cual contiene características necesarias que se podrían atribuir a un programa creado mediante el sistema de guiado por interacción física A éste se le denomina base y como etapa inicial se crea con la capacidad de generar 100 movimientos; cada movimiento requiere que la base cuente con dos movimientos, uno lineal y otro circular, para dar la posibilidad de elegir el tipo de trayectoria. De manera que la base tiene 200 comandos de movimiento, alternando uno circular y uno lineal. Todos los comandos se encuentran vacíos, es decir, no están relacionados a ninguna posición. En la figura 3.19 se presenta el proceso para generar el programa de tipo TPE.

3.7. Desarrollo de porta-sensor

Para tener la posibilidad de mostrar el funcionamiento del sistema de programación por guiado, se fabricó un porta-herramientas que permite contar con dos platos de montaje, uno de ellos es utilizado para montar el sensor de fuerza y en el otro se puede montar una herramienta, que es la que demostrará si el sistema tiene la capacidad de ser utilizado con un propósito definido.

El diseño se realizó buscando economizar tanto procesos de manufactura como costos de materiales. Lo anterior sin prescindir de los siguientes criterios de diseño necesarios:

- Que el porta-herramientas sea liviano.
- Que su aplicación no limite en exceso la movilidad del robot.
- Que el plato de montaje destinado para el sensor de fuerza, permita un desmontaje rápido del mismo, con el fin de que durante la etapa de pruebas, el sensor pueda ser removido evitando al 100% la posibilidad de colisión con el robot.

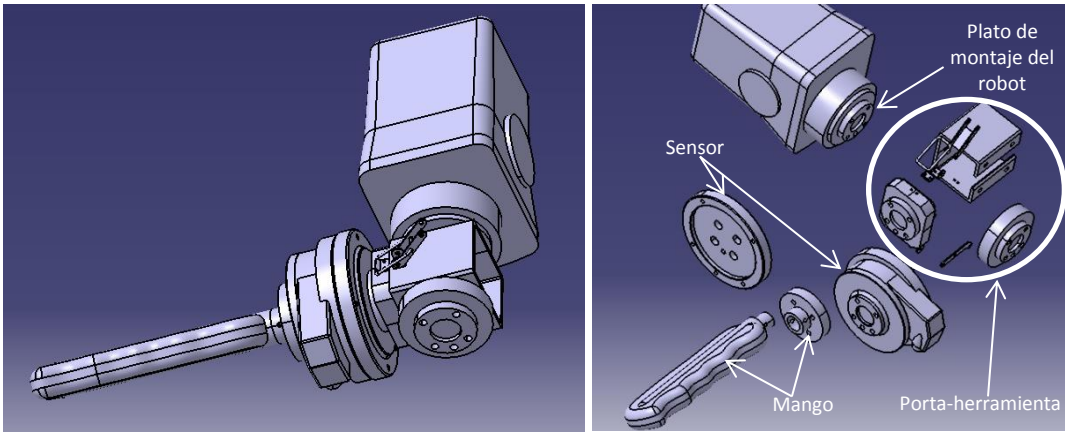


Figura 3.20. Modelo CAD de la porta-herramienta. En esta imagen se puede ver el sistema de montaje propuesto, que consiste en sujetar el plato del sensor al porta-herramientas mediante unos broches que mediante tensión, sujetan de manera efectiva el plato de montaje del sensor.

Se consideraron diversas estrategias como, uniones controladas magnéticamente y mecánicas, sin embargo los diseños mecánicos resultaron más viables. En la figura 3.20 se muestra el diseño más apropiado de acuerdo a los criterios preestablecidos.

Debido a las limitaciones de manufactura no fue posible reproducir fielmente el modelo, sin embrago se generó un diseño que cumplía de diferente manera todos los criterios de diseño.

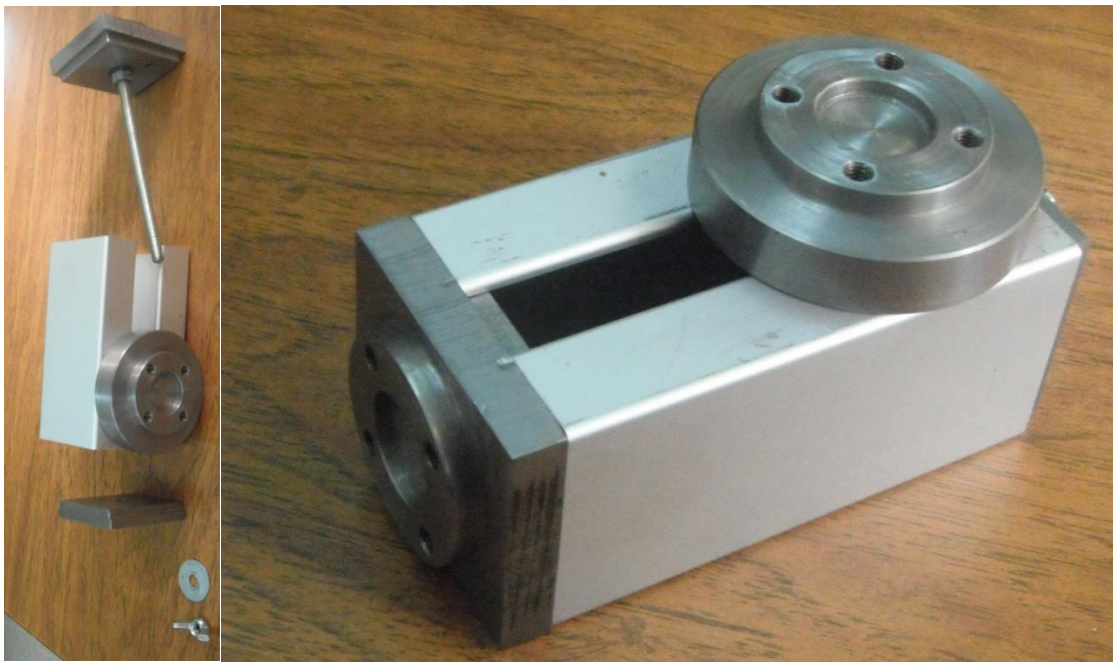


Figura 3.21. Izquierda, Conjunto de piezas que conforman el porta-herramientas. El desmontaje rápido en el diseño final, se logró mediante un perno pasante que se fija mediante una tuerca de mariposa. Derecha, ensamble final del porta-herramientas.

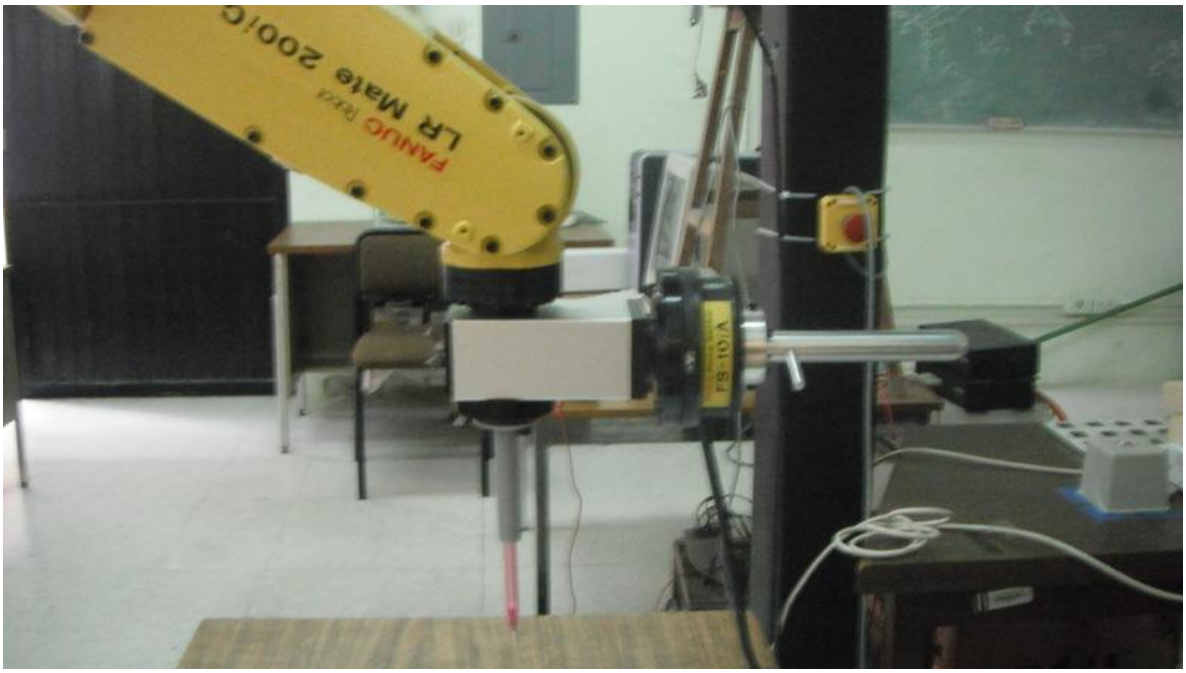


Figura 3.22. Montaje del porta-herramientas en el robot. La herramienta consiste en un lapicero plegable mediante un resorte, para evitar colisiones sobre la mesa de trabajo. Con esta herramienta se podrá verificar si el sistema de programación por guiado resultó más eficiente que los convencionales.

3.8. Uso del sistema

El procedimiento de programación se muestra mediante el esquema de la figura 3.25 que consiste en tres etapas, las cuales se detallan a continuación.

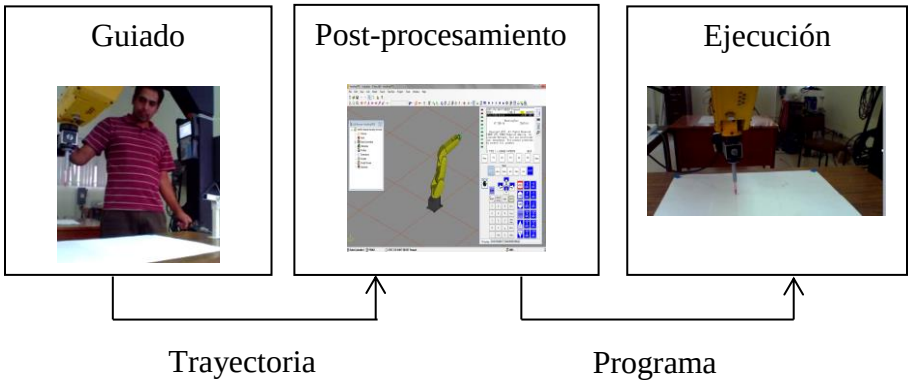


Figura 3.23. Etapas para la creación de un programa

3.8.1. Guiado

Consiste en llevar manualmente el robot a través de la trayectoria deseada para ubicarlo en ciertas posiciones de interés que serán guardadas. Para efecto de comunicar al sistema las acciones a tomar durante la etapa de guiado, se generó un algoritmo en el que, por medio de cuatro botones, se puede establecer cuándo realizar seis acciones diferentes, las cuales se dividen en acciones del "modo guiando" y acciones del "modo grabando". Para obtener más detalles ver el apéndice B.

3.8.2. Post-procesamiento

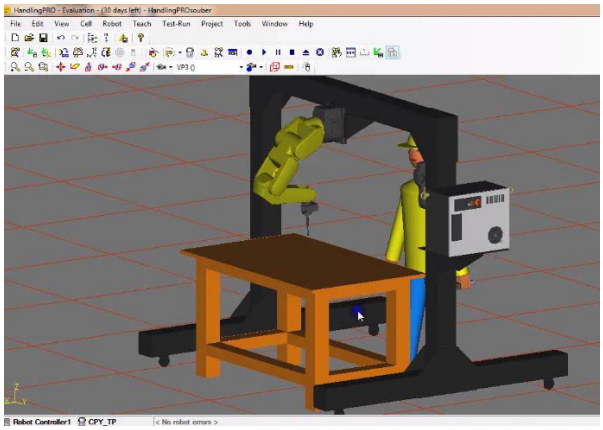


Fig. 3.24. Celda de Manufactura virtual HandlingPRO.

En esta etapa es necesario transferir el programa de tipo TPE, generado en la etapa anterior, a una computadora donde mediante la plataforma ROBOGUIDE se puede realizar o modificar un programa partiendo de la trayectoria ya generada en la etapa de guiado. Dicha plataforma ofrece la posibilidad de modificar, usando una terminal de enseñanza virtual. Además, en esta etapa se pueden hacer pruebas de funcionalidad de los programas

en una celda de manufactura virtual la cual puede, mediante diseño CAD, reproducir de manera muy fiel el estado de la celda de manufactura con la que se cuente en la realidad, como se ve en la figura 3.24. En esta etapa se puede modificar totalmente la trayectoria, velocidad, y terminación o bien agregar comandos referentes a la activación de herramientas, ciclos, pausas en espera de señales, entre otros. De manera que la implementación de las trayectorias sea funcional en algún proceso de manufactura, segura para el operador y para el equipo y se ahorre tiempo de uso de la celda de manufactura real.

3.8.3. Ejecución

Para esta etapa es necesario exportar, desde la PC hacia el sistema robótico, el programa tipo TPE modificado y probado en la etapa anterior. Qué tan segura es la ejecución respecto de lo observado en la celda de manufactura virtual, dependerá de la presión con que se reprodujo la celda de manufactura real. Sin embargo, siempre es recomendable hacer una ejecución a baja velocidad, sobre todo en los puntos críticos por ejemplo, aquellos en los que el robot se desplace muy próximo a la pieza de trabajo o a estructuras de la celda de manufactura. Además, el usuario ha de permanecer atento y cercano a cualquiera de los paros de emergencia, para que en caso de cualquier falla, sea posible detener al sistema inmediatamente. En esta etapa, de acuerdo al proceso de aplicación, es muy posible que se requiera de un proceso de calibración de la trayectoria para obtener un funcionamiento adecuado.

3.9. Conclusión del capítulo

En este capítulo se ha presentado la metodología seguida para implementar un sistema de guiado activo para un robot de modelo específico. No obstante, bajo pequeñas variaciones en las problemáticas que se puedan presentar, una metodología similar puede aplicarse a otros modelos que se encuentren en el mismo nivel de desarrollo tecnológico.

Capítulo 4

4. Pruebas y discusión de resultados

El éxito en el desarrollo de cualquier sistema, particularmente en el caso de una plataforma de arquitectura cerrada como la utilizada en el presente trabajo, se centra en el conocimiento de las capacidades de dicha plataforma. Al ser las nuevas aplicaciones no convencionales, los proveedores del equipo utilizado no suelen aportar la información necesaria, además que, en muchas ocasiones, esta información se debe comprobar debido a la naturaleza no común de las aplicaciones.

En este capítulo se presentan pruebas realizadas durante el desarrollo del sistema, necesarias para tomar decisiones y se presentan implementaciones seleccionadas de acuerdo a las capacidades que se observaron tanto en el sistema robótico como en los algoritmos utilizados. Finalmente, se realizaron pruebas al sistema de programación desarrollado, con el objetivo de verificar su utilidad en comparación con el sistema convencional del proveedor.

4.1. Frecuencia de muestreo del sensor de fuerza

Un aspecto muy importante en los sistemas basados en información provista por sensores, es con qué frecuencia esta información se genera. El sensor utilizado es parte de un sistema robótico integral, por lo que el uso de la señal no requiere tarjetas de adquisición de datos ni procesamiento basado en PC. En su lugar, mediante una tarjeta integrada en el controlador se adquiere la señal que es procesada por el controlador para asociar la información a una variable del sistema. A esta variable se puede tener acceso mediante una sentencia que asigna el valor de ésta, a una variable que debe ser del tipo arreglo de seis elementos reales, donde los tres primeros elementos del arreglo son las fuerzas presentes en cada una de las direcciones del sistema coordenado del sensor, mientras que los tres restantes son los pares presentes en las mismas direcciones.

De acuerdo a lo anterior, se puede establecer que la frecuencia de muestreo es suficientemente alta para el sistema de guiado y, dado que éste es un dato que el proveedor del equipo no otorga, es necesario verificarlo. Es posible conocer la frecuencia de muestreo del sensor, mediante un programa en el lenguaje KAREL donde simplemente se imprima de forma continua el estado de esfuerzo del sensor y el valor de un cronómetro en un archivo de texto. Analizando los datos arrojados se puede obtener de manera preliminar el valor de la frecuencia de muestreo.

El cronómetro del sistema se actualiza cada 8 milisegundos y, dado que la tarea de imprimir en un archivo la información deseada le toma al intérprete fracciones de milisegundo, el resultado en el archivo texto es una lista de valores que se repiten por cierto periodo de tiempo y cada línea corresponde a un ciclo del programa. Es conocido que los valores del cronómetro se repiten durante 8 milisegundos, lo que dará la pauta para saber la

frecuencia de muestreo del sensor, con base en la cantidad de valores que se repitan del mismo. Los resultados de la prueba son los siguientes:

- Se realizan, en promedio, 41.88 ciclos cada 8 milisegundos, es decir que la frecuencia de ejecución del ciclo programado para esta prueba es de 5.236 kHz.
- La información del sensor permanece fija en promedio durante 21 ciclos del programa, o sea que el sensor muestrea cada 4 milisegundos o, lo que es lo mismo, tienen una frecuencia de muestreo de 0.25 kHz

Es importante considerar que durante la prueba no existen comandos de movimiento ni otras operaciones, además de los necesarios para imprimir la información deseada. Por tal motivo es necesario realizar pruebas en programas más complejos para ver de qué manera influyen en este aspecto.

4.2. Algoritmo de Runge-Kutta

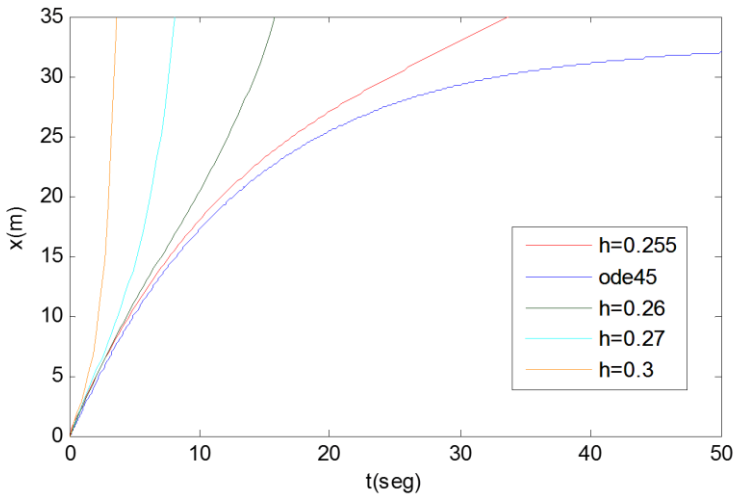


Figura 4.1. Comparación de algoritmo R-K segundo orden a diferentes pasos, con el ode45 de Matlab.

Es un método numérico iterativo utilizado para resolver ecuaciones diferenciales cuya convergencia depende, entre otros factores, del paso utilizado en la solución y de la docilidad del conjunto de ecuaciones diferenciales. El paso está relacionado con la variable independiente de la ecuación a resolver, es decir, al tiempo de nuestra ecuación masa-resorte-amortiguador. Como se había comentado en el capítulo anterior, el peligro de implementar este algoritmo con la posibilidad de utilizar un paso grande, implica movimientos del robot a gran velocidad en trayectos largos mientras que estos movimientos se ejecutan indiferentes ante la interacción. Por tal razón se debe verificar en qué rango se puede establecer el paso, asegurando la convergencia del algoritmo de integración numérica.

Para verificar la convergencia del algoritmo de R-K de segundo orden, se resuelve la ecuación diferencial (3.1) mediante el R-K de segundo orden (4.3 y 4.4), usando

diferentes pasos y con el ode45 de matlab. Para comparar el resultado, se resolvió el caso particular de un sistema sobreamortiguado: $M = 5,\ K = 1,\ B = 40,\ F = 10\ kgf$. En la figura 4.1 se ve cómo desde pasos mayores a 0.255 segundos, la solución por R-K se aleja del comportamiento correcto, representado por la solución del ode45.

Se repitió el experimento anterior para diferentes factores de amortiguamiento, manteniendo los parámetros de rigidez y masa constantes. Se notó que la capacidad del algoritmo varía dependiendo los valores de los parámetros de impedancia; mientras más alta la razón de amortiguamiento se requiere de pasos más pequeños. Por ejemplo, para sistemas subamortiguados con $\xi = 0.1$ es posible usar pasos de hasta 0.4 segundos y mientras aumenta ξ es posible usar pasos más grandes. Este comportamiento se mantiene hasta donde el sistema se hace críticamente amortiguado ($\xi = 1$), donde es posible usar el mayor paso de 0.9 segundos. A medida que se eligen sistemas sobreamortiguados, el paso deberá ser más pequeño hasta, por ejemplo, en $\xi = 10$ se requiere un paso de 0.12 segundos.

Se observó el comportamiento del algoritmo respecto a la razón de amortiguamiento, dado que éste es uno de los principales discriminantes del comportamiento del sistema dinámico, correspondiente a la ecuación diferencial que se está resolviendo. Sin embargo, es importante mencionar que la solución por medio de R-K de segundo orden es muy sensible ante cambios en el parámetro de masa, en especial, cuando se eligen parámetros de masa pequeños. De acuerdo a lo anterior, se debe utilizar un paso pequeño que garantice convergencia del algoritmo y que se ajuste a los requerimientos de la implementación en el sistema robótico. Es decir que, el paso sea similar al tiempo de procesamiento requerido para generar un movimiento.

4.3. Selección de parámetros de impedancia

En trabajos anteriores [65, 66], los parámetros de impedancia se habían seleccionado por medio del método de prueba y error, sin embargo, tal método resultaba complicado y peligroso. Es conveniente recordar que el control de impedancia cinemático se caracteriza por definir el comportamiento del efector final del robot, de acuerdo a una relación dinámica entre las fuerzas de interacción y la desviación en trayectoria que en éste se genera. Dicha relación dinámica es la que corresponde a un sistema idealizado masa-resorte-amortiguador. Las constantes de cada uno de los elementos del sistema se seleccionan de manera que se obtenga cierta respuesta deseada. A continuación se mencionan algunos conceptos mediante los cuales se pueden establecer pautas para la selección los valores de cada parámetro.

4.3.1. De acuerdo a la razón de amortiguamiento

El principal discriminante para el comportamiento de un sistema masa-resorte-amortiguador es la razón de amortiguamiento. Ésta distingue el comportamiento del sistema en tres posibilidades: subamortiguado, donde la razón de amortiguamiento tiene un valor menor a la unidad, sobreamortiguado, donde la razón de amortiguamiento tiene un valor mayor a la unidad y amortiguamiento crítico, donde la razón de amortiguamiento tiene un valor igual a la unidad.

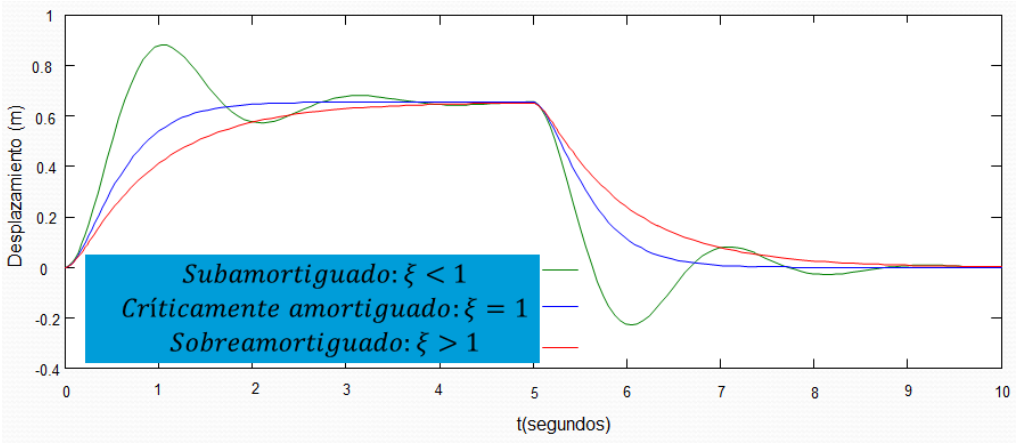


Fig. 4.2. Comportamiento de un sistema masa-resorte-amortiguador ante una fuerza constante aplicada durante 5 segundos para diferentes razones de amortiguamiento.

En la selección de parámetros de impedancia, modulando la masa, rigidez y amortiguamiento, es difícil obtener cierta razón de amortiguamiento, debido a que durante la selección de cada parámetro, ésta se modifica. Una estrategia más deseable es seleccionar la razón de amortiguamiento en lugar del parámetro de amortiguamiento. Como se puede ver en la figura 4.2, el sistema subamortiguado presenta oscilaciones indeseables, por lo que se considera que la opción más adecuada corresponde al de un sistema que puede ser críticamente amortiguado o sobreamortiguado. La razón de amortiguamiento se relaciona con los parámetros de impedancia del sistema de la siguiente manera:

$$\xi = \frac{B_d}{2\sqrt{K_d M_d}} \tag{4.1}$$

De donde se obtiene que la constante del amortiguador B_d está dada por:

$$B_d = 2\xi\sqrt{K_d M_d} \tag{4.2}$$

Se puede ver en (4.2), que el amortiguamiento B_d depende de la razón de amortiguamiento ξ, de la masa M_d y de la constante de rigidez del resorte M_d, de manera que al seleccionarlos, se calcula un amortiguamiento que se ajuste a la razón de amortiguamiento deseada.

4.3.2. De acuerdo a la masa o inercia

En física se dice que un sistema tiene más inercia cuando resulta más difícil lograr un cambio en el estado físico del mismo, es decir, que cuando menor sea la masa del sistema será más fácil alterar su estado de reposo o movimiento. En las ecuaciones (4.3 y 4.4), que son las respuestas del sistema obtenidas con Runge-Kutta de segundo orden, se puede observar que mientras M_d sea mayor, menor es el cambio en la respuesta del sistema x_a y $\dot{x}_a$. Por lo que, de acuerdo a lo anterior, se puede confirmar que valores de masa

pequeños son recomendables para obtener mayor docilidad, resaltando que cuando el valor de la masa se aproxima a cero el cambio en la respuesta tiende al infinito, lo que es un caso no deseado. El parámetro "b" se detalla en la sección 4.6.

$$x_a = x_0 + \tau b\left(\dot{x}_0 + \frac{\tau}{2M_d}(F_e - B_d\dot{x}_0 - K_d x_0)\right) \tag{4.3}$$

$$\dot{x}_a = \dot{x}_0 + \frac{\tau b}{M_d}\left\{F_e - B_d\left[\dot{x}_0 + \frac{\tau}{2M_d}(F_e - B_d\dot{x}_0 - K_d x_0)\right] - K_d\left(x_0 + \frac{\tau\dot{x}_0}{2}\right)\right\} \tag{4.4}$$

En la figura 4.3, se muestra como se modifica el comportamiento del sistema al variar la masa, manteniendo rigidez y la razón de amortiguamiento constantes ante un pulso de fuerza. Se puede notar como la respuesta del sistema es más rápida para valores de masa pequeños.

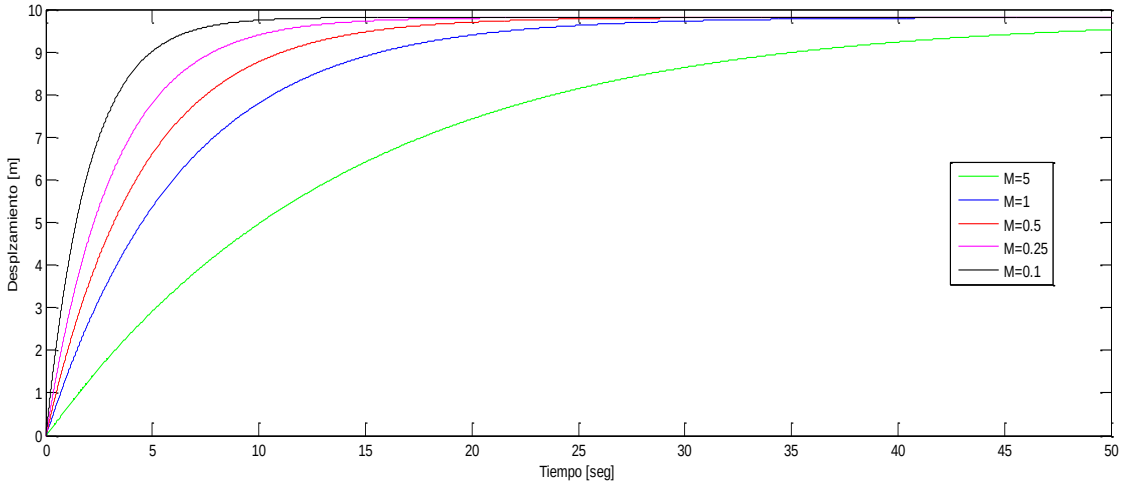


Figura 4.3. Respuesta del sistema masa-resorte-amortiguador para diferentes valores de masa. Rigidez y razón de amortiguamiento constantes.

4.3.3. De acuerdo a la rigidez

Si se mantienen contantes la razón de amortiguamiento y la masa del sistema, al modificar la rigidez, ésta influye en la rapidez del movimiento del sistema al generar desplazamientos pequeños en el caso de rigidez alta y desplazamientos grandes en el caso de rigidez baja, como se ve en la figura 4.4. Por lo que este parámetro se puede utilizar para modular la precisión del movimiento durante el proceso de guiado, pues se debe recordar que la intención del usuario se reproduce de forma más precisa si se ejecuta a baja velocidad, como es en el caso de sistemas teleoperados por joystick.

Es necesario establecer un estado de rigidez de referencia, donde lo que se busca es obtener una docilidad alta, mientras que se evita el caso en el que el movimiento del robot resulta peligroso para el usuario, debido a las velocidades que se pueden generar.

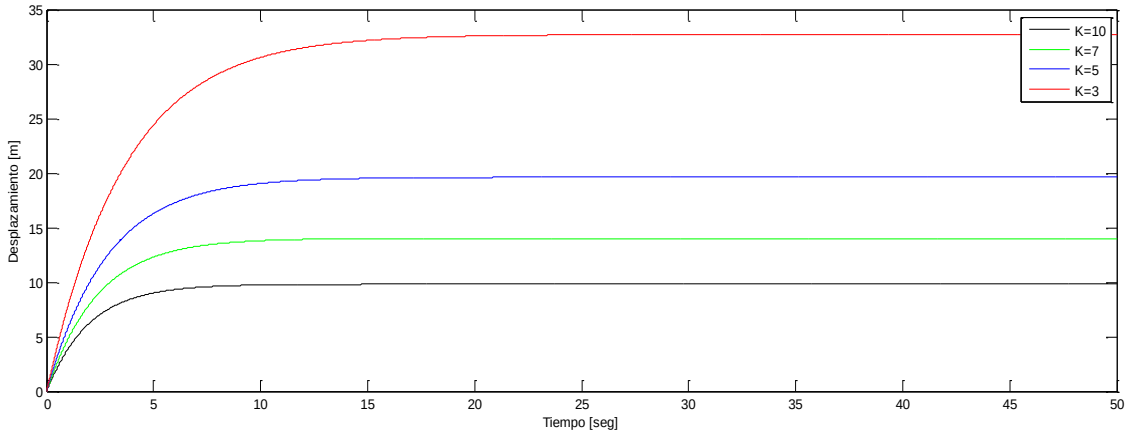


Figura 4.4. Respuesta del sistema masa-resorte-amortiguador para diferentes valores de rigidez, con masa y razón de amortiguamiento constantes, ante un pulso de fuerza.

4.3.4. Implementación de parámetros

Lo discutido hasta ahora sólo da pistas basadas en la teoría, acerca de en qué orden de magnitud deberían localizarse los valores de cada parámetro sin embargo, el entorno, en este caso el usuario del sistema de programación, no ofrece estímulos tan sencillos como los mostrados en las gráficas, por lo que la selección de los parámetros se realiza finalmente mediante experimentación, procurando permanecer dentro de los rangos que la teoría describe como favorables o seguros.

La implementación de los parámetros en el algoritmo se realizó en un programa que permite modificar los tres parámetros de razón de amortiguamiento, masa y rigidez durante la interacción, para identificar una disposición de éstos que sea favorable en tareas de guiado. El movimiento del robot se rige por 6 parámetros; tres están relacionados a la interacción en traslación y tres a la interacción en orientación. Sin embargo, una sintonización adecuada para traslación, es adecuada de manera homóloga a los parámetros de orientación. La razón de amortiguamiento, al ser adimensional, puede implementarse indistintamente en ambos casos y, mediante un factor de proporción, se pueden encontrar valores de inercia y rigidez homólogos para el movimiento en rotación. De esta manera la sintonización se reduce a la búsqueda de cuatro parámetros: razón de amortiguamiento, masa (inercia), rigidez y factor de proporción. Dado que el rango de movimiento en traslación es más amplio que en rotación, la sintonización se realizó dejando fija la orientación.

Como ya se había comentado, son preferibles sistemas que van de críticamente amortiguados a sobreamortiguados. Por tal motivo, la razón de amortiguamiento inicialmente se estableció en 1 para que, durante la etapa de experimentación, aumentar su valor gradualmente. En cuanto a los valores de masa y rigidez, se iniciaron con valores pequeños M=5 kg y K = 1 N/m, con el objetivo de obtener un sistema dócil sin causar problemas de convergencia en el algoritmo de R-K, por lo que se utilizó un paso de 0.01 segundos. Para extremar precauciones, la interacción se inició regulando la velocidad a un 25% y, una vez que se identificó un comportamiento seguro, se aumentó gradualmente la velocidad hasta el 100%.

Se realizó el experimento para diferentes valores de razón de amortiguamiento entre 1 y 20; en cada valor se modificaba la masa y la rigidez entre valores de 5 a 50 y 1 a 10, para en ese rango identificar una configuración que ofrezca docilidad y maniobrabilidad. Aunque este método depende de la experiencia y la apreciación del realizador, se logró obtener parámetros que otorgan un control que permitió, a usuarios sin experiencia, evaluar el funcionamiento del sistema. Los parámetros seleccionados son los siguientes: $\xi = 1$, $K = 3\,\frac{N}{m}$, $M = 15$ kg.

4.4. Sistema masa-amortiguador

Otra posibilidad es utilizar un sistema masa-amortiguador; en este caso no hay resorte, por lo que el robot no tendera a regresar hacia el punto de referencia, condición que puede ser favorable para un sistema de guiado. En este caso las ecuaciones (4.3 y 4.4) quedan de la siguiente forma.

$$x_a = x_0 + \tau b\left(\dot{x}_0 + \frac{\tau}{2M_d}(F_e - B_d\dot{x}_0)\right) \tag{4.5}$$

$$\dot{x}_a = \dot{x}_0 + \frac{\tau b}{M_d}\left\{F_e - B_d\left[\dot{x}_0 + \frac{\tau}{2M_d}(F_e - B_d\dot{x}_0)\right]\right\} \tag{4.6}$$

Como se ve en las ecuaciones (4.5 y 4.6) el sistema se sintoniza con la constante de masa y la de amortiguamiento. Para el caso de la masa aplica el mismo razonamiento expresado arriba, y el amortiguamiento puede elegirse de acuerdo a la velocidad operación deseada. En la implementación se usaron los mismos valores encontrados para el sistema masa-resorte-amortiguador. El parámetro “b” se detalla en la sección 4.6.

4.5. Pruebas de desempeño de la interacción

Se realizaron pruebas de interacción Humano-Robot al sistema de programación para la parte de traslación. Los datos recabados son: el tiempo y la posición y fuerzas en los ejes x, y, z del sistema WORLD para cada iteración. Durante la prueba el usuario realiza un movimiento arbitrario dentro del área de trabajo del robot sujetando el sensor.

Del análisis de los datos obtenidos durante el experimento se puede mencionar que, en cada iteración, el sensor muestra lecturas diferentes, es decir, que el muestreo del sensor es igual o más rápido que la ejecución de cada iteración. En promedio cada iteración dura aproximadamente 6 milisegundos. Lo que comprueba que el procesamiento no afecta la frecuencia de muestreo del sensor, que previamente se había calculado de aproximadamente 4 milisegundos.

Se pudo comprobar, por un lado, que el control de impedancia en efecto generaba la trayectoria deseada por el operador. Además, se comparó gráficamente la trayectoria generada por el robot y la generada por el sistema dinámico resuelto en una PC ante las mismas fuerzas de interacción. A continuación presentan las gráficas donde se aprecia que ambas curvas son muy similares.

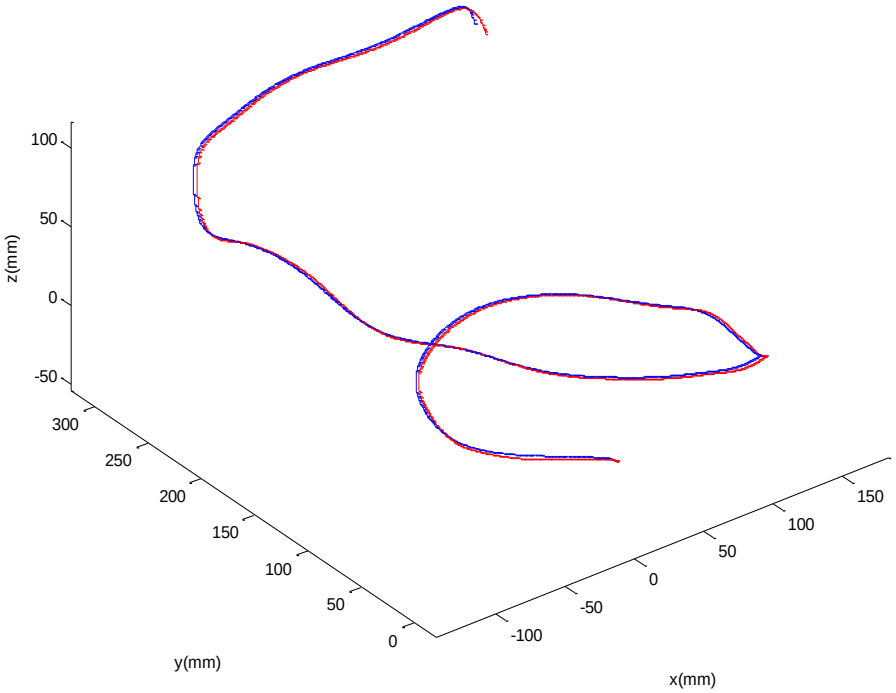


Figura 4.5. Comparación de trayectorias, en azul se muestra la trayectoria que siguió el robot y en rojo se muestra la trayectoria que de acuerdo a la dinámica establecida en el controlador, el robot debiera seguir.

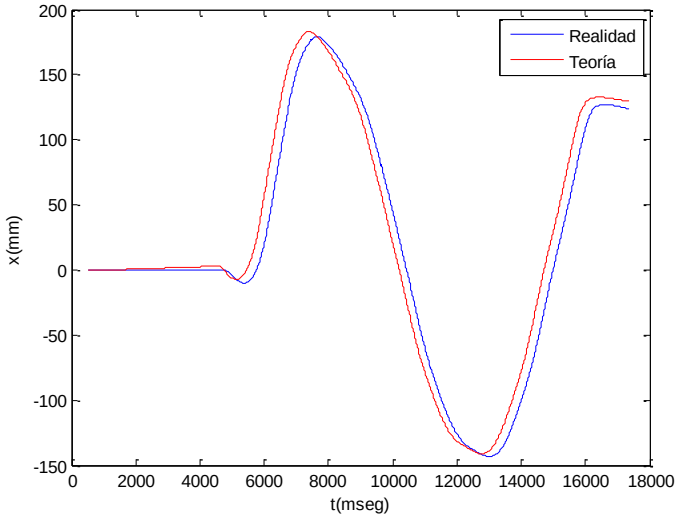


Figura 4.6. Comparación de posición vs tiempo, para un grado de libertad cartesiano.

La variación encontrada en las gráficas puede atribuirse al tipo de terminación del movimiento en el robot denominado "NODECEL", que causa imprecisiones ya que este tipo de terminación no garantiza que la herramienta llegue a la posición comandada. Sin embargo, este tipo de terminación es útil debido a que propicia movimientos más suaves del robot, dado que se eliminan las etapas de desaceleración-aceleración entre cada punto.

4.6. Eliminación de movimientos bruscos

Durante la selección de parámetros se entra en un conflicto entre docilidad y precisión. Es decir, cuando se establecen parámetros en el sistema de manera que éste responda rápidamente ante la interacción, se pierde la capacidad de ubicar la punta de la herramienta con precisión. El efecto que se obtiene cuando se pretende ubicar la herramienta en un punto bien definido, operando el sistema dócilmente, son una serie de oscilaciones alrededor de este punto, que pueden llegar a causar movimientos bruscos si el usuario se empeña en detener las oscilaciones. Esto ocurre debido a que en el instante que el operador trata de detener el movimiento se genera un cambio de fuerza muy grande que genera como respuesta una velocidad y desplazamiento grande.

Se propuso atacar este problema mediante tres estrategias:

1. Estableciendo límites superiores e inferiores a la señal del sensor.
2. Aplicando un filtro que arroja el promedio la medición actual más un paquete de mediciones anteriores consecutivas.
3. Aplicando una condición que solo permite respuesta cuando la diferencia entre dos lecturas consecutivas permanece bajo cierto valor máximo [63].

Se aplicaron las tres estrategias en código y, mediante una evaluación de desempeño que consistía en buscar generar las condiciones en que se produce el evento no deseado para identificar que tanto se lograba mitigar este efecto, se llegó a la conclusión de que la tercera opción presentó mejor desempeño. A continuación se muestra en la figura 4.7 la forma en que se implementó.

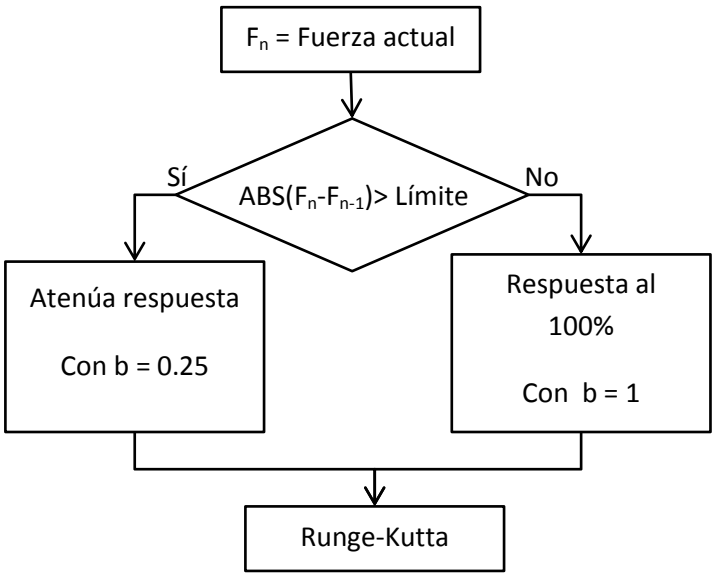


Figura 4.7 Diagrama de flujo de la implementación de la condición para eliminar movimientos bruscos.

El parámetro "b" multiplica en Runge-Kutta como se puede ver en las ecuaciones (4.3-4.6), atenuando la respuesta en el instante que se generó un ΔF grande.

4.7. Pruebas finales

Se realizaron pruebas de uso del sistema de programación con 10 personas, hombres y mujeres en edades entre 16 y 29 años con el objetivo de obtener una evaluación del sistema y además de retroalimentación por parte de los participantes, la cual será considerada en la búsqueda de mejoras al sistema de programación.

4.7.1. Tareas

El sistema se evaluó con respecto del sistema convencional (vía terminal de enseñanza TE), ya que éste representa el sistema más utilizado en la actualidad. Por tanto, las pruebas consisten en indicar a cada participante que realice la misma tarea con ambos sistemas y medir cuanto tiempo le tomó en cada caso.

Las tareas asignadas fueron 2:

1. Manipulación de objetos.
   Consiste en llevar cuatro objetos de diferente forma y dimensiones desde cierta posición en un estante (P1), donde se encuentran ensambladas, hacia otras posiciones en un estante (P2), donde han de ensamblar, figura 4.8.
2. Realización de un programa.
   Consiste en realizar un programa que lleve uno de los objetos desde su posición en P1 hasta su posición P2.

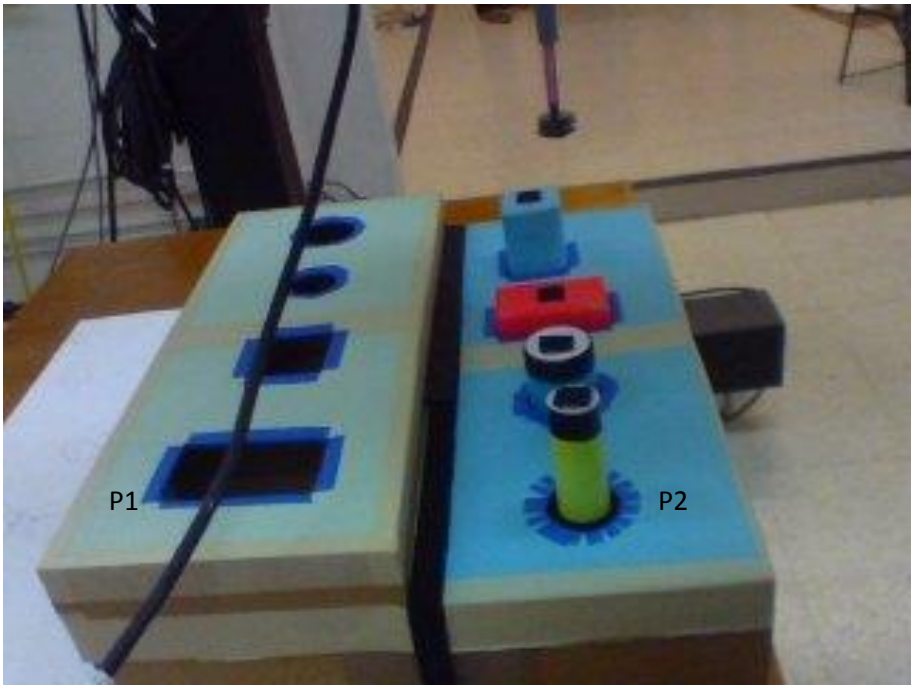


Figura 4.8. Plataforma de experimentación.

Ya que la mayoría de los participantes tenían poco o nulo conocimiento sobre robots, a cada participante se le dio una breve descripción del sistema robótico. Después se

les dieron los conocimientos necesarios para poder realizar la tarea 1 que consiste solo en manipulación de objetos. Tras haber realizado la tarea 1, se les instruyó en lo necesario para realizar la tarea 2, la cual requería manipulación de objetos a la vez que se debía generar un programa funcional para que el robot lo pudiese ejecutar.

4.7.2. Método de evaluación

Hay literatura [61] donde se evalúa el nivel de estrés al que se someten los operadores que trabajan en contacto o cerca de equipo robótico en movimiento. Una de las técnicas más sencillas de aplicar es mediante cuestionarios diferencial semánticos [62]. Con esta prueba se evaluó el nivel de estrés además de obtener información acerca de que tan intuitivo les había resultado cada sistema. También, con el fin de obtener información más detallada a cada uno de los participantes, se le instó a dejar sus comentarios y sugerencias. Se cuidó que el cuestionario fuera breve para evitar el tedio en los participantes, que ocasiona respuestas no meditadas.

Según [61], mediante las tres emociones temor, sorpresa e incomodidad, se puede evaluar que tanto estrés genera la interacción con un equipo robótico (Parte 1 de la encuesta). Por otro lado, para evaluar la opinión de los participantes acerca de que tan natural (intuitivo) percibieron el sistema, se les pregunta que tan fácil resultó el uso del sistema y que tan rápido fue su adaptación, es decir, que tan rápido se familiarizaron con el sistema (Parte 2 de la encuesta).

A continuación se muestran las preguntas realizadas en la encuesta la cual se compone de dos partes. La primera se utilizó para evaluar el estrés y la segunda para evaluar qué tan intuitivo les pareció cada sistema de programación.

Parte 1:

Indique en qué nivel las siguientes emociones, describirían su experiencia durante la tarea asignada para el sistema.

Temor:

Nada ⓪ ① ② ③ ④ ⑤ Mucho

Sorpresa:

Nada ⓪ ① ② ③ ④ ⑤ Mucho

Incomodidad:

Nada ⓪ ① ② ③ ④ ⑤ Mucho

Parte 2:

¿Cómo describiría el uso del sistema de programación?

Fácil ⓪ ① ② ③ ④ ⑤ Difícil

Considera que el tiempo para familiarizarse con el uso del sistema resulto:

Corto ⓪ ① ② ③ ④ ⑤ Largo

4.7.3. Resultados

En la tabla 1, se muestran los resultados de promedio y desviación estándar. El parámetro estrés se obtiene de la suma de los resultados de temor, sorpresa e incomodidad para cada sujeto, bajo una escala de 0 a 15, donde 0 es nada y 15 es mucho. De manera similar para naturalidad se suma facilidad y adaptación, donde la escala es: 0 muy Natural o intuitivo y 15 es poco intuitivo.

Se realizaron pruebas de significancia estadística, para los resultados entre guiado manual contra guiado TE, mediante la prueba estadística t-student. En la tabla 2, se presentan los resultados, donde usualmente para valores de $p < 0.1$ [64], se puede decir que hay una diferencia significativa entre ambos grupos de datos.

Tabla 1: Resultados de las pruebas (promedios y desviaciones estándar)

Parámetro	Guiado Manual		Guiado TE	
	Promedio	Desv. Std.	Promedio	Desv. Std.
Temor	1.2	1.14	1	1.6
Sorpresa	2.8	1.4	1.41	1.07
Incomodidad	0.9	0.88	2	2.11
Estrés(0-15)	4.9	2.64	4.6	3.92
Facilidad	0.3	0.48	1.7	1.34
Adaptación	1.3	0.82	2	1.76
Naturalidad (0-15)	1.6	0.84	3.7	3.02
Rapidez T1 [seg]	230.5	64.79	482.1	103.63
Rapidez T2 [seg]	117.7	35.83	177.3	37.99

Tabla 2: Resultados de las pruebas de significancia estadística

Parámetro	p
Temor	0.73
Sorpresa	0.04**
Incomodidad	0.14
Estrés	0.84
Facilidad	0.006*
Adaptación	0.27
Naturalidad	0.04*
Rapidez T1	0.000004*
Rapidez T2	0.002*

*hay una diferencia estadísticamente significativa a favor del guiado manual
** hay una diferencia estadísticamente significativa a favor del guiado TE

A continuación, por medio de gráficas se muestran los resultados para cada tarea. Las barras representan la media de cada grupo de datos.

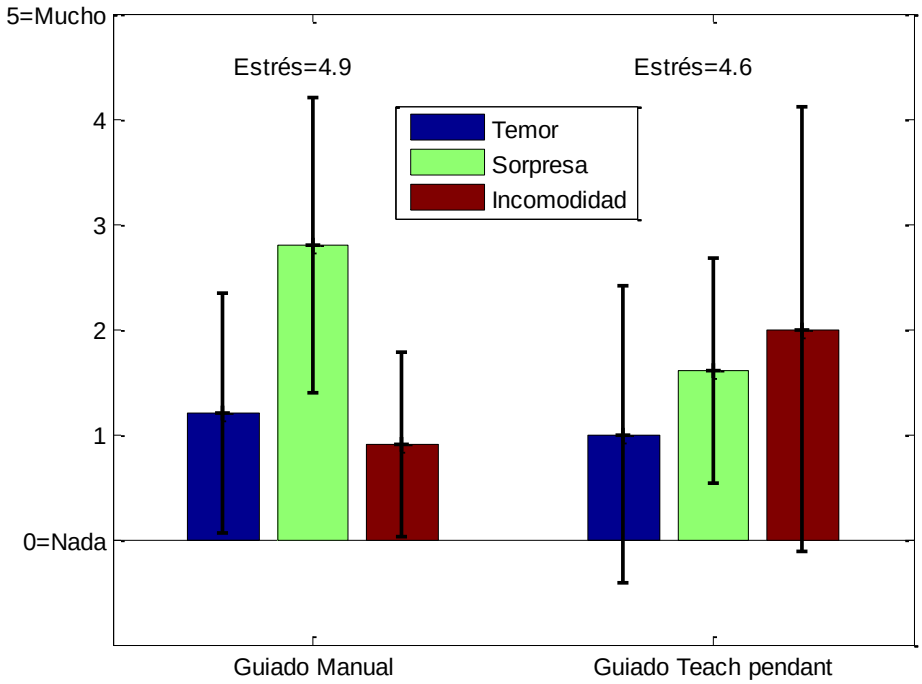


Figura 4.9. Nivel de estrés: El dato estrés es la suma de los tres parámetros (temor sorpresa e incomodidad).

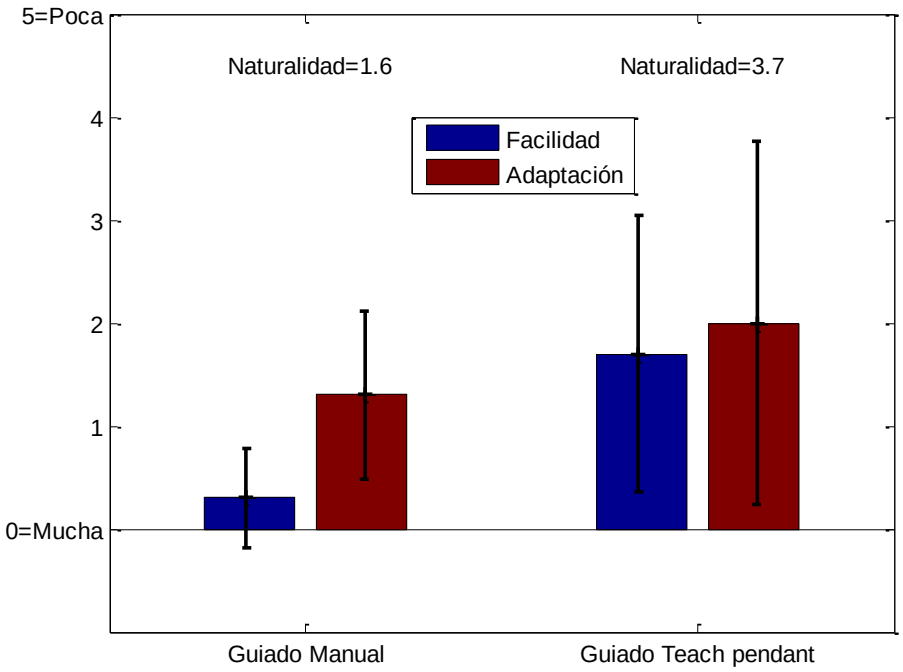


Figura 4.10. Nivel de Naturalidad.

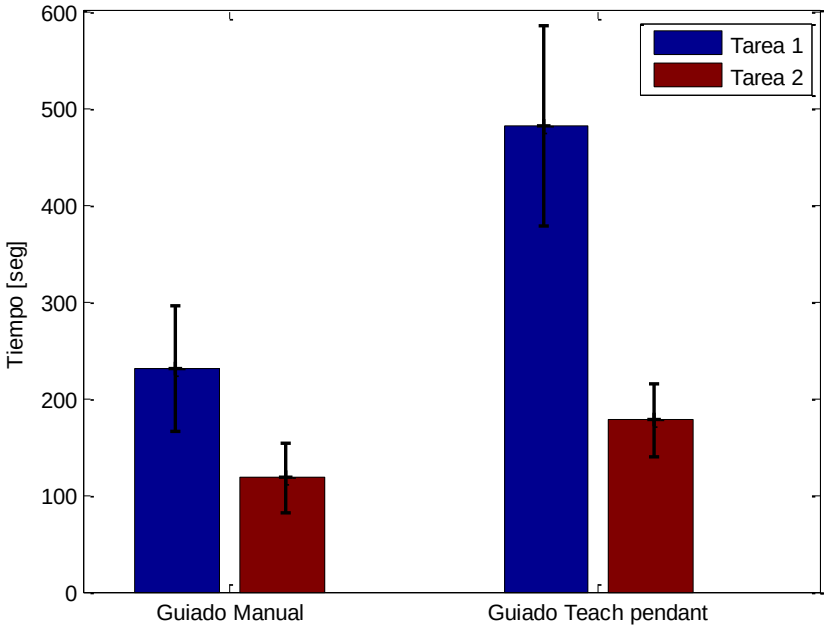


Figura 4.11. Resultados del tiempo de ejecución para ambas tareas.

4.8. Conclusiones del capítulo

Los resultados de las pruebas preliminares realizadas permitieron albergar la posibilidad de someter dicho sistema a experimentación con personas no familiarizadas con sistemas robóticos industriales, teniendo en mente el objetivo de obtener retroalimentación por parte de los participantes la cual será considerada en la búsqueda de mejoras al sistema de programación. Además, está la posibilidad de considerar el estado actual del desarrollo del sistema como un requerimiento para implementar un sistema de programación de aprendizaje vía demostración.

Finalmente, se logró el objetivo de la tesis al haber implementado el sistema de programación de manera aceptable. Aunque el sistema presenta varias áreas de oportunidad, se puede decir que de manera preliminar cumple con su cometido. Los resultados de las pruebas demuestran contundentemente, en cuanto a tiempo y sencillez, que el sistema desarrollado da una ventaja considerable respecto del sistema convencional. En cuanto al nivel de estrés, como se esperaba, el sistema desarrollado genera niveles altos, debido a que el usuario está en contacto con el robot mientras éste está en movimiento, sin embargo, se probó estadísticamente que no hay diferencia entre el estrés causado con el sistema de guiado manual y el de TE.

Por otro lado, las pruebas hechas al sistema de programación ratificaron su funcionalidad ya que mediante el sistema se logró programar trayectorias a través de puntos definidos de manera sencilla debido a que el comportamiento del robot resultó estable y mucho menos complejo que los sistemas tradicionales.

Conclusiones

Se logró establecer un sistema funcional de programación por guiado activo, en un sistema robótico de arquitectura cerrada, mediante el esquema de control cinemático de impedancia. Esto fue logrado gracias a que fue posible identificar y aprovechar las capacidades que el equipo robótico proporciona.

El sistema desarrollado permite crear trayectorias definidas por curvas básicas (líneas y arcos de circunferencia), que pasan a través de puntos en el espacio cartesiano, seleccionables por el usuario mediante una terminal de comunicación, de uso sencillo, que consta de 4 botones. El robot es guiado por el usuario hacia las posiciones deseadas mediante interacción física con una mano mientras que, con la otra mano, se sujeta la terminal de comunicación para presionar el botón de hombre muerto, acondicionado para oprimirse con cualquiera de los dedos exceptuando el pulgar, el cual queda libre para presionar cada uno de los botones dispuestos para la administración de la interacción y la selección de trayectorias. La administración de la interacción se realiza mediante tres parámetros que modifican el comportamiento del control del robot. Estos permiten establecer condiciones que facilitan el posicionamiento del robot en el lugar y orientación requeridos.

El sistema se diseñó con la posibilidad de aplicarse tanto a procesos industriales como de servicio. Desde el punto de vista del usuario, las características del sistema permiten que éste sea utilizado por personas sin experiencia en programación, como es el caso del uso de robots en el sector servicio, particularmente en la enseñanza de terapias de rehabilitación. En aplicaciones industriales, el uso del sistema reduce la necesidad de adiestramiento y el tiempo de programación. Si el sistema se analiza con base en las capacidades del mismo, pareciera estar limitado a la generación de trayectorias básicas, sin embargo, el programa generado puede editarse vía el sistema convencional de programación o mediante sistemas fuera de línea, para agregar atributos necesarios en aplicaciones industriales. De este modo, el sistema se puede considerar de uso general, estableciendo una manera sencilla de indicar los movimientos necesarios para una tarea a realizar por medio de un robot.

La generación de trayectorias por medio de líneas y arcos de círculo, a través de puntos definidos por el usuario, permite eliminar en buena medida la inconsistencia humana, ya que para el humano es imposible generar trayectorias perfectamente rectas o que describan círculos. Con el sistema propuesto el usuario se limita a seleccionar los puntos característicos y trayectoria que seguirá el robot a través de ellos. De esta manera, el sistema resulta conveniente para su empleo en procesos industriales, donde las trayectorias usualmente han de reproducirse en piezas de formas regulares.

La resolución de los movimientos del robot a la mínima velocidad es de 0.02 mm, que consiste en un movimiento que el ser humano no alcanza a detectar visualmente por lo que la precisión con que se definan las trayectorias recae en la apreciación y el tacto del usuario. Éste, mediante su sentido de la vista, definirá cuándo el robot ha llegado a la posición requerida. La tarea de ubicar el robot con precisión se hace más fácil a menor velocidad y éste es uno de los parámetros que se controla en la administración de la interacción. Mediante un factor se modifican los parámetros de impedancia para que, durante las etapas de posicionamiento preciso, sea posible obtener parámetros de impedancia mayores, reduciendo la velocidad de la interacción. Durante las etapas que no se requiere posicionamiento preciso los parámetros se reducen para aumentar la velocidad de interacción y así reducir el tiempo necesario para mover el robot en trayectos largos.

Una problemática importante en la implementación de equipos robóticos, radica en los costos que éstos representan. Por tal motivo, el diseño de los dispositivos requeridos para la implementación como son: la terminal de comunicación y el portasensor, se realizó con especial atención a la reducción de costos, sin demerito del cumplimiento de su función. Además, se presenta como alternativa la implementación de interacción por medio de contacto la cual permite sustituir el sensor de fuerza por un dispositivo más económico, constituido por interruptores para identificar la dirección de la intención del usuario.

En el uso e implementación de sistemas robóticos, aún más crítico que el costo resulta el tema de la seguridad, tanto para el usuario como para el equipo. En ese contexto fue necesario implementar alternativas a las medidas de seguridad con la que cuenta el robot. En el caso de la terminal de comunicación, ésta se diseñó con un botón de hombre muerto, de manera que el usuario debe mantenerlo presionado para habilitar el funcionamiento del sistema, haciendo más fácil deshabilitarlo en caso de una situación de peligro. Además, se agregaron botones de paro de emergencia, dispuestos en el área de trabajo al alcance del usuario durante una interacción. En cuanto a la seguridad del equipo, el diseño del portasensor se realizó con la posibilidad de un desmontaje rápido, para hacer más rápidas las transiciones de enseñanza a pruebas de ejecución, evitando la posibilidad de colisiones con el sensor. Para las pruebas, se implementó una herramienta retráctil de plástico, la cual permite errores durante la aproximación de la misma a los objetos a manipular. De esta manera los usuarios lograron aprender durante la misma prueba, sin dañar la herramienta o las piezas a manipular, debido a los errores iniciales propios de una primera experiencia.

En esta tesis se presentó la investigación que se realizó para obtener el conocimiento para lograr el objetivo planteado. Con este propósito se realizó una revisión de los avances en el desarrollo de los sistemas de programación, además se presentaron las diferentes clasificaciones de dichos sistemas, lo que hace posible identificar el sistema de programación del presente trabajo como un sistema de guiado activo por medio de interacción física, que además se caracteriza por ser un sistema de programación automática y constituye un paso importante en el desarrollo de un sistema de programación por demostración, uno de los niveles más altos en programación de robots industriales.

Uno de los tópicos más importantes en este tipo de estrategias es la manera en que la persona interacciona con el robot, para guiarlo a través de la posiciones requeridas, ya que esto definirá que tan intuitivo el sistema resulta. De este modo se abordaron los

aspectos concernientes a los avances en interacción humano-robot, con especial atención a la interacción física, la cual es puesta en práctica en el presente trabajo de tesis. La información presentada permite entender las implicaciones que existen en el desarrollo de sistemas robóticos, al describir los posibles escenarios y agruparlos de acuerdo a variados enfoques. En la robótica no se puede prescindir del planteamiento de interacción; ya sea con el entorno, otros robots o con humanos. Por tal motivo el uso o implementación de nuevas estrategias para la incorporación de estos equipos a la resolución de cualquier tarea, requiere el análisis de una gran cantidad de factores que, de no guiarse en alguna estructura, se corre el riesgo de pasar por alto consideraciones importantes. La principal problemática se encuentra en la comunicación, dadas las reducidas capacidades de los equipos robóticos para comunicarse y recibir información de los usuarios, en comparación con la comunicación interpersonal. Esto obliga al usuario a adquirir cierto nivel de conciencia de lo que ocurre o va a ocurrir en determinada situación. Por su parte, el desarrollador debe tener pleno conocimiento de las capacidades del equipo y, de acuerdo con el uso que se dará del mismo, definir la metodología a seguir para generar una aplicación funcional.

En este trabajo se ha presentado la metodología seguida para implementar un sistema de guiado activo para un robot de modelo específico. Sin embargo, bajo pequeñas variaciones en las problemáticas que se puedan presentar. Como puede ser: que el sistema robótico no cuente con un sensor propio de la marca, diferencias en las capacidades de procesamiento del controlador y diferencias en capacidades del lenguaje de alto nivel propio del modelo, entre otras. Una metodología similar puede aplicarse a otros modelos que se encuentren disponibles comercialmente, con un nivel de desarrollo tecnológico similar debido que se han identificado dispositivos y software genéricos que favorecen la portabilidad de este tipo de desarrollos.

Los resultados de las pruebas preliminares realizadas permitieron albergar la posibilidad de someter dicho sistema a experimentación con personas no familiarizadas con sistemas robóticos industriales, teniendo en mente el objetivo de obtener retroalimentación por parte de los participantes, aspecto que será considerado en la búsqueda de mejoras al sistema de programación. Además, está la posibilidad de considerar el estado actual del desarrollo del sistema como un requerimiento para implementar un sistema de programación de aprendizaje vía demostración.

Finalmente, se logró el objetivo de la tesis al haber implementado el sistema de programación de manera exitosa. Aunque el sistema presenta varias áreas de oportunidad, se puede decir que de manera preliminar cumple con su cometido. Los resultados de las pruebas demuestran contundentemente que, en cuanto a tiempo y sencillez, el sistema desarrollado da una ventaja considerable con respecto al sistema convencional. En cuanto al nivel de estrés, como se esperaba, el sistema desarrollado genera niveles altos, debido a que el usuario está en contacto con el robot mientras éste está en movimiento. Sin embargo, se probó estadísticamente que no hay diferencia entre el estrés causado con el sistema de guiado manual y el que se tiene al utilizar el medio tradicional, que consistente del uso de una terminal de enseñanza.

Apéndices

A. Cinemática del Robot

A.1. Cinemática directa

Para este proceso es necesario tener la información dimensional del robot. En la figura A.1 se muestra un esquema que permite conocer la longitud de cada eslabón del robot.

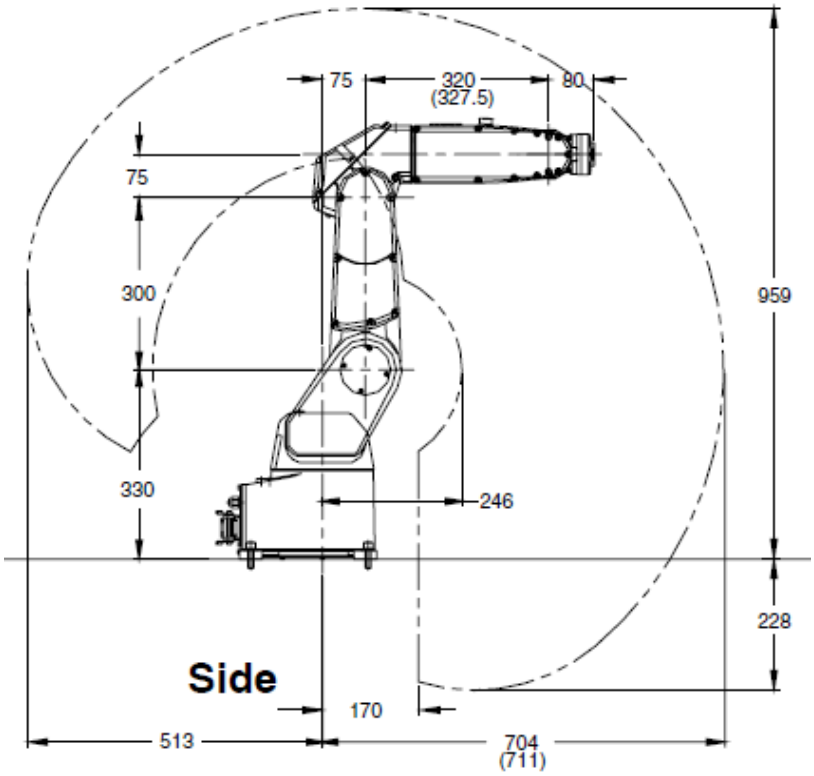


Figura A.1. Robot FANUC LR Mate 200iC (dimensiones en mm, las dimensiones en paréntesis corresponden a otro modelo).

Con esta información se eligió la posición de cada marco de referencia, correspondiente a cada junta, de acuerdo a la convención de Denavit-Hartenberg (D-H) [68]. En la figura A.2 se muestra lo mencionado.

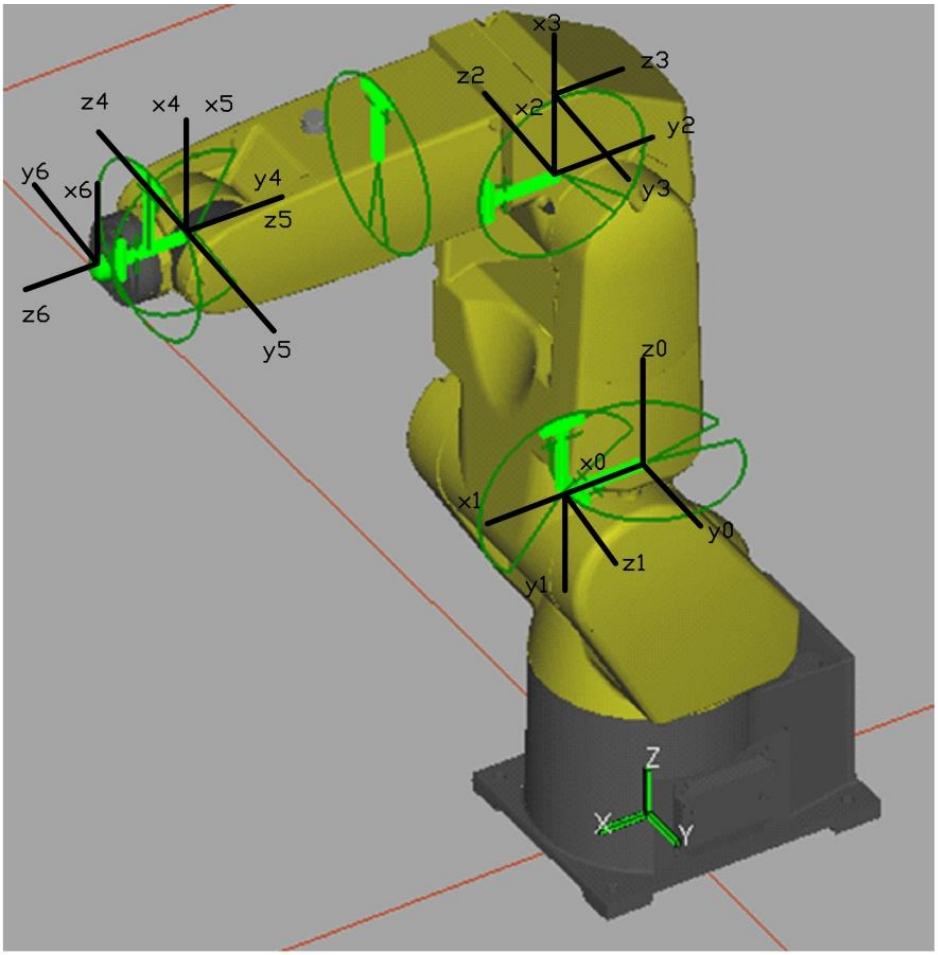


Figura A.2. Ubicación de los marcos de referencia para el Robot FANUC LR Mate 200iC.

Tabla A.1. Parámetros de (D-H)				
Junta	Θ(rad)	d(mm)	a(mm)	α(rad)
1	Θ_1	0	75	-π/2
2	Θ_2-π/2	0	300	π
3	Θ_3+ Θ_2	0	75	-π/2
4	Θ_4	-320	0	π/2
5	Θ_5	0	0	-π/2
6	Θ_6	-80	0	π

Con los parámetros que se muestran en la tabla A.1 se obtienen las matrices A_i, para i=1, 2…6 (grados de libertad del robot), donde:

$$A_i = \begin{pmatrix} \cos(\theta_i) & -\sin(\theta_i)\cdot\cos(\alpha_i) & \sin(\theta_i)\cdot\sin(\alpha_i) & a_i\cdot\cos(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i)\cdot\cos(\alpha_i) & -\cos(\theta_i)\cdot\sin(\alpha_i) & a_i\cdot\sin(\theta_i) \\ 0 & \sin(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{(A.1)}$$

Y la posición del efector final del manipulador con respecto a su base está dado por: $x_0 = Tx_6$.

$$T = A_1A_2A_3A_4A_5A_6 \quad \text{(A.2)}$$

Si se considera que la orientación (w, p, r) está dada respecto de un marco de referencia cero (sistema $x_0y_0z_0$). Entonces la orientación se obtiene por medio de los ángulos de Euler 3-2-1, es decir.

$$T = \begin{bmatrix} R & d \\ \mathbf{0} & 1 \end{bmatrix} \quad \text{(A.3)}$$

donde:

$$R = \begin{pmatrix} \cos(p)\cdot\cos(r) & \sin(p)\cdot\cos(r)\cdot\sin(w) - \sin(r)\cdot\cos(w) & \sin(w)\cdot\sin(r) + \sin(p)\cdot\cos(r)\cdot\cos(w) \\ \cos(p)\cdot\sin(r) & \cos(r)\cdot\cos(w) + \sin(p)\cdot\sin(r)\cdot\sin(w) & \sin(p)\cdot\sin(r)\cdot\cos(w) - \sin(w)\cdot\cos(r) \\ -\sin(p) & \cos(p)\cdot\sin(w) & \cos(p)\cdot\cos(w) \end{pmatrix} \quad \text{(A.4)}$$

De tal manera que la orientación se puede obtener mediante la función atan2 como se muestra a continuación.

$$w = atan2(R_{3,3}, R_{3,2}) \quad \text{(A.5)}$$

$$p = atan2\left[\sqrt{(R_{1,1})^2 + (R_{2,1})^2}, -R_{3,1}\right] \quad \text{(A.6)}$$

$$r = atan2(R_{1,1}, R_{2,1}) \quad \text{(A.7)}$$

Con lo anterior se ha resuelto la cinemática directa del Robot FANUC LR Mate 200iC.

Durante el desarrollo de la cinemática directa se identificó un comportamiento particular en el modelo FANUC LR Mate 200iC, que quizá esté presente en todos los modelos Fanuc, el cual consiste en un acoplamiento de la junta 2 con la 3, generando que al girar la junta 2 cierto ángulo, la junta 3 girará la misma cantidad en sentido opuesto. Éste efecto se conoce como movimiento en paralelogramo.

Para comprobar que el procedimiento anterior esta correcto se elaboró un programa en el software Mathcad donde se verificó concordancia entre los datos obtenidos del programa y los obtenidos del robot.

A.2. Cinemática inversa

Para la cinemática inversa, considere que se conoce $d\ y\ R$, la posición y orientación del efector final, y por medio de (A.3) se conoce la matriz "T".

De (A.1) se obtuvieron las matrices A_i: con Θ32=Θ3+Θ2

$$A_1 = \begin{pmatrix} \cos(\theta 1) & 0 & -\sin(\theta 1) & a1\cdot\cos(\theta 1) \\ \sin(\theta 1) & 0 & \cos(\theta 1) & a1\cdot\sin(\theta 1) \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad A_2 = \begin{pmatrix} \sin(\theta 2) & -\cos(\theta 2) & 0 & a2\cdot\sin(\theta 2) \\ -\cos(\theta 2) & -\sin(\theta 2) & 0 & -a2\cdot\cos(\theta 2) \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$A_3 = \begin{pmatrix} \cos(\theta 32) & 0 & -\sin(\theta 32) & a3\cdot\cos(\theta 32) \\ \sin(\theta 32) & 0 & \cos(\theta 32) & a3\cdot\sin(\theta 32) \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad A_4 = \begin{pmatrix} \cos(\theta 4) & 0 & \sin(\theta 4) & 0 \\ \sin(\theta 4) & 0 & -\cos(\theta 4) & 0 \\ 0 & 1 & 0 & d4 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$A_5 = \begin{pmatrix} \cos(\theta 5) & 0 & -\sin(\theta 5) & 0 \\ \sin(\theta 5) & 0 & \cos(\theta 5) & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad A_6 = \begin{pmatrix} \cos(\theta 6) & \sin(\theta 6) & 0 & 0 \\ \sin(\theta 6) & -\cos(\theta 6) & 0 & 0 \\ 0 & 0 & -1 & d6 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Considere

$$U_n = A_n A_{n+1} \dots A_6 \tag{A.8}$$

Observando (A.2) y (A.8) se nota que:

$$U_1 = T \tag{A.9}$$

Luego U_n, para $n > 1$, se puede obtener con (A.10):

$$U_n = (A_{n-1})^{-1} U_{n-1} \tag{A.10}$$

La idea presentada en [69], consiste en de (A.10) para cada n desde 2 a 6 buscar elemento por elemento una o dos relaciones que solo dependan de una o dos variables. De cada una de estas ecuaciones se puede resolver un valor de junta. Por ejemplo para $n = 2$, se tiene:

$$A_2 A_3 A_4 A_5 A_6 = (A_1)^{-1} T \tag{A.11}$$

Si se desarrolla (A.11), se nota que el lado derecho solo depende de θ_1, por lo que en el lado izquierdo se han de buscar elementos que permitan obtener una ecuación o sistema de ecuaciones de la que se pueda despejar θ_1.

De (A.11) se obtiene: las ecuaciones (A.12) y (A.13)

$$-\sin(\theta 1)\, T_{1,3} + \cos(\theta 1)\, T_{2,3} = \sin(\theta 4)\sin(\theta 5) \tag{A.12}$$

$$-\sin(\theta 1)\, T_{1,4} + \cos(\theta 1)\, T_{2,4} = -d6\sin(\theta 4)\sin(\theta 5) \tag{A.13}$$

Al combinar (A.12) y (A.13) obtiene:

$$\sin(\theta 1)\left(d6\, T_{1,3} + T_{1,4}\right) - \cos(\theta 1)\left(d6\, T_{2,3} + T_{2,4}\right) = 0 \tag{A.14}$$

Que es una ecuación de la forma:

$$\mathrm{A}\sin(\theta) + B\cos(\theta) + C = 0 \tag{A.15}$$

Cuya solución es:

$$\theta = 2\,\mathrm{atan}\left(\frac{-A \pm \sqrt{A^2 + B^2 - C^2}}{C - B}\right) \tag{A.16}$$

De manera que $\theta 1$ se obtiene con (A16) sustituyendo: $A = \left(d6\, T_{1,3} + T_{1,4}\right)$, $B = -\left(d6\, T_{2,3} + T_{2,4}\right)$, y $C = 0$

Ahora $\theta 1$ y U_2 son conocidas.

Se repitió el mismo procedimiento para $n = 3$, sin éxito ya que $U_3 = (A_2)^{-1}U_2$ no generó ecuaciones que dependieran únicamente de $\theta 2$. Así mismo para $n = 4$, con $U\_4 = (A_3)^{-1}(A_2)^{-1}U_2$, no se logró encontrar un sistema de ecuaciones que dependiera únicamente de $\theta 2\ y\ \theta 3$.

Para resolver $\theta 2\ y\ \theta 3$, se optó por aplicar el método de desacoplamiento dinámico, lo que es posible ya que el modelo del robot, tiene la característica de que sus últimos tres ejes se intersectan en un punto llamado centro de la muñeca. Lo que hace que la posición del TCP del robot dependa sólo de $\theta 1, \theta 2\ y\ \theta 3$, si se elimina la traslación del eslabón seis, postmultiplicando ambos lados de (A.9) por $(A_6)^{-1}$.

De lo anterior se obtiene el siguiente sistema de tres ecuaciones:

$$\cos(\theta 1)\cdot\left(a_1 + a_2\cdot\sin(\theta 2) - a_3\cdot\sin(\theta 3) - d_4\cdot\cos(\theta 3)\right) = T_{1,4} + T_{1,3}\cdot d_6 \tag{A.17}$$

$$\sin(\theta 1)\cdot\left(a_1 + a_2\cdot\sin(\theta 2) - a_3\cdot\sin(\theta 3) - d_4\cdot\cos(\theta 3)\right) = T_{2,4} + T_{2,3}\cdot d_6 \tag{A.18}$$

$$a_2\cdot\cos(\theta 2) + a_3\cdot\cos(\theta 3) - d_4\cdot\sin(\theta 3) = T_{3,4} + T_{3,3}\cdot d_6 \tag{A.19}$$

Ya que el valor de Θ1 es conocido, se verificó que (A.17) y (A.18) son la misma ecuación. Para resolver Θ2 se considera (A.18) y (A.19) y se ordenan de la siguiente manera:

$$a_2 \cdot \sin(\theta 2) - f_1 = a_3 \cdot \sin(\theta 3) + d_4 \cdot \cos(\theta 3) \tag{A.20}$$

$$a_2 \cdot \cos(\theta 2) - f_2 = -a_3 \cdot \cos(\theta 3) + d_4 \cdot \sin(\theta 3) \tag{A.21}$$

Donde

$$f_1 = \left(\frac{T_{2,4} + T_{2,3} \cdot d_6}{\sin(\theta 1)} - a_1 \right) \tag{A.22}$$

$$f_2 = \left(T_{3,4} + T_{3,3} \cdot d_6 \right) \tag{A.23}$$

Se elevan al cuadrado ambos lados de (A.20) y (A.21). Sumando y simplificando se obtiene:

$$2 \cdot \sin(\theta 2) \cdot a_2 \cdot f_1 + 2 \cdot \cos(\theta 2) \cdot a_2 \cdot f_2 - \left(f_1\right)^2 - \left(f_2\right)^2 - \left(a_2\right)^2 + \left(a_3\right)^2 + \left(d_4\right)^2 = 0 \tag{A.24}$$

Que tiene la forma de (A.15) y se resuelve con (A.16) donde:

$$A = 2 \cdot a_2 \cdot f_1, \quad B = 2 \cdot a_2 \cdot f_2, \quad C = -\left(f_1\right)^2 - \left(f_2\right)^2 - \left(a_2\right)^2 + \left(a_3\right)^2 + \left(d_4\right)^2 \tag{A.25}$$

Ahora que se conoce Θ2, de (A.20) o bien de (A.21) se puede obtener Θ3 tal como se hizo para Θ2. Por ejemplo con (A.20), Θ3 se obtiene con (A.16) donde:

$$A = a_3, \quad B = d_4, \quad C = f_1 - a_2 \cdot \sin(\theta 2) \tag{A.26}$$

Ahora que se conoce Θ1, Θ2 y Θ3 se puede retomar el método que se planteó en un principio, en el cual por medio de (A.10) con $n = 5$, se puede encontrar una ecuación que permita resolver Θ4. Se tiene que U_4 es conocida y que $U_5 = (A_4)^{-1} U_4$. Si se igualan los elementos (3,1) y (3,2) de la ecuación (A.10) se obtienen las siguientes expresiones:

$$-\sin(\theta 6) = \left(U_4\right)_{1,1} \cdot \sin(\theta 4) - \left(U_4\right)_{2,1} \cdot \cos(\theta 4) \tag{A.27}$$

$$\cos(\theta 6) = \left(U_4\right)_{1,2} \cdot \sin(\theta 4) - \left(U_4\right)_{2,2} \cdot \cos(\theta 4) \tag{A.28}$$

Se elevan al cuadrado ambos lados de (A.27) y (A.28) y suman los resultados para obtener:

$$2 g_3 \cdot \sin(2\theta 4) + \left(g_2 - g_1\right) \cdot \cos(2\theta 4) - \left(g_2 + g_1\right) + 2 = 0. \tag{A.29}$$

Donde:

$$g_1 = \left[\left(U_4\right)_{2,1}\right]^2 + \left[\left(U_4\right)_{2,2}\right]^2, \quad g_2 = \left[\left(U_4\right)_{1,1}\right]^2 + \left[\left(U_4\right)_{1,2}\right]^2$$
$$g_3 = \left(U_4\right)_{1,1} \cdot \left(U_4\right)_{2,1} + \left(U_4\right)_{1,2} \cdot \left(U_4\right)_{2,2} \tag{A.30}$$

Y Θ4 se resuelve por medio de (A.16) con:

$$A = 2g_3, \quad B = g_2 - g_1, \quad C = 2 - \left(g_1 + g_2\right) \tag{A.31}$$

En este caso se comprobó que $A^2 + B^2 - C^2 = 0$, por lo que la solución, sin pérdida de generalidad, se reduce a:

$$\theta 4 = \operatorname{atan}\left(\frac{-A}{C - B}\right) \tag{A.32}$$

Ahora se puede resolver Θ5 a partir de la ecuación $A_6 = (A_5)^{-1} U_5$ obtenida de (A.10) con $n = 6$. Del elemento (1,4) se obtiene:

$$\left(U_5\right)_{1,4} \cdot \cos(\theta 5) + \left(U_5\right)_{2,4} \cdot \sin(\theta 5) = 0 \tag{A.33}$$

Θ5 se obtiene con (A.16), sustituyendo:

$$A = \left(U_5\right)_{2,4}, \quad B = \left(U_5\right)_{1,4}, \quad C = 0. \tag{A.34}$$

Ahora que U_5 es conocido Θ6 se puede obtener fácilmente utilizando la función atan2 ya mencionada.

$$\theta 6 = \operatorname{atan2}\left[\left(U_5\right)_{3,2}, -\left(U_5\right)_{3,1}\right] \tag{A.35}$$

De este modo, se ha resuelto la cinemática inversa del robot FANUC LR Mate 200iC. Sin embargo las juntas obtenidas por medio de (A.16) presentan una dualidad de soluciones, para las que en algunos casos se puede seleccionar una de ellas de acuerdo a la configuración del robot o considerando algún índice de manipulabilidad.

B. Manual de usuario

Manual del usuario para un sistema de programación por guiado mediante interacción física.

Sistema desarrollado en:

UNIVERSIDAD AUTÓNOMA DE SAN LUIS POTOSÍ

LABORATORIO DE ROBÓTICA

Por:

Ing. Reynaldo Soubervielle Montalvo

Bajo la asesoría del:

Dr. Emilio Jorge González Galván

Robot:

FANUC LR Mate 200*i*C

B.1. Introducción

La programación de robots industriales se puede definir como el proceso mediante el cual se indica a éste la secuencia de acciones que deberá llevar a cabo durante la realización de una tarea. Estas acciones consisten generalmente en moverse a puntos predefinidos y manipular objetos del entorno.

En este manual se muestra de que manera ha de usarse un sistema de programación alternativo, implementado en un robot FANUC LR Mate 200iC para generar la información necesaria para que el robot reproduzca una trayectoria deseada, dicho sistema tiene como objetivo facilitar la enseñanza de trayectorias prescindiendo de vastos conocimientos en programación.

B.2. Seguridad para el operador

La idea principal de un sistema de programación por guiado mediante interacción física, representa violar una regla de seguridad básica en el uso de robots industriales, aquella que prohíbe el contacto físico del usuario con el robot cuando este se encuentra en operación. Sin embargo mediante el aprovechamiento de las capacidades del sistema robótico de FANUC, ha sido posible establecer lo necesario para asegurar la integridad física del usuario.

El sistema cuneta con tres paros de emergencia los cuales están distribuidos como se muestra en la figura B.2 De los cuales 2 consisten en botones en forma de hongo, como el mostrado en la figura B.1, que al ser pulsados permanecen enclavados y el botón regresa a su estado original aplicando un giro en sentido horario del mismo. El tercer paro de emergencia consiste en un botón de hombre muerto ubicado en la terminal de comunicación el cual debe permanecer pulsado mientras el sistema está en funcionamiento, al soltar el botón el sistema entrará en estado de paro. En los tres casos además de corregir el estado del botón es necesario pulsar la tecla “RESET” de la terminal de enseñanza para restablecer el estado del sistema.

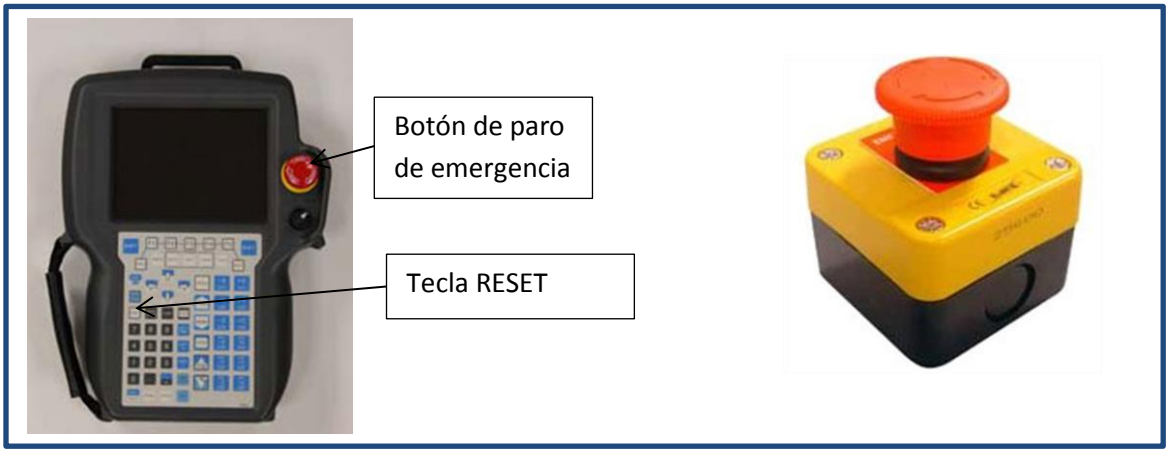


Figura B.1. Izquierda, ubicación del botón de paro de emergencia y tecla RESET en la terminal de enseñanza. Derecha, botón de paro de emergencia externo.

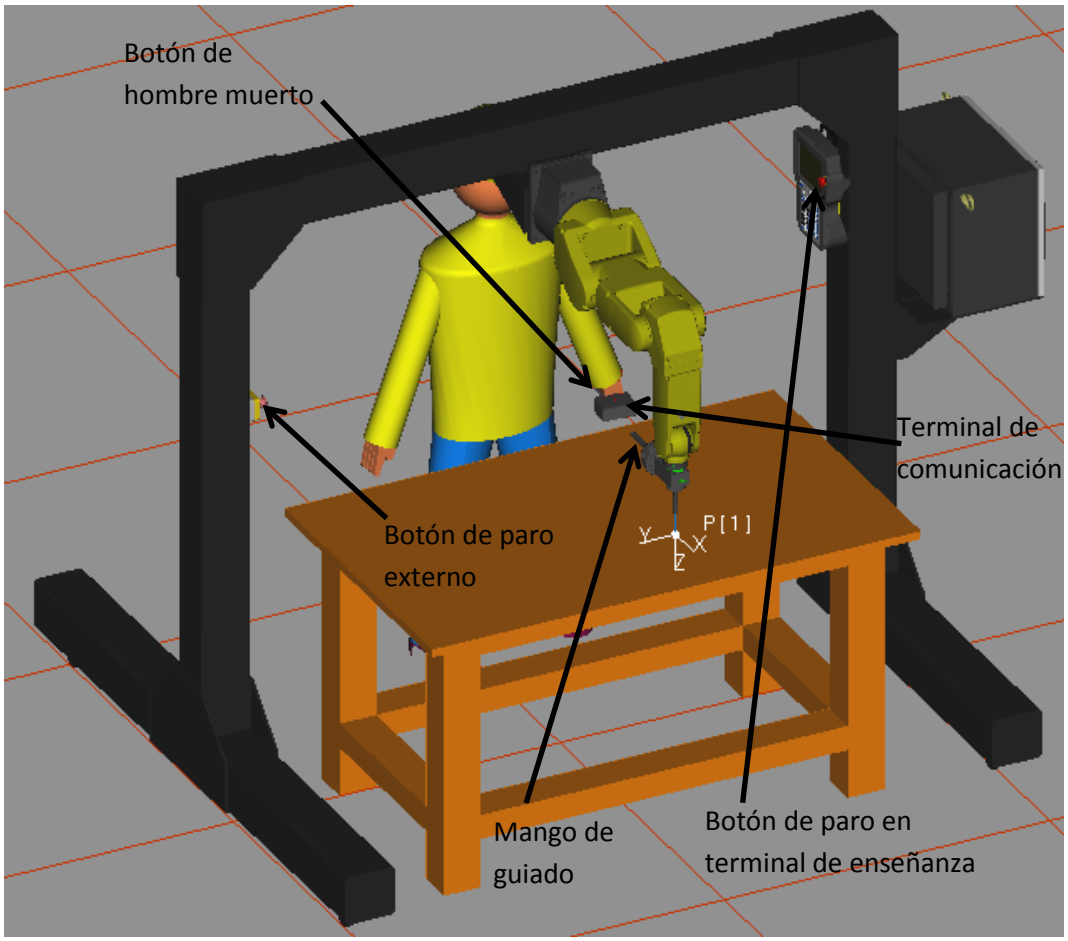


Figura B.2. Ubicación de botones de paro de emergencia en la celda de manufactura.

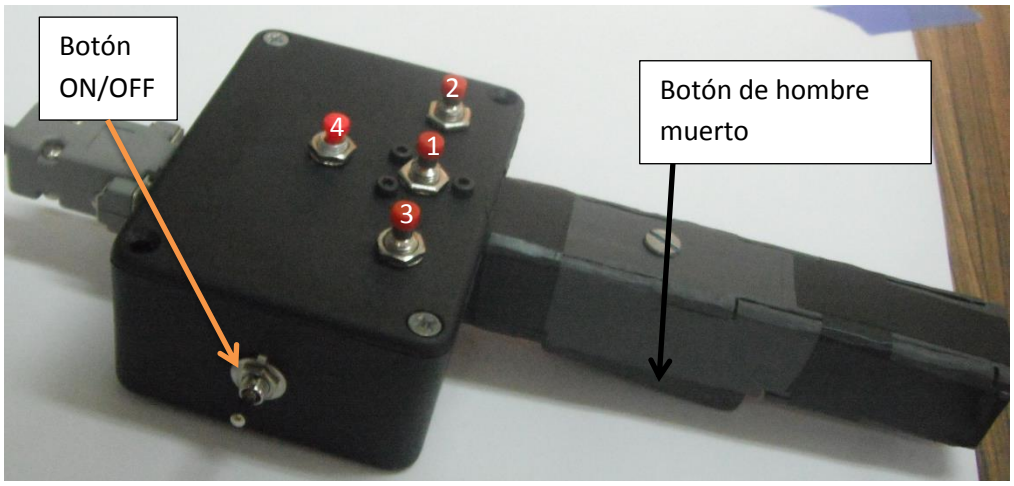


Figura B.3. Ubicación del botón de hombre muerto en la terminal de comunicación.

Por medio de estos botones el usuario puede detener el funcionamiento instantáneamente en caso de detectarse algún comportamiento peligroso en el movimiento del robot que ponga en riesgo de accidente a su persona o al robot.

B.3. Interfaces de programación

El sistema funciona por medio de tres interfaces de comunicación: interacción física Humano-Robot, (mango unido al sensor, figura B.2) que permite al usuario guiar al robot hacia las posiciones de interés, interacción por medio de botones (terminal de comunicación, figura B.3) mediante la cual el usuario indica al sistema en que momento realizar diferentes procedimientos como pueden ser: guardar cierta posición o seleccionar tipo de movimiento, finalmente el sistema cuenta con una interfaz visual (pantalla de la terminal de enseñanza, figura B.1) cuyo objetivo es el de informar al usuario el estado del sistema y confirmar que los procedimientos seleccionados se realizaron correctamente.

A continuación se describe el procedimiento que se debe seguir para realizar un programa con el objetivo de explicar de manera detallada el uso y funcionamiento de cada interface.

Para poner en marcha el sistema de programación mediante guiado manual se deben seguir las siguientes instrucciones:

- Encender el sistema robótico FANUC LR Mate 200*i*C. Se pone el botón ON/OFF del controlador en ON con el interruptor de modalidad del controlador en AUTO y el interruptor ON/OFF de la terminal de enseñanza en OFF.
- En la terminal de enseñanza oprimir la tecla SELECT para navegar en la lista hasta encontrar y seleccionar el programa TPFRK9 y oprima la tecla ENTER. En la esquina superior derecha de la pantalla sobre un fondo verde hay un valor en porcentaje, si este no es 100% oprima la tecla +% hasta llevarlo a ese valor.
- En la terminal de comunicación poner el botón ON/OFF en ON, luego oprimir el botón de hombre muerto (éste no se soltara hasta finalizar el programa). Lo anterior causara una señal de alarma la cual se elimina presionando la tecla RESET en la terminal de enseñanza.
- Finalmente se presiona el botón CYCLE START en el controlador.

El sistema de programación funciona mediante dos modos: MODO GUIANDO y MODO GRABANDO. La selección del modo en el sistema representa una acción y además cada modo cuenta con tres acciones. A continuación se menciona como identificar cada una en las tres interfaces.

Acción 1 (Botón 1: Cambio de modo) Se realiza cada que se pulsa el botón 1 en la terminal de comunicación, en la cual se han colocado etiquetas que de forma gráfica indican que acción realiza cada botón Figura B.4. En fondo verde están las acciones correspondientes a guiado y en fondo azul las de grabando. Después de presionar la tecla CYCLE START en el controlador el sistema inicia en modo guiando por lo que en la en interfaz

visual se mostrara una ventana denominada USER con un mensaje que dice “Guiando” seguido de “Se ha establecido la posición actual como fija”

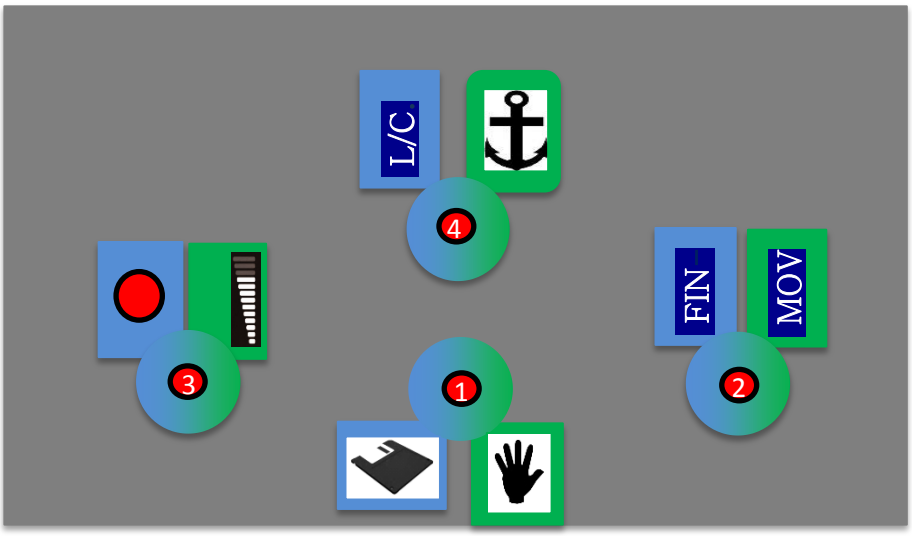


Figura B.4. Etiquetas para fácil identificación en la terminal de comunicación

En este modo el usuario puede mover el robot sujetándolo por el mango de guiado, al soltar el mango de guiado el robot regresará muy lentamente hacia la posición que tenía el robot cuando en la interfaz visual apareció el mensaje “Se ha establecido la posición actual como fija”. Las acciones relacionadas a botones dos a tres en este modo son:

Acción 2 (Botón 2: tipo de movimiento). Selecciona entre tres opciones: guiar solo posición, solo orientación o ambas. Al inicio el sistema tiene la capacidad de guiar ambas y cada vez que se presione el botón se cambiará a guiar posición luego a guiar orientación para finalmente regrese al estado inicial. En la interfaz visual el resultado serán una serie de mensajes que confirman que se realizó cada cambio de operación de la siguiente manera.

Tipo de operación: posición
Tipo de operación: orientación
Tipo de operación: total

Con el correspondiente y obvio resultado en el movimiento del robot al ser comandado por el operador.

Acción 3 (Botón 3: Cambia la velocidad de guiado). Selecciona entre 4 niveles de velocidad establecidos subjetivamente por los mensajes que aparecen en la interfaz visual. Al inicio el sistema opera a una velocidad “alta” y cada vez que se presione el botón se cambiará a un nivel de velocidad menor hasta llegar al nivel “muy baja”, donde el cambio se realizará a un nivel mayor hasta llegar al estado inicial. En la interfaz visual el

resultado serán una serie de mensajes que confirman que se realizó cada cambio en la velocidad de operación de la siguiente manera.

Velocidad: media
Velocidad: baja
Velocidad: muy baja
Velocidad: baja
Velocidad: media
Velocidad: alta

Como consecuencia de lo anterior el robot opondrá diferentes resistencias al movimiento comandado por el usuario generando diferentes velocidades de interacción.

Acción 4 (Botón 4: Establece la posición actual del robot como la nueva posición fija). En la interfaz visual saldrá el mensaje “Se ha establecido la posición actual como fija” es de mucha importancia que este botón se presione solo cuando el robot no esté en contacto con el operador o el entorno. Una manera de evitar tal caso, es presionar este botón con la mano utilizada para guiar el robot mediante interacción.

Después de presionar este botón el robot no se moverá hasta que el operador tenga contacto con el mango de guiado y al soltarlo regresará muy lentamente a la posición que tenía al presionar el botón.

Hasta este punto se han descrito las acciones relacionadas con el modo de guiado, lo que da el conocimiento necesario para guiar al robot hasta una posición de interés para la tarea a programar. Una vez que la posición requerida es alcanzada se realiza el cambio de modo presionando el botón 1 de la terminal de comunicación. El cambio de modo se confirma en la pantalla de la terminal de enseñanza con el mensaje “Grabando”, en este modo el robot permanece inmóvil es decir la interfaz de interacción humano-robot queda inactiva y se activan las acciones correspondientes al modo grabando como son:

Acción 5 (botón 4: tipo de trayectoria). Selecciona entre dos tipos de trayectoria (lineal o circular) que el robot seguirá durante la ejecución del comando de movimiento en la tarea. Al iniciar el sistema el tipo de trayectoria activa es lineal por lo que al presionar el botón 4, en la pantalla de la terminal de enseñanza aparecerá el mensaje “Tipo de trayectoria: circular”. Se regresa al estado inicial presionando el mismo botón con lo que aparecerá el mensaje “Tipo de trayectoria: lineal”.

Acción 6 (botón 3: Captura de posición) esta acción permite indicar al sistema que dicha posición se asignara a un comando de movimiento su mensaje de confirmación dependerá de qué tipo de trayectoria está seleccionada, si la trayectoria es lineal el mensaje será: “se guardó un punto con trayectoria lineal”. Para el caso de trayectoria circular el comando de movimiento se terminará hasta que capturen dos posiciones alternando una etapa de guiado. Después de capturar la primera posición aparecerá el mensaje: “Se capturó la posición intermedia para una trayectoria circular. Pase al modo guiando y capture el punto final

del arco". Al capturar la segunda posición se mostrará el mensaje: "Se guardó el punto final para una trayectoria Circular.

Acción 7 (botón 2: finaliza el programa) esta acción permite finalizar la programación de la tarea, al presionar el botón se mostrará el mensaje: "La enseñanza ha finalizado con éxito". Además el programa generado quedará seleccionado, en caso de que se haya capturado por lo menos una posición. De no ser así se mostrará el mensaje: "No se ha enseñado ninguna posición"

C. Conexión de paros de emergencia

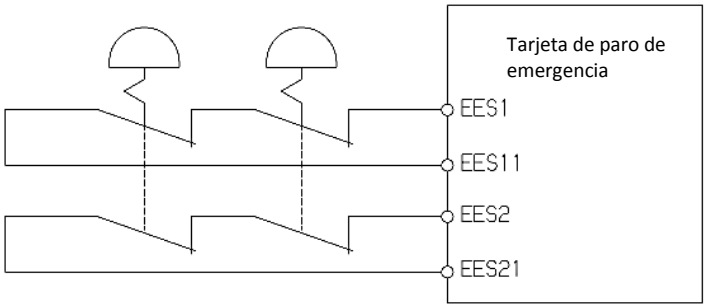


Figura C.1. Conexión de botones de paro de emergencia en serie, uno se ubica fijo en una posición estratégica, el otro queda móvil para aplicaciones no definidas.

Se conectaron dos paros de emergencia extras utilizando las entradas utilizadas para la puerta de la valla de seguridad. Se utilizaron botones en forma de hongo normalmente cerrados que, al ser pulsados, se abren y permanecen enclavados mecánicamente. El botón regresa a su estado original aplicando un giro en sentido horario en el mismo.

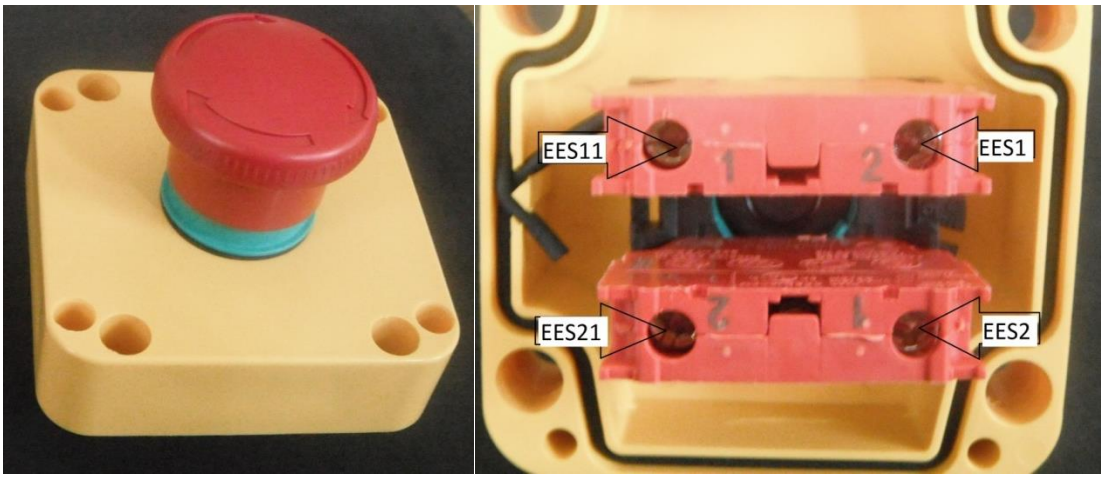


Figura C.2. Izquierda: botón del tipo hongo, derecha: conexión en los interruptores.

D. Programas

D.1. Programa principal

```
PROGRAM nonstop
%COMMENT = 'Pro1'
%RWACCESS
%STACKSIZE = 4000
%NOLOCKGROUP
%NOPAUSE=ERROR+COMMAND+TPENABLE
%ENVIRONMENT uif
%ENVIRONMENT sysdef
%ENVIRONMENT memo
%ENVIRONMENT kclop
%ENVIRONMENT bynam
%ENVIRONMENT fdev
%ENVIRONMENT flbt
%ENVIRONMENT STRNG
%INCLUDE klevccdf
%INCLUDE klevkeys
%INCLUDE klevkmsk

VAR

m,n,o,p              :BOOLEAN
xd,x,xc              :XYZWPR
STATUS,entry,clock   :INTEGER
v1,v0,juntas,j       :ARRAY[6] OF REAL
F,G,Fb,Gb            :VECTOR
jpos                 :JOINTPOS
indi,cont,v,r,c,mo   :INTEGER
prog_index,fst       :INTEGER
file_var, file_var1, file_var2, file_var3:FILE
Xa,Ya,Za,x0,y0,z0,Vx,Vy,Vz,vx0,vy0,vz0:REAL
Wa,Pa,Ra,w0,p0,r0,Vw,Vp,Vr,vw0,vp0,vr0:REAL
Md,Bd,Kd,Id,Cd,Kr,Fx,Gx,t,bp,bo,bpl,bol:REAL
fa,fe:REAL

ROUTINE rungekpos
BEGIN
  Xa=x0+t*bp*(vx0+0.5*t*(F.x-Bd*vx0-Kd*x0)/Md)
  Ya=y0+t*bp*(vy0+0.5*t*(F.y-Bd*vy0-Kd*y0)/Md)
  Za=z0+t*bp*(vz0+0.5*t*(F.z-Bd*vz0-Kd*z0)/Md)
  Vx=vx0+t*bp*(F.x-Bd*(vx0+0.5*t*(F.x-Bd*vx0-Kd*x0)/Md)-Kd*(x0+0.5*t*vx0))/Md
  Vy=vy0+t*bp*(F.y-Bd*(vy0+0.5*t*(F.y-Bd*vy0-Kd*y0)/Md)-Kd*(y0+0.5*t*vy0))/Md
  Vz=vz0+t*bp*(F.z-Bd*(vz0+0.5*t*(F.z-Bd*vz0-Kd*z0)/Md)-Kd*(z0+0.5*t*vz0))/Md
END rungekpos

ROUTINE rungekori
BEGIN
  Wa=w0+t*bo*(vw0+0.5*t*(G.x-Cd*vw0-Kr*w0)/Id)
  Pa=p0+t*bo*(vp0+0.5*t*(G.y-Cd*vp0-Kr*p0)/Id)
  Ra=r0+t*bo*(vr0+0.5*t*(G.z-Cd*vr0-Kr*r0)/Id)
  Vw=vw0+t*bo*(G.x-Cd*(vw0+0.5*t*(G.x-Cd*vw0-Kr*w0)/Id)-Kr*(w0+0.5*t*vw0))/Id
  Vp=vp0+t*bo*(G.y-Cd*(vp0+0.5*t*(G.y-Cd*vp0-Kr*p0)/Id)-Kr*(p0+0.5*t*vp0))/Id
  Vr=vr0+t*bo*(G.z-Cd*(vr0+0.5*t*(G.z-Cd*vr0-Kr*r0)/Id)-Kr*(r0+0.5*t*vr0))/Id
END rungekori

ROUTINE fumbral
```

```
BEGIN
  IF F@F < 10 THEN
    F.x=0; F.y=0; F.z=0
  ENDIF
  IF G@G < 0.05 THEN
    G.x=0; G.y=0; G.z=0
  ENDIF
END fumbral

ROUTINE backslash
BEGIN
  IF ABS(F@F-Fb@Fb)>bpl THEN
    bp=fa
  else
    IF bp <> 1 THEN
      bp=bp+0.25
    ENDIF
  ENDIF
  IF ABS(G@G-Gb@Gb)>bol THEN
    bo=fa
  else
    IF bo <> 1 THEN
      bo=bo+0.25
    ENDIF
  ENDIF
  Fb=F
  Gb=G
END backslash

ROUTINE ic --condicion 5
BEGIN
  ----condiciones inciales del sistema R-K------------
  x0=0; y0=0; z0=0; vx0=0; vy0=0; vz0=0
  w0=0; p0=0; r0=0; vw0=0; vp0=0; vr0=0
  Xa=0; Ya=0; Za=0; Vx=0;  Vy=0;  Vz=0
  Wa=0; Pa=0; Ra=0; Vw=0;  Vp=0;  Vr=0
  Fb.x=0; Fb.y=0; Fb.z=0; Gb.x=0; Gb.y=0; Gb.z=0
  ------------------------------------------------
  DELAY(500)
  GET_VAR(entry,'*SYSTEM*','$CCC_GRP[1].$FS_FORCE', v0, STATUS)
  xd=CURPOS(0,0)-- Get the current cartesian position of the TCP
  x=xd
  WRITE TPDISPLAY('Se ha establecido la posición',CR,'actual como fija',CR)
  ENABLE CONDITION[4]
END ic

ROUTINE interaction
BEGIN
  ic
  WHILE o DO
    IF n THEN
      GET_VAR(entry,'*SYSTEM*','$CCC_GRP[1].$FS_FORCE', v1, STATUS)
      F.x=(v1[1]-v0[1])*9.81;     F.y=(V1[2]-V0[2])*9.81;     F.z=(V1[3]-
V0[3])*9.81
      G.x=(v1[4]-v0[4])*9.81/100; G.y=(V1[5]-V0[5])*9.81/100; G.z=(V1[6]-
V0[6])*9.81/100
      Fx=F@F; Gx=G@G
      backslash
      fumbral
      Md=fe*15*r; Bd=fe*94*r; Kd=fe*3*r; Id=fe*0.25*r; Cd=fe*2*r; Kr=fe*0.125*r
      SELECT mo OF
        CASE(1):
          rungekpos
```

```
          rungekori
        CASE(2):
          rungekpos
        CASE(3):
          rungekori
      ENDSELECT
      x.x=Xa*1000
      x.y=Ya*1000
      x.z=Za*1000
      x.w=Wa*180/PI
      x.p=Pa*180/PI
      x.r=Ra*180/PI
      CHECK_EPOS(xd:x,$UFRAME,$UTOOL,fst)
      IF fst <> 0 THEN
        n=FALSE
        WRITE TPDISPLAY(CHR(128),CHR(137),'Grabando',CR)
        DISABLE CONDITION[8]
        ENABLE CONDITION[7]
      else
        WITH $SEG_TIME=v MOVE TO xd:x   NOWAIT
        x0=Xa; vx0=Vx; y0=Ya; vy0=Vy; z0=Za; vz0=Vz;
        w0=Wa; vw0=Vw; p0=Pa; vp0=Vp; r0=Ra; vr0=Vr;
      ENDIF
    ENDIF
  ENDWHILE
END interaction

ROUTINE getp --condicion 3
BEGIN
  DELAY(500)
  indi=indi+1
  IF p THEN --si p es verdadera guarda el punto lineal
    jpos=CURJPOS(0,0)
    CNV_JPOS_REL(jpos,juntas,STATUS)
    WRITE file_var
(juntas[1],juntas[2],juntas[3],juntas[4],juntas[5],juntas[6],CR)
    WRITE file_var2 (0,indi,0,0,0,0,CR)
    WRITE file_var3 (0,0,0,0,0,0,CR)
    WRITE TPDISPLAY(CHR(128),CHR(137),'Grabando',CR,'Se guardó un puno',CR,'con
trayectoria lineal',CR)
  else
            --si p no es verdadera garda el punto intermedio y final de un
circulo
    cont=cont+1
    IF cont <> 2 THEN
      jpos=CURJPOS(0,0)
      CNV_JPOS_REL(jpos,juntas,STATUS)
      WRITE file_var3
(juntas[1],juntas[2],juntas[3],juntas[4],juntas[5],juntas[6],CR)
      WRITE file_var2 (1,indi+1,0,0,0,0,CR)
      WRITE TPDISPLAY(CHR(128),CHR(137),'Grabando',CR,'Se guardó el puno
intermedio',CR,'para una trayectoria circular.',CR)
      WRITE TPDISPLAY('Pase al modo de guiando y...',CR,'capture el punto final
del arco',CR)
    else
      jpos=CURJPOS(0,0)
      CNV_JPOS_REL(jpos,juntas,STATUS)
      WRITE file_var
(juntas[1],juntas[2],juntas[3],juntas[4],juntas[5],juntas[6],CR)
      cont=0
      indi=indi-1
      WRITE TPDISPLAY('Grabando',CR,'Se guardó el puno final...',CR,'para una
trayectoria circular',CR)
```

```
    ENDIF
  ENDIF
  ENABLE CONDITION[3]
  interaction
END getp

ROUTINE velup --CONDICION 2
BEGIN
  c=c+2
  r=c*c
  SELECT c OF
    CASE(3):
      WRITE TPDISPLAY('Velocidad: media',CR)
    CASE(5):
      WRITE TPDISPLAY('Velocidad: baja',CR)
    CASE(4):
      WRITE TPDISPLAY('Velocidad: muy baja',CR)
  ENDSELECT
  IF c=5 THEN
    ENABLE CONDITION[1]
  else
    ENABLE CONDITION[2]
  ENDIF
END velup

ROUTINE veldown --CONDICION 1
BEGIN
  c=c-2
  r=c*c
  SELECT c OF
    CASE(1):
      WRITE TPDISPLAY('Velocidad: alta',CR)
    CASE(3):
      WRITE TPDISPLAY('Velocidad: media',CR)
    CASE(31):
      WRITE TPDISPLAY('Velocidad: baja',CR)
  ENDSELECT
  IF c=1 THEN
    ENABLE CONDITION[2]
  else
    ENABLE CONDITION[1]
  ENDIF
END veldown

ROUTINE guia --condicion 7
BEGIN
  WRITE TPDISPLAY(CHR(128),CHR(137),'Guiando',CR)
  ic
END guia

ROUTINE gra --condicion 8
BEGIN
  WRITE TPDISPLAY(CHR(128),CHR(137),'Grabando',CR)
END gra

ROUTINE moli--rutina reproducida por la condicion 10
BEGIN
  IF cont <> 1 THEN
    p=TRUE
    WRITE TPDISPLAY('Tipo de trayectoria: lineal',CR)
    ENABLE CONDITION[4]
  else
    WRITE TPDISPLAY('No puede cambiar el tipo de movimiento',CR)
```

```
    WRITE TPDISPLAY('hasta enseñar el punto final del arco',CR)
    ENABLE CONDITION[10]
  ENDIF
END moli

ROUTINE moci--rutina reproducida por la condicion 4
BEGIN
  p=FALSE
  WRITE TPDISPLAY('Tipo de trayectoria: Circular',CR)
  ENABLE CONDITION[10]
END moci

ROUTINE cambiamo
BEGIN
  mo=mo+1
  IF mo=4 THEN
    mo=1
  ENDIF
  SELECT mo OF
    CASE(1):
       WRITE TPDISPLAY('Tipo de operacion: total',CR)
    CASE(2):
       WRITE TPDISPLAY('Tipo de operacion: Posición',CR)
    CASE(3):
       WRITE TPDISPLAY('Tipo de operacion: Orientación',CR)
   ENDSELECT
  ENABLE CONDITION[11]
END cambiamo

ROUTINE fin
BEGIN
  IF cont <> 1 THEN
    o=FALSE
  else
    WRITE TPDISPLAY('No puede finalizar el programa hasta',CR)
    WRITE TPDISPLAY('enseñar el punto final del arco',CR)
    ENABLE CONDITION[9]
  ENDIF
END fin
-------------------principal----------------------
BEGIN
$USE_WJTURNS=FALSE
$GROUP[1].$USEMAXACCEL=TRUE
$GROUP[1].$UTOOL=$MNUTOOL[1,6]--con porta-herramineta[1,3]--
$GROUP[1].$UFRAME=$MNUFRAME[1,1]
$MOTYPE=JOINT
$TERMTYPE=NODECEL
FORCE_SPMENU(TP_PANEL,SPI_TPUSER,1)
SET_FILE_ATR(file_var,ATR_IA)
OPEN FILE file_var ('RW','puntos.kl')
SET_FILE_ATR(file_var1,ATR_IA)
OPEN FILE file_var1 ('RW','indice.kl')
SET_FILE_ATR(file_var2,ATR_IA)
OPEN FILE file_var2 ('RW','mov.kl')
SET_FILE_ATR(file_var3,ATR_IA)
OPEN FILE file_var3 ('RW','via.kl')
-----------------------------------------------------------------
----------------definicion de los condition handlers------------
-----------------------------------------------------------------

--seleccionan la funciónn de los botones-----------------------

--tareas relacionadas al modo guiando
```

```
CONDITION[7]:
  WHEN DIN[103]+ DO n=TRUE, guia
  ENABLE CONDITION[8]
ENDCONDITION

--tareas relacionadas al modo grabando
CONDITION[8]:
  WHEN DIN[103]+ DO n=FALSE, gra
  ENABLE CONDITION[7]
ENDCONDITION
-------------------------------------------------------------
--------------------condiciones guiando----------------------
-------------------------------------------------------------
--reinicia el sistema-----------------------------------------1
CONDITION[5]:
  WHEN DIN[101]+ AND n=TRUE DO ic
  ENABLE CONDITION[5]
ENDCONDITION

--reduce la velocidad de operación----------------------------2
CONDITION[2]:
  WHEN DIN[102]+ AND n=TRUE DO velup
ENDCONDITION

--aumenta la velocidad de operación---------------------------3
CONDITION[1]:
  WHEN DIN[102]+ AND n=TRUE DO veldown
ENDCONDITION

--seleciona que grados de libertad son manipulables-----------4
CONDITION[11]:
  WHEN DIN[104]+ AND n=TRUE DO cambiamo
ENDCONDITION

-------------------------------------------------------------
--------------------condiciones grabando---------------------
-------------------------------------------------------------
--indica cuando finalizar el programa-------------------------1
CONDITION[9]:
  WHEN DIN[104]+ AND n=FALSE DO fin
ENDCONDITION

--indica cuando se captura un punto---------------------------2
CONDITION[3]:
  WHEN DIN[102]+ AND n=FALSE DO getp
ENDCONDITION

--seleccionan el tipo de movimiento: lineal o circular--------3
--Lineal
CONDITION[10]:
  WHEN DIN[101]+ AND n=FALSE AND p=FALSE DO moli
ENDCONDITION
--Circular
CONDITION[4]:
  WHEN DIN[101]+ AND n=FALSE AND p=TRUE DO moci
ENDCONDITION

-------------------------------------------------------------
--inicia el programa-------------------------------------------
-------------------------------------------------------------
Md=5; Bd=40; Kd=3; Id=1; Cd=8; Kr=.5; r=1; c=1; mo=1
indi=0; cont=0; t=0.005; v=4 ; m=TRUE; n=m; o=m; p=m; bp=1; bo=1;
bpl=40; bol=.5; fa=0; fe=0.5;
```

```
ENABLE CONDITION[2]
ENABLE CONDITION[3]
ENABLE CONDITION[4]
ENABLE CONDITION[5]
ENABLE CONDITION[8]
ENABLE CONDITION[9]
ENABLE CONDITION[11]
WRITE TPDISPLAY(CHR(128),CHR(137),'Guiando',CR)
interaction
WRITE file_var1 (indi, 0, 0, 0, 0, 0)
CLOSE FILE file_var
CLOSE FILE file_var1
CLOSE FILE file_var2
CLOSE FILE file_var3
IF indi <> 0 THEN
  WRITE TPERROR(CHR(128),CHR(137),'La enseñanza ha finalizado')
  WRITE TPDISPLAY('La enseñanza ha finalizado',CR,'con éxito')
  CALL_PROG('gentpe1',prog_index)
else
   WRITE TPDISPLAY('No se ha enseñado ninguna posición')
ENDIF
END nonstop
```

D.2. Programa que genera TPE

```
PROGRAM GENTPE1
%COMMENT = 'Proguia2'
%ENVIRONMENT sysdef
%ENVIRONMENT TPE -- necesario para los xxx_TPE built-ins

VAR
       creastat  : INTEGER
       STATUS    : INTEGER
       STATUS1   : INTEGER
       STATUS2   : INTEGER
       STATUS3   : INTEGER
       STATUS4   : INTEGER
       STATUS5   : INTEGER
       X         : JOINTPOS
       open_id   :INTEGER
jpos,jpos1  :JOINTPOS
junta,junta1,junta2: ARRAY[6] OF REAL
t,i,m:INTEGER
file_var, file_var1,file_var2,file_var3:FILE


BEGIN
  $GROUP[1].$UTOOL=$MNUTOOL[1,7]
  CREATE_TPE('SOU_1',PT_MNE_UNDEF,creastat)
  COPY_TPE('base','SOU_1',TRUE,STATUS1)
  OPEN_TPE('SOU_1',TPE_RWACC,TPE_NOREJ,open_id,STATUS2)
  SET_FILE_ATR(file_var1,ATR_FIELD)
  OPEN FILE file_var1('RO','indice.kl')
  READ file_var1(junta[1],junta[2],junta[3],junta[4],junta[5],junta[6])
  i=ROUND(junta[1])
  CLOSE FILE file_var1
  SET_FILE_ATR(file_var2,ATR_FIELD)
  OPEN FILE file_var2('RO','mov.kl')

  FOR t=1 TO i DO
    READ file_var2 (junta1[1],junta1[2],junta1[3],junta1[4],junta1[5],junta1[6])
    m=ROUND(junta1[2])
    DEL_INST_TPE(open_id,m,STATUS)
```

```
  ENDFOR

  WHILE STATUS=0 DO
    DEL_INST_TPE(open_id,i+1,STATUS)
  ENDWHILE

  SET_FILE_ATR(file_var,ATR_FIELD)
  OPEN FILE file_var('RO','puntos.kl')
  SET_FILE_ATR(file_var3,ATR_FIELD)
  OPEN FILE file_var3('RO','via.kl')

  jpos = CURPOS(0,0)  --Get the current POSITION
  jpos1=jpos
  FOR t=1 TO  i DO
    WRITE TPERROR(CHR(128),CHR(137),t)
    --puntos para movimiento lineal o final de arco de movimiento circular
      READ file_var (junta[1],junta[2],junta[3],junta[4],junta[5],junta[6])
      --posiciones via para movimiento circular
      READ file_var3
(junta2[1],junta2[2],junta2[3],junta2[4],junta2[5],junta2[6])
      CNV_REL_JPOS(junta,jpos, STATUS)
      CNV_REL_JPOS(junta2,jpos1, STATUS)  -- Convierte  de real a juntas
    SET_JPOS_TPE(open_id,2*t-1,jpos1,STATUS3)
      SET_JPOS_TPE(open_id,2*t,jpos,STATUS4)
  ENDFOR
  CLOSE FILE file_var
  CLOSE FILE file_var2
  CLOSE FILE file_var3
  SELECT_TPE('SOU_1',STATUS)
END GENTPE1
```

Bibliografía

[1] http://www.ifr.org/history/. History of industrial robots, revisada en 2013

[2] Gruver, W. A., Soroka, B. I., Craig, J. J., & Turner, T. L. (1984). Industrial robot programming languages: a comparative evaluation. Systems, Man and Cybernetics, Memorias de la IEEE. No 4, pp. 565-570.

[3] Lozano-Perez, T. (1983). Robot programming. Congreso de la IEEE, Vol 71, No. 7, pp. 821-841.

[4] Biggs, G., & MacDonald, B. (2003). A survey of robot programming systems. En el congreso de la Australasian conference on robotics and automation. pp. 1-3.

[5] Faverjon, B. (1986). Object level programming of industrial robots. In Robotics and Automation. Conferencia Internacional de la IEEE, Vol. 3, pp. 1406-1412.

[6] http://www.osha.gov/dts/osta/otm/otm_iv/otm_iv_4.html, página oficial de la Occupational Health and Safety Administration, U.S. Department of Labor, revisada en Enero del 2013.

[7] http://en.wikipedia.org/wiki/Mobile_Robot_Programming_Toolkit, Mobile robot programming toolkit, revisada en Enero del 2013.

[8] Abrate, F., Indri, M., Lazzero, I., & Bottero, A. (2011). Efficient solutions for programming and safe monitoring of an industrial robot via a virtual cell. In Advanced Intelligent Mechatronics (AIM), 2011 IEEE/ASME Conferencia Internatciona. pp. 434-439

[9] Baturone, A. O. (2006). Robótica: Manipuladores y robots móviles. Marcombo. pp. 341-345.

[10] Gaiotto, Cesare; Gaiotto, Alvise. System for programming electric robots by direct learning. Patente Europea No EP 0520128, 27, Mar. 1996.

[11] Davini, Giorgio. Light-weight program controller. Patente de U.S.A. No 4,239,431, 16, Dic. 1980.

[12] Harjar, Martin J.; NOSS, Jeffrey S. Apparatus for calibrating link position transducers of a teaching robot and a work robot. Patente U.S.A. No 4,372,721, 8 Feb. 1983.

[13] Pan, Z., Polden, J., Larkin, N., Van Duin, S., & Norrish, J. (2012). Recent progress on programming methods for industrial robots. Robotics and Computer-Integrated Manufacturing, 28, No. 2, pp. 87-94.

[14] Delson, N., & West, H. (1994). Robot programming by human demonstration: The use of human inconsistency in improving 3D robot trajectories. In Intelligent Robots and Systems. Advanced Robotic Systems and the Real World, Conferncia de la IEEE. Vol. 2, pp. 1248-1255.

[15] Delson, N., & West, H. (1996). Robot programming by human demonstration: Adaptation and inconsistency in constrained motion. In Robotics and Automation, Proceedings, Conferencia Internacional de la IEEE. Vol. 1, pp. 30-36.

[16] Schraft, R. D., & Meyer, C. (2006). The need for an intuitive teaching method for small and medium enterprises. VDI BERICHTE, vol. 1956, p. 95.

[17] Skoglund, A., Iliev, B., Kadmiry, B., & Palm, R. (2007). Programming by demonstration of pick-and-place tasks for industrial manipulators using task primitives. In Computational Intelligence in Robotics and Automation. CIRA 2007. Simposio Internacional. pp. 368-373.

[18] Maeda, Y., Ushioda, T., & Makita, S. (2008). Easy robot programming for industrial manipulators by manual volume sweeping. In Robotics and Automation, 2008. ICRA 2008. Conferencia Internacional de la IEEE. pp. 2234-2239.

[19] Argall, B. D., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. Robotics and Autonomous Systems, Vol. 57, No. 5, pp. 469-483.

[20] Hollmann, R., Rost, A., Hagele, M., & Verl, A. (2010). A HMM-based approach to learning probability models of programming strategies for industrial robots. In Robotics and Automation (ICRA), Conferencia Internacional de la IEEE. pp. 2965-2970.

[21] http://www.dlr.de/rm/en/desktopdefault.aspx/tabid-3808/6234_read-8998/ The DLR SpaceMouse, revisada en Febrero del 2013.

[22] Choi, M. H., & Kim, S. J. (2000). A force/moment direction sensor and its use for human-robot interface in robot teaching. In Systems, Man, and Cybernetics, Conferencia Internacional de la IEEE. Vol. 3, pp. 2222-2227.

[23] Gini, G., & Gini, M. (1982). ADA: a language for robot programming?. Computers in Industry, vol. 3, No. 4, pp. 253-259.

[24] Volz, R. A., Mudge, T. N., & Gal, D. A. (1984). Using Ada as a programming language for robot-based manufacturing cells. Systems, Man and Cybernetics, Memorias de la IEEE, No. 6, pp. 863-878.

[25] Dixon, K. R., Dolan, J. M., & Khosla, P. K. (2004). Predictive robot programming: Theoretical and experimental analysis. The International Journal of Robotics Research, Vol. 23, No. 9, pp. 955-973.

[26] Stoica, M., Calangiu, G. A., Sisak, F., & Sarkany, I. (2010). A method proposed for training an artificial neural network used for industrial robot programming by demonstration. In Optimization of Electrical and Electronic Equipment (OPTIM), Conferencia Internacional de la IEEE. pp. 831-836.

[27] Lozano-Pérez, T., & Brooks, R. A. (1985). An approach to automatic robot programming. Springer US. pp. 293-328.

[28] Nilsson, K., & Johansson, R. (1999). Integrated architecture for industrial robot programming and control. Robotics and Autonomous Systems, Vol. 29, No. 4, pp. 205-226.

[29] Zuhlke, D., Mobius, F., & Schroder, C. (1997). Symbols facilitate programming of industrial robots. In Robotics and Automation, Conferencia Internacional de la IEEE Vol. 4, pp. 3037-3042.

[30] Wahl, F. M., & Thomas, U. (2002). Robot programming-from simple moves to complex robot tasks. Institute for Robotics and Process Control, Technical University of Brawnschweig.

[31] Samaka, M. (2005). Robot task-level programming language and simulation. Congreso of World Academy of Science, Engineering and Technology. Vol. 9, pp. 99-103.

[32] Pires, J. N., & Sá da Costa, J. M. G. (2000). Object-oriented and distributed approach for programming robotic manufacturing cells. Robotics and computer-integrated manufacturing, Vol. 16, No. 1, pp. 29-42.

[33] Myers, D. R., Pritchard, M. J., & Brown, M. D. (2001). Automated programming of an industrial robot through teach-by showing. In Robotics and Automation, 2001. Proceedings 2001 ICRA. Conferencia Internacional de la IEEE, Vol. 4, pp. 4078-4083.

[34] Hein, B., Hensel, M., & Worn, H. (2008). Intuitive and model-based on-line programming of industrial robots: A modular on-line programming environment. In Robotics and Automation, ICRA 2008. Conferencia Internacional de la IEEE, pp. 3952-3957.

[35] Park, C., Park, K., Park, D. I., & Kyung, J. H. (2009). Dual arm robot manipulator and its easy teaching system. In Assembly and Manufacturing, ISAM 2009. Congreso Internacional de la IEEE, pp. 242-247.

[36] Park, C., Kyung, J. H., & Park, D. I. (2010). Development of an industrial robot manipulator for the easy and safe human-robot cooperation. In Control Automation and Systems (ICCAS), Conferencia Internacional de la IEEE, pp. 678-681.

[37] Lambrecht, J., Kleinsorge, M., & Kruger, J. (2011, September). Markerless gesture-based motion control and programming of industrial robots. In Emerging Technologies & Factory Automation (ETFA), Conferencia 16 IEEE, pp. 1-4.

[38] Pires, J. N. (2005). Robot-by-voice: experiments on commanding an industrial robot using the human voice. Industrial Robot: An International Journal, Vol. 32, No. 6, pp. 505-511.

[39] Akan, B., Ameri, A., Curuklu, B., & Asplund, L. (2011, May). Intuitive industrial robot programming through incremental multimodal language and augmented reality. In Robotics and Automation (ICRA), Conferencia Internacional de la IEEE, pp. 3934-3939.

[40] Freund, E., & Luedemann-Ravit, B. (2002). A system to automate the generation of program variants for industrial robot applications. In Intelligent Robots and Systems, Conferencia Internacional de la IEEE/RSJ, Vol. 2, pp. 1856-1861.

[41] Kavraki, L. E., Svestka, P., Latombe, J. C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. Robotics and Automation, Memorias de la IEEE, Vol. 12, No. 4, pp. 566-580.

[42] Pettersen, T., Pretlove, J., Skourup, C., Engedal, T., & Lkstad, T. (2003). Augmented reality for programming industrial robots. In Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality, p. 319.

[43] Marin, Roque; Sanz, Pedro J.; Sanchez, J. S. A very high level interface to teleoperate a robot via web including augmented reality. En Robotics and Automation, 2002. Proceedings. ICRA'02. Conferencia Internacional de la IEEE, pp. 2725-2730.

[44] Mörtl, A., Lawitzky, M., Kucukyilmaz, A., Sezgin, M., Basdogan, C., & Hirche, S. (2012). The role of roles: Physical cooperation between humans and robots. The International Journal of Robotics Research, Vol. 31, No. 13, pp. 1656-1674.

[45] Scholtz, Jean. Human-robot interactions: Creating synergistic cyber forces.Multi-robot systems: from swarms to intelligent automata. Kluwer, (2002).

[46] Scholtz, J. (2003). Theory and evaluation of human robot interactions. In System Sciences, 36th Annual Hawaii International Conference, p. 10.

[47] Yanco, Holly A.; DRURY, Jill. (2004). Classifying human-robot interaction: an updated taxonomy. En Systems, Man and Cybernetics, Conferencia Internacional de la IEEE, pp. 2841-2846.

[48] Ogorodnikova, Olesya. (2008). Creating an active awareness system for humans in robotic workcell. Acta Polytechnica Hungarica, vol. 5, no 2, pp. 11-20.

[49] Hancock, P. A., Billings, D. R., Schaefer, K. E., Chen, J. Y., De Visser, E. J., & Parasuraman, R. (2011). A meta-analysis of factors affecting trust in human-robot

interaction. Human Factors: The Journal of the Human Factors and Ergonomics Society, Vol. 53, No. 5, pp. 517-527.

[50] Das, Biman. (2001). Ergonomics considerations and management action in the implementation of industrial robots. Human Factors and Ergonomics in Manufacturing & Service Industries, vol. 11, no 3, pp. 269-285.

[51] Steinfeld, A., Fong, T., Kaber, D., Lewis, M., Scholtz, J., Schultz, A., & Goodrich, M. (2006). Common metrics for human-robot interaction. InProceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction, pp. 33-40.

[52] Goodrich, M. A., & Schultz, A. C. (2007). Human-robot interaction: a survey. Foundations and Trends in Human-Computer Interaction, vol. 1 no. 3, pp. 203-275.

[53] Karlsson, N. (1999). A capacitance sensor for safeguarding operators of industrial robots. Robotica, vol. 17, no. 1, pp. 33-39.

[54] Moon, S., Rhim, S., Cho, Y. J., Park, K. H., & Virk, G. S. (2013). Summary of recent standardization activities in the field of robotics. Robotica, vol. 1no. 1, pp. 1-8.

[55] Breazeal, C. (2002). Regulation and entrainment in human-robot interaction.The International Journal of Robotics Research, vol. 21, no. 10-11, pp. 883-902.

[56] Argall, B. D., & Billard, A. G. (2010). A survey of tactile human–robot interactions. Robotics and autonomous systems, vol. 58, no. 10, pp. 1159-1176.

[57] De Santis, A., Siciliano, B., De Luca, A., & Bicchi, A. (2008). An atlas of physical human–robot interaction. Mechanism and Machine Theory, vol. 43, no. 3, pp. 253-270.

[58] N. Hogan, (1985). "Impedance Control: An Approach to Manipulation: PartI-Theory, Part II-Implementation and Part III-Aplications". Journal of Dynamic Systems, Measurement, and Control. Vol. 107, pp. 1-24.

[59] D. A. Lawrence, (1988). "Impedance Control Stability Properties in Common Implenentations", Proceedings of the IEEE International Conference on Robotics and Automation,

[60] Isela Bonilla-Gutiérrez. Control de interacción de robots manipuladores en tareas industrials y de rehabilitación. Tesis doctoral, Universidad Autónoma de San Luis Potosí, (Ago 2011).

[61] Arai, T., Kato, R., & Fujita, M. (2010). Assessment of operator stress induced by robot collaboration in assembly. CIRP Annals-Manufacturing Technology, vol. 59, no. 1, pp. 5-8.

[62] http://es.wikipedia.org/wiki/Diferencial_semantico, revisada en Febrero del 2013

[63] Schraft, R. D., Meyer, C., Parlitz, C., & Helms, E. (2005, April). PowerMate–A safe and intuitive robot assistant for handling and assembly tasks. In Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference, pp. 4074-4079.

[64] http://es.wikipedia.org/wiki/Prueba_t_de_Student, revisada en Marzo del 2013

[65] L. N. Rojas García, "Desarrollo de software para el control de impedancia de robots industriales", Tesis de Maestría, Universidad Autónoma de San Luis Potosí, (Febrero 2012).

[66] L. I. Avila Martínez, "Sistema de apoyo para rehabilitación usando un robot industrial", Tesis de maestria, Universidad Autónoma de San Luis Potosí, (Septiembre 2012).

[67] M. O. Mendoza Gutiérrez, "Control de un sistema de teleoperación para su aplicación en terapias robóticas de rehabilitación," Tesis doctoral, Universidad Autónoma de San Luis Potosí, (Agosto 2011).

[68] Denavit, J., and Hartenberg, R. S., (1955), A kinematic notation for lower pair mechanisms based on matrices, J. Applied Mechanics, vol. 22, pp. 215-221

[69] Richard P. Paul, Bruce Shimano, Gordon E. Mayer, (1981). Kinematic Control Equations for Simple Manipulators, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-11.