UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO

INSTITUTO DE CIENCIAS BÁSICAS E INGENIERÍA

INTEGRACIÓN DE LA ESTACIÓN ROBÓTICA INDUSTRIAL YAMAHA YK550H-QRCH AL SISTEMA DE MANUFACTURA FLEXIBLE DE LA EMPRESA NISSAN TIME DE MÉXICO

TESIS PARA OBTENER EL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

POR

DANIEL ALBERTO RAMÍREZ CONTRERAS
ASESOR: Dr. OMAR ARTURO DOMÍNGUEZ RAMÍREZ

PACHUCA, HIDALGO

2005

Resumen

Los nuevos sistemas de manufactura están constituidos por robots industriales en la mayor parte del proceso, como es el caso del centro de ensamblado automatizado de mecanismos de la empresa Nissan Time de México S.A. de C.V., el cual tiene una distribución celular con 12 estaciones robotizadas de ensamblado y en el que se requería integrar al proceso una estación robotizada que se encargue del embalaje de los mecanismos una vez ensamblados, para esto se contaba con un robot manipulador Yamaha YK550H de alta velocidad y configuración SCARA, una unidad de control Yamaha QRCH una botonera de programación (teach pendant) y un ordenador PC con procesador Athlon 2200 a 1.5 MHz sobre el cual se ejecuta el entorno de comunicación VIP Windows. El lenguaje de programación tiene comandos muy similares al lenguaje BASIC.

Esto con el fin de automatizar el área de embalaje y liberar de esta tarea repetitiva a la persona encargada de dicha función y colocarla en otra labor más humana como sería la de control de calidad.

Como el problema se trata de mover el brazo a dos distintos puntos, en un entorno estático y libre de obstáculos, se utilizó un modelo geométrico para la localización de los puntos para la programación de la tarea de carga y descarga, teniendo presente el criterio de obtener la trayectoria mínima entre los puntos y de que se programara el menor número de movimientos con el fin de agilizar el proceso de carga y descarga para que no se acumulen las piezas en la zona de recolección.

Como la tarea esta definida por trayectorias lineales en el espacio de trabajo se utilizo el modelo cinemático directo e inverso para el estudio en matlab del movimiento del robot cuando recorre las trayectorias.

Agradecimientos

Deseo expresar mi agradecimiento más sincero al catedrático e investigador de la Universidad Autónoma del Estado de Hidalgo Dr. Omar Arturo Domínguez Ramírez por su ayuda en la preparación de este manuscrito.

Agradezco a la familia Hernández, por el apoyo y la confianza que me tuvo al aceptarme en su empresa, también agradezco a su gerente de planta Ing. Pablo Torres Velázquez, cuya participación en la primera fase de la elaboración de este documento fue crucial para la obtención de la información necesaria.

Gracias también a los miembros del consejo universitario por su valioso apoyo.

A mí papá y mamá
por su extrema paciencia, comprensión y apoyo.

"Si no tienes ganas de ser frustrado jamás en tus deseos, no desees sino aquello que depende de ti."

Epícteto

Contenido

CAPÍTULO 4

EL ROBOT Y CONTROLADOR YAMAHA

CAPÍTULO 5

SISTEMA INTEGRADO Y PRUEBAS EXPERIMENTALES

CAPÍTULO 6

CONCLUSIONES Y PERSPECTIVAS

Índice de figuras

Índice de tablas

Capítulo 1

INTRODUCCIÓN

1.1 Objetivo general

El objetivo de esta tesis es la integración del robot manipulador industrial Yamaha YK550H tipo SCARA, en el sistema de manufactura flexible existente mediante la programación del PLC Yamaha QRCH para realizar la tarea de carga y descarga, planificada a partir de obtener la trayectoria mínima entre dos puntos mediante métodos geométricos, utilizando sensores de proximidad, cuyo propósito es el embalaje de los mecanismos para relojes de pared de la empresa Nissan Time de México S.A de C.V.

1.2 Objetivos específicos

- Integrar la estación robotizada encargada del empaquetamiento de los mecanismos para relojes de pared de la empresa Nissan Time de México.

- Obtención del modelo cinemático directo e inverso de posición del robot Yamaha YK550H.

- Programar la tarea en el PLC Yamaha QRCH a partir de obtener la trayectoria mínima entre dos puntos mediante métodos geométricos

- Apertura de la arquitectura del robot Yamaha YK550H.

- Apertura de la arquitectura del PLC Yamaha QRCH.

- Introducir al lector en el manejo de los robots industriales manipuladores mediante la operación del Yamaha robot controller QRCH series.

- También pretende ofrecer al lector un ejemplo fehaciente de la aplicación en las empresas de tecnologías de vanguardia como lo son los PLC`s y la robótica que incrementan la productividad, y liberan de tareas repetitivas a las personas.

1.3 Antecedentes

Con el paso del tiempo la tecnología ha ido mejorando, desde la invención de la imprenta hasta el desarrollo de máquinas especializadas para tareas, tales como poner tapones a las botellas. Sin embargo esas máquinas no tienen la flexibilidad y versatilidad del brazo humano por lo que no podían tomar y colocar objetos en la posición deseada [8].

Los nuevos sistemas de manufactura están constituidos por robots industriales en la mayor parte del proceso, como es el caso de la industria automovilística, que es la pionera en el uso de robots industriales [7].

Existen muchas aplicaciones para los robots industriales, entre las que destacan: la soldadura por puntos, la pintura en spray y la carga y descarga de piezas. Esta última aplicación es la que plantearemos en este proyecto.

Un ejemplo real de la aplicación de sistemas de manufactura flexibles en nuestro estado es el centro de ensamblado automatizado de la empresa Nissan Time de México S.A de C.V., el cual tiene una distribución de tipo celular, es decir, cuenta con 12 estaciones de ensamblado, conformado por robots Yamaha YK440A y YK420A-I tipo SCARA, los que se encargan de la colocación de las piezas para el armado de los mecanismos de relojes publicitarios de pared, estos

robots son controlados con los PLC's DRHC y QRCH de Yamaha respectivamente, sus efectores finales son pinzas neumáticas (ver apéndice B). Para el área donde se recolectaban los mecanismos una vez ensamblados estos eran tomados por 2 ventosas y colocados en el conveyor que los arrojaba hacia las cajas, después la persona encargada del control de calidad retiraba la caja una vez que estaba llena, para cambiarla por una vacía. Por lo que se decidió colocar en esta área el robot que se había adquirido a Yamaha Robotics, para liberar a la persona de esta tarea y dejarla encargada exclusivamente del control de calidad de los mecanismos.

1.4 Breve estado del arte

Los robots industriales se consideran según el instituto de robótica de América (RIA) como un manipulador automático servo-controlado, reprogramable, capaz de posicionar y orientar piezas, siguiendo trayectorias variables reprogramables, para la ejecución de tareas variadas. Normalmente tiene la forma de uno o varios brazos terminados en una muñeca. Su unidad de control incluye un dispositivo de memoria y ocasionalmente de percepción del entorno. Normalmente su uso es el de realizar una tarea de manera cíclica, pudiéndose adaptar a otra sin cambios permanentes en su estructura.

Los robots pueden ser clasificados como: industriales y móviles. El robot industrial se utiliza para llevar a cabo tareas que puedan ser peligrosas para el hombre, o para mejorar la producción de la industria realizando tareas repetitivas. El robot móvil, tiene la capacidad de desplazarse de un lugar a otro con la ayuda de un sistema de locomoción, siguiendo su camino de acuerdo a la información recibida a través de un sistema de sensores. [11]

Los brazos robóticos han tratado de asemejar el movimiento del brazo humano, por lo que se han construido en base a las articulaciones de estos [11].

Las articulaciones de un brazo robótico suelen moverse mediante motores eléctricos o dispositivos neumáticos. En la mayoría de los robots, la pinza se mueve de una posición a otra cambiando su orientación, el controlador calcula los ángulos de las articulaciones necesarios para llevar la pinza a la posición deseada, este proceso es conocido como cinemática inversa. [4]

1.5 Planteamiento del problema

Este proyecto surge como necesidad de integrar la estación robótica, es decir poner en funcionamiento el PLC y el robot para realizar la tarea de colocar el mecanismo de las siguientes dimensiones 5X5cm con un espesor de 1cm y un peso de aproximadamente 5gr. una vez ensamblado, de la banda transportadora en cajas, para liberar a la persona encargada de esta tarea el robot tiene que evitar la acumulación de los mecanismos en la banda transportadora. Es decir, se trata de encontrar un camino desde una posición y orientación inicial hasta una posición y orientación final, tarea que era desarrollada por un operador humano.

Cabe mencionar que una vez colocados los mecanismos en las cajas, estos serán inspeccionados por el personal de control de calidad, para revisar que estén funcionando correctamente, por lo que no es necesario ordenarlos a la hora de ponerlos en las cajas.

1.6 Solución propuesta

En los robots manipuladores, la trayectoria en el espacio de trabajo suele especificarse por la posición y orientación asociado al efector final.

Un método clásico de planificación de trayectorias que permite obtener los caminos más cortos libres de obstáculos es el de los grafos de vértices o líneas de visibilidad. En el que se elige el camino más corto, formado por segmentos que conectan los vértices de los objetos [11].

De forma más concreta, se puede especificar el movimiento de las siguientes formas:

1) Por puntos consecutivos.

2) Especificando el camino que debe seguir, mediante una trayectoria, tal como una línea recta o un círculo.

Teniendo en mente el marco teórico expuesto y gracias al teach pendant con el que se pueden obtener las coordenadas fácilmente, que posteriormente se introducirán en el código del programa, claro que esto se puede realizar por otros métodos, pero por lo práctico se utilizo el teach pendant para la localización de las coordenadas que a juicio del operador se consideran sean las más cortas de la posición y orientación inicial hasta la posición y orientación final, estas son las que se van a utilizar para la programación de la rutina en el PLC del robot manipulador Yamaha YK550H.

Como la tarea esta definida por trayectorias lineales en el espacio de trabajo se utilizo el modelo cinemático directo e inverso para el estudio del movimiento del robot cuando recorre las trayectorias.

1.7 Justificación

La historia nos demuestra que el mundo industrial ha pasado por diversos períodos tecnológicamente revolucionarios. Como la revolución industrial caracterizada por las aplicaciones de la electricidad, la mecánica y la química en los procesos industriales mecanizados para la fabricación de bienes a gran escala [8].

Actualmente con las llamadas nuevas tecnologías, entre las que hay que destacar a la microelectrónica, las telecomunicaciones, la bioingeniería y la robótica.

Estas están provocando que las empresas adopten tecnologías de vanguardia que consuman poca energía y que incrementen la productividad y la calidad del producto final, pero estas tecnologías generan desempleos, un ejemplo fehaciente de esto son los robots a los que se les asignan trabajos duros, peligrosos y repetitivos. Por lo que los obreros que son sustituidos por los robots deberán ser ocupados en trabajos de mayor nivel intelectual y humano, por ello se requerirán expertos y técnicos a nivel operativo para los sistemas robotizados [7].

Es por esto, que esta tesis pretende introducir y familiarizar al lector con términos, conceptos y operación de sistemas robotizados.

El desarrollo de este trabajo de tesis surge ante la necesidad de automatizar el área de empaquetado en el centro de ensamble automatizado de la empresa Nissan Time de México, esto debido a que se necesitaba agilizar el proceso y al mismo tiempo liberar a la persona encargada de esta tarea repetitiva y colocarla en el área de control de calidad, para lograr esto se contó con el robot Yamaha YK550H y el controlador Yamaha QRCH los cuales se integraron al sistema de manufactura flexible existente en la planta.

1.8 Organización de la tesis

Este trabajo esta formado por seis capítulos, los cuales se integran de la siguiente manera:

Capítulo 1: en este se inicia planteando el objetivo general de la tesis y los objetivos específicos de la misma, una descripción de los antecedentes de los sistemas de manufactura automatizados, se da a conocer la problemática, así como la solución propuesta y la justificación de la elaboración de esta tesis, por último se describe la organización del texto y su contenido.

Capítulo 2: este capítulo hace referencia a la teoría acerca del control de los robots manipuladores industriales, inicia con una reseña histórica, presenta la estructura, las configuraciones, las aplicaciones, los modelados matemáticos de su cinemática, algunas estrategias de control, la planificación de trayectorias y concluye mencionando los lenguajes de programación de los robots manipuladores industriales.

Capítulo 3: aquí se mencionan aspectos de los controladores lógicos programables, se inicia con una perspectiva histórica, el principio del funcionamiento de los PLC, se presentan las formas en que se programan los PLC´s, los tipos de PLC y sus aplicaciones.

Capítulo 4: contiene la información acerca del controlador y del robot utilizado para la solución de la problemática, en lo referente al controlador del robot se inicia describiendo las partes constitutivas, su funcionamiento y su modo de operación. En cuanto al robot se menciona su configuración, sus datos técnicos y su lenguaje de programación.

Capítulo 5: son los resultados y puesta al punto de la rutina, menciona la integración de la estación robótica, la programación de la tarea en el PLC, el modelado cinemático directo e inverso del robot para la obtención del algoritmo para programar en matlab el comportamiento de los movimientos del robot cuando ejecuta la tarea de carga y descarga

Capítulo 6: contiene las conclusiones y perspectivas de esta tesis.

Finaliza con los apéndices que contienen la información de la operación de la interfaz para programar el robot, del optoacoplador que utiliza el controlador, así como información acerca del efector final que se utilizo en el robot manipulador, el listado del programa en matlab del modelado cinemático del robot, un apartado de los acrónimos y abreviaturas, un glosario de términos utilizados y concluye haciendo referencia a la bibliografía consultada para la elaboración de esta tesis.

Capítulo 2

ROBOTS MANIPULADORES INDUSTRIALES

2.1 Introducción

En este capítulo se presenta el marco teórico referencial acerca de los robots manipuladores en general comenzando con una introducción acerca de los robots manipuladores industriales, sus características, métodos de obtención de la cinemática directa e inversa para determinar y representar la geometría espacial de los robots, con respecto a un sistema de referencia fijo, se describen algunas estrategias de control para robots manipuladores y se da a conocer como planificar una trayectoria, concluye describiendo la manera de programación de los robots.

2.2 Reseña histórica de la robótica

La palabra robot es de origen eslavo. En ruso significa trabajo. En checo significa trabajo forzado. En 1921 el escritor checo Karol Capek escribió Rossums Universal Robots, una obra teatral de gran éxito en Broadway. En esta obra unos androides trabajaban más del doble que un ser humano. El término robotics (robótica) se debe a Isaac Asimov. El primer brazo articulado o manipulador fue construido por Harold Roselund de la compañía Devilviss, ahora desaparecida, en 1938. Se usaba para pintar en spray. De más esta decir que al no existir en aquella época computadoras digitales, la programación de una tarea dada era altamente laboriosa por lo cual el primer manipulador no tuvo éxito. Durante la segunda guerra mundial fue desarrollado el teleoperador, es decir, dos manos mecánicas controladas a distancia por un ser humano mediante encadenamientos mecánicos[5].

George Devol es considerado como el padre del robot. En 1946 inventó un aparato mecánico que permitía repetir una secuencia de movimientos a una máquina herramienta. Ese mismo año Eckert y Mauchly construyeron la primera computadora digital, ENIAC. En 1952 se construyo en el MIT la primera máquina de control numérico. En 1952 Devol patento el primer brazo con memoria. Esta máquina era capaz de mover la herramienta de trabajo de punto a punto. En 1957 la corporación Planet produjo el primer robot comercial. En 1960 Devol vendió su patente a la corporación Condec que empezó a producir el robot Unimate en su subsidiaria Unimation. En 1962 General Motors instaló el primer Unimation en su planta de fundición por troquel [5].

A partir de 1965 varios centros importantes de investigación tales como el MIT, etc. Empezaron la investigación en robótica y áreas asociadas. El primer robot controlado por microcomputador fue producido por Cincinnati Milacron en 1973. En 1978 Unimation desarrollo el PUMA (Programmable Universal Machine for Assambling) [5].

2.3 Estructura de los Robots manipuladores.

Los robots manipuladores son, brazos articulados que están montados sobre una base que está sujeta al suelo.

Un robot manipulador operado individualmente necesita como mínimo los siguientes componentes:

a) El brazo consiste en un sistema de articulaciones mecánicas (eslabones, engranajes, transmisión por cadena o correa), actuadores (motores eléctricos o hidráulicos) y sensores de posición usados en el sistema de control de bucle cerrado.

b) El controlador, generalmente basado en un microcomputador, que recibe las señales de los sensores de posición y envía comandos a la fuente de potencia controlada.

c) La unidad conversora de potencia que alimenta los motores que actúan las articulaciones.

Dependiendo del tipo de aplicación, un robot puede contar también con sensores externos, de los cuales el más poderoso es un sistema de visión consistente en cámara de video, interfaz y microcomputador procesador de imágenes. La información obtenida por este sistema pasa al microcomputador que controla el robot y es usada para dirigir el movimiento de éste [5].

2.3.1 Tipos de articulaciones.

Existen diferentes tipos de articulaciones. Las más utilizadas en robótica son las que se indican en la figura 2.2.1

La articulación de rotación suministra un grado de libertad consiste en una rotación alrededor del eje de la articulación. Está articulación es, la más empleada.

En la articulación prismática el grado de libertad consiste en una traslación a lo largo del eje de la articulación.

En la articulación cilíndrica existen dos grados de libertad: una rotación y una traslación.

La articulación planar está caracterizada por el movimiento de desplazamiento en un plano, existiendo por lo tanto, dos grados de libertad.

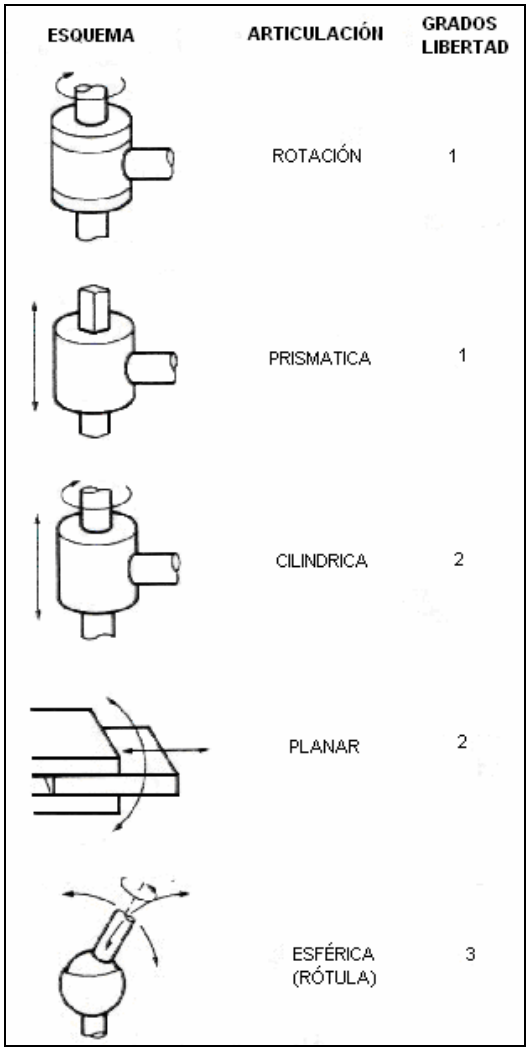


Figura. 2.2.1 Tipos de articulaciones robóticas [4].

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

Los grados de libertad son el número de parámetros independientes que fijan la posición del efector final. El número de grados de libertad suele coincidir con el número de eslabones de la cadena cinemática [11].

2.3.2 Estructuras básicas.

La estructura típica de un manipulador consiste en un brazo compuesto por elementos con articulaciones entre ellos. En el último enlace se coloca un órgano terminal o efector final tal como una pinza o un dispositivo especial para realizar operaciones [11].

Se consideran, en primer lugar, las estructuras más utilizadas como brazo de un robot manipulador. Estas estructuras tienen diferentes propiedades en cuanto a espacio de trabajo y accesibilidad a posiciones determinadas. En la figura 2.2.2 se muestran cuatro configuraciones básicas [11].

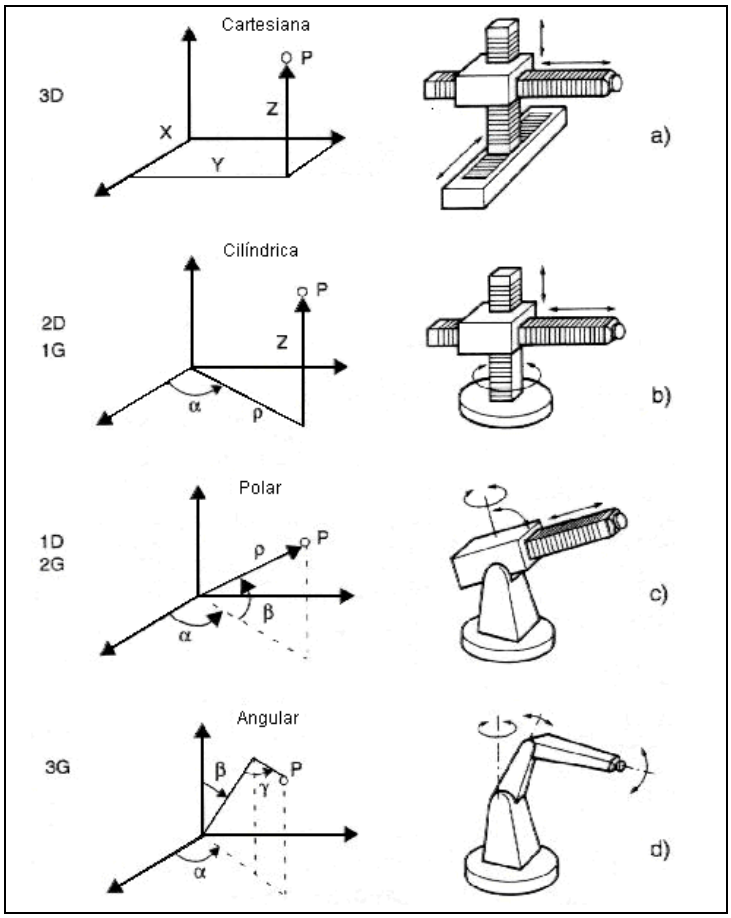


Figura. 2.2.2 Estructuras básicas de manipuladores [4].

El espacio de trabajo es el conjunto de puntos en los que puede situarse el efector final del manipulador. Corresponde al volumen encerrado por las superficies que determinan los puntos a los que accede el manipulador con su estructura totalmente extendida y totalmente plegada [11].

Por otra parte, todos los puntos del espacio de trabajo no tienen la misma accesibilidad. Los puntos de accesibilidad mínima son los que las superficies que delimitan el espacio de trabajo ya que a ellos solo puede llegarse con una única orientación [11].

Configuración cartesiana.

La configuración tiene tres articulaciones prismáticas (3D o estructura PPP). Esta configuración es bastante usual en estructuras industriales, tales como pórticos, empleadas para el transporte de cargas voluminosas. La especificación de posición de un punto se efectúa mediante las coordenadas cartesianas (x, y, z). Los valores que deben tomar las variables articulares corresponden directamente a las coordenadas que toma el extremo del brazo. Esta configuración no resulta adecuada para acceder a puntos situados en espacios relativamente cerrados y su volumen de trabajo es pequeño cuando se compara con el que puede obtenerse con otras configuraciones [4].

Configuración cilíndrica.

Esta configuración tiene dos articulaciones prismáticas y una de rotación (2D, 1G). La primera articulación es normalmente de rotación (estructura RPP). La posición se especifica de forma natural en coordenadas cilíndricas. Esta configuración puede ser de interés en una célula flexible, con el robot situado en el centro de la célula sirviendo a diversas máquinas dispuestas radialmente a su alrededor. El volumen de trabajo de esta estructura RPP (o de la PRP),

suponiendo un radio de giro de 360 grados y un rango de desplazamiento de l, es el de un toro de sección cuadrada de radio interior l y radio exterior $2l$. Se demuestra que el volumen resultante es: $3\pi l^3$ [4].

Configuración polar o esférica.

Está configuración se caracteriza por dos articulaciones de rotación y una prismática (2G, 1D o estructura RRP). En este caso las variables articulares expresan la posición del extremo del tercer enlace en coordenadas polares.

En un manipulador con tres enlaces de longitud l, el volumen de trabajo de esta estructura, suponiendo un radio de giro de 360 grados y un rango de desplazamiento de l, es el que existe entre una esfera de radio $2l$ y otra concéntrica de radio l Por consiguiente el volumen es $(28 / 3)\pi l^3$ [4].

Configuración angular.

Esta configuración es una estructura con tres articulaciones de rotación (3G o RRR). La posición del extremo final se especifica de forma natural en coordenadas angulares.

La estructura tiene un mejor acceso a espacios cerrados y es fácil desde el punto de vista constructivo. Es muy empleada en robots manipuladores industriales, especialmente en tareas de manipulación que tengan una cierta complejidad. La configuración angular es la más utilizada en educación y actividades de investigación y desarrollo. En esta estructura es posible conseguir un gran volumen de trabajo. Si la longitud de sus tres enlaces es de l, suponiendo un radio de giro de 360 grados, el volumen de trabajo sería el de una esfera de radio $2l$ [4].

Configuración SCARA.

Esta configuración está especialmente diseñada para realizar tareas de montaje en un plano. Está constituida por dos articulaciones de rotación con respecto a dos ejes paralelos, y una de desplazamiento en sentido perpendicular al plano. El volumen de trabajo de este robot, suponiendo segmentos de longitud ℓ, un radio de giro de 360 grados y un rango de desplazamiento de ℓ, es de $4\pi\ell^3$ [4].

2.4 Aplicaciones de los robots industriales

En la actualidad se ha generalizado el uso de robots manipuladores en muchas aplicaciones que requieren movimientos repetitivos sencillos, pero la industria automovilística que fue la pionera en el uso de robots, continua siendo la que más los usa, siendo los tipos principales de aplicación los siguientes: soldadura de puntos, pintura, manipulación de objetos. En ninguna de estas es necesario un sistema de visión, siempre y cuando se organice el trabajo en forma tal que las partes a soldar, pintar, etc. se encuentren en el lugar determinado.

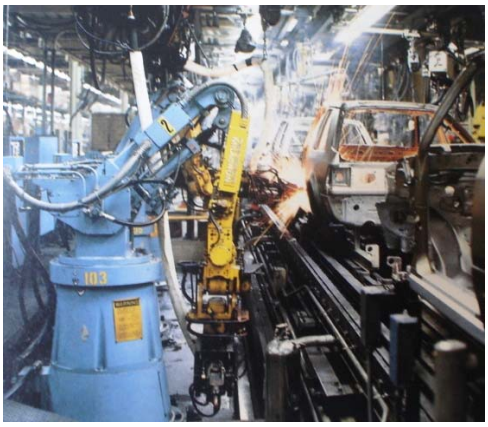


Figura 2.4 Soldadura con robots [7]

Figura 2.4.1 Aplicación en pick and place en Nissan Time

2.5 Modelado matemático de robots manipuladores

La cinemática estudia el movimiento pero no la manera de controlar este movimiento. Para el control de un brazo manipulador necesitamos, además de su descripción cinemática, una formulación matemática de su dinámica [5].

Un brazo manipulador tiene varios grados de libertad, generalmente tres sin contar los del efector final o mano. El movimiento de cada eslabón puede referirse al sistema de coordenadas del eslabón anterior, yendo desde el efector final hacia la base (o punto inmóvil de referencia). En esta forma seremos capaces de describir el movimiento deseado de la herramienta de trabajo en función del movimiento de cada una de las articulaciones. Para lograr esto necesitaremos definir los parámetros de eslabón y en base a ellos las matrices homogéneas de cada eslabón y del espacio de trabajo [5].

Cada eslabón, que será definido más adelante, posee su propio sistema de coordenadas, con el origen sobre la articulación que controla el movimiento. El movimiento puede ser rotatorio o de traslación [4].

Por otra parte, la cinemática del robot trata también de encontrar las relaciones entre las velocidades del movimiento de las articulaciones y las del extremo. Esta relación viene dada por el modelo diferencial expresado mediante la matriz Jacobiana [11].

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

La cinemática del brazo del robot trata con el estudio analítico de la geometría del movimiento de un robot con respecto a un sistema de coordenadas de referencia fijo como una función del tiempo sin considerar las fuerzas-momentos que originan dicho movimiento. Así pues, trata con la descripción analítica del desplazamiento espacial del robot como función del tiempo, en particular las relaciones entre variables espaciales de tipo de articulación y la posición y orientación del efector final del robot [1].

Representación de la posición

La cinemática del robot RR (dos articulaciones rotacionales) es más difícil de analizar que la del robot LL (dos articulaciones lineales) [4].

La posición del extremo del brazo se puede representar de varias formas. Una de ellas es utilizar los dos ángulos de articulación θ_1 y θ_2. Este procedimiento se conoce como representación en espacio de articulación y se puede definir como: $P_j(\theta_1, \theta_2)$ (2.1)

Otra forma de definir la posición del brazo es en espacio universal. Este procedimiento utiliza el sistema de coordenadas cartesianas que es externo al robot. El origen del sistema se localiza en la base del robot. La posición del extremo del brazo se definirá dentro de un espacio universal como $P_w(x, y)$. Este concepto de un punto en el espacio universal se puede extender fácilmente a tres dimensiones; esto es $P_w(x, y, z)$ (2.2)

Con el objeto de utilizar ambas representaciones, se deberá ser capaz de realizar transformaciones desde una a la otra. La transformación desde el espacio de articulaciones al espacio universal se denomina transformación directa. La

transformación desde el espacio universal al espacio de articulación se denomina transformación inversa [4].

Cinemática directa.

Se utiliza fundamentalmente el álgebra vectorial y matricial para representar y describir la localización de un objeto en el espacio tridimensional con respecto a un sistema de referencia fijo. Dado que un robot puede considerarse como una cadena cinemática formada por objetos rígidos o eslabones unidos entre sí mediante articulaciones, se puede establecer un sistema de referencia fijo situado en la base del robot y describir la localización de cada uno de los eslabones con respecto a dicho sistema de referencia. De esta forma, el problema cinemático directo se reduce a encontrar una matriz homogénea de transformación T que relacione la posición y orientación del extremo del robot respecto del sistema de referencia fijo situado en la base del mismo. Esta matriz T será función de las coordenadas articulares. Un sistema de mando de robot causa a un manipulador remoto, seguir una trayectoria de referencia estrechamente en un marco de referencia cartesiano en el espacio de trabajo, sin el recurso a un modelo matemático intensivo de dinámica del robot y sin el conocimiento del robot y parámetros de carga. El sistema, derivado de la teoría lineal multivariable, utiliza a los manipuladores delanteros relativamente simples y controladores de retroalimentación con modelo y adaptable de referencia del mando. El sistema requiere dimensiones de posición y velocidad del extremo manipulador del efector. Éstos pueden obtenerse directamente de los sensores ópticos o por cálculo que utiliza las relaciones de la cinemática conocidas entre el manipulador modelado y el extremo de la juntura de la posición del efector. Derivando las ecuaciones de control, las ecuaciones diferenciales no lineales acopladas a la dinámica del robot, expresan primero la forma general de la cinemática, entonces la linealización por cálculo de perturbaciones sobre una especifica operación del punto en las coordenadas cartesianas del extremo del efector. El modelo matemático resultante

es un sistema multivariable lineal de orden de $2n$ (donde n = es el número de coordenadas espaciales independientes del manipulador) esto expresa la relación entre los incrementos del actuador de n voltajes de control (las entradas) y los incrementos de las coordenadas de n, la trayectoria de extremo del efector (los rendimientos). La trayectoria del efector incrementa la referencia, la trayectoria se incrementa: esto requiere la retroalimentación independiente y controladores de manipulación. Para este propósito, le basta aplicar posición y retroalimentación de velocidad a través de la matriz de n x n posición y velocidad, la matriz de ganancia de retroalimentación [1].

Resolución del problema cinemático directo mediante matrices de transformación homogénea.

La resolución del problema cinemático directo consiste en encontrar las relaciones que permiten conocer la localización espacial del extremo del robot a partir de los valores de sus coordenadas articulares. Así, si se han escogido coordenadas cartesianas y ángulos de Euler para representar la posición y orientación del extremo de un robot de seis grados de libertad, la solución al problema cinemático directo vendrá dada por las relaciones [1]:

$$x = f(q_1, q_2, q_3, q_4, q_5, q_6) \quad (2.2.1)$$

$$y = f_y(q_1, q_2, q_3, q_4, q_5, q_6) \quad (2.2.2)$$

$$z = f_z(q_1, q_2, q_3, q_4, q_5, q_6) \quad (2.2.3)$$

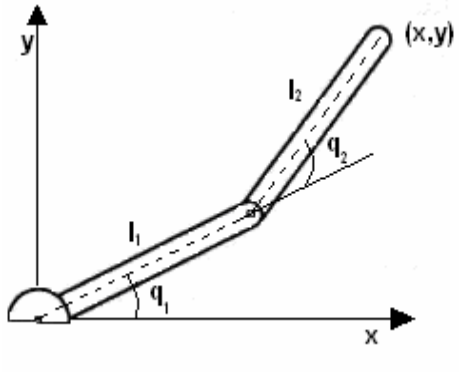
$$\alpha = f_\alpha(q_1, q_2, q_3, q_4, q_5, q_6) \quad (2.2.4)$$

$$\beta = f_\beta(q_1, q_2, q_3, q_4, q_5, q_6) \quad (2.2.5)$$

$$\gamma = f_\gamma(q_1, q_2, q_3, q_4, q_5, q_6) \quad (2.2.6)$$

La obtención de estas relaciones no es en general complicada, siendo incluso en ciertos casos (robots de pocos grados de libertad) fácil de encontrar

mediante simples consideraciones geométricas. Por ejemplo, para el caso de un robot con 2 grados de libertad es fácil comprobar que [1]:



$$x = l_1 \cos(q_1) + l_2 \cos(q_1 + q_2) \qquad (2.2.7)$$

$$y = l_1 \cos(q_1) + l_2 \cos(q_1 + q_2) \qquad (2.2.8)$$

Para robots de más grados de libertad puede plantearse un método sistemático basado en la utilización de las matrices de transformación homogénea. En general, un robot de n grados de libertad esta formado por n eslabones unidos por n articulaciones, de forma que cada par articulación-eslabón constituye un grado de libertad. A cada eslabón se le puede asociar un sistema de referencia solidario a él y utilizando las transformaciones homogéneas, es posible representar las rotaciones y traslaciones relativas entre los distintos eslabones que componen el robot. Normalmente, la matriz de transformación homogénea que representa la posición y orientación relativa entre los sistemas asociados a dos eslabones consecutivos del robot se le suele denominar ${}^{(i-1)}A_i$. Así pues, ${}^{0}A_i$ describe la posición y orientación del sistema de referencia solidario al primer eslabón con respecto al sistema de referencia solidario a la base, ${}^{1}A_2$ describe la posición y orientación del segundo eslabón respecto del primero, etc. Del mismo modo, denominando ${}^{0}A_k$ a las matrices resultantes del producto de las matrices ${}^{(i-1)}A_i$ con i desde 1 hasta k, se puede representar de forma total o parcial la cadena cinemática que forma el robot. Así, por ejemplo, la posición y orientación del sistema solidario con el segundo eslabón del robot con respecto al sistema de coordenadas de la base se puede expresar mediante la matriz ${}^{0}A_2$ [1]:

$$^{0}A_{2}=^{0}A_{1}(^{1}A_{2}) \tag{2.3}$$

De manera análoga, la matriz $^{0}A_{3}$ representa la localización del sistema del tercer eslabón:

$$^{0}A_{3}=^{0}A_{1}(^{1}A_{2})(^{2}A_{3}) \tag{2.3.1}$$

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

$$T=^{0}A_{6}=^{0}A_{1}(^{1}A_{2})(^{2}A_{3})(^{3}A_{4})(^{4}A_{5})(^{5}A_{6}) \tag{2.3.2}$$

Cinemática inversa.

El objetivo del problema cinemático inverso consiste en encontrar los valores que deben adoptar las coordenadas articulares del robot q= (q1, q2,..., qn)^{T} para que su extremo se posicione y oriente según una determinada localización espacial. Así como es posible abordar el problema cinemático directo de una manera sistemática a partir de la utilización de matrices de transformación homogéneas, e independientemente de la configuración del robot, no ocurre lo mismo con el problema cinemático inverso, siendo el procedimiento de obtención de las ecuaciones fuertemente dependiente de la configuración del robot. Los métodos geométricos permiten tener normalmente los valores de las primeras variables articulares, que son las que consiguen posicionar el robot. Para ello utilizan relaciones trigonometrícas y geométricas sobre los elementos del robot. Se suele recurrir a la resolución de triángulos formados por los elementos y articulaciones del robot [1].

Resolución del problema cinemático inverso por métodos geométricos.

Como se ha indicado, este procedimiento es adecuado para robots de pocos grados de libertad o para el caso de que se consideren solo los primeros grados de libertad, dedicados a posicionar el extremo. El procedimiento en si se basa en encontrar suficiente número de relaciones geométricas en las que intervendrán las coordenadas del extremo del robot, sus coordenadas articulares y las dimensiones físicas de sus elementos. Para mostrar el procedimiento a seguir se va a aplicar el método a la resolución del problema cinemático inverso de un robot con 3 grados de libertad de rotación (estructura típica articular). El dato de partida son las coordenadas (P_x, P_y, P_z) referidas a (S_0) en las que se requiere posicionar su extremo. El robot posee una estructura planar, quedando este plano definido por el ángulo de la primera variable articular q_1. El valor de q_1 se obtiene inmediatamente como [1]:

$$q_1 = tg^{-1}\left(\frac{P_y}{P_x}\right) \tag{2.3.3}$$

Considerando ahora únicamente los elementos l_2 y l_3 (ver fig. 2.5.1) que están situados en un plano y utilizando el teorema del coseno, se tendrá [1]:

$$r^2 = (P_x)^2 + (P_y)^2 \tag{2.3.4}$$

$$r^2 + (P_x)^2 = (l_2)^2 + (l_3)^2 + 2(l_2)(l_3)\cos(q_3) \tag{2.3.5}$$

$$\cos(q_3) = \frac{P_x^2 + P_y^2 + P_z^2 - l_2^2 - l_3^2}{2(l_2)(l_3)} \tag{2.3.6}$$

Esta expresión permite obtener q_1 en función del vector de posición del extremo P. No obstante, por motivos de ventajas computacionales, es más conveniente utilizar la expresión del arco tangente en lugar del arco seno. Puesto que [1]:

$$sen(q_3) = \pm\sqrt{(1 - cos^2(q_3))} \tag{2.4}$$

Se tendrá que:

$$q_3 = tg^{-1}\left(\frac{\sqrt{(1 - cos^2(q_3))}}{cos(q_3)}\right) \tag{2.4.1}$$

$$cos(q_3) = \frac{P_x^2 + P_y^2 + P_z^2 - l_2^2 - l_3^2}{2(l_2)(l_3)} \tag{2.4.2}$$

Como se ve, existen dos posibles soluciones para q_3 según se tome el signo positivo o negativo de la raíz. Estas corresponden a las configuraciones de codo arriba (fig. 2.5.1.a) y codo abajo del robot (fig. 2.5.1-b) [1].

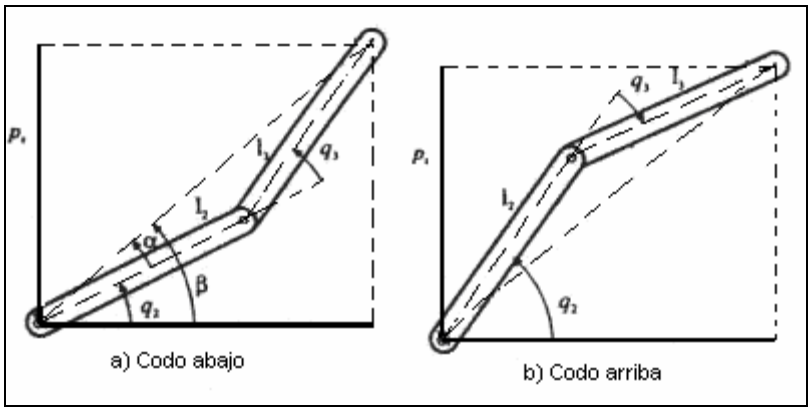


Figura 2.5.1 Posiciones de codo arriba y codo abajo del brazo

El cálculo de q_2 se hace a partir de la diferencia entre β y α [1]:

$$q_2 = \beta - \alpha \tag{2.4.3}$$

Siendo:

$$\beta = tg^{-1}\left(\frac{P_z}{r}\right) = tg^{-1}\left(\frac{P_z}{\pm\sqrt{\left(P_x^2 + P_y^2\right)}}\right) \tag{2.4.4}$$

$$\alpha = tg^{-1}\left(\frac{l_3 sen(q_3)}{l_2 + l_3 \cos(q_3)}\right) \tag{2.4.5}$$

Luego finalmente:

$$q_2 = tg^{-1}\left(\frac{P_z}{\pm\sqrt{\left(P_x^2 + P_y^2\right)}}\right) - tg^{-1}\left(\frac{l_3 sen(q_3)}{l_2 + l_3 \cos(q_3)}\right) \tag{2.4.6}$$

De nuevo los dos posibles valores según la elección del signo dan lugar a dos valores diferentes de q_2 correspondientes a las configuraciones codo arriba y abajo [1].

Matriz Jacobiana.

El modelado cinemático de un robot busca las relaciones entre las variables articulares y la posición (expresada normalmente en forma de coordenadas cartesianas) y orientación del extremo del robot. En esta relación no se tienen en cuenta las fuerzas o pares que actúan sobre el robot (actuadores, cargas, fricciones, etc.) y que pueden originar el movimiento del mismo. Sin embargo, si que debe permitir conocer, además de la relación entre las coordenadas articulares y del extremo, la relación entre sus respectivas derivadas. Así, el sistema de control del robot debe establecer que velocidades debe imprimir a cada articulación (a través de sus respectivos actuadores) para conseguir que el

extremo desarrolle una trayectoria temporal concreta, por ejemplo, una línea recta a velocidad constante. Para este y otros fines, es de gran utilidad disponer de la relación entre las velocidades de las coordenadas articulares y las de posición y orientación del extremo del robot. La relación entre ambos vectores de velocidad se obtiene a través de la denominada matriz Jacobiana [1].

La matriz Jacobiana directa permite conocer las velocidades del extremo del robot a partir de los valores de las velocidades de cada articulación. Por su parte, la matriz Jacobiana inversa permitirá conocer las velocidades determinadas en el extremo del robot [1].

Relaciones diferenciales.

El método más directo para obtener la relación entre las velocidades articulares y del extremo del robot consiste en diferenciar las ecuaciones correspondientes al modelo cinemático directo [1].

2.6 Estrategias de control para robots manipuladores

Existen varios métodos de control de robots manipuladores, el más sencillo de los cuales es el control de posición por bucle cerrado de cada articulación, ignorando los efectos de acoplamiento. Como la mayoría de las articulaciones, si se aproximan por un sistema de segundo orden, son sumamente subamortiguados, cuando se usa control de posición independiente para cada articulación, se añade control de velocidad (retroalimentación tacométrica), lo cual mejora la respuesta notablemente. Existen métodos de control adaptivo. Otro método es el de momento calculado. Por último, se han propuesto métodos de control por medio de redes neuronales [5].

Si usamos un sistema de control simple, tal como un PID (proporcional – integral – derivativo) para el control del robot en bucle cerrado, cada articulación es controlada independientemente, como si no existiera acoplamiento entre ellas, y por lo tanto sólo se necesita un modelo dinámico simple para cada eslabón, esto es, un sistema de segundo orden. Este método, aunque relativamente satisfactorio, no funciona bien si se necesita controlar la posición de la herramienta en forma continua, mientras el brazo está en movimiento, por ejemplo la soldadura autógena [5].

Control de posición

Para aplicar el método de control de posición, deben conocerse la inercia, coeficiente de fricción y coeficiente de torsión para cada articulación. Generalmente el coeficiente de torsión es despreciable en robots industriales. La inercia puede calcularse para cada eslabón conociendo las dimensiones y el material, y el coeficiente de fricción puede determinarse experimentalmente. El motor que mueve las articulaciones a través de un tren de engranajes, también tiene parámetros conocidos. Con estos datos se tiene la información necesaria para diseñar un sistema de control de posición para cada articulación.Si deseamos usar un método más refinado, debemos obtener un modelo dinámico que tome en cuenta el acoplamiento entre eslabones [5].

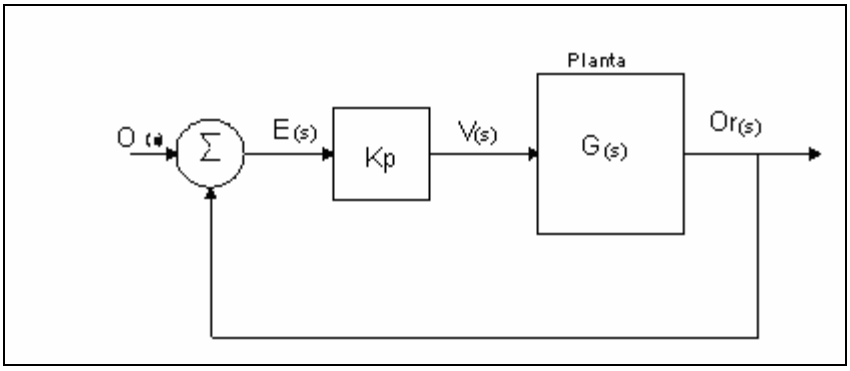


Figura 2.6 Sistema de control de posición con retroalimentación unitaria [5]

En la figura 2.6 se muestra un sistema de control de posición con retroalimentación unitaria. En este sistema E(s) es la diferencia entre el ángulo de referencia y el de salida (en el eje del motor) en el dominio ´s´. Kp es la constante de proporcionalidad, que se obtiene del amplificador de potencia (fuente controlada de c.c.) y G(s) es el motor con el tren de engranajes y la carga reflejada a su eje. Con este sistema de control se obtiene que $\xi = 0.149$; evidentemente este valor es demasiado bajo (para amortiguación critica $\xi = 1$). Podría lograrse un coeficiente de amortiguación más alto reduciendo Kp (haciéndolo menor que uno). Sin embargo, esto tendría un efecto negativo en el error estacionario para entrada "rampa" (velocidad constante), como cuando el robot sigue una trayectoria determinada (sabemos de teoría de control que para un sistema con una integración, el error estacionario es cero para entrada escalón e inversamente proporcional a Kp para entrada rampa). La única forma de aumentar el coeficiente de amortiguación sin afectar el ancho de banda del sistema es utilizando retroalimentación tacométrica o añadiendo a Kp un término proporcional a la derivada del error. La retroalimentación tacométrica puede implementarse en un sistema analógico mediante un tacómetro en el bucle de retroalimentación. La respuesta con retroalimentación tacométrica es subamortiguada y tiene un tiempo de asentamiento muy inferior al sistema con sólo Kp. Esto nos indica que si elegimos controlar el brazo con un simple sistema de control de posición, es imprescindible incluir retroalimentación derivativa. Si además se incluye retroalimentación integrativa (un término Ki/s) el error estacionario para entrada rampa será cero. Sin embargo, la retroalimentación integrativa tiene un efecto desestabilizador en el sistema (ahora es de tercer orden, en vez de segundo orden) y debe diseñarse cuidadosamente para que sea estable [5].

En el desarrollo anterior no hemos tomado en cuenta el efecto de la gravedad. Este efecto puede considerarse como una perturbación, una entrada indeseada al sistema. Esta perturbación ocurre en la parte mecánica del brazo, como se indica en la figura 2.6.1 a la entrada del bloque inercia y fricción [5].

Es posible compensar, al menos una parte el efecto de la gravedad por medio de “alimentación adelantada” (feed-forward compensation). En el mismo punto donde ocurre la perturbación, se aplica una señal igual a D(s) con signo positivo.

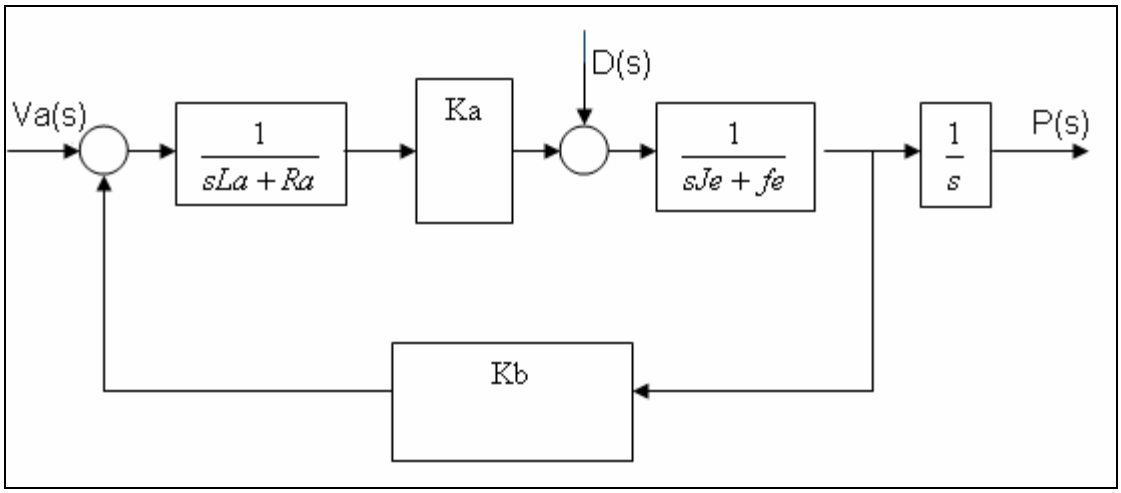


Figura 2.6.1 El motor con la carga mecánica, la entrada (Va) y el disturbio D(s) [5]

El problema es que D(s) no es una constante, ya que al variar la posición del brazo y al variar la carga, varia el efecto de la gravedad. En general puede decirse que aunque sea imperfecta, la compensación de gravedad produce una prestación mejor que si no se aplica. También hay que recordar que en aplicaciones en las cuales el robot sólo debe moverse entre puntos, sin seguir una trayectoria determinada, este efecto tiene poca importancia. Como se mencionó más arriba, si se desea que el error estacionario para entrada rampa (movimiento continuo siguiendo una trayectoria) sea cero, debemos incorporar control integrativo en el sistema [5].

Control por momento computado

Debido a que es posible obtener un modelo dinámico relativamente exacto del robot que toma en cuenta las interacciones debido a las fuerzas centrifugas, gravedad y de Coriolis, etc.. Usando este método puede diseñarse un sistema de control para el conjunto del robot más efectivo que el control de posición de cada articulación. Este sistema de control se llama control por momento computado ya que hace uso del modelo dinámico que expresa el valor de momentos en función de la posición instantánea de los eslabones [5].
De acuerdo al modelo dinámico tenemos:

$$\tau(t) = D[q(t)]q''(t) + h[q, q'] + c[q(t)] \qquad (2.4.7)$$

$D[q(t)]q''(t)$ es el término que “mueve el brazo”

$h[q, q']$ representa el efecto indeseado de fuerzas centrifugas y de Coriolis

$c[q(t)]$ representa la carga debida a la gravedad

Otros métodos de control, no se detallan por que requieren conocimientos avanzados, especialmente de control adaptivo. Como ya se ha dicho la gran mayoría de los robots industriales usan un sistema de control PID [5].

2.7 Planificación de tareas en robots manipuladores.

En aplicaciones industriales, hay casos en los cuales es esencial seguir una trayectoria determinada. Ejemplos sencillos y prácticos son: soldadura autógena, corte de materiales, etc. Pero en generalmente estamos interesados en mover la herramienta desde una posición a una nueva posición, de tal forma que no se produzcan colisiones y se optimice un cierto criterio. El criterio más usual es la minimización de la longitud del camino [4].

El problema, en su planteamiento general, es muy complejo, por lo que suelen adoptarse una serie de hipótesis simplificadoras entre las cuales las más comunes son [11]:

- Entorno conocido (modelo previo)
- Entorno estático
- Obstáculos modelados como polígonos o mediante segmentos lineales
- Robot modelado como un punto
- Robot desplazándose lentamente
- Robot ejecutando perfectamente los caminos

Las dos primeras hipótesis son muy frecuentes en los métodos de planificación que emplean modelos [11].

La tercer hipótesis implica un planteamiento del problema en dos dimensiones. Este planteamiento es valido para robots móviles navegando en el plano pero, en general no es admisible para manipuladores [11].

De igual forma el modelado del robot como un punto es una condición que sólo sirve como una referencia para obtener resultados que después pueden ser generalizados mediante el empleo de modelos apropiados del robot y su entorno.

Las hipótesis de movimientos lentos y de ejecución perfecta de los caminos transfieren los problemas de comportamiento dinámico y precisión a los niveles inferiores de control del robot y obviamente su aproximación a la realidad depende de las condiciones de operación y prestaciones exigidas el sistema de control de movimientos. En cualquier caso resulta imprescindible tenerlas en cuenta para establecer condiciones seguras en la planificación [11].

La generación de trayectorias puede plantearse tanto en el espacio cartesiano como en el espacio de las articulaciones. En el espacio cartesiano la especificación de caminos se realiza en términos de las posiciones y orientaciones del efector final con respecto a un sistema de referencia, sin tener que realizar previamente la transformación cinemática inversa [11].

Existen diferentes problemas geométricos que limitan la aplicación de los métodos en el espacio cartesiano. Entre estos cabe mencionar [11]:

1) Puntos intermedios inalcanzables.

Para un manipulador con dos articulaciones, el problema es que el enlace 2 es más corto que el 1, por tanto, el espacio de trabajo tiene un agujero en el centro cuyo radio es la diferencia de radios.

El movimiento de A a B es posible en el espacio articular. Sin embargo, la línea recta del espacio cartesiano contiene puntos que no son alcanzables. Existen sistemas de control que notifican al operador esta circunstancia. En otros comienza el movimiento hasta que alcanza el límite de una articulación, momento en el cual se para.

2) Velocidades articulares altas cerca de las singularidades.

El manipulador esta siguiendo una línea recta y aproximándose a una singularidad. En este caso, una o más velocidades crecen hacia infinito.

Se trata de mover el extremo final del manipulador a velocidad lineal constante a lo largo de la línea recta. Todos los puntos son alcanzables pero en cuanto el robot pasa por la mitad, la velocidad de la primera articulación es muy alta. Esta velocidad es tanto más alta cuanto más se aproxima el camino al eje de la primera articulación. Para resolver el problema se puede limitar la velocidad al impuesto por el límite de velocidad de las articulaciones.

3) Puntos alcanzables sólo con diferentes orientaciones.

Considérese que se especifica un punto alcanzable en el interior del espacio de trabajo al cual sólo puede accederse con determinadas orientaciones. Si se especifica una orientación distinta, el planificador podría detectar el problema evitando que el controlador intente ejecutarlo [11].

Métodos en el espacio cartesiano.

Un método clásico de planificación de trayectorias que permite obtener los caminos más cortos libres de obstáculos es el de los grafos de vértices o líneas de visibilidad. Suponiendo que se trabaja en 2D, que el objeto móvil es un punto y que los obstáculos son convexos, se elige el camino más corto, formado por segmentos que conectan los vértices de los objetos. Otro método muy utilizado es el de búsqueda heurística en el espacio cartesiano. Se trata de establecer un camino que optimice una función de evaluación que suponiendo que al objeto móvil en un estado x (posición y orientación) viene dada por $f(x) = g(x) + h(x)$ siendo [11]:

x el estado (posición y orientación del robot)

f la estimación del camino desde el estado inicial hasta el estado final

g el camino del estado inicial a x

h el camino de x al estado final

El método se define inicialmente para un punto o una forma arbitraria móvil que mantiene en su movimiento una orientación fija. El espacio se considera dividido en rejillas. Comenzando en la posición inicial se van explorando todos los vecinos n-conectados, típicamente 8 o 16 en 2D, definiéndose una estructura de datos tipo árbol en cuyos nodos se evalúa la función. Se sitúan todos los puntos

conectados en una lista abierta de acuerdo con el valor de la función. La operación se repite para otro punto conectado que presente un valor mínimo de f y así hasta que se alcance al punto final mediante una serie de puntos conectados. Cuando se exploran todos los caminos se elige el de mínima distancia [11].

2.8 Programación de robots

Como se sabe la traducción de un lenguaje de programación de computadoras puede realizarse mediante compiladores o interpretes. Muchos de los lenguajes de programación son interpretados, con lo cual se facilita la depuración interactiva. De hecho, para conseguir las propiedades de edición y puesta a punto es necesario utilizar traducción interpretada. Sin embargo esto reduce las prestaciones durante la ejecución. Por ello, a veces se emplea un intérprete para escritura y puesta a punto y posteriormente un compilador para traducir el código ya verificado [3].

En la programación textual se define la tarea mediante lenguajes que formalmente recuerdan a los lenguajes de programación de computadora [4].

La definición de la tarea conduce a un texto de un programa en el que existen órdenes especiales para especificar los movimientos del robot, acceder a la información de los sensores y actuar sobre los objetos del entorno [11].

La mayor parte de los lenguajes de programación de robots se definieron en los años setenta y ochenta y su sintaxis es similar a lenguajes tales como el BASIC o el Pascal [4].

Las características y capacidades de los lenguajes textuales de programación de los robots se pueden listar del siguiente modo [4]:

1) Control de movimiento
2) Capacidades de sensor avanzadas. Tratan con más que señales binarias simples (todo o nada) y controlan los dispositivos por medio de los datos del sensor.
3) Inteligencia limitada. Utiliza la información recibida sobre el entorno de trabajo para modificar el comportamiento del sistema de una manera programada.
4) Comunicaciones y procesamiento de datos. Suelen tener medios para interactuar con computadoras y bases de datos con el propósito de generar informes y controlar las actividades de la célula de trabajo.

En los lenguajes de programación de robots existen sentencias para especificar el movimiento de un robot indicando la duración del movimiento o la velocidad. Así podría emplearse una sentencia tal como [4]:

Mover BRAZO a C con duración 3 segundos

En caso de no especificarse una duración, el movimiento se realiza con una velocidad previamente establecida en el sistema (velocidad por defecto).

2.9 Conclusiones

En este capítulo se abordaron aspectos de modelado cinemático, de estrategias de control y de modelos para planificar trayectorias en robots manipuladores. Esta teoría, corresponde a los aspectos básicos, para el diseño de brazos manipuladores, teniendo este marco teórico en mente se puede decir que para el diseño de un manipulador, los tres primeros eslabones son los críticos para la dinámica y el control. Ya que la herramienta o efector final tiene pequeñas dimensiones y masa, comparada con los eslabones y por lo tanto no influye tanto como estás en la dinámica y por ende en el control.

En lo referente al control, en la mayoría de los manipuladores industriales se utiliza un control PID debido a que la acción integradora reduce el error estacionario sin afectar la constante de proporcionalidad obtenida de la fuente de potencia y el motor, y con la acción derivativa se logra un tiempo de asentamiento aceptable.

En cuanto a la planificación de trayectorias, nos hace ver que un método geométrico es práctico a la hora de planificar el camino, teniendo en mente el criterio de obtener la trayectoria mínima entre dos puntos.

Por otra parte nos hace reflexionar acerca de cómo los robots nos pueden liberar de trabajos peligrosos, duros y repetitivos, para ocuparnos en áreas de mayor nivel intelectual como lo seria la investigación o bien en tareas tales como control de calidad o en el mantenimiento y programación de los robots.

El marco teórico expuesto en este capítulo nos sirve para familiarizarnos con los robots industriales, así como también nos sirve de referencia para saber que es lo que se necesita para el diseño de robots, en dado caso que nos inclinemos a realizar estudios de postgrado en esta área.

Capítulo 3

CONTROLADOR LÓGICO PROGRAMABLE

3.1 Introducción

La mayor parte de los procesos de fabricación requieren la operación de un sistema, que necesita una secuencia de operaciones. La secuenciación del proceso se puede realizar manualmente o empleando algún tipo de controlador.

Por los requerimientos de la industria y ante los nuevos avances en la electrónica y con el fin de reducir los costos de los sistemas basados en relés, aparece el PLC, que es de lo que trata este capítulo.

3.2 Reseña de los PLCs

El origen del PLC se remonta a 1968, año en que la división Hydramatic de la General Motors Corp. de los Estados Unidos especificó los criterios básicos para su diseño [16].

La intención de la empresa al encargar el desarrollo de este aparato fue la de dar una solución a una gran cantidad de problemas que tenían las automatizaciones basadas en componentes electromecánicos o electrónicos (relés, temporizadores, contadores, etc.), o en circuitos electrónicos específicamente diseñados para una tarea [16].

Algunos de estos problemas eran:

• El alto costo asociado a los circuitos de relés por su baja confiabilidad, dificultoso mantenimiento, paradas imprevistas, etc.

• La falta de flexibilidad de circuitos electromecánicos o electrónicos diseñados para una aplicación especial. No se podía, o era muy difícil, utilizar una misma máquina para varios modelos de piezas. Un nuevo diseño significaba adaptar todo el circuito (es decir, docenas de componentes) o reemplazarlo por uno nuevo [16].

Las especificaciones mínimas que debía cumplir el primer PLC según la solicitud de General Motors fueron [16]:

- Programable: El aparato debía adaptarse fácilmente a una gran variedad de aplicaciones.
- Sencillo: Tanto la programación como el mantenimiento y la instalación debían estar a cargo de técnicos o ingenieros de planta sin un entrenamiento específico.
- Lógico: Como solo se pretendía reemplazar tableros electromecánicos o electrónica dedicada sencilla, se requería un aparato que efectuara un control lógico binario (ON/OFF).
- Reutilizable: Si bien no era en ese momento la condición de mayor peso, se pretendía que un mismo aparato pudiera reutilizarse [16].

General Motors encargó la tarea de desarrollo a este equipo a una consultora llamada Bedford Associates. La implementación de este desarrollo fue liderada por Richard Morley, quien construyó en 1969 el primer controlador lógico programable: el Modicon 084. Con una capacidad de 255 entradas/salidas, y una memoria de 4KB, el Modicon 084 tenía un peso de 46 Kg [16].

Años más tarde, la firma Allen Bradley iniciaba la producción de sus propios equipos, dándole el nombre de PLC (programmable logic controller). Este nombre se universalizaría como denominación de los controladores lógicos programables.

Durante los primeros años de vida se efectuaron infinidad de modificaciones en el desarrollo original, algunas orientadas a corregir problemas, y otras a fin de apuntar a un mercado industrial más amplio [16].

Los primeros PLC se programaban en un lenguaje de tipo secuencial muy primitivo. Hacia 1972 aparecieron equipos que utilizaban un lenguaje de lógica de escalera (RLL, relay ladder logic), semejante a los esquemas funcionales eléctricos utilizados en los circuitos lógicos de relés. Este lenguaje tuvo gran aceptación entre los técnicos de ingeniería y mantenimiento [16].

Con el paso del tiempo (1970-1974), los equipos comenzaron a ofrecer cada vez más ventajas, como la de almacenar programas en cintas magnéticas (para facilitar la reprogramación), las posibilidades de conexión a campo se hicieron más variadas ya que las señales podían ser de distintos tipos y niveles, algunos equipos eran conectables a computadoras y otros periféricos, etc [16].

Al igual que en otros ámbitos industriales, un hecho clave que aceleró el desarrollo de los PLCs fue la invención del microprocesador, y su aplicación en el diseño de los nuevos PLCs. Los equipos pudieron manejar mayor cantidad de datos y realizar mayor cantidad de operaciones, incluso aritmética con números enteros [16].

Los avances entre 1975 y 1979 continuaron incesantemente. El desarrollo de memorias con cada vez mayor capacidad de almacenamiento en menor espacio hizo que los PLCs disminuyeran en tamaño y aumentaran en potencia. Así fue posible el desarrollo de programas mucho más grandes y complejos [16].

A partir de los comienzos de la década del 80’s, los PLC’s ofrecieron una nueva ventaja: las entradas/salidas remotas. El hecho de poder colocar elementos de forma y emisión de señales de campo en forma distribuida y a veces a grandes

distancias del controlador significa un ahorro muy grande de dinero en instalación y cableado y por consiguiente una gran sencillez de mantenimiento [16].

3.3 Estructura de un PLC:

Para poder interpretar la estructura de un PLC utilizaremos un sencillo diagrama en bloques. En la figura 3.3 se muestran las tres partes fundamentales: la CPU, las entradas y las salidas [16].

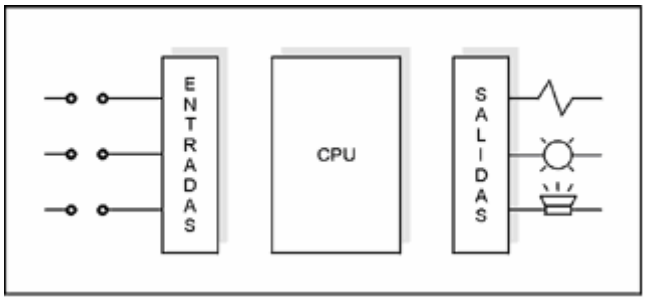


Figura 3.3 Diagrama a bloques de los elementos básicos de un PLC [16].

La CPU es el cerebro del PLC, responsable de la ejecución del programa desarrollado por el usuario. Estrictamente, la CPU está formada por uno o varios procesadores; en la práctica, puede abarcar también a la memoria, puertos de comunicaciones, circuitos de diagnóstico, fuentes de alimentación, etc [16].

Las entradas (interfases o adaptadores de entrada) se encargan de adaptar señales provenientes del campo a niveles que la CPU pueda interpretar como información. En efecto, las señales de campo pueden implicar niveles y tipos de señal eléctrica diferentes a los que maneja la CPU. En forma similar, las salidas (interfases o adaptadores de salida) comandan dispositivos de campo en función de la información enviada por la CPU [16].

La CPU se comunica con las interfases de entrada por medio de un bus paralelo. De esta forma se cuenta con un bus de datos y un bus de direcciones.

Adicionalmente, un bus de alimentación provee alimentación eléctrica a las interfases de entrada [16].
A las entradas se conectan sensores, que pueden ser:

• Pulsadores
• Llaves
• Termostatos
• Presostatos
• Límites de carrera
• Sensores de proximidad
• Otros elementos que generan señales binarias (ON-OFF)

Las salidas comandan distintos equipos, por ejemplo:
• Lámparas
• Sirenas y bocinas
• Contactores de mando de motores
• Válvulas solenoide
• Otros elementos comandados por señales binarias

Cuando un sensor conectado a una entrada se cierra, permite que aparezca entre los bornes de esa entrada una tensión, por ejemplo: 24 Vcc. Esta tensión es adaptada por la interfaz de entrada al nivel y tipo de tensión que la CPU puede leer a través del bus de datos. Cuando la CPU lee este nivel de tensión, recibe la información de que dicha entrada está en el estado activo, o sea en el estado lógico 1 [16].

Cada entrada es reconocida por la CPU mediante una identificación. Si la entrada activada se denomina X1, podemos decir que X1 está en estado lógico 1 (X1=1). Cuando el sensor conectado al borne de entrada se abra, X1 estará en estado 0 (X1=0) [16].

En forma similar, cuando la CPU desea que una salida se active (pase a estado lógico 1), modifica los niveles de tensión en el bus de datos. La tarjeta de salida, que está conectada al bus de datos, cierra entonces el circuito de conexión, energizando el dispositivo de campo. Cada salida está identificada, por ejemplo una salida podría denominarse Y2. Podemos decir entonces que salida Y2 está energizada (Y2=1) o desenergizada (Y2=0) [16].

La identificación que la CPU utiliza para cada punto de entrada/salida en la memoria se conoce como direccionamiento (o addressing) de la entrada/salida. Un programa muy sencillo podría ser: "Cuando X1=1, se debe hacer que Y2=1". Este podría ser el caso en que se enciende una lámpara al presionar un pulsador. El pulsador deberá estar conectado a la entrada X1, y la lámpara a la salida Y2. Al presionar el pulsador, la CPU leerá en la interfaz de entrada que X1=1. Resolverá el programa, y pondrá un 1 en Y2. Como consecuencia, la salida cerrará el circuito de conexión y encenderá la lámpara [16].

La convención por la cual un "1" indica la presencia de señal, mientras que un "0" indica su ausencia; se denomina lógica positiva. En forma inversa, la lógica negativa utiliza un "0" para indicar la presencia de señal, y un "1" para indicar su ausencia [16].

Las interfases de entrada/salida presentadas pueden tomar solo uno de los dos estados: "1" ó "0".

Otras interfases pueden tener como entrada o salida a variables analógicas, las que se caracterizan por tomar valores intermedios en forma continua entre dos límites. Un ejemplo de variable analógica puede ser la presión de un reactor, que varía en forma continua entre 0 y 10 kg/cm² (g). Dado que la naturaleza de una señal de presión no es eléctrica, se requiere un transmisor de presión. Este

convierte la presión medida en una señal eléctrica, que puede ser de 4 a 20 mA, 0 a 10 Vcc, etc [16].

La interfaz de entrada analógica convierte una señal analógica eléctrica en un número binario, cuya cantidad de dígitos depende de la resolución de la interfaz de entrada/salida (por ejemplo, un rango de 00000000 a 11111111, con un resolución de 8 bits). Es evidente que la interfaz maneja en realidad valores digitales, pese a lo cual se las denomina interfases de entradas/salidas analógicas [16].

El conjunto de entradas y salidas se denomina a veces "estructura de entradas/salidas", o también "periferia de entradas/salidas", aunque es más frecuente que se lo abrevie como E/S, o I/O por su siglas en inglés (input/output). Cada entrada o salida se denomina canal o punto de E/S.

3.4 Principio de funcionamiento de un PLC

CPU (Unidad Central de Proceso)

La CPU está compuesta por dos partes fundamentales: el procesador y la memoria. Puede contener también otros elementos, como puertos de comunicación, o incluso la fuente de alimentación [2].

Procesador

El procesador tiene como tarea principal ejecutar el programa de aplicación escrito por el usuario. También cumple con otras tareas importantes, como son las de administrar las tareas de comunicación y ejecutar programas de autodiagnóstico [16].

Los PLCs más sencillos poseen un solo procesador, pero en la medida que su capacidad de control aumenta pueden tener varios procesadores dedicados a tareas específicas como resolución de lazos, comunicaciones, diagnóstico, etc.

Para poder gobernar todo el sistema, el procesador necesita de un programa escrito por el fabricante (el mismo contiene el conjunto de instrucciones utilizado para ejecutar el programa de aplicación, una rutina de autodiagnóstico y el sistema básico de interacción con los periféricos: tarjetas de E/S, puertos de comunicaciones, etc.). A este programa se lo denomina programa ejecutivo o sistema operativo. El sistema operativo no es accesible al usuario y se encuentra almacenado en la memoria no volátil que forma parte de la CPU. Las tareas asignadas al procesador son ejecutadas por éste en forma secuencial incesantemente mientras el equipo está conectado a la alimentación. Esta secuencia se denomina barrido o scan [16].

Una secuencia típica de barrido o scan consistiría en (Figura 3.4):

• Consultar el estado de las entradas y almacenar estos valores en la memoria.
• Resolver el programa de aplicación.
• Atender las comunicaciones con módulos inteligentes.
• Atender las comunicaciones de los puertos de la CPU.
• Ejecutar un autodiagnóstico.
• Actualizar las salidas a partir de los resultados almacenados en la memoria.
• Volver a empezar el ciclo.

El tiempo que necesita el procesador para llevar al cabo es ciclo se denomina tiempo de barrido o de scan time. Los fabricantes en general informan un tiempo promedio necesario para ejecutar un programa de aplicación que contiene 1K (1024) instrucciones de lógica booleana. Los PLCs más rápidos

tienen un tiempo de barrido de menos de medio milisegundo para 1K de instrucciones [16].

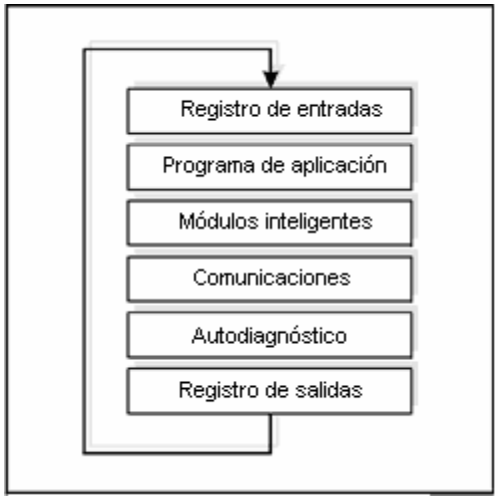


Figura 3.4 Diagrama de flujo del barrido de programa o scan del PLC [16].

Sin embargo, la comparación de distintos PLCs en función del tiempo de barrido indicado en la especificación puede ser engañosa. Esto se debe a que no está normalizado el tipo de instrucciones a utilizar en este ensayo, y a que los distintos PLCs pueden tener distinta rapidez en la resolución de una determinada instrucción. Algunos PLCs que parecen muy rápidos deben de serlo cuando ejecutan operaciones aritméticas o instrucciones complejas.

Por otra parte, los PLCs de mayor capacidad suelen tener tiempos de barrido más rápidos por Kb de instrucciones, pero son también los que deben ejecutar programas de mayor volumen. La determinación exacta del tiempo real de barrido de un programa de aplicación requiere el cálculo del tiempo que le insume al procesador la ejecución de cada una de las instrucciones utilizadas, así como el tiempo consumido por las demás funciones que ejecuta la CPU [16].

Con frecuencia se asocia un corto tiempo de barrido a una rápida ejecución de una aplicación. Esto es incorrecto, ya que otros factores suman su influencia.

Consideremos los pasos necesarios para encender una lámpara desde un pulsador [16]:

• Debe cerrarse el contacto del pulsador.

• La tarjeta de entrada debe leer que el contacto está cerrado (tiempo de respuesta de la entrada).

• La CPU debe leer la tarjeta de entrada, resolver el programa de aplicación y escribir el resultado en la tarjeta de salida. El tiempo total para estas tareas es el tiempo de barrido.

• La tarjeta de salida debe cerrar el circuito de conexión (tiempo de respuesta de la salida).

• La lámpara debe encenderse.

El tiempo total que todas estas tareas insumen se denomina tiempo total de respuesta, o throughput. La figura 3.4.1 muestra la distribución del tiempo total de respuesta para una aplicación de pequeña magnitud, con E/S discretas en corriente alterna [16].

Se puede apreciar que una mejora en el tiempo de barrido (aún cuando ésta sea importante) no mejorará demasiado la respuesta total del sistema. Concentrarse en el tiempo insumido por los dispositivos de campo puede ser, a veces, más importante [16].

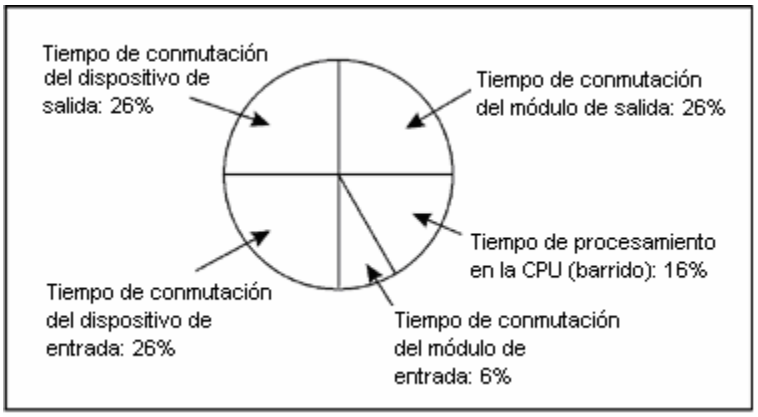


Figura 3.4.1 Componentes del tiempo de respuesta para una aplicación de pequeña magnitud [16]

3.5 Lenguajes de programación de un PLC

Para la programación de PLC existen diferentes formas de programar, que las podemos reducir en lo siguiente [18]:

Lenguajes Básicos:

- Diagramas escalera o ladder
- Funciones lógicas o mnemónico

Lenguajes de alto nivel:

- Bloques funcionales Grafcet
- Sentencias en idioma inglés

El lenguaje básico esta limitado a un conjunto de instrucciones que realizarán funciones elementales de control como [18]:

- Reemplazar a relés
- Temporización

- Conteo

Los lenguajes de alto nivel, tienen instrucciones más poderosas que van más allá del ON-OFF, realizando operaciones como:

- Control análogo
- Manejo de datos

Estas distintas representaciones se conocen como lenguajes de programación de un PLC. El lenguaje utilizado para la programación de un PLC depende de su diseño, por lo que varía en los distintos equipos. Algunos equipos permiten programar indistintamente en uno u otro lenguaje, siendo posible la traducción de un programa de aplicación de un PLC a otro [18].

Para poder implementar una aplicación en un PLC se debe conocer el lenguaje de programación que éste utiliza. También se debe estudiar la forma en que se denominan las variables en la memoria del mismo. Por ejemplo, para denominar las E/S, algunas marcas mencionan un número que indica la posición del módulo en el chasis o base de montaje, seguido de otro número que indica él número de borne del campo correspondiente. Otros identifican con una X las entradas y con una Y las salidas, seguidas de un número que identifica la dirección de memoria que contiene a su estado. En los equipos pequeños sólo se utilizan números [18].

3.6 Clasificación de PLC´s industriales

Si deseamos establecer una clasificación de PLCs, podemos considerar distintos aspectos [16]:

• Por su construcción:

Integral

Modular

• Por su capacidad:

Nivel 1: Control de variables discretas y pocas analógicas, operaciones aritméticas y capacidad de comunicación elementales.

Nivel 2: Control de variables discretas y analógicas. Matemáticas de punto flotante. E/S inteligentes. Conexión en red. Gran capacidad de manejo de datos analógicos y discretos.

• Por cantidad de E/S:

microPLC (hasta 64 E/S)
PLC pequeño (65 a 255 E/S)
PLC mediano (256 a 1023 E/S)
PLC grande (más de 1024 E/S)

Clasificación por construcción:

La clasificación por construcción distingue a los PLCs que integran todas sus partes (E/S, CPU, fuentes, puertos de comunicaciones, etc.) en una misma caja o gabinete, de los que están formados por módulos [16].

Llamaremos PLC integral a aquel que integre todas sus partes en una misma caja o gabinete (Figura 3.6). Se suele utilizar también la denominación de compacto, pero la aparición de PLCs modulares de pequeño tamaño hace que éste resulte inadecuado.

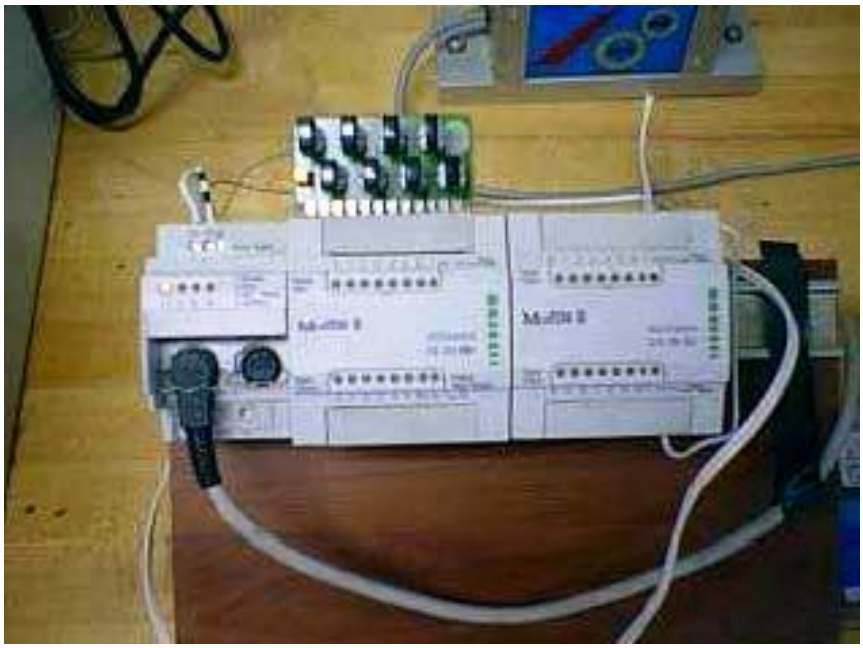


Figura 3.6 PLC integral TP-02 [15]

El PLC integral suele tener muy pocas E/S, clasificándose en general como micro PLC. Tiene como ventajas un bajo costo y un pequeño tamaño. Una desventaja es la imposibilidad de expandir un equipo en forma gradual. En general se parte de un equipo básico que puede ampliarse mediante el agregado de unas pocas unidades de expansión con cantidad y tipo de E/S. Otra desventaja es la escasa variedad disponible de tipos de E/S, ya que, al estar éstas integradas en un gabinete, es imposible cubrir una amplia gama de opciones [15].

Un PLC modular, como su nombre lo indica, está formado por módulos. El equipo se arma sobre un bastidor o base de montaje (también llamada chasis o rack) sobre el cual se instalan la CPU, los módulos de entrada, los módulos de salida y otros periféricos (figura 3.6.1) [16].

El chasis contiene en su parte posterior los buses de datos, direcciones y alimentación del PLC, con conectores apropiados a los que se conectan los distintos módulos [16].

Figura 3.6.1 PLC modular T2small [15]

Por la forma que tienen estos módulos, es usual que se los denomine tarjeta. Es así que es muy frecuente encontrar la frase tarjeta de entrada/salida en referencia a los módulos de entrada/salida (Figura 3.6.2) [16].

Figura 3.6.2 Módulos E/S (fuente: Siemens)

La principal ventaja de un PLC modular frente a uno integral es evidente, el usuario puede componer su equipo con la cantidad y tipo de entradas y salidas que necesite, y luego puede ampliarlo agregando los módulos necesarios [16].

La desventaja, en equipos pequeños, es su mayor costo. En general, este mayor costo tiene dos razones; mayor cantidad y costo de los componentes utilizados en la fabricación y ensamblado del equipo (conectores, chasis, plaquetas, etc.), y la mayor capacidad que suele tener un PLC modular. Esta mayor capacidad se evidencia en un lenguaje de programación más potente y con instrucciones para aplicaciones más complejas, con mayor capacidad de comunicaciones, etc [16].

Clasificación por capacidad:

La clasificación por capacidad distingue dos niveles, en función de la complejidad de las instrucciones que el PLC puede manejar. El nivel 1 identifica a un PLC con construcciones sencillas y no muy potentes, mientras que el nivel 2 identifica a los PLCs con funciones de mayor complejidad [16].

Algunas de las instrucciones que podemos encontrar en un PLC de nivel 2, y que en general no estarán en un PLC de nivel 1 son: raíz cuadrada, logaritmo, antilogaritmo, aritmética de doble precisión y de punto flotante, funciones trigonométricas, diferenciación e integración lazos PID, etc [16].

Clasificación por cantidad de E/S

La clasificación por cantidad de E/S es arbitraria. A pesar de ello, este parámetro es el indicador que habitualmente define un PLC. Los fabricantes ofrecen características tales como la capacidad de memoria, operaciones

aritméticas, etc., en directa relación a la cantidad de entradas y salidas que el controlador puede manejar. Así, por ejemplo, suele haber una directa relación entre la clasificación de PLCs como integrales, y los clasificados como microPLC por la cantidad de E/S. Más aún, este PLC clasificado como integral por su construcción y como microPLC por su cantidad de E/S, probablemente deba ser clasificado como de nivel 1 en cuanto a su capacidad [16].

3.7 Aplicaciones

Originalmente, los PLCs fueron esencialmente orientados a la industria manufacturera o de procesos discontinuos, como la fabricación de automóviles, industria del plástico, etc. Luego encontraron aplicación en la industria de procesos continuos, como equipo auxiliar que reemplazó a los paneles de relés, en el enclavamiento de bombas, válvulas on/off, control de seguridad de calderas, etc [16].

Al ser complementados con manejo de señales analógicas y control regulatorio en el mismo equipo, los PLCs ampliaron su campo de acción en la industria de procesos continuos, en estos casos desplazaron a equipos más costosos como los sistemas formados por instrumentos neumáticos o electrónicos analógicos y paneles de relés. Esto resultó particularmente cierto a partir de la combinación de PLCs, computadoras y software para control de procesos, combinación que tuvo su origen a comienzos de la década de los 80’s [16].

Los PLC´s están actualmente involucrados en cualquier proceso automatizado, las aplicaciones de estos pueden ser desde la operación de una sola máquina, hasta controlar un grupo de máquinas como lo muestran las fotos 3.7 y 3.7.1 (cortesía de Nissan Time de México S.A de C.V.).

Figura 3.7 Máquina de inyección y robot controlados por PLC.

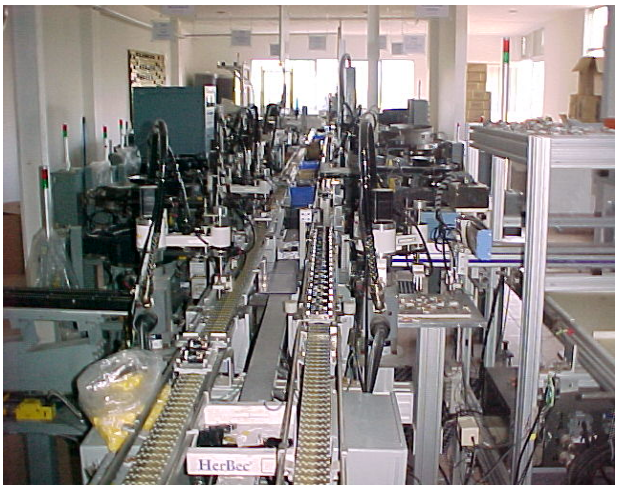


Figura 3.7.1 Centro de ensamble automatizado

3.8 Conclusiones.

En este capítulo se presento de manera general aspectos de los controladores lógicos programables, tales como su principio de funcionamiento y modo de programación, esto con el fin de tener un marco teórico de referencia, para cuando tratemos aspectos relacionados acerca del controlador lógico programable que se va a programar para manipular el robot industrial.

El controlador lógico programable surge ante la necesidad de reducir los costos de los sistemas que eran controlados por relés.

La estructura de un controlador lógico programable es muy similar a la de una computadora personal solo que los controladores lógicos programables se pueden conectar directamente a los sensores y actuadores, funcionan en ambientes industriales (donde las condiciones de temperatura y vibraciones son mayores a las que opera un computador personal) Su función es la de tratar las señales procedentes de los sensores, para emitir señales de control hacia los actuadores.

Capítulo 4

EL ROBOT Y CONTROLADOR YAMAHA

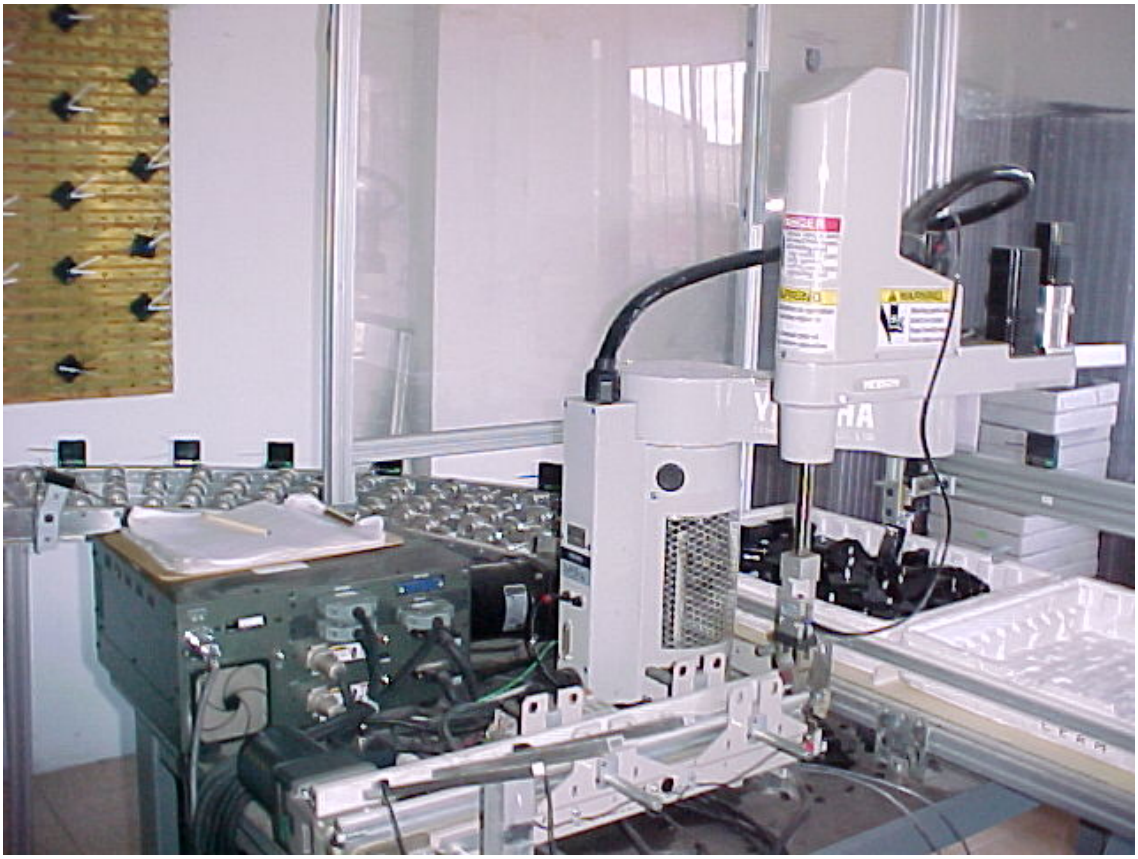


Figura 4 Controlador y robot manipulador Yamaha (fuente: Nissan Time)

4.1 Introducción.

El presente capítulo se refiere, a la operación, al principio de funcionamiento y a los aspectos técnicos tanto del robot como del controlador que se va a utilizar para realizar la tarea planteada en el primer capítulo. También, nos presenta el lenguaje de programación del controlador, para programar la rutina.

4.2 El controlador del robot YAMAHA

El controlador Yamaha de la serie QRCH esta diseñado para utilizarse con el robot Yamaha tipo SCARA o robot cartesiano XY, que su principal función es la de ensamblar, cargar y descargar piezas.

El robot Yamaha YK550H es un robot SCARA (Selective Compliance Assambly Robot Arm) cuyas articulaciones de hombro y codo giran alrededor de ejes verticales. Esta configuración proporciona una importante rigidez para el robot en la dirección vertical, pero una elasticidad en el plano horizontal, esto le hace ideal para tareas de montaje.

El QRCH es un controlador multi-ejes, es decir, puede controlar los movimientos en los tres ejes (XYZ) si lo vemos como un plano cartesiano. El controlador cuenta con las siguientes funciones:

1.- Multitareas. Esto quiere decir, que puede ejecutar hasta 8 tareas al mismo tiempo. Las tareas se ejecutan de acuerdo a su prioridad, este punto va a quedar más claro en la sección dedicada a la programación.
2.- Lenguaje de programación. Los comandos son muy similares al BASIC basado en JIRA, este hace que tanto los movimientos complejos como las multitareas sean fácilmente programados. Utiliza el método compilador para una rápida ejecución.

4.2.1 Partes constitutivas

El controlador esta constituido por una unidad central de proceso, compuesta de un procesador i386 EX a 25 MHz (encerrado en circulo en la figura 4.1) y un co-procesador matemático i387SX (encerrado en el cuadrado en la figura 4.1), con una capacidad de memoria de 100 KB.

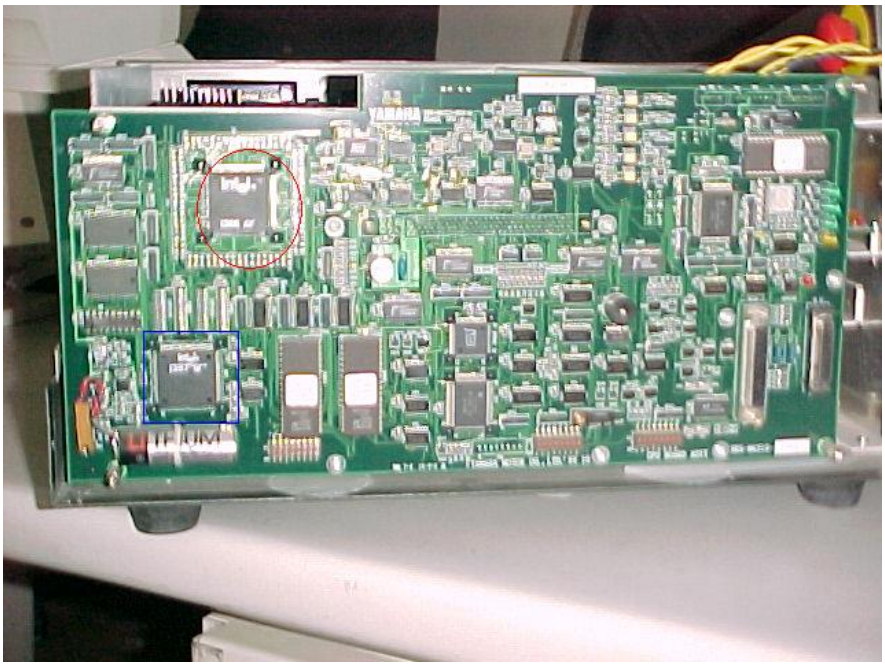


Figura 4.1 Tarjeta madre del controlador (fuente: Nissan Time)

Las tarjetas de control (driver unit), se encargan del control de los servomotores, cada conector de los servomotores acepta las señales para controlar el movimiento en dos ejes ya sea en X-Y o bien Z-R. También en estas tarjetas se encuentran los conectores utilizados para la retroalimentación entre el servomotor y las señales de los sensores.

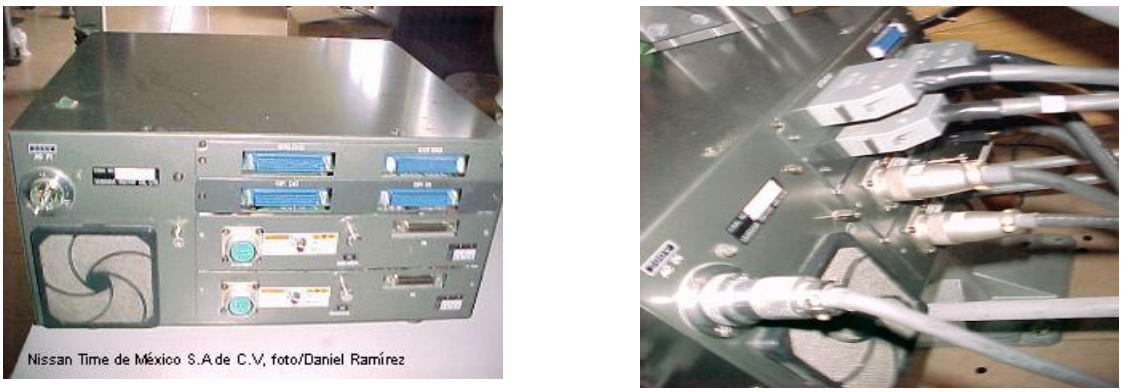


Figura 4.2 Conectores externos de las tarjetas de control

Partes del panel frontal

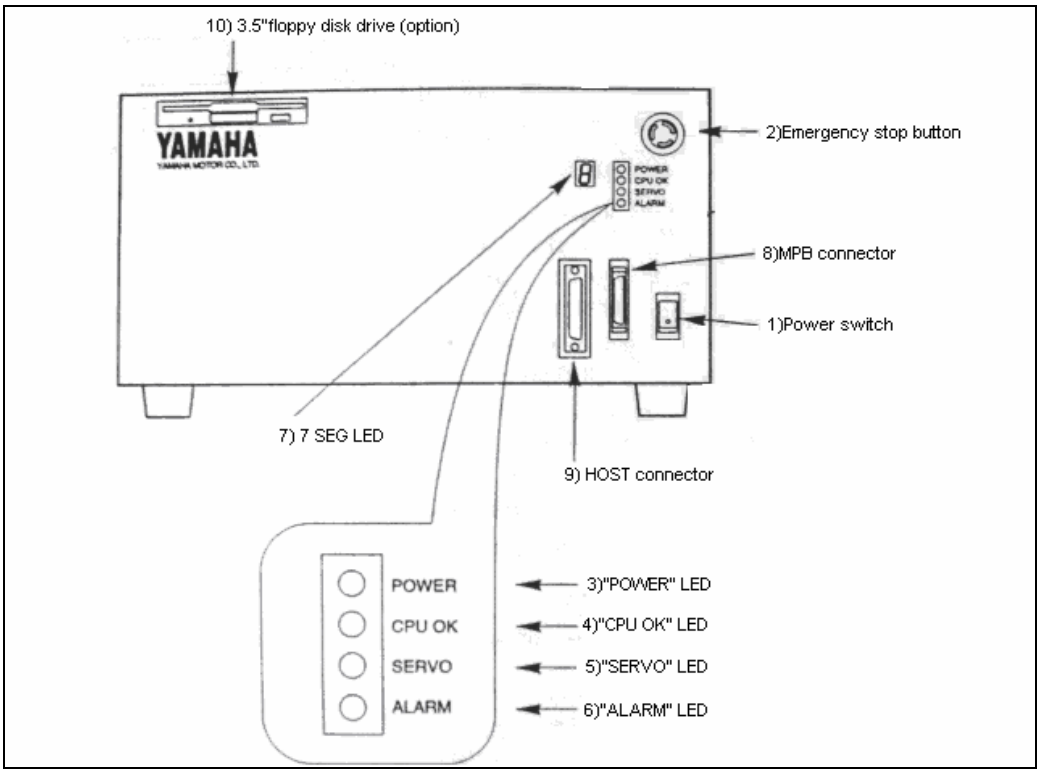


Figura 4.2.1.1 Panel frontal del controlador [14].

1 Interruptor de encendido/apagado

2 Botón de paro de emergencia

3 “Power” LED: ilumina cuando se enciende el controlador

4 “CPU OK” LED: Enciende cuando el controlador funciona normal y se apaga cuando ocurre un error critico

5 “SERVO” LED: Ilumina cuando el servo del robot se enciende.

6 “ALARM” LED: Enciende cuando ocurre algún error.

7 “7SEG” LED: Este detalla el error de la alarma.

8 “TEACH PENDANT connector: En este puerto se conecta la botonera de programación.

9 HOST connector: Puerto para conectar la Computadora.

Panel posterior

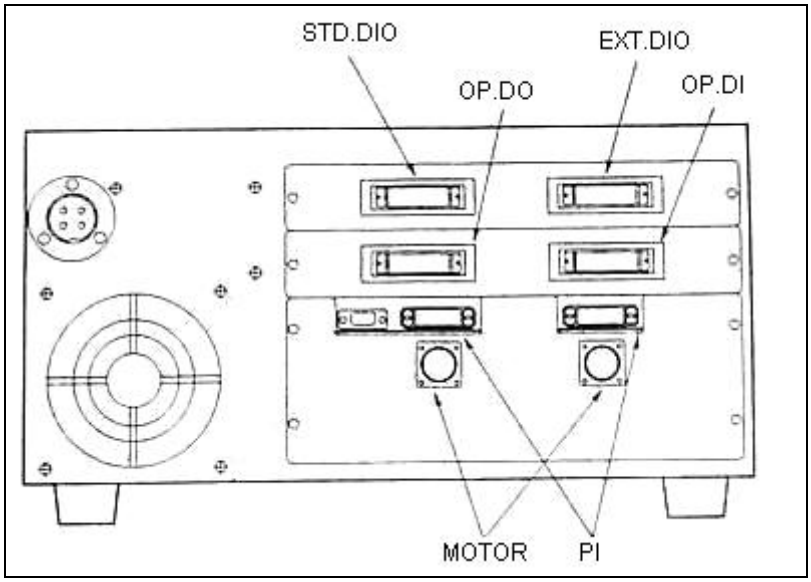


Figura 4.2.1.2 Panel posterior del controlador [14].

En esta parte se encuentran las conexiones para los servos del robot, así como los sensores y actuadores que se van a controlar durante el proceso, que en la tabla siguiente se especifican las funciones de los conectores.

Nombre del conector	Función
AC IN	Este se conecta a la fuente de AC
STD.DIO	Este conector es usado para las entradas y salidas tanto de propósitos generales como para las I/O establecidas.
OP.DI	Este conector es usado como extensión de las entradas de propósito general.
OP.DO	Este conector es usado para extender las salidas de propósitos generales
MOTOR	Estos conectores son utilizados para controlar los servo motores. Cada conector acepta las señales de 2 ejes.
PI	Este conector es usado para la retroalimentación del servo con las señales de los sensores. Cada conector acepta las señales para 2 ejes.
EXT.DIO	Este es utilizado para extender las I/O de propósito general

Tabla 1. Función de los conectores de las tarjetas de control

Botonera de programación (teach pendant)

Esta se conecta al controlador como lo muestra la figura 4.2.1.3, la botonera permite editar y ejecutar programas.

Su pantalla esta compuesta por 4 áreas que se describen a continuación:

1) La primer línea indica en el modo en que se esta operando el controlador.
2) La segunda línea, es la línea de mensajes.
3) De la tercera línea a la séptima es el área de los datos es decir, aquí se edita el programa.
4) En la octava línea se muestra el menú de las funciones activas.

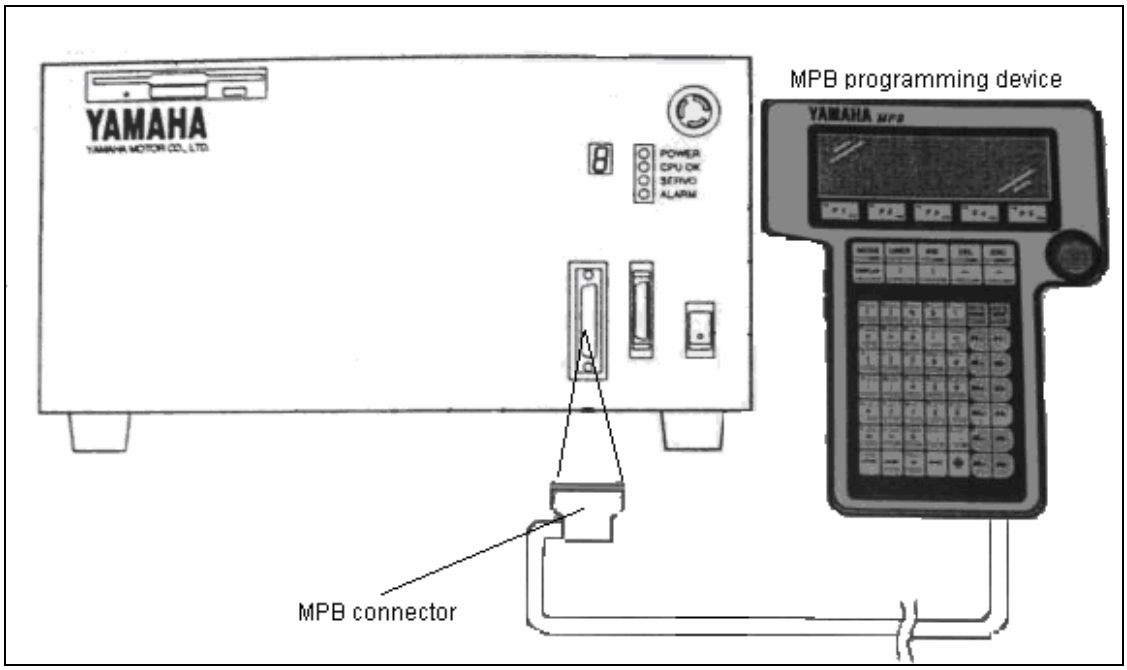


Figura 4.2.1.3 Conexión del teach pendant [14].

Cuenta con tres tipos de teclas estas son:

1) Teclas de funciones.
2) Teclas de control.
3) Teclas de datos.

Las teclas de funciones sirven para seleccionar del menú de la octava línea de la pantalla la función a ejecutar.

Las teclas de control, son las que controlan los movimientos del cursor, las que controlan el cambio de pantallas y las de inserción y borrado de caracteres y líneas.

Las teclas de datos son aquellas que se utilizan para editar el programa. En la siguiente imagen se detalla mejor lo comentado.

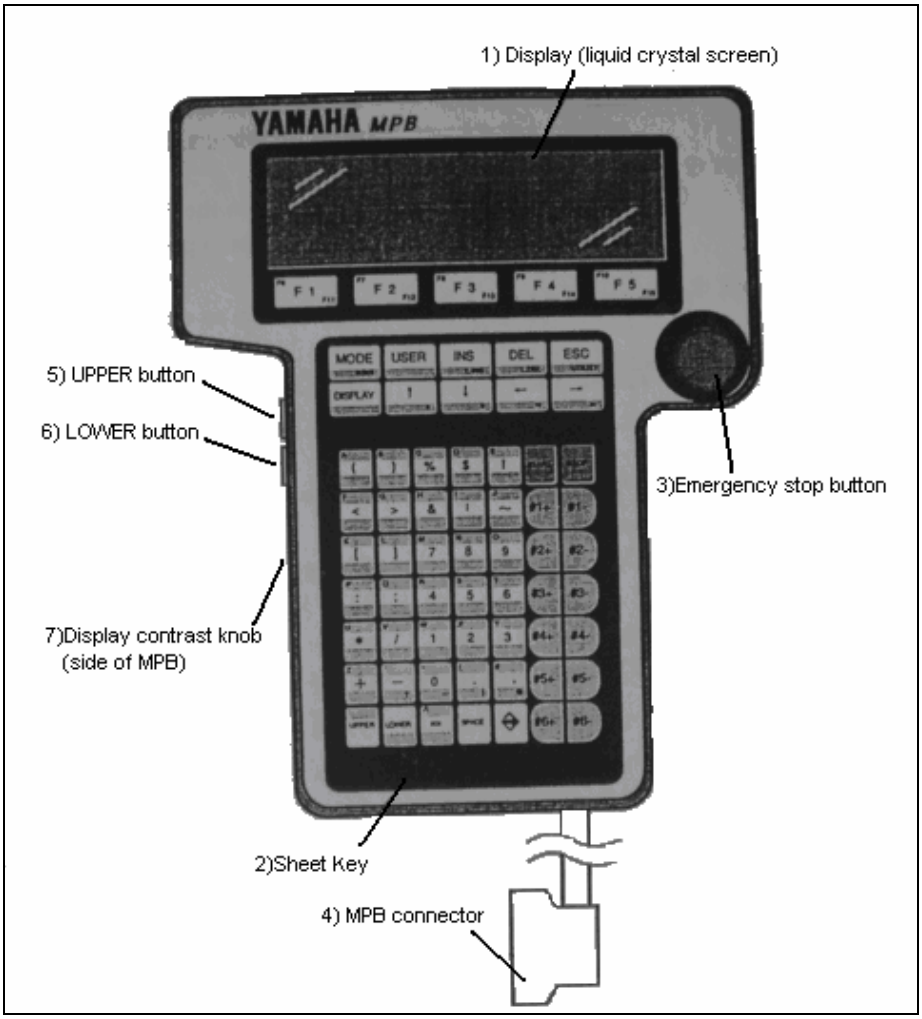


Figura 4.2.1.4 Partes del teach pendant [14].

1) Esta pantalla de LCD tiene 40 caracteres x 8 líneas, muestra la información.
2) Son las teclas encargadas de introducir el programa.
3) Cuando este botón se presiona mientras esta operando el robot, este para inmediatamente.
4) Es el encargado de conectar el TEACH PENDANT al controlador.
5) Activa la función superior de la tecla.
6) Activa la operación que aparece en la parte inferior de la tecla.
7) Este botón sirve para ajustar el contraste de la pantalla.

Interfaz de entrada/salida

El puerto de entrada/salida tiene nueve entradas asignadas y dieciséis entradas de propósito general, mientras que en la salida tiene diez asignadas y ocho de propósito general, esto en el conector estándar de I/O (STD.DIO). En conjunto con los conectores externo y opcional hacen un total de ochenta entradas de propósito general y sesenta y cuatro salidas de propósitos generales.

El arreglo de pines en el conector estándar de I/O se muestra en la siguiente tabla.

PIN	I/O No.	Nombre
1	DI 00	Paro de emergencia
2	DI 01	Encendido del servo
3	DI 10	Secuencia de control
4	DI 11	Interlock
5	DI 12	Arranque
6	DI 13	Automático
7	DI 14	Regreso a origen
8	DI 15	Reiniciar programa
9	DI 16	Manual
10	DI 17	Reinicio total
11	DI 20	Entrada de propósito general 20

12	DI 21	Entrada de propósito general 21
13	DI 22	Entrada de propósito general 22
14	DI 23	Entrada de propósito general 23
15	DI 24	Entrada de propósito general 24
16	DI 25	Entrada de propósito general 25
17	DI 26	Entrada de propósito general 26
18	DI 27	Entrada de propósito general 27
19	DI 30	Entrada de propósito general 30
20	DI 31	Entrada de propósito general 31
21	DI 32	Entrada de propósito general 32
22	DI 33	Entrada de propósito general 33
23	DI 34	Entrada de propósito general 34
24	DI 35	Entrada de propósito general 35
25	DI 36	Entrada de propósito general 36
26	DI 37	Entrada de propósito general 37
27	COMMON	Relevador
28	DO 1b	CPU_OK(B contact)
29	DO 1ª	CPU_OK(A contact)
30	DO 2b	Servo ON(B contact)
31	DO 2ª	Servo ON(A contact)
32	DO 3b	Alarma (B contact)
33	DO 3ª	Alarma (A contact)
34	DO 10	Modo de operación automático
35	DO 11	Regreso a origen
36	DO 12	1 during operation
37	DO 13	2 during operation
38	DO 14	Estado de reinicio de programa
39	DO 20	Salida propósito general 20
40	DO 21	Salida propósito general 21
41	DO 22	Salida propósito general 22
42	DO 23	Salida propósito general 23
43	DO 24	Salida propósito general 24
44	DO 25	Salida propósito general 25
45	DO 26	Salida propósito general 26
46	DO 27	Salida propósito general 27
47	DC24V	DC+24V
48	DC24V	DC+24V
49, 50	GND	Tierra

Tabla 2. Arreglo de pines en el conector estándar de I/O

Interfaz de comunicación

Como se mencionó en apartados anteriores el controlador puede comunicarse con dispositivos externos como lo es el teach pendant o la computadora, usando la interfaz RS-232C.

Normas de la interfaz

V.24 Son las especificaciones para las comunicaciones que incluyen la asignación de pin. Esta es usada junto con la norma V.28 para definir las especificaciones de la comunicación asíncrona o síncrona

V.28 Son las especificaciones que definen las características de las señales eléctricas.

RS-232C es una combinación de V.24 y V.28

Aplicaciones

Una de las más comunes aplicaciones de la interfaz es en el puerto COM y el puerto serial de diversos aparatos que utilizan el modo asíncrono de comunicación (ASYNC)

Tipos de conectores utilizados.

DB25 este conector es utilizado en ambos tipos de comunicación, ya sea asíncrona o síncrona. En la tabla de abajo se muestra la función de cada pin, las señales utilizadas por la gran mayoría de aparatos están resaltadas en negro.

Señal	DB25 Contacto No
Shield	**1**
TD	**2**
RD	**3**
RTS	**4**
CTS	**5**
DSR	6
Signal Ground	**7**
DCD	**8**
+ VOLTAGE	9
- VOLTAGE	10

Unassigned	11
Secondary DCD	12
Secondary CTS	13
Secondary TD	14
Transmitter Signal Element Timing (clock line)	**15**
Secondary RD	16
Receiver Signal Element Timing (clock line)	**17**
Local Loop back	18
Secondary RTS	19
DTR	**20**
Remote Loop back	21
RI	22
Data Signal Rate Selector	23
Transmit Signal Element Timing	24
Test Mode	25

Tabla 3. Arreglo de pines en el conector DB25

El conector DB9 es utilizado solo en conexiones asíncronas como lo es el puerto de comunicaciones de la computadora.

En la siguiente tabla se muestra las configuraciones de los pines. Las señales más utilizadas están resaltadas en negro.

Signal Name	DB9 Contact
DCD	**1**
RD	**2**
TD	**3**
DTR	**4**
Signal Ground	**5**
DSR	6
RTS	**7**
CTS	**8**
RI	9

Tabla 4. Arreglo de pines en conector DB9

El conector DB25 se utiliza para la comunicación con la computadora a 19200 bps y el DB9 se utiliza para conectar el controlador con el teach pendant (a 9600 bps) como lo muestra la figura 4.2.1.5

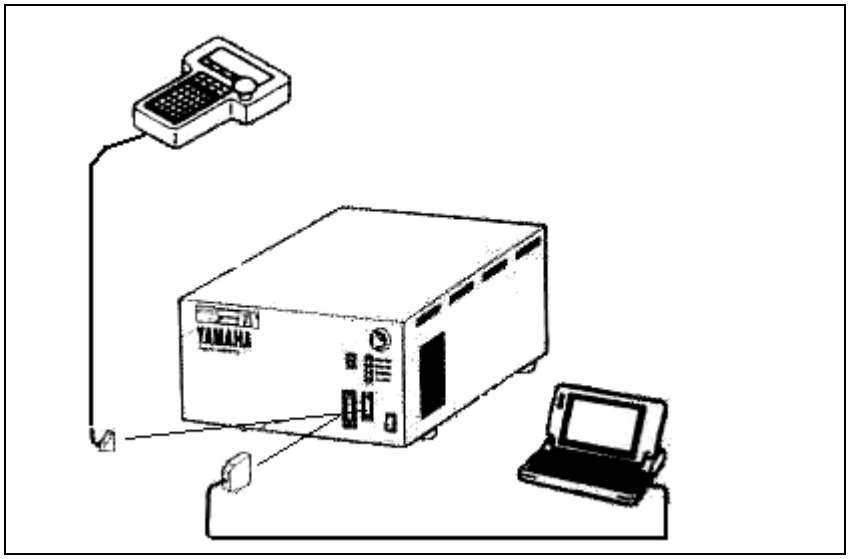


Figura 4.2.1.5 Conexión del equipo de programación en los puertos de comunicación [14].

Para comunicarse se utiliza los siguientes modos:

1) Mediante el comando SEND

Ejemplos:

SEND A TO CMU ------ La variable A se transmite al dispositivo externo

SEND CMU TO P100 ------ El punto P100 es recibido del dispositivo externo

2) Con la función ONLINE se puede transmitir varios comandos directamente desde el teach pendant.

Ejemplos:

@ RUN ------ Ejecuta el programa

@ MOVE P, P123, SPEED=30 ------ Mueve el robot al punto P123 al 30% de su velocidad máxima.

En el primer caso se trata de una programación estática u off-line, en el segundo caso es una programación directa u on-line.

4.2.2 Modos de operación

La operación del robot, la podemos clasificar en cuatro modos básicos de la siguiente manera:

1) Modo automático “AUTO”
2) Modo programa “PROGRAM”
3) Modo manual “MANUAL”
4) Modo sistema “SYSTEM”

Cada modo de operación es seleccionado mediante la tecla de función correspondiente en el teach pendant

1.- El modo “AUTO” es seleccionado para la ejecución del programa. Esto se logra presionando la tecla “START” del teach pendant.
2.- En el modo “PROGRAM” se pueden crear o corregir los programas.
3.- El modo “MANUAL” es seleccionado para controlar manualmente el robot. En este modo se pueden introducir nuevas coordenadas.
4.- El modo “SYSTEM” es para el control de los parámetros de todo el sistema, como el modelo de robot utilizado, o bien realizar un autodiagnóstico del controlador.

4.2.3 Principio de funcionamiento

Ejecuta una serie de instrucciones introducidas en su memoria en forma de programa y, por tanto, se asemeja a las computadoras. No obstante, existen tres características fundamentales que lo diferencian claramente de las computadoras estas son:
– Pueden conectarse directamente a los sensores y actuadores mediante sus puertos de entrada/salida para equipos industriales,

– Su diseño permite que funcionen en ambientes industriales duros (temperatura, vibraciones, etc.),

– Por último, la programación se basa en lenguajes específicamente desarrollados para el tratamiento de funciones de automatismo, de modo que ni su instalación ni su uso requieren conocimientos de informática.

Estructura básica

La estructura básica del controlador programable se fundamenta en tres elementos funcionales principales: procesador, memoria y entradas/salidas “Todo o Nada”. El enlace eléctrico de estos elementos se realiza por medio de un bus. Un bloque de alimentación proporciona las tensiones necesarias para el funcionamiento del conjunto.

Procesador

El cometido principal del procesador, o unidad central (UC), consiste en tratar las instrucciones que constituyen el programa de funcionamiento de la aplicación. Además de esta tarea, la UC desempeña las siguientes funciones:

– Gestión de entradas/salidas

– Control y diagnóstico del autómata mediante una serie de pruebas que se ejecutan en el momento del encendido o cíclicamente, durante el funcionamiento del sistema,

– Diálogo con el terminal de programación, tanto durante las fases de escritura y depuración del programa como durante su explotación, para realizar tareas de verificación y ajuste de datos.

Uno o varios microprocesadores ejecutan las funciones mediante el microsoftware previamente programado en una memoria de control o de sistema. Esta memoria muerta define la funcionalidad del controlador y no es accesible para el usuario.

Memoria de usuario

Permite almacenar las instrucciones que conforman el programa de funcionamiento del automatismo y los datos, que pueden ser de los siguientes tipos:

– Información susceptible de variar durante la ejecución de la aplicación. Por ejemplo, resultados de cálculos realizados por el procesador que se guardan para su uso posterior. Estos datos se denominan variables internas o palabras internas,

– Información que no varía durante la ejecución pero que el usuario puede modificar: las coordenadas de los movimientos, etc. Se denominan palabras constantes,

– Memorias de estado de las entradas/salidas, actualizadas por el procesador en cada turno de ejecución del programa.

El elemento básico de la memoria es el bit (abreviatura del inglés binary digit: dígito binario), que admite dos estados lógicos: 0 y 1. Los bits se agrupan en palabras (16 bits) o en bytes (8 bits) que se identifican mediante una dirección.

Para cada una de las partes (programa y datos), el volumen de la memoria se expresa en K palabras (1 K palabra = 210 palabras = 1024 bits) o en K bytes.

El controlador utiliza dos tipos de memoria:

– **Memoria viva**, o memoria RAM (Random Access Memory: memoria de acceso aleatorio). El contenido de este tipo de memoria puede leerse y modificarse en cualquier momento, pero se pierde en caso de falta de tensión (memoria volátil). Por tanto, necesita una batería de seguridad. La memoria viva se utiliza para escribir y poner a punto los programas y para almacenar los datos.

– **Memoria muerta,** cuyo contenido se conserva (no volátil) en caso de falta de tensión y que sólo puede leerse. Su escritura requiere el borrado total previo por medio de un procedimiento especial externo al controlador, por rayos ultravioletas (memorias EPROM y REPROM) o eléctrico (Memorias EEPROM). Se utilizan para almacenar los programas previamente depurados.

La memoria de programa se ubica en uno o varios cartuchos que se insertan en el módulo procesador o en un módulo de ampliación de memoria. La memoria de datos, y en ocasiones la memoria de programa, se integra en el procesador (memoria “on board”).

Entradas/salidas

Las entradas/salidas garantizan la integración directa del controlador en el entorno industrial. Sirven como enlace entre el procesador y el proceso. Todas cumplen una doble función básica:

– Función de interfaz para recibir y tratar señales procedentes del exterior (sensores, pulsadores, etc.) y para emitir señales hacia el exterior (control de

actuadores, alarmas, etc.). El diseño de estos interfaces, con desacoplamiento optoelectrónico (PS2801 ver ficha técnica en el apéndice B), asegura la protección del controlador contra señales parásitas. Como lo muestra la siguiente figura.

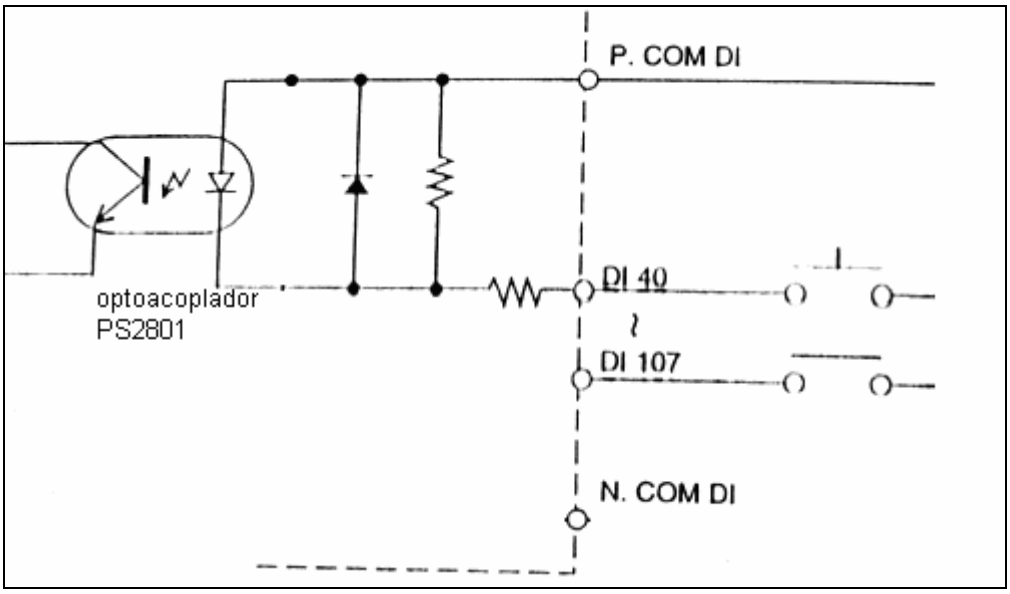


Figura 4.2.2 Optoacoplamiento de las señales de I/O [14].

– Función de comunicación para el intercambio de señales con la unidad central por medio de un bus de entradas/ salidas.

Bus

El bus consiste en un conjunto de conductores que enlazan entre sí los distintos elementos del controlador, que consta de conectores a los que se enchufan los distintos módulos: procesador, ampliación de memoria, interfaces y acopladores, que en el figura 4.2.3 se representa en un diagrama de bloques. Se organiza en varios subconjuntos que gestionan distintos tipos de tráfico:

1. Bus de datos para las señales de entrada/salida,

2. Bus de direcciones de las entradas/salidas,

3. Bus de control para las señales de servicio, por ejemplo, los topes de sincronización, el sentido de los intercambios, el control de validez de los intercambios, etc.,

4. Bus de distribución de las tensiones generadas por el bloque de alimentación.

Alimentación

Genera las tensiones internas que se distribuyen a los módulos del controlador a partir de una red de 110 o 220 V en corriente alterna.

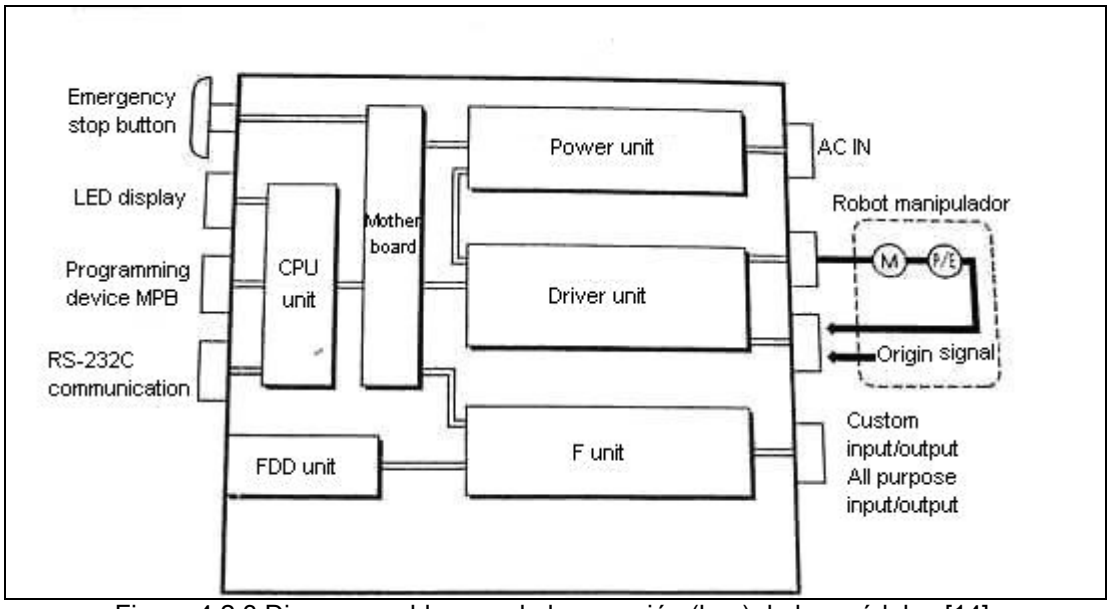


Figura 4.2.3 Diagrama a bloques de la conexión (bus) de los módulos [14].

4.3 El robot Yamaha

El robot Yamaha modelo YK550, es un robot de configuración SCARA con cuatro grados de libertad que son rotacional, rotacional, prismático y rotacional (RRPR).

Esta configuración está especialmente diseñada para realizar tareas de montaje en un plano. Está constituida por dos articulaciones de rotación con respecto a dos ejes paralelos, y una de desplazamiento en sentido perpendicular al plano.

Los robots SCARA son unidades articuladas que hacen la tarea de tomar y colocar (pick and place), que es la rutina que se va a ejecutar. En la figura 4.3.1 se muestra la foto del robot.

Figura 4.3.1 Robot Yamaha YK550H (fuente Nissan Time).

4.3.1 Datos técnicos

En la siguiente tabla se especifican sus datos técnicos del robot manipulador.

Modelo			YK550H
Especificaciones de los ejes	X eje	Longitud de brazo	300mm
		Rango de rotación	±90°
	Y eje	Longitud de antebrazo	250mm
		Rango de rotación	± 145°
	Z eje	Carrera	200mm
	R eje	Rango de rotación	± 180°
Motor		X eje	800W
		Y eje	200W
		Z eje	200W
		R eje	100W
Velocidad máxima		X, Y combinados	6.7m / s
		Z eje	2.3m / s
		R eje Alta velocidad	1700°/s
Repetibilidad*1		X,Y eje	± 0.02mm
		Z eje	± 0.01mm
		R eje	± 0.005°
Carga máxima			5kg
Standard cycle time(con 2kg carga util)			0.38s
Momento de inercia*2 permitido en el eje R		R eje Alta velocidad	0.05kgm2 (0.5kgfcms2)
Cableado usado			0.2sq × 20 cables
User tubing			ø6 × 2
Robot cable			Standard 3.5m Option: 5m, 10m
Controller, power supply capacity			RCX142-R 1700VA
Peso			46kg
*1: A temperatura ambiente constante. (X, Y ejes) *2: Hay limites para el coeficiente de aceleración.			

Tabla 5. Datos técnicos del robot Yamaha YK550H

En el siguiente diagrama (figura 4.3.2, obtenida de la pagina en internet de Yamaha Robotics) se detallan sus dimensiones del brazo del robot manipulador Yamaha YK550H, así como su área de trabajo típica en un robot con una configuración tipo SCARA

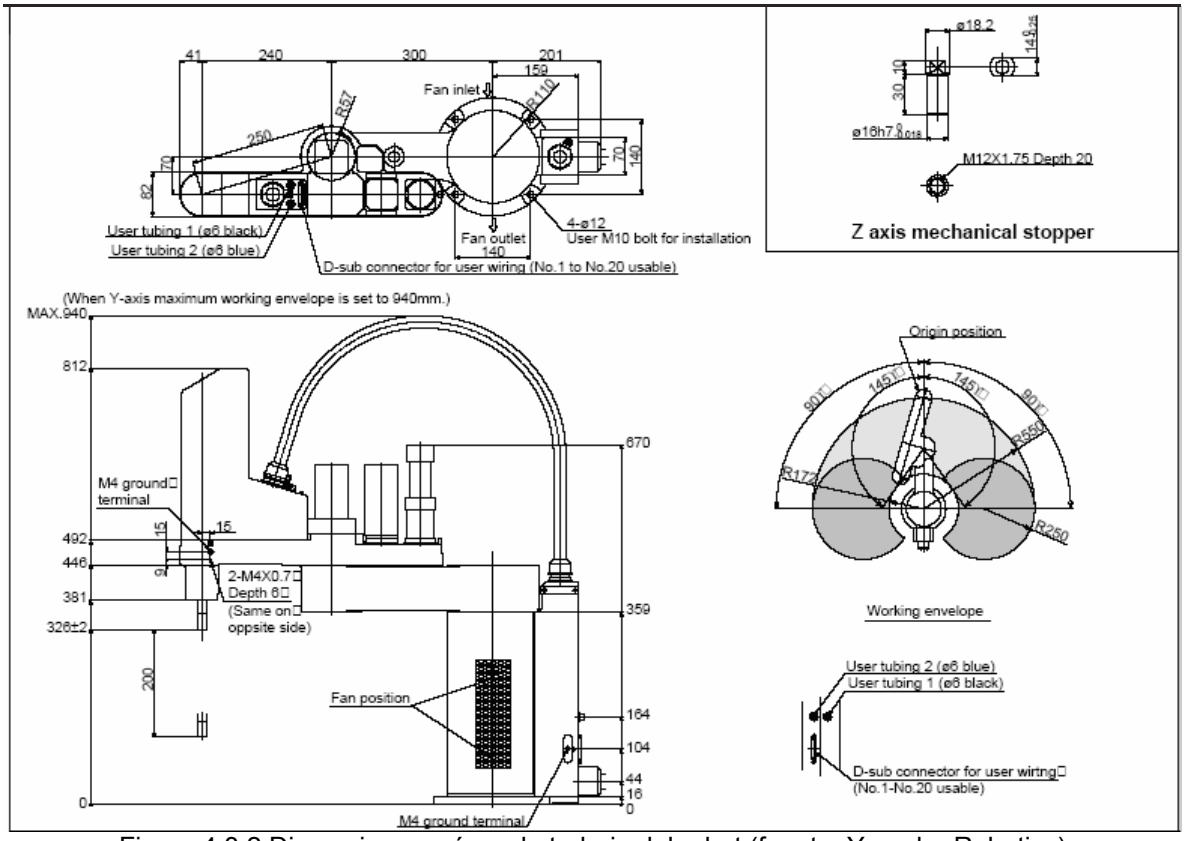


Figura 4.3.2 Dimensiones y área de trabajo del robot (fuente: Yamaha Robotics)

4.3.2 Lenguaje de programación

El lenguaje del robot Yamaha es un lenguaje desarrollado por Yamaha Motor Company para una simple y eficiente programación.

Los comandos son muy similares a los del lenguaje BASIC y esto hace que cualquier movimiento complejo sea fácil de programar.

Como se sabe existen dos formas para traducir los lenguajes de alto nivel a un lenguaje más conveniente para la máquina, estos son los intérpretes y los compiladores.

Los lenguajes son interpretados si cada vez que se lee una orden se traduce a lenguaje máquina. Esto permite ejecutar inmediatamente y corregir sobre la marcha.

Cuando se traduce todo el programa una sola vez cuando está terminado, se llaman compilados y funcionan más rápidamente que los interpretados, pero no permiten modificaciones fácilmente por que el más mínimo cambio obliga a compilar todo de nuevo.

Este último método es el utilizado por el controlador para una rápida ejecución del programa.

Al igual que un lenguaje natural, un lenguaje de programación tiene una gramática, un conjunto de reglas sintácticas, que veremos a continuación.

El formato de las sentencias o enunciados, siempre se escriben en una línea y son arreglados de la siguiente forma:

[< etiqueta >:] <expresión> [<operando>]

[ ] Muestra elementos que no tienen que ser incluidos en el enunciado.
<> Muestra elementos que deben ser escritos en un formato especifico.

- La etiqueta no tiene que incluirse en el comando. Todas las etiquetas comienzan con un asterisco y terminan con dos puntos.
- El operando puede ser emitido en ciertos comandos.

Constantes o numerales

Los numerales de tipo carácter se encierran entre comillas, ejemplos: “Yamaha robot”, “a”

Los numerales de tipo entero comprenden los valores entre -9999999 a +9999999.

Los numerales binarios son de hasta 8 bits, y para definir que es un número binario se escribe al inicio &B.

Los numerales hexadecimales van de 0 a FFF y para denotarlos se escribe al inicio del número &H.

Los numerales de tipo real utilizan siete dígitos incluyendo enteros y decimales y comprenden el rango de -999999.9 a + 999999.9. En forma exponencial va de $-1.0x10^{38}$ a $+ 1.0x10^{38}$.
Ejemplos:-1.23456E-12, 1E5

Variables dinámicas

Una variable es una letra con la que se representa un número. Cualquier letra puede fungir como nombre de variable. Sin embargo hay secuencias de caracteres reservadas que no pueden ser un nombre legal de variables. He aquí que no se pueden utilizar variables que comiencen con FN, DIn, DOn, Pn o Sn (con n=0 a 9).

El tipo de variable se especifica al final del nombre, con un carácter. Ejemplo:
COUNT% -------- variable de tipo entero

COUNT! ---------- variable de tipo real

STRING$ --------- variable de tipo carácter

Variables estáticas

Variables para definir puntos.

Los puntos son definidos por números enteros o alguna expresión, su sintaxis es la siguiente: Pnnn o P[expresión] n=0 a 9

Ejemplos:

P0

P[A]

P1=1000 -2000 0 0

MOVE P, P1

HALT

Variable de cambio de coordenada

Sintaxis: Sn o S[expresión] n=0 a 9

Ejemplos:

S1

S[A], S[BASE]

Variables de entrada

Esta variable expresa el estado de la señal de entrada.

Sintaxis:

DIm(b,b,…,b) m: número de puerto 0 al 7, 10 al 13

DI(mb,mb,…,mb) b: definición del bit 0 al 7

Los bits se definen en orden ascendente de derecha a izquierda. Ejemplos:

A%= DI5(7,4,0)

El valor de la variable A% es substituido por los estados de entrada DI(57), DI(54) y DI(50).

Variables de salida

Esta define los valores de salida a un puerto.

Sintaxis:

DOm(b,b,…,b) m: número de Puerto 2 al 7,10,11

DO(mb,mb,…mb) b: bit 0 al 7

Ejemplo: A%=DO5(7,4,0)

Los valores de A% son sustituidos por los estados de salida DO(57), DO(54) y DO(50)

A continuación se sintetiza en una tabla la lista de comandos utilizados para la programación del robot.

Tipo de declaración	Sintaxis	Ejemplo
Declaración de cambio de asignación	Sm=<expresión de cambio>	S1=S0 S[A]=S1+S2
Saltos	GOTO <etiqueta>	GOTO *loop
Decisión	IF <expresión> THEN <etiqueta> <expresión> ELSE <etiqueta> <expresión>	IF A=1 THEN *Prgend IF A=1 THEN *L1 ELSE *L2

	IF <expresión> THEN
	.
	.
	ELSE
	.
	.
	ENDIF
		IF DI3(1)=1 THEN MOVE P,P1
		DO(30)=1
		ELSE
		MOVE P,P2
		DO(30)=0
		ENDIF

Múltiple decisión	SELECT[CASE]<expresión> CASE<expresión list 1> [Block1] CASE<expresión list 2> [Block 2] . . CASE ELSE [Block n] END SELECT	SELECT CASE DI3() CASE 1 CALL *EXEC(1,10) CASE 2 CALL *EXEC(21,30) CASE ELSE END SELECT
Subrutinas	GOSUB <etiqueta> . . . RETURN	DECLARE *INITIALIZE GOSUB *SUBROUTIN GO SUB *INITIALIZE . RETURN
Salto condicionado	ON<expresión> GOTO <etiqueta>	ON A GOTO *L10, *L20
Subrutina condicionada	ON <expresión> GOSUB <label>	ON A GOSUB *SB10, SB20
Bucles	FOR <variable>=<expresión 1> TO <expresión 2> STEP <expresión 3> NEXT <variable>	FOR A=10 TO 4 STEP2 . NEXT A
Terminación del bucle	EXIT FOR	EXIT FOR
Bucle condicionado	WHILE <expresión> . . WEND	WHILE A>10 WHILE DI1()=&B1100101 . WEND
Llamada a un procedimiento	CALL <etiqueta> (parámetros)	CALL *DISTANCE CALL *AREA(2.5,X,REF)
Fin del procedimiento	EXIT SUB	EXIT SUB
Suspensión temporal	HOLD <expresión>	HOLD “ERROR STOP”
Fin del movimiento	HALT <expresión>	HALT “PROGRAM STOP”
Cambio de programa	SWI <nombre del programa>	SWI<ABC>
Movimiento del robot	MOVE P<definición del punto> L<opcion> C<opción>	MOVE P,P100 MOVE L,P1, STOPON DI3(0)=1
Movimiento axial	DRIVE <expresión>,<expresión>	DRIVE(1,100.00)

	SPEED= <expresión> S	DRIVE(2,100.00),(1,50.00) DRIVE(A,P10),S=10
Control del servo	SERVO ON <expresión> OFF FREE	SERVO ON SERVO OFF SERVO FREE(2)
Espera a que se cumpla una condición	WAIT <expresión de la condición>	WAIT DI2()=&B11010110 WAIT DI2(3,2)=&B10,1000
Conjunto de condiciones	SET <DO variable> <variable> <MO variable> <LO variable>	SET DO2(7,5,2) SET MO(27,25,23) SET LO0(7,5)
Impresión	PRINT <expresión>	PRINT “COUNT=”;C PRINT A$
Entradas	INPUT <variable> <variable>	INPUT A INPUT S0,S1
Comunicación	SEND <file 1> TO <file 2>	SEND PGM TO CMU SEND <000000> TO SCR
Retardo	DELAY <expresión>	DELAY 1000 DELAY A*10
Cambio de velocidad	SPEED <expresión>	SPEED 70 SPEED A*10
Tolerancia en la ubicación	TOLE (<expresión>)=<expresión>	TOLE(1)=10
Aceleración	ACCEL<expresión> ACCEL(<expresión>)=<expresión>	ACCEL 100 ACCEL(1)=100
Selección del modo de comunicación	ONLINE OFFLINE	ONLINE OFFLINE
Cambio de coordenadas	SHIFT <variable>	SHIFT S1 SHIFT S[A]
Inicio de tarea	START <etiqueta>, Tn	START *IOTAREA,T2
Paro de tarea	CUT Tn	CUT T2
Suspender tarea	SUSPEND Tn	SUSPEND T2
Reiniciar tarea	RESTART Tn	RESTART T2
Terminar tarea	EXIT TASK	EXIT TASK

Tabla 6. Compendio de comandos del lenguaje VIP Windows

4.4 Conclusiones

En este capítulo se presenta la información de cómo es que opera el controlador del robot, esto nos permite ir asociando la teoría aprendida en clases con la aplicación en el campo laboral, para que de esta manera podamos resolver cualquier tipo de dificultades que se nos presente a la hora de integrar el equipo en el sistema de manufactura flexible existente.

Además la información aquí presentada tal como la apertura de la arquitectura del controlador lógico programable y la del robot que nos permite familiarizamos con el equipo a utilizar, así como el de aprender los comandos del lenguaje de programación del controlador que son muy similares a los comandos utilizados en el lenguaje de programación de alto nivel BASIC, para poder controlar el robot.

Debido a que el controlador lógico programable traduce el código del programa utilizando el método de compilador, es decir, traduce todo el programa una sola vez esto permite que se ejecute rápidamente.

Capítulo 5

SISTEMA INTEGRADO Y PRUEBAS EXPERIMENTALES

5.1 Introducción

En el capítulo 2 se hablo de los métodos de planificación de tareas en los robots, en este capítulo se mostrara una manera práctica de planificarla, auxiliándonos de conceptos básicos de geometría analítica. Por otra parte se menciona la integración de la estación en el sistema existente, así como también la obtención del modelo matemático de la cinemática del robot.

5.2 Integración de la estación robótica.

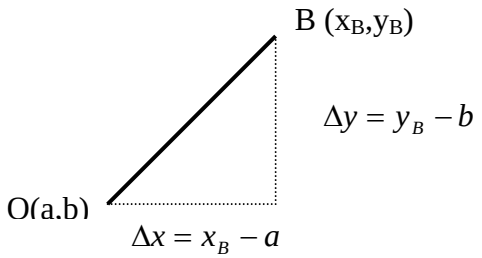
Para integrar la estación robotizada de embalaje en el sistema de manufactura existente se estuvo restringido a la distribución del centro de ensamblaje existente y a las dimensiones de la planta, por lo que para la colocación de la banda que va a transportar las cajas se dejo un espacio de 2m entre la pared y la barrera protectora (ver figura 5.2), con el fin de dejar espacio para el área de control de calidad en el que se colocara el almacén de las cajas llenas con los mecanismos inspeccionados. Una vez colocada la banda, para instalar el robot se tuvo que considerar que quedara en un lugar donde los puntos de recolección y descarga estuvieran dentro de el lugar geométrico que describe el robot, para esto se hizo la siguiente consideración matemática, partiendo de conceptos de geometría tenemos que sea:

A (x_A,y_A) el punto donde se recogerá la pieza.
B (x_B,y_B) el punto donde se descargara la pieza
O(a,b) la base del robot (el origen)

r = la longitud del manipulador (550mm)

Para que los puntos A & B estén en el lugar geométrico que describe el robot, es decir, que estén en los puntos del plano que se encuentran a la distancia fija ‘r’ del punto ‘O’.

no es necesario determinar si la distancia ‘O’ a ‘A’ esta en el lugar geométrico que describe el robot ya que la distancia de ‘O’ a ‘A’ es conocida por las razones ya expuestas, por lo tanto sabemos que si es parte del lugar geométrico. Para determinar la distancia de ‘O’ a ‘B’.



Denotando por $\left|\overline{OB}\right|$ la distancia de ‘O’ a ‘B’, se utilizan las barras de valor absoluto por que la longitud es un número no negativo. Usando el teorema de Pitágoras tenemos:

$$\left|\overline{OB}\right|^2 = \left|\Delta x\right|^2 + \left|\Delta y\right|^2 \tag{5.1}$$

$$\left|\overline{OB}\right| = \sqrt{(x_B - a)^2 + (y_B - b)^2} \tag{5.2}$$

La formula (5.2) la podemos generalizar para el calculo de la distancia entre dos puntos cualquiera.

Como el movimiento del robot genera una circunferencia el punto ‘B’ debe estar en la circunferencia con radio igual a ‘r’, para ello debe cumplirse que $d(B,O) \leq r$, es decir, de la formula de la distancia entre dos puntos ahora tenemos a ‘r’ en lugar de ‘B’ y sustituyendo queda $r \geq \sqrt{(x_B - a)^2 + (y_B - b)^2}$ (5.3)

Ahora del punto B (que es donde se descargara la pieza) conocemos las coordenadas sustituyendo los valores en la formula (5.3) tenemos que:

$$550 \geq \sqrt{(x_B - 0)^2 + (y_B - 0)^2} \tag{5.4}$$

$$550 \geq \sqrt{(x_B^2 + y_B^2)} \tag{5.5}$$

Teniendo esto en mente se coloco el robot a una distancia de 210mm de la banda que transporta las cajas (lugar de descarga) y a una distancia de 210mm del lugar de donde se recogen los mecanismos. Para evitar accidentes se coloco una barrera de protección, esto debido a que si se produjera un error en el sistema de control, el robot podría lastimar al que pasara en ese momento ya que el área de paso esta dentro de el lugar geométrico que describe el robot.

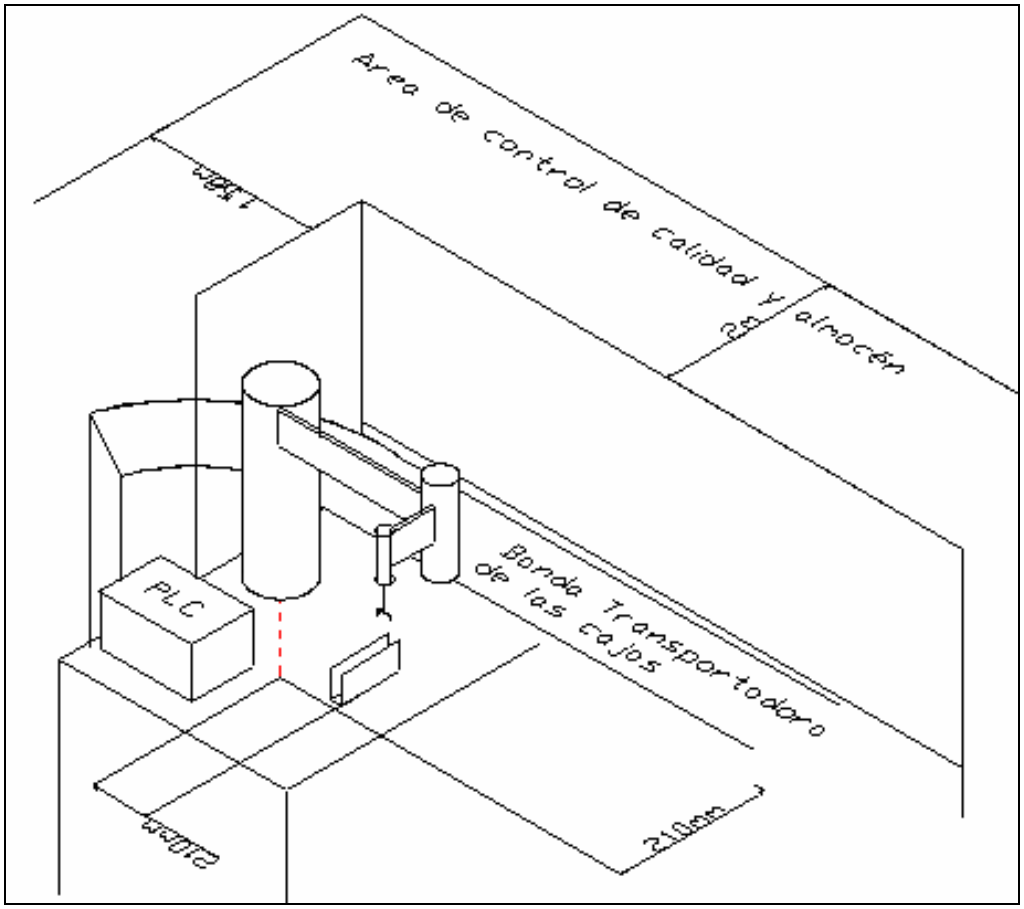


Figura 5.2 Bosquejo de la estación robótica de embalaje

5.3 Robot – Controlador

Los componentes del sistema son un robot manipulador Yamaha YK550H de alta velocidad y configuración SCARA, una unidad de control Yamaha QRCH una botonera de programación (teach pendant) y un ordenador PC con procesador Athlon 2200 a 1.5 MHz sobre el cual se ejecuta el entorno de comunicación VIP Windows. El lenguaje de programación tiene comandos muy similares al lenguaje BASIC.

Las características más importantes de este manipulador son: esta formado por 4 elementos. La base, el brazo (“upper arm”), el antebrazo (“lower arm”), y la muñeca (“wrist”).

El manipulador tiene 4 articulaciones (“joints”): el hombro (“shoulders”), codo (“elbow”), muñeca (“wrist”) y la rotación del elemento final (“tool rotate”).

La unidad de control del robot, a parte de suministrar las señales de potencia y control al robot manipulador, tiene una conexión al ordenador de 19200 bps y una conexión con la botonera de 9600 bps. También permite la entrada/salida de señales digitales accesibles desde el programa

5.3.1 Composición del robot

Como en este proyecto se trata de manipular un brazo robótico y no el de diseñarlo, se utilizo el de la marca Yamaha modelo YK550H, que cuenta con 4 grados de libertad, entre los cuales tiene 3 movimientos principales que son: el hombro, codo y la muñeca, esta tiene un movimiento rotacional y otro prismático.

El movimiento del hombro tiene un rango de rotación de 90º. El codo su rango de movimiento rotacional es de 145º. El movimiento rotacional de la muñeca esta en el rango de 180º con un desplazamiento de 200mm.

Para los movimientos del robot se utilizan cuatro servos, manufacturados por servosystems, del tipo smartmotors y modelo animatics 3410. Que son utilizados uno para mover el brazo, otro para el antebrazo y dos para los movimientos (rotacional y prismático) de la muñeca, el cierre y apertura del efector final se efectúa neumáticamente.

5.3.2 Funcionamiento

La arquitectura cerrada del PLC tiene incluidos elementos de control, como lo son la tarjeta de control de potencia y las tarjetas para el control de movimiento de los motores, que cuentan con registros para datos de posición velocidad y aceleración, así como un filtro PID.

Una vez programada la rutina y cargada en el controlador este se encarga de ejecutarla, enviando mediante el bus de comunicación la configuración y trayectoria del movimiento al circuito de control de movimiento que genera las señales de PWM que pasan por la estructura de puente H que permite el control bidireccional de los servos. El encoder proporciona la retro para el control en lazo cerrado, es decir, este entrega las señales que indican la posición y sentido de giro del motor, y el filtro PID se encarga de corregir el error entre la posición deseada y la real. También recibe las señales de los sensores de proximidad, para la retroalimentación de los movimientos, además envía las señales de control al motor de la banda transportadora de las cajas y a la pinza neumática (efector final) para el cierre y apertura de la misma.

5.4 Planificación de la tarea

El movimiento del brazo se restringe a las dimensiones, debido a las siguientes consideraciones:

1) El punto donde se recogerá la pieza es conocido y tiene las siguientes coordenadas (274.22, -45.90, 85.98) en los ejes (x,y,z) respectivamente las dimensiones son en milímetros (mm), tomando como origen la base del robot.
2) El movimiento del hombro del brazo no debe exceder el rango de 0º a 90º, debido a que pasando este rango el movimiento rotatorio sería más largo, por lo que se requeriría mayor tiempo para el recorrido.
3) Las cajas donde se van a colocar los mecanismos tienen un dimensión de 30X50 cm de ancho y largo respectivamente, la banda donde se transportan las cajas tiene una longitud de 4.30 m, que se distribuirán de la siguiente forma 2m para colocar cajas vacías, 50cm para el espacio donde estará la caja para ser llenada y los 1.8m restantes será el espacio que ocupen las cajas llenas.
4) Del lugar donde se va a colocar la pieza sólo se conoce una distancia que corresponde a la coordenada y_B, que es la distancia de la base del robot a la banda que contendrá la caja en donde se depositarán los mecanismos, y_B es una coordenada que va a variar.

Como el problema se trata de mover el brazo a dos distintos puntos, en un entorno estático y libre de obstáculos. Se opto por un método cartesiano para la planificación de la tarea, en la que se buscara el camino más corto que cumpla con las restricciones arriba mencionadas. Para esto se ocuparan conceptos de métodos geométricos para su planificación, es decir buscar la trayectoria de menor recorrido y debido a que el tiempo, es un factor

importante, ya que se necesitaba desocupar a la persona encargada de empaquetar, para colocarla en otra actividad, lo que se hizo fue tomar medidas de tres puntos en la coordenada y y otros tres en la coordenada x, estos puntos fueron: en y las dos esquinas de la caja y el centro de la misma, así como sus respectivas coordenadas en x, el valor de estas coordenadas, se obtuvieron mediante el teach pendant, al escoger estos tres puntos se pretende que se llenen las cajas y también hacer que los movimientos del robot manipulador sean pocos con el fin de agilizar el proceso, es decir, que al hacer que se tenga el menor número de movimientos se tendrá un menor consumo de tiempo a la hora de recoger la pieza, para evitar la acumulación de los mecanismos en el conveyor.

Con la finalidad de analizar el movimiento del robot cuando esta recorriendo la trayectoria se utilizo el modelo cinemático directo e inverso.

5.5 Programación de la tarea.

A continuación se mostrara el listado del programa para la ejecución de la rutina, en los comentarios se describe que es lo que sucede en cada línea, hay que recordar que el lenguaje utilizado es muy similar al BASIC.

```
NAME=ROBOT
START *ROBOT
*ROBOT:
WAIT DI2(3)=1 'ESPERA POR SEÑAL DE PRESENCIA DE CAJA EN LA BANDA TRANSPORTADORA
IF DI2(3)=0 THEN
    GOSUB *ALARM
ENDIF
WAIT DI2(0)=1 'ESPERA POR SEÑAL DE AUTO/STEP.
P2=P12  'POSICION
P3=P13
P4=P14
P5=P15
P6=P8       'PONE P6 EN 0.
FOR C=1 TO 5
FOR A=1 TO 5
MOVE P, P1, Z=0    'MOVER A LA POSICION DE AGARRE CON, Z=0
WAIT DI2(0)=1 'ESPERA POR SEÑAL DE AUTO/STEP.
```

```
WAIT DI2(2)=1 'ESPERA HASTA QUE HAY PRESENCIA DE PIEZA.
MOVE P, P0    'MUEVER A LA POSICION DE AGARRE BAJANDO LA PINZA.
DELAY 250
WAIT DI2(0)=1
DO2(1)=1      'CIERRA TENAZA.
DELAY 250
MOVE P, P1 'SUBE EL EFECTOR FINAL DE LA POSICION DE AGARRE DE LA PIEZA.
WAIT DI2(0)=1
MOVE P, P2 'MUEVE ARRIBA DE LA 1er ESQUINA DE LA CAJA.
WAIT DI2(0)=1
WAIT DI2(3)=1 '.
WAIT DI2(0)=1
MOVE P, P3 'BAJA EL EFECTOR PARA LA DESCARGA.
WAIT DI2(0)=1
DELAY 25
WAIT DI2(4)=1, 1500
IF DI2(4)=0 THEN
  GOSUB *ALARM
ENDIF
WAIT DI2(0)=1
DO2(1)=0      'ABRE TENAZA.
DELAY 150
MOVE P, P2'SUBE EL EFECTOR FINAL.
P2=P2+P16    'INCREMENTA EL VALOR DE LA COORDENADA Y PARA .
P3=P3+P16    'NUEVA POSICION DE DESCARGA.
NEXT A   'INCREMENTA EL VALOR DE A.
IF A=5 THEN GOTO*RB2
*RB2:
FOR B=1 TO 05
MOVE P, P1, Z=0    'COLOCA EL EFECTOR FINAL EN LA POSICION DE AGARRE , Z=0
WAIT DI2(0)=1
MOVE P, P0 'BAJA EL EFECTOR FINAL
WAIT DI2(2)=1
DELAY 250
WAIT DI2(0)=1
DO2(1)=1      'CIERRA TENAZA
DELAY 250
WAIT DI2(0)=1
MOVE P, P1  'SUBE EL EFECTOR FINAL
WAIT DI2(0)=1
MOVE P, P4     'MOVER ARRIBA DE LA POSICION DE DESCARGA  CALCULADA.
WAIT DI2(0)=1
WAIT DI2(3)=1
MOVE P,P5 'BAJA EL EFECTOR EN LAPOSICION DE DESCARGA .
WAIT DI2(0)=1
DO2(1)=0       'ABRE TENAZA PARA DESCARGA
DELAY 150
MOVE P,P4 'SUBE EL EFECTOR.
P4=P4+P16     'ESQUINA SUPERIORDE CAJA Y.
P5=P5+P16     'NUEVA COORDENADA EN  Y.
NEXT B
P6=P6+P7
P2=P12+P6      'NUEVA COORDENADA EN X.
P3=P13+P6      'NUEVA COORDENADA EN X.
P4=P14+P6      'NUEVA COORDENADA EN X.
P5=P15+P6
NEXT C
WAIT DI2(0)=1
MOVE P, P9     'SUBE EL PRISMATICO POR SEGURIDAD MIENTRAS CORRE LA BANDA.
DELAY 10
```

```
DO2(5)=1      'SEÑAL DEL SENSOR PARA CORRER BANDA TRANSPORTADORA.
DELAY 1100   'ESPERA 1.1 SEC PARA QUE LA BANDA CORRA
DO2(5)=0
GOTO*ROBOT
*ALARM:
DO(26)=1 'ENCIENDE LUZ
WAIT DI(20)=0
DO(26)=0 'APAGA LUZ
RETURN
P0   = 274.22  -45.90   85.98   72.86    0.00    0.00
P1   = 274.19  -47.35    0.00   72.86    0.00    0.00
P2   =  -1.90  569.73    0.00  253.28    0.00    0.00
P3   =  -1.90  569.73   56.10  253.28    0.00    0.00
P4   =  15.05  319.98    0.00  251.52    0.00    0.00
P5   =  15.05  319.98   55.76  251.52    0.00    0.00
P6   =   0.00    0.00    0.00    0.00    0.00    0.00
P7   =  49.50    0.00    0.00    0.00    0.00    0.00
P8   =   0.00    0.00    0.00    0.00    0.00    0.00
P9   = 218.57  463.19    0.00   72.06    0.00    0.00
P10  = 195.89  494.39    0.00  253.43    0.00    0.00
P11  =   0.00    0.00    0.00    0.00    0.00    0.00
P12  =  -1.90  212.23    0.00  253.28    0.00    0.00
P13  =  -1.90  212.23   56.10  253.28    0.00    0.00
P14  =  15.05  176.98    0.00  251.52    0.00    0.00
P15  =  15.05  176.98   55.76  251.52    0.00    0.00
P16  =   0.00   71.50    0.00    0.00    0.00    0.00
P20  =   0.00    0.00    0.00    0.00    0.00    0.00
```

5.6 Resultados experimentales

Una propuesta para obtener las gráficas del movimiento, esto con el fin de analizar el movimiento del robot cuando esta recorriendo la trayectoria de la posición de carga hasta la posición de descarga se utilizo la cinemática inversa, esto sería hallar la matriz Jacobiana para obtener la velocidad del efector final a partir de las posiciones articulares (q_1, q_2, q_3) y de las velocidades articulares $\left(\dot{q}_1, \dot{q}_2, \dot{q}_3\right)$ como a continuación se muestra, a partir de la siguiente deducción de la cinemática del robot se obtuvo el algoritmo para programar en matlab el comportamiento del robot en el recorrido de la trayectoria que describe en su tarea de carga y descarga.

Obtención de la matriz Jacobiana para el robot SCARA

De la cinemática directa de posición se tiene que:

$$x = l_3 \cos(q_1 + q_2) + l_2 \cos(q_1) \tag{5.1}$$

$$y = l_3 \sin(q_1 + q_2) + l_2 \sin(q_1) \tag{5.2}$$

$$z = l_1 - q_3 \tag{5.3}$$

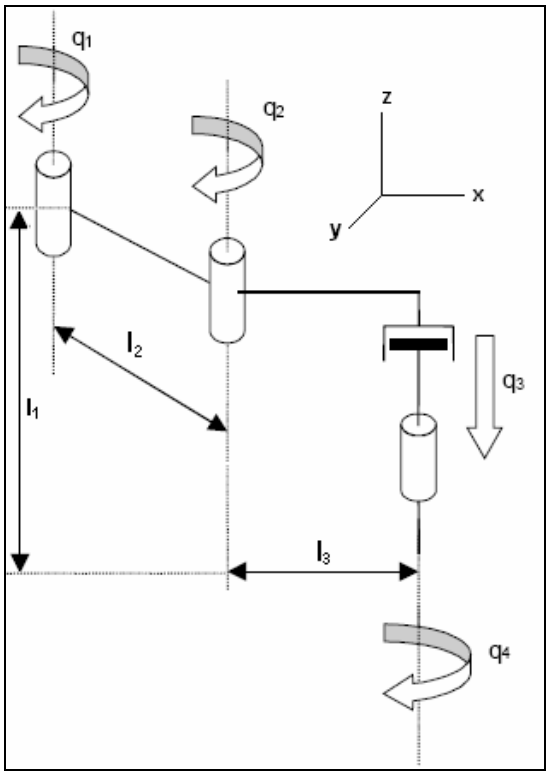


Figura 5.5 Esquema cinemático del robot SCARA

Donde:

l_1, l_2, l_3: Longitudes de los eslabones.

q_1, q_2: Articulaciones de revolución.

q_3: Articulación prismática.

Derivando ambos miembros de las ecuaciones (5.1), (5.2) y (5.3) se tendrá:

$$\dot{x} = -l_3 \sin(q_1 + q_2)(\dot{q}_1 + \dot{q}_2) - l_2 \sin(q_1)(\dot{q}_1) \tag{5.4}$$

$$\dot{y} = l_3 \cos(q_1 + q_2)(\dot{q}_1 + \dot{q}_2) + l_2 \cos(q_1)(\dot{q}_1) \tag{5.5}$$

$$\dot{z} = -\dot{q}_3 \tag{5.6}$$

Por lo que la matriz Jacobiana será: $\dot{x} = J\dot{q}$ (5.7)

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -(l_3 S_{12} + l_2 S_1) & -l_3 S_{12} & 0 \\ (l_3 C_{12} + l_2 C_1) & l_3 C_{12} & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} \tag{5.8}$$

Y el determinante de la matriz Jacobiana es:

$$|J| = -[-l_3 C_{12}(l_3 S_{12} + l_2 S_1) + l_3 S_{12}(l_3 C_{12} + l_2 C_1)] \tag{5.9}$$

Simplificando la expresión queda:

$$|J| = -l_2 l_3 S_2 \tag{5.10}$$

Donde las singularidades se encuentran cuando $|J| = 0$, esto es para $S_2 = 0$, es decir cuando $q_2 = 0, \pi$

De la cinemática inversa de posición se tiene:

$$x^2 + y^2 = l_2^2 + l_3^2 - 2l_2 l_3 \cos(180 - q_2) \tag{5.11}$$

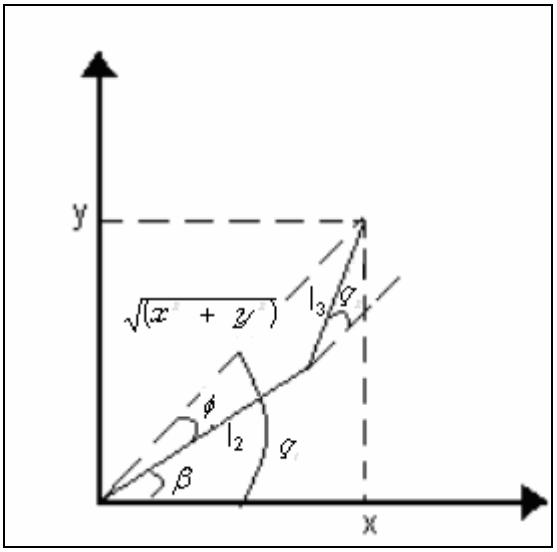


Figura 5.5.1 diagrama de la vista superior de los eslabones

Despejando q_2 de la ecuación

$$q_2 = \cos^{-1}\left\{\frac{x^2 + y^2 - (l_2^2 + l_3^2)}{2l_2 l_3}\right\} \tag{5.12}$$

$$tg\beta = \frac{y}{x} \tag{5.13}$$

$$\beta = a\tan 2(y, x) \tag{5.14}$$

$$l_3^2 = x^2 + y^2 + l_2^2 - 2l_2 \cos\phi\sqrt{(x^2 + y^2)} \tag{5.15}$$

$$\phi = \cos^{-1}\left\{\frac{x^2 + y^2 - (l_2^2 + l_3^2)}{2l_2 l_3}\right\} \tag{5.16}$$

Posición de codo abajo para:

$$q_1 = \beta - \phi; \quad q_2 \leq 0 \tag{5.17}$$

Posición de codo arriba para:

$$q_1 = \beta + \phi; \quad q_2 \geq 0 \tag{5.18}$$

$$q_3 = l_2 - z \tag{5.19}$$

Modelo cinemático inverso de velocidad (MCIV) $\dot{q} = J^{-1}\dot{x}$ (5.20)

$$\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = -\frac{1}{l_2 l_3 S_2} \begin{bmatrix} -l_3 C_{12} & -l_3 C_{12} - l_2 C_1 & 0 \\ l_3 S_{12} & l_3 S_{12} + l_2 S_1 & 0 \\ 0 & 0 & l_2 l_3 (C_1 S_{12} - C_{12} S_1) \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{5.21}$$

Modelo cinemático directo de aceleración (MCDA) $\ddot{x} = J\ddot{q} + \dot{J}\dot{q}$ (5.22)

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} -(l_3 S_{12} + l_2 S_1) & -l_3 S_{12} & 0 \\ (l_3 C_{12} + l_2 C_1) & l_3 C_{12} & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \end{bmatrix} \tag{5.23}$$

$$+ \begin{bmatrix} -2l_3 C_{12} - l_2 C_1 & -2l_3 C_{12} & 0 \\ -2l_3 S_{12} - l_2 S_1 & -2l_3 C_{12} & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix}$$

Modelo cinemático inverso de aceleración (MCIA) $\ddot{q} = J^{-1}[\ddot{x} - \dot{J}\dot{q}]$ (5.24)

$$\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = -\frac{1}{l_2 l_3 S_2} \begin{bmatrix} -l_3 C_{12} & -l_3 C_{12} - l_2 C_1 & 0 \\ l_3 S_{12} & l_3 S_{12} + l_2 S_1 & 0 \\ 0 & 0 & l_2 l_3 (C_1 S_{12} - C_{12} S_1) \end{bmatrix}$$

$$\left\{ \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} - \begin{bmatrix} -2l_3 C_{12} - l_2 C_1 & -2l_3 C_{12} & 0 \\ -2l_3 S_{12} - l_2 S_1 & -2l_3 C_{12} & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} \right\} \qquad (5.25)$$

Para la simulación en matlab de la geometría del movimiento que describe el robot al recorrer las trayectorias que cumplen con la tarea de carga y descarga, se comenzó por describir las trayectorias entre el punto de recolección y los puntos de descarga como se aprecia en la figura 5.5.2

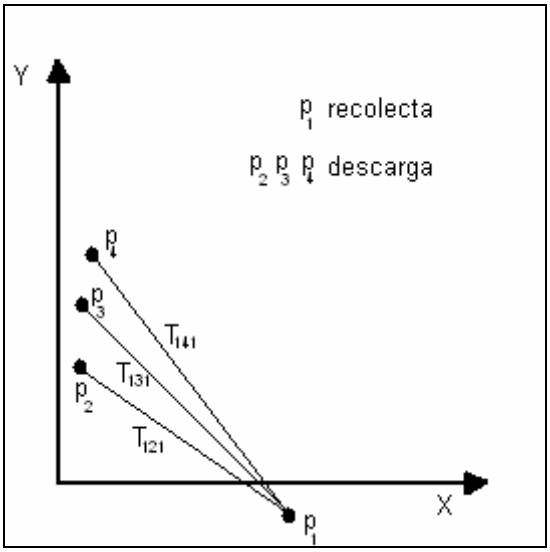


Figura 5.5.2 Trayectorias que sigue el robot.

Denominaremos como T_{121} a la trayectoria que describe el robot cuando va del punto de recolección $p_1(274.22,-45.9)$ al primer punto de descarga $p_2(1.90,212.23)$ físicamente este punto corresponde a la esquina de la caja que se encuentra próxima a la base del robot. T_{131} es la trayectoria que recorre el

robot desde p_1 hasta el $p_3(1.90,283.73)$ que es la coordenada correspondiente a el centro de la caja. T_{141} se denomina a la trayectoria que va del punto de recolección p_1 a la otra esquina de la caja $p_4(15.05,319.98)$.

Debido a que el entorno donde se desplazara el robot esta libre de obstáculos, basto con unir los puntos en los que se localizara el robot para cumplir con la tarea propuesta, ya que conocemos $\overline{p_1p_2}(x_2 - x_1, y_2 - y_1)$ se tiene que: $tg\alpha = \frac{y_2 - y_1}{x_2 - x_1}$ (5.26)

Sabemos de geometría analítica que $tg\alpha = m$ (5.27)

Por lo tanto podemos representar a las trayectorias descritas por el robot como segmentos de línea, mediante la ecuación de una recta conociendo un punto y la pendiente tenemos que para:

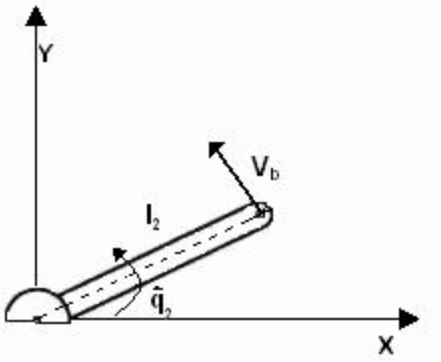
$$T_{121} \mid y = -0.9478x + 0.21403 \quad (5.28)$$

$$T_{131} \mid y = -1.2104x + 0.28602 \quad (5.29)$$

$$T_{141} \mid y = -1.4117x + 0.34122 \quad (5.30)$$

Por otra parte la velocidad operacional $V_d = 6.7\ m/s$ la conocemos por que la proporciona el fabricante en los datos técnicos, la cual toma por defecto si no se indica otra en el código del programa.

Analizando el movimiento tanto en el brazo o eslabon 2 (l_2) como en el antebrazo o eslabon 3 (l_3) tenemos que considerando el movimiento del brazo como un movimiento de rotación alrededor de un eje fijo obtenemos:



$$V_b = l_2 \dot{q}_2 \tag{5.30}$$

El movimiento del antebrazo se considera como un movimiento plano general (figura 5.5.3). La velocidad V_D del punto donde se une el efector final debe tener la dirección de la recta que describe la trayectoria. Descomponiendo el movimiento del antebrazo en una traslación y una rotación se obtiene:

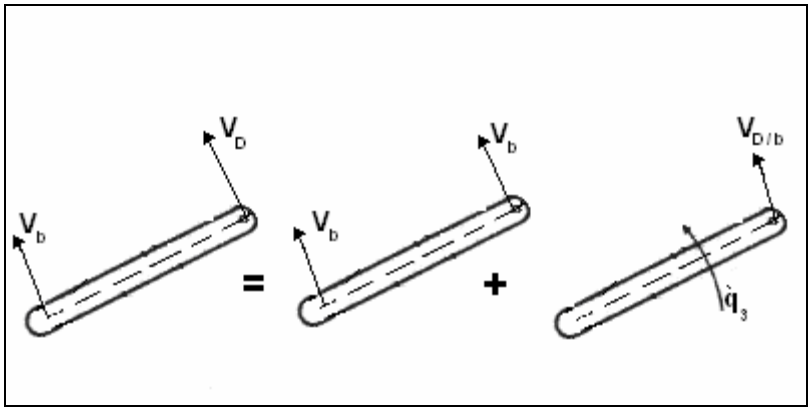


Figura 5.5.3 Movimiento plano general del antebrazo

La relación entre las velocidades $V_D, V_b, V_{D/b}$ es:

$$V_D = V_b + V_{D/b} \tag{5.31}$$

Para T_{121} se tiene que, para $y = 0$ sustituyendo en (5.28) se tiene la coordenada en X, esto es:

$x = \frac{214.0308}{0.9478} = 225.81$, este dato se resta de la abscisa de p_1, para obtener el ángulo de la dirección de V_D, esto es $\theta_D = tg^{-1}\left(\frac{45.9}{16.12}\right) = 70.65^o$, lo mismo se realiza para las otras trayectorias, con estos datos ahora se calculan las componentes en i y en j de V_D y obtenemos:

$$T_{121}|\ \dot{x} = 6.7\cos 70.65 = 2.22\ m/s \quad (5.32)$$

$$T_{121}|\ \dot{y} = 6.7\sin 70.65 = 6.32\ m/s \quad (5.33)$$

$$T_{131}|\ \dot{x} = 6.7\cos 50.44 = 4.26\ m/s \quad (5.34)$$

$$T_{131}|\ \dot{y} = 6.7\sin 50.44 = 5.16\ m/s \quad (5.35)$$

$$T_{141}|\ \dot{x} = 6.7\cos 54.69 = 3.87\ m/s \quad (5.36)$$

$$T_{141}|\ \dot{y} = 6.7\sin 54.69 = 5.47\ m/s \quad (5.37)$$

Y del MCIP conocemos a q_1 y q_2 que se obtuvieron de las ecuaciones (5.12) y (5.17), estos valores se sustituyen en la ecuación (5.28) del MCIV para obtener los valores de $\dot{q}$ ahora estos datos se utilizan en la ecuación (5.25) del MCIA para conocer $\ddot{q}$.

Con los pasos anteriores se obtiene el código en matlab para obtener las gráficas de la cinemática de los movimientos realizados en cada trayectoria de la tarea de carga y descarga (ver apéndice C para el listado del código del programa).

La figura 5.5.4 muestra la simulación en matlab de las trayectorias (segmentos de recta) que describe el robot Yamaha YK550H cuando realiza la

tarea de carga y descarga de los mecanismos, donde P1 representa el punto de carga y los puntos P2,P3, y P4 son los puntos en donde realizara la descarga.

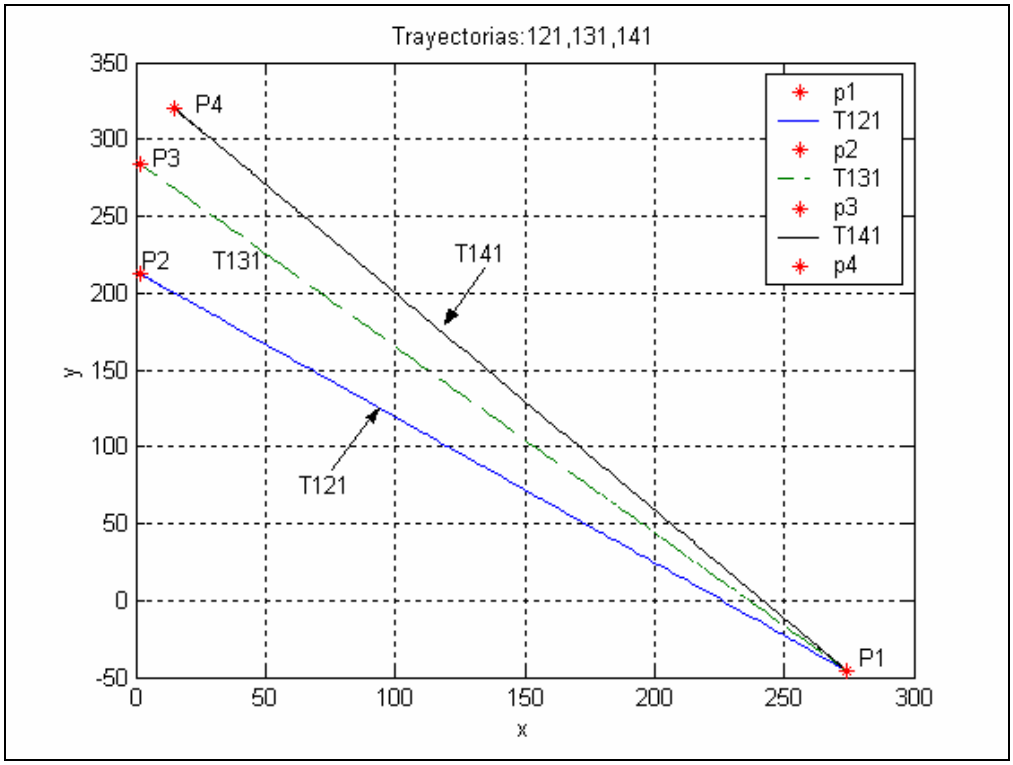


Figura 5.5.4 Simulación de las trayectorias

A continuación se muestran las gráficas de la cinemática del robot, no hay que olvidar, que el robot no se mueve a lo largo de estas curvas; se desplaza según una recta. Puesto que la derivada de una función mide la pendiente de la curva correspondiente. En las gráficas 5.5.5, 5.5.6 y 5.5.7 se describe la cinemática del recorrido que describe el robot cuando se dirige del punto de carga al punto de descarga.

Un análisis de las tres gráficas de la figura 5.5.5, muestra el movimiento articular del robot desde $t = 0$ hasta $t = 0.2$:

1.- El robot parte de la posición de carga con las articulaciones rotativas en las siguientes posiciones, para $q_1 = 2.6rad$, $q_2 = 2.3rad$ y $q_3 = cte.$ Desde $t = 0$ hasta $t = 0.2$ q_1 crece (negativamente por que va rotando en sentido contrario a las manecillas del reloj) por que sufre una aceleración $\ddot{q}_1$, el mismo comportamiento presenta q_2.

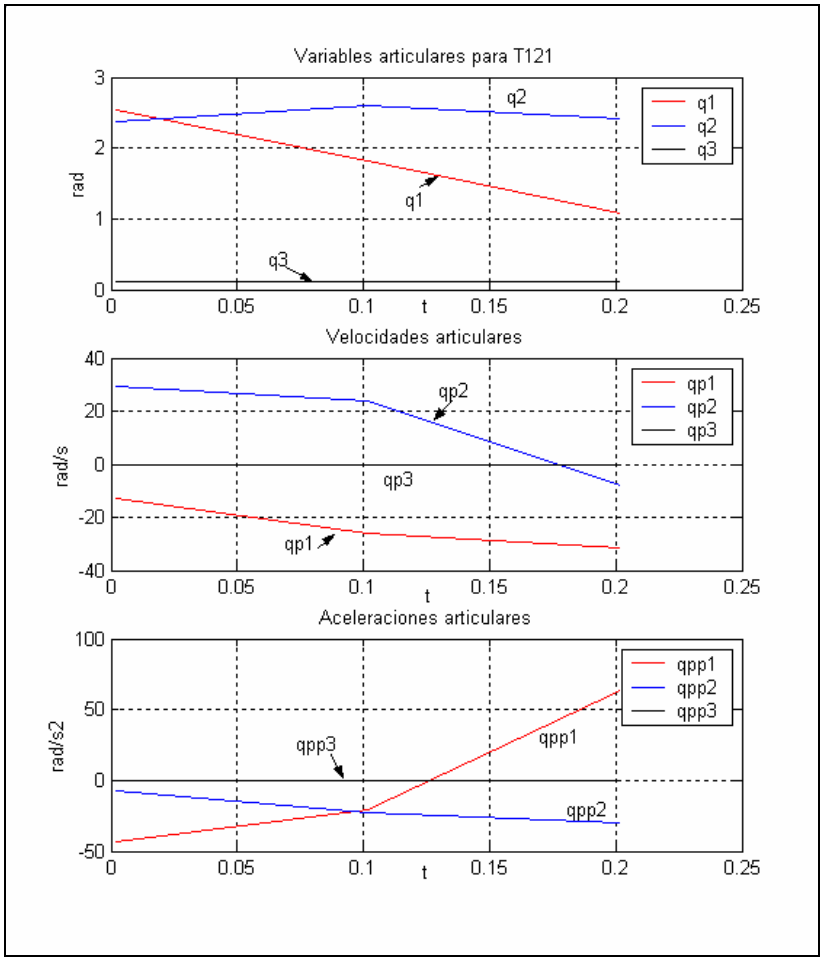


Figura 5.5.5 Gráficas de los movimientos articulares para la trayectoria 121

Analizando las gráficas de la figura 5.5.6, que muestran el movimiento articular del robot desde $t = 0$ hasta $t = 0.2$ para la trayectoria 131 se tiene que:

1.- De $t = 0$ a $t = 0.1$ q_1 se mueve en sentido antihorario y q_2 permanece moviendo en sentido horario, en cuanto a q_1 aumenta casi constantemente, por lo que sufre muy poca aceleración y q_2 decrece, es decir sufre una deceleración uniforme.

2..- De $t = 0.1$ a $t = 0.2$ q_1 crece rápidamente, es decir sufre una mayor aceleración y q_2 decrece con mayor rapidez, esto es debido a la deceleración que presenta.

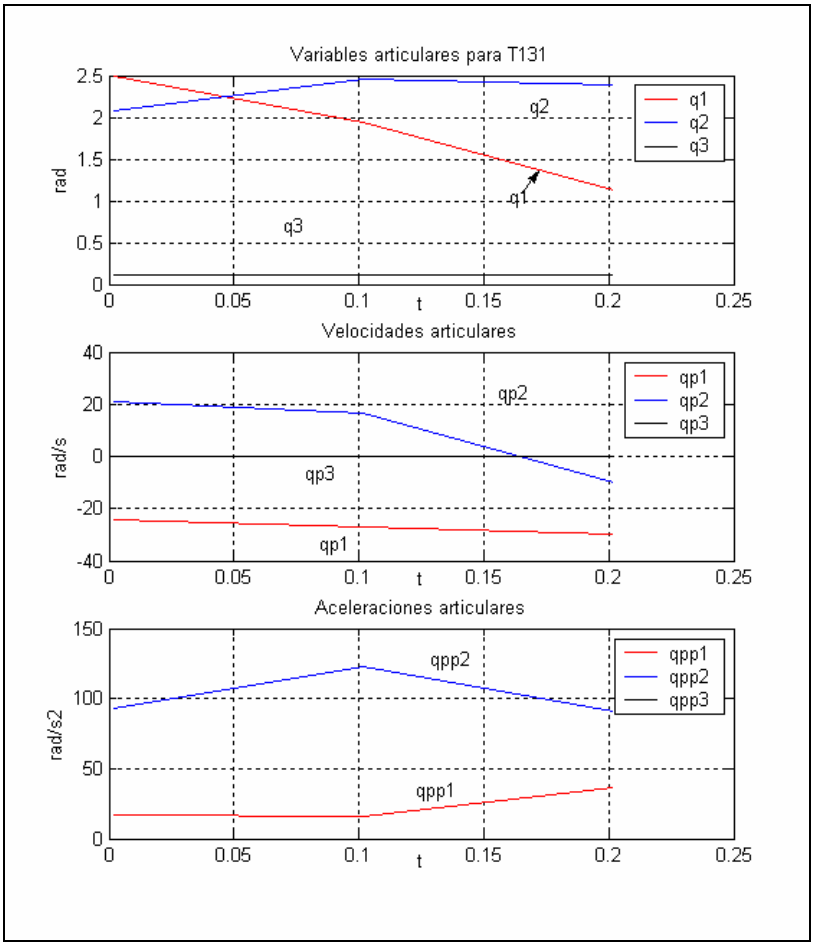


Figura 5.5.6 Gráficas de los movimientos articulares para la trayectoria 131

Analizando las gráficas de la figura 5.5.7, que muestran el movimiento articular del robot desde $t = 0$ hasta $t = 0.2$ para la trayectoria 141 se tiene que:

1.- De $t = 0$ a $t = 0.1$ q_1 se mueve en el mismo sentido a q_2, en cuanto a q_1 aumenta casi constantemente, por lo que sufre muy poca aceleración y q_2 decrece, es decir sufre una deceleración uniforme.

2..- De $t = 0.1$ a $t = 0.2$ q_1 crece rápidamente, es decir sufre una mayor aceleración y q_2 decrece con mayor rapidez, esto es debido a la deceleración que presenta.

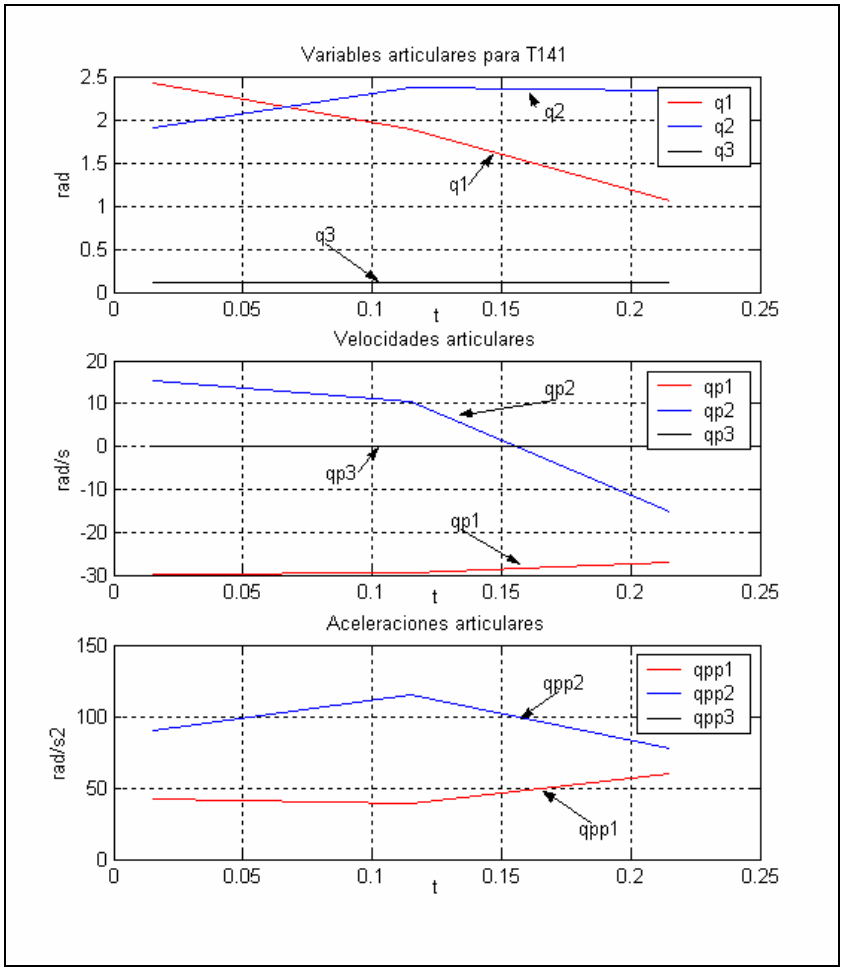


Figura 5.5.7 Gráficas de los movimientos articulares para la trayectoria 141

En lo que respecta a la ejecución del programa, cada ciclo de recolección y descarga lo ejecuta aproximadamente en diez segundos, es decir, recoge y descarga 6 piezas en 10 segundos, esto se debe a que esta funcionando solo un servo que es el que esta moviendo el peso del brazo, antebrazo y la muñeca esto en la primer posición de descarga (1.90,212.23,56.1, ver figura 5.6-b) y la segunda posición que es que corresponde a la mitad de la caja(1.90,283.73,55,76, ver figura 5.6-c) para la tercer posición que corresponde a la otra esquina de la caja(1.90,391.48,55.76, ver figura 5.6-d) en esta actúan los servos del brazo y del antebrazo por lo que la velocidad se incrementa.

5.6-a) Posición de agarre

5.6-b) Posición 1 de descarga

5.6-c) Posición 2 de descarga

5.6-d) Posición 3 de descarga

Figura 5.6 Posiciones de la tarea carga y descarga

Con estos movimientos se cumple uno de los objetivos que es el de agilizar el proceso de colocar los mecanismos en las cajas ya que no se tuvo el problema de cuello de botella en esta área, es decir, no se acumularon los mecanismos en

el conveyor y además se logro liberar de esta tarea repetitiva y monótona a la persona encargada de realizar este trabajo, para colocarla en otra área.

5.7 Conclusiones

Si bien es cierto que para la planificación de trayectorias, existen diversos modelos como los mencionados en el capítulo 2 de este trabajo, y debido a las razones ya mencionadas, pero principalmente el tiempo que en la industria es uno de los factores primordiales, por lo que para la solución del problema se utilizaron métodos prácticos, esto debido a que al robot se le puede introducir las coordenadas que se obtuvieron de la programación on-line, es decir que con el teach pendant se colocaba al robot en la posición que se consideraba la de menor recorrido y con las coordenadas mostradas se introdujeron en el listado del programa de la rutina.

El modelado matemático de la cinemática del robot fue útil para obtener el algoritmo de la programación en matlab que realizara la simulación de la cinemática del robot cuando realiza la tarea

De las simulaciones en matlab se aprecia que para la primer trayectoria recorrida por el robot el tiempo es menor que aquellas en las que el recorrido de la trayectoria era mayor, pero aun así se cumplió el objetivo de que no se acumularan las piezas en la banda transportadora.

Capítulo 6

CONCLUSIONES Y PERSPECTIVAS

6.1 Conclusiones

- La instalación de la banda que transportara las cajas que serán llenadas con los mecanismos, quedó restringida su localización al espacio disponible entre el edificio y las estaciones robóticas existentes.

- La localización del robot en el área de trabajo, se restringe a las dimensiones del edificio y a las de las instalaciones existentes. Por lo que sólo se tuvo que considerar que el robot quedara instalado en un sitio tal que los puntos de recolección y descarga estuvieran dentro del lugar geométrico que describe el robot manipulador.

- Aunque el modelo del controlador no es el indicado para utilizarse con el robot YK550H, este si cumple con los requerimientos demandados por el robot, es decir, cumple con la demanda de potencia eléctrica requerida por los servos.

- Ya montado el equipo se localizaron los puntos de recolección y descarga, mediante el teach pendant, para los puntos de descarga se utilizo el criterio de realizar el menor número de movimientos para agilizar el proceso, por lo que se escogieron tres puntos que fueron las esquinas y el punto medio de la caja.

- La rapidez con la que se lleva a cabo la tarea de recolección y descarga de las piezas (mecanismos) depende del número de

movimientos realizados, así como de las distancias entre la posición y orientación de los puntos de recolección y descarga.

- La rutina programada cumplió con las expectativas, debido a que las coordenadas planeadas lograron que se llenara casi en su totalidad las cajas.

- Debido a que el número de movimientos fueron pocos, se cumplió con el objetivo de agilizar el proceso de colocar los mecanismos en las cajas, logrando que no se hiciera un cuello de botella a la hora de recoger las piezas.

- Con la utilización de robots en la industria, se irán sustituyendo los trabajos físicos por trabajos de mayor nivel intelectual y humanos como programadores, expertos control de calidad, etc.

6.2 Perspectivas

Podemos observar que existe una necesidad en la industria por agilizar el proceso de programación y por optimizar el recorrido de las trayectorias, mejorando su tiempo de recorrido. Como vimos en este trabajo el ejemplo de una aplicación real en el proceso de ensamblado, específicamente en la última etapa del mismo que es la de empaquetar el producto final. Sin embargo, existen diversos puntos que pueden mejorarse, debido a que la rutina se programo bajo el criterio de una sola persona, esto no quiere decir que el algoritmo de la programación es incorrecto, si no que proporciona una posible solución al problema.

Glosario

Actuador: Es el motor que convierte la energía eléctrica en fuerza de movimiento [10]

Usaremos este término para indicar motores eléctricos, hidráulicos o cualquier otro tipo de mecanismo capaz de accionar o actuar los ejes de los eslabones ya sea directamente o mediante algún acoplamiento mecánico. [1]

Algoritmo: Conjunto de reglas especificas para la solución de un problema en un número finito de pasos. [10]

Ángulo: Región del plano comprendida entre dos líneas que parten de un mismo punto. [10]

Articulación: Es la unión de dos cuerpos rígidos en una estructura mecánica. [10]

Base: Es la plataforma o estructura en la cual el brazo del robot esta sujeto. [10]

Brazo robótico: Es el conjunto interconectado de eslabones y articulaciones que constituyen a un manipulador el cual soporta o mueve una muñeca.[10]

Consistente en un sistema de articulaciones mecánicas (eslabones, engranajes, transmisión por cadena o correa), actuadores (motores eléctricos) y sensores de posición usados en el sistema de control de bucle cerrado. [5]

Cadena cinemática: Es la combinación de articulaciones rotativas y/o de traslación, o ejes de movimiento. [10]

CIM: Es la integración de varias tecnologías y áreas funcionales

Cinemática: Es el estudio del movimiento sin importar las fuerzas y momentos que lo producen. [10]

Estudia el movimiento (en nuestro caso, de estructuras mecánicas multiarticuladas) pero no la manera de controlar este movimiento. [5]

Configuración: Posicionamiento de las articulaciones en diferentes maneras, formando así una posición especifica del robot. [10]

Control: Es el proceso de regulación manual o automática sobre una variable o sistema de variables conforme a lo que se desea. [10]

Controlador: Dispositivo con el que cuenta el robot, para el manejo de circuitos encargados del movimiento eléctrico. [11]

Generalmente basado en un microcomputador, que recibe las señales de los sensores de posición y envía comandos a la fuente de potencia controlada. [5]

Comando: Son aquellas palabras reservadas para la sintaxis de un lenguaje de programación. [2]

Coordenada articular: Especifican la posición de los ejes en pulsos de codificador. [10]

Desplazamiento: En el plano cartesiano son los giros, las traslaciones y las simetrías con respecto de una recta. [12]

Efector final: También conocido como órgano terminal, es la herramienta que esta montada en el robot, la cual puede ser una pinza, herramientas de pintura, soldadura y dispositivos de medida láser. [10]

Eje: Cada una de las dos o más líneas que sirven para determinar la posición de los puntos de un espacio. [9]

Eje rotacional: También conocido como eje giratorio, es cuando en una cadena conectada con dos cuerpos rígidos en un mecanismo permite rotar alrededor de un eje. [10]

Encoder: O Codificador óptico es un transductor utilizado para convertir posiciones lineales o rotativas en datos digitales. [10]

EPROM: Son chips de memoria que se programan después de su fabricación. Son un buen método para que los fabricantes inserten códigos que cambian constantemente.

Eslabón: Pieza o cuerpo rígido en una cadena cinemática. [10]

Espacio de trabajo: Es la zona donde el robot puede posicionarse y está limitada por las dimensiones físicas del manipulador. [10]

Grado de libertad: Son los posibles movimientos básicos independientes, ya sean giratorios o de desplazamiento, que el robot puede realizar. [12]

Jacobiano: Es la matriz que describe la relación entre las velocidades de las articulaciones y el efector final. [9]

Límite operacional: Son las fronteras alrededor del espacio de trabajo del robot. [10]

Longitud: Dimensión que expresa el valor de una distancia. [12]

Manipulabilidad: Es la posibilidad de moverse libremente en todas las direcciones del espacio de trabajo. [10]

Manipulador: Es un mecanismo que usualmente consiste en una serie de eslabones articulados o desplazamiento relativo a otro eslabón, para agarrar o mover objetos, por lo regular tiene varios grados de libertad. [10]

Manufactura: Producción fabril con empleo de maquinaria movida por energía mecánica y con una división compleja de trabajo. [10]

Marco de referencia: Corresponde al marco ortonormal base, basado en los vectores unitarios i, j, k sobre el cual se define el vector de posición y orientación del efector final.

Parámetro: Es una variable que dentro de un programa toma un valor constante durante la ejecución del mismo. [2]

PLC: Es un dispositivo con memoria para almacenar instrucciones para el control de una gran variedad de tipos de máquinas, considerando a estas ultimas como unidades de entrada y salida [18].

Planificación de movimientos: Es la manera de encontrar una ruta libre de colisiones de una posición inicial y una posición final dentro de un espacio que puede o no tener obstáculos. [9]

Pitch: Es también conocido como inclinación perpendicular de la muñeca. [10]

Precisión: La capacidad de que el robot posicione el efector final a un punto programado. Es la característica que diferencia a la posición que va automáticamente el efector final del robot entre la posición grabada originalmente. [10]

Programabilidad: Es la característica que posee el controlador de las diversas formas de programar el robot. [10]

RAM: Memoria que puede ser escrita y leída por el microprocesador u otros dispositivos de hardware. [2]

Rango operacional: véase límite operacional.

Repetibilidad: Es la variación en la posición de la herramienta del robot para ciclos repetidos bajo las mismas condiciones. Esta se obtiene de la desviación de la posición y la orientación alcanzada al final de varios ciclos similares. [10]

Robot: Un dispositivo mecánico que puede ser programado para que realice una variedad de tareas de manipulación y locomoción baja control automático. [10]

Robot antropomórfico: También conocido como brazo articulado y es un robot con todas las articulaciones rotativas y movimientos similares a los del brazo de una persona. [10]

Robótica: Ciencia de diseño, construcción y aplicación de robots. [10]

Simulación digital: Representación del comportamiento de algo, expresado en magnitudes numéricas mediante el computador. [9]

Singularidad: Puntos y zonas que son inadmisibles dentro y fuera del espacio de trabajo respectivamente para el robot. [10]

Sintaxis: Conjunto de reglas que definen las secuencias correctas de los elementos de un lenguaje de programación. [2]

Tareas del robot: Es una rutina que ha sido programada con el fin de que el robot la realice cuando sea ejecutada. [10]

Transductor: Es un dispositivo que convierte una forma, por ejemplo convertir el movimiento a lo largo de una cierta distancia a un número de pulsos eléctricos que puedan ser contados. [10]

Trayectoria: Está formada de puntos que realiza el robot o pasa a través de una operación dependiendo de la programación. [10]

Es una relación temporal de posición velocidad y aceleración para cada uno de los grados de libertad del robot. [5]

Teach pendant: También conocido como panel portátil, es un dispositivo de programación portable conectado el controlador del robot y está constituido por un número de botones. Esté es usado para programación en línea. [10]

Acrónimos y abreviaturas

ASYNC asynchronous mode of communications (modo de comunicación asíncrona)

BASIC Beginners All-purpose Symbolic Instruction Code (Código de Instrucciones Simbólicas de Uso General para Principiantes).

BIT Binary digit (Digito binario)

CPU Unidad central de proceso

EEPROM Electrical ereasable programmable read only memory (memoria de solo lectura programable y borrable eléctricamente)

ENIAC Electronic Numerical Integrator And Computer (Integrador y Computador Electrónico Numérico)

LED Light-Emitting Diode(Diodo emisor de luz)

MCDA Modelado cinemático directo de aceleración

MCIA Modelado cinemático inverso de aceleración

MCIV Modelado cinemático inverso de velocidad

MIT Massachusetts Institute of Technology (Instituto Tecnológico de Massachusetts)

PID Proporcional-integral-derivativo

PLC Programmable Logic Controller (Controlador lógico programable)

PUMA Programmable Universal Machine for Assambling (Brazo Manipulador Universal Programable para Montaje)

PWM Pulse Width Modulation (Modulación por ancho de pulso)

RAM Random Access Memory (Memoria de Acceso Aleatorio)

RIA Robot Institute of America (instituto de robótica de América)

RLL Relay Ladder Logic (lenguaje de lógica de escalera)

ROM Read only memory (memoria de solo lectura)

SCARA Selective Compliance Assambly Robot Arm (robot para montaje con acomodación selectiva)

UC Unidad central

Apéndice A Interfaz para la programación

VIP Windows
PC Offline Programming & Communication

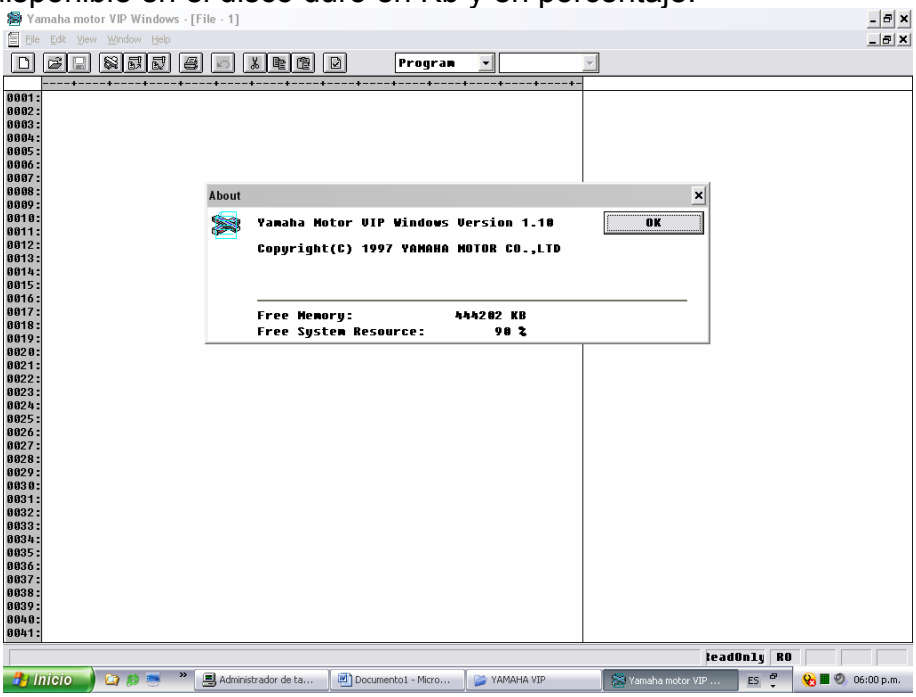
La interfaz para la programación del controlador llamada VIP Windows es un software de aplicación el cual sirve para la creación y edición de programas para la operación de robots, que utiliza un lenguaje de alto nivel y por lo mismo no es tan sencillo programar ya que actualmente hay interfases más amigables porque su ambiente de trabajo es grafico lo que los hace más intuitivos y sólo requieren de la introducción de datos tales como las coordenadas y los tiempos.

Los requerimientos mínimos para la instalación del software son una computadora con procesador 486, memoria RAM de 64MB y Windows 3.11

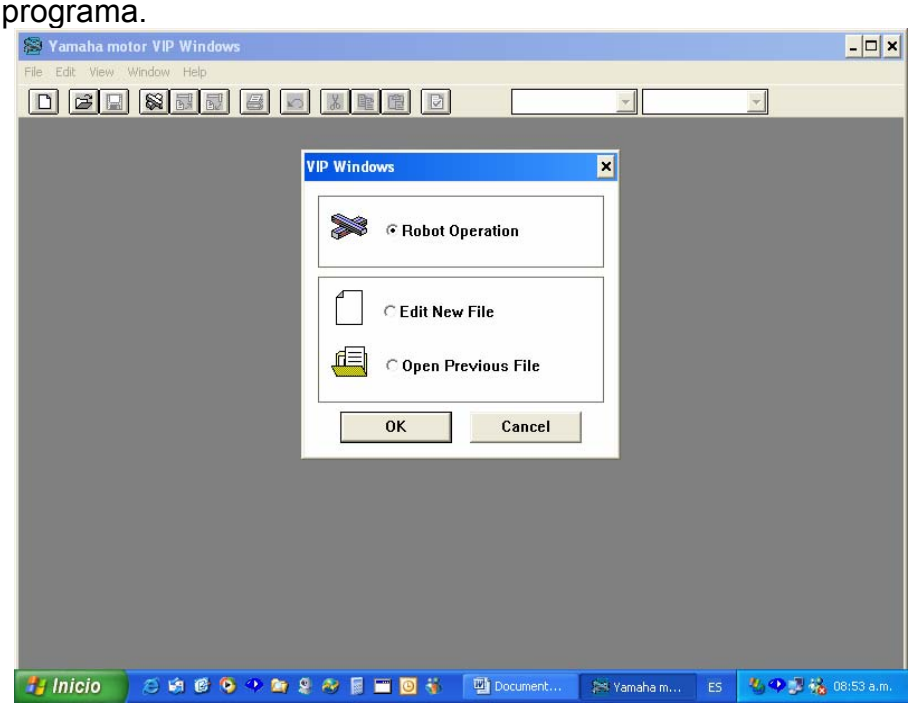
Esta interfaz fue creada para el control del robot manipulador y trabaja de acuerdo a las coordenadas que el usuario coloque. Una vez terminado el programa se conecta la computadora al controlador mediante un cable DB25 para la comunicación entre la computadora y el controlador que recibe el programa.

A continuación describiré paso a paso como es que cualquier persona fácilmente puede tener acceso a esta herramienta.

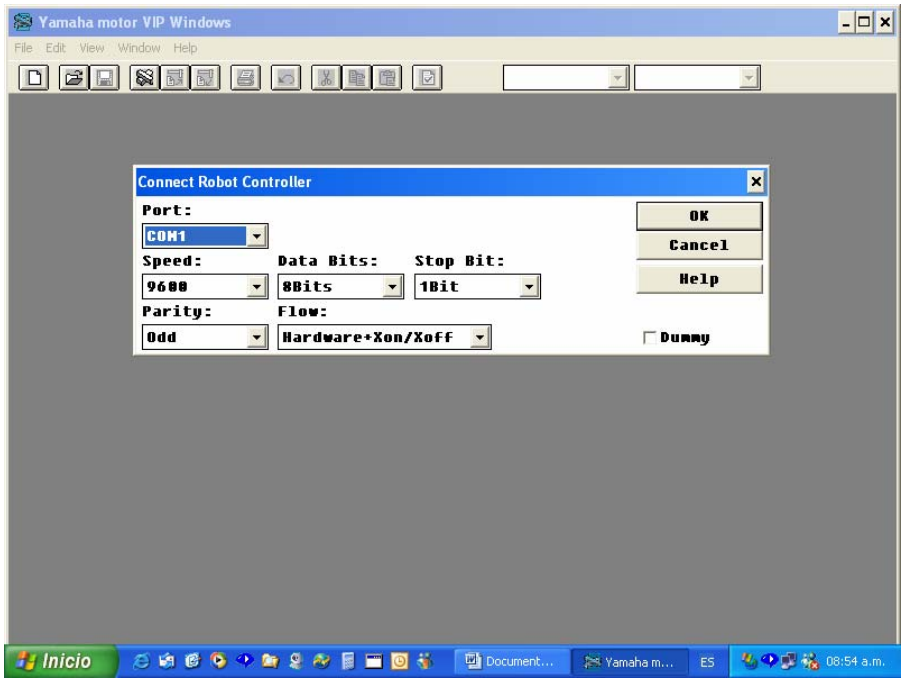
En la siguiente figura aparece la pantalla de inicio de la interfaz, en la cual aparece información acerca de la versión de la interfaz, su copyright, así como el espacio disponible en el disco duro en Kb y en porcentaje.



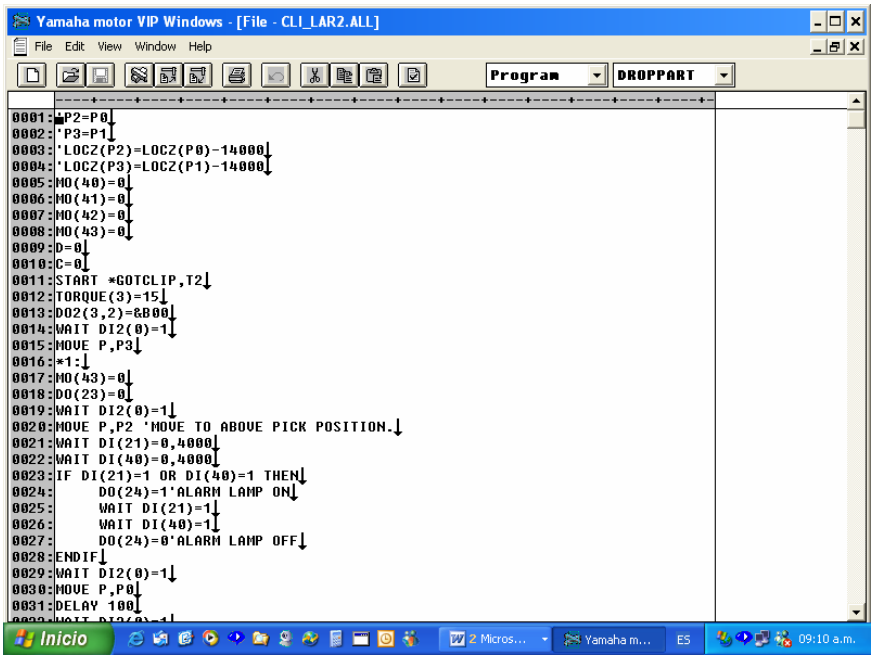
En esta figura se presenta el menú, en el que podemos decidir si vamos a editar, corregir un programa o si vamos a comunicarnos con el controlador, para enviar el programa.



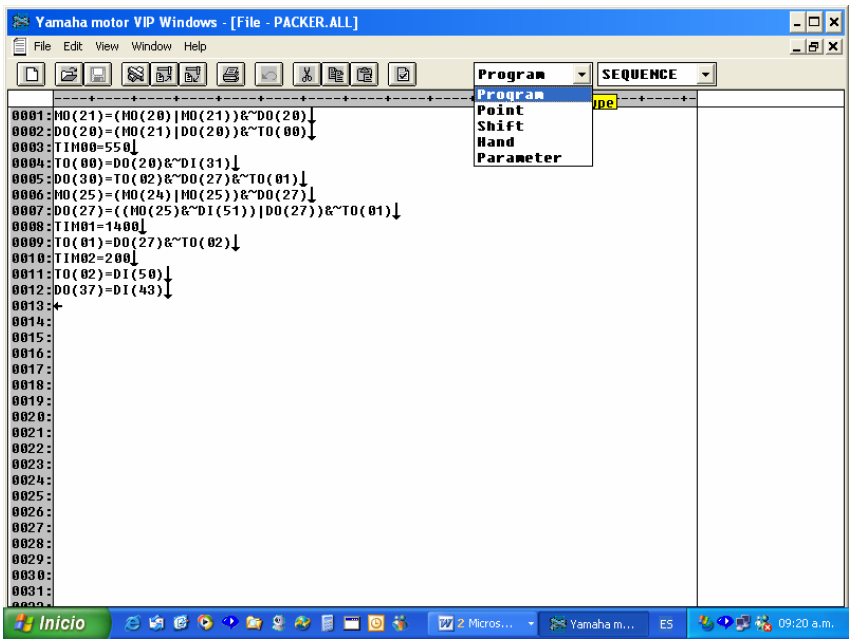
La siguiente pantalla muestra la ventana de comunicación con el controlador.



En la figura siguiente se muestra la opción de edición y corrección de programas.



Menú que nos permite seleccionar que tipo de datos queremos ver en el listado del programa, ya sean datos generales, o los puntos de las coordenadas o saltos.



Apéndice B Fichas técnicas

B.1 Datos técnicos del optoacoplador

NEC PS2801-1,PS2801-4

ABSOLUTE MAXIMUM RATINGS (T_A = 25 ° C, unless otherwise specified)

Parameter		Symbol	Ratings		Unit
			PS2801-1	PS2801-4	
Diode	Forward Current (DC)	I_F	50		mA
	Reverse Voltage	V_R	6		V
	Power Dissipation Derating	ΔP_D/°C	0.6	0.8	mW/°C
	Power Dissipation	P_D	60	80	mW/ch
	Peak Forward Current *1	I_{FP}	1		A
Transistor	Collector to Emitter Voltage	V_{CEO}	80		V
	Emitter to Collector Voltage	V_{ECO}	6		V
	Collector Current	I_C	50		mA/ch
	Power Dissipation Derating	ΔP_C/°C	1.2		mW/°C
	Power Dissipation	P_C	120		mW/ch
Isolation Voltage *2		BV	2 500		Vr.m.s.
Operating Ambient Temperature		T_A	−55 to +100		°C
Storage Temperature		T_{stg}	−55 to +150		°C

*1 PW = 100 μs, Duty Cycle = 1 %

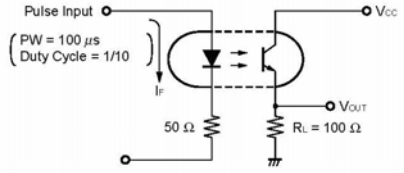
*2 AC voltage for 1 minute at T_A = 25 ° C, RH = 60 % between input and output

NEC PS2801-1,PS2801-4

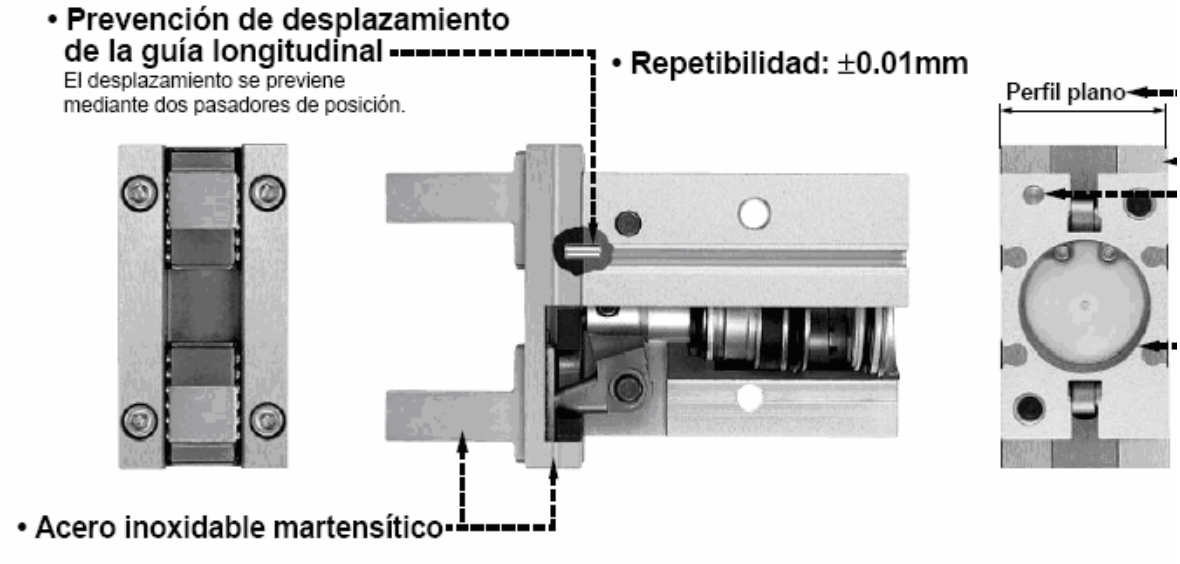
ELECTRICAL CHARACTERISTICS (T_A = 25 ° C)

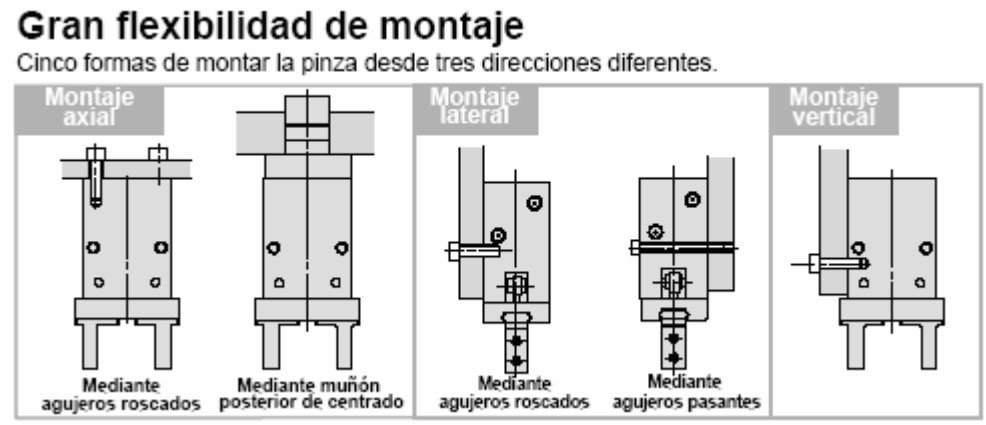
Parameter		Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Diode	Forward Voltage	V_F	I_F = 5 mA		1.1	1.4	V
	Reverse Current	I_R	V_R = 5 V			5	μA
	Terminal Capacitance	C_t	V = 0 V, f = 1.0 MHz		30		pF
Transistor	Collector to Emitter Dark Current	I_{CEO}	V_{CE} = 80 V, I_F = 0 mA			100	nA
Coupled	Current Transfer Ratio (I_C/I_F)	CTR	I_F = 5 mA, V_{CE} = 5 V	80		600	%
	Collector Saturation Voltage	$V_{CE(sat)}$	I_F = 10 mA, I_C = 2 mA			0.3	V
	Isolation Resistance	R_{I-O}	V_{I-O} = 1.0 kV_{DC}	10^{11}			Ω
	Isolation Capacitance	C_{I-O}	V = 0 V, f = 1.0 MHz		0.4		pF
	Rise Time *1	t_r	V_{CC} = 5 V, I_C = 2 mA, R_L = 100 Ω		3		μs
	Fall Time *1	t_f			5		

*1 Test circuit for switching time



B.2 Datos técnicos del efector final





Apéndice C listado del programa en matlab de la cinemática

```
clear all
close all
clc

l1=0.2;
l2=0.3;
l3=0.25;
z=0.0859;
x=0.0019:0.1:0.27422;
t=x;
px1=0.27422;
py1=-0.0459;
px=0.0019;
py=0.21223;
y=-0.9478*x+0.2140308;
xp_aux=4.86;
yp_aux=4.60;
zp_aux=0;
maux2=ones(size(y));
xp=xp_aux*maux2;
yp=yp_aux*maux2;
zp=zp_aux*maux2;
q2=acos((x.^2+y.^2-(l2.^2+l3.^2))/(2*l2*l3));
beta=atan2(y,x);
fee=acos((x.^2+y.^2+l2.^2-l3.^2)/(2*l2*sqrt(x.^2+y.^2)));
if q2<0
    q1=beta-fee;
else q1=beta+fee;
end
q3_aux=l1-z;
maux=ones(size(y));
q3=q3_aux*maux;
DJ=-l2*l3*sin(q2);
Jinv11=-l3*cos(q1+q2)./DJ;
Jinv12=(-l3*cos(q1+q2)-l2*cos(q1))./DJ;
Jinv13=0;
Jinv21=l3*sin(q1+q2)./DJ;
Jinv22=(l3*sin(q1+q2)+l2*sin(q1))./DJ;
Jinv23=0;
Jinv31=0;
Jinv32=0;
Jinv33=(-l3*l2*cos(q1+q2).*sin(q1)+(l3*l2*cos(q1).*sin(q1+q2)))./DJ;
qp1=Jinv11.*xp+Jinv12.*yp+Jinv13.*zp;
```

```
qp2=Jinv21.*xp+Jinv22.*yp+Jinv23.*zp;
qp3=Jinv31.*xp+Jinv32.*yp+Jinv33.*zp;
a=(-2*l3*cos(q1+q2)-l2*cos(q1)).*qp1-2*l3*cos(q1+q2).*qp2;
b=-(2*l3*sin(q1+q2)-l2*sin(q1)).*qp1-2*l3*cos(q1+q2).*qp2;
c=-2*l3*cos(q1+q2)-l2*cos(q1);
d=-2*l3*sin(q1+q2)-l2*sin(q1);
e=-2*l3*cos(q1+q2);
f=-2*l3*sin(q1+q2);
JinJder11=Jinv11.*c+Jinv12.*d;
JinJder12=Jinv11.*e+Jinv12.*f;
JinJder21=Jinv21.*c+Jinv22.*d;
JinJder22=Jinv21.*e+Jinv22.*f;
qpp1=JinJder11.*qp1+JinJder12.*qp2;
qpp2=JinJder21.*qp1+JinJder22.*qp2;
qpp3=0;
figure(1)
plot(x,y,'b',px1,py1,'*r',px,py,'*r');
title('Trayectoria:121');
xlabel('x');
ylabel('y');
grid
legend('T121','p2')
figure(2)
subplot(3,1,1);
plot(t,q1,'r',t,q2,'b',t,q3,'k');
title('Variables articulares para T121');
xlabel('t');
ylabel('rad');
grid
legend('q1','q2','q3')
subplot(3,1,2);
plot(t,qp1,'r',t,qp2,'b',t,qp3,'k');
title('Velocidades articulares');
xlabel('t');
ylabel('rad/s');
grid
legend('qp1','qp2','qp3')
subplot(3,1,3);
plot(t,qpp1,'r',t,qpp2,'b',t,qp3,'k');
title('Aceleraciones articulares');
xlabel('t');
ylabel('rad/s2');
grid
legend('qpp1','qpp2','qpp3')
```

Bibliografía

[1] Barrientos A., Peñin L. F., Balaguer C.,Aracil R., “Fundamentos de robótica”, McGrawHill, 1997

[2] Editorial patria, “Guía escolar VOX informática”, 1993

[3] Goldstein, Larry, Goldstein Martín.” Programación y aplicaciones en BASIC”, Prentice-Hall, 1986

[4] Groover P., M., Weiss M., Nayel, R., Odrey, N., “Robótica industrial”, McGrawHill, 1990

[5] Iñigo M. R., Vidal I. E., “Robots industriales manipuladores”, Alfa omega, 2004.

[6] Mark Spong F.L. Lewis C.T. Abdallah (1993). Robot Control. Dynamic, Motion Planning and Analysis. New York, IEEE

[7] Mompin Poblet, J., “Aplicaciones de la electrónica Vol. 36”, Marcombo Orbis, 1986

[8] Mompin Poblet, J, “Electrónica aplicada Vol.1”, Marcombo Orbis,1986

[9] Murray, R., Li, Z., Sastry, S., “A Mathematical Introduction to Robotic Manipulation”, CRC press LLC. 1994

[10] Nof, S. “Handbook of industrial Robotics”, John Wiley & Sons, INC 2da Edition, 1999

[11] Ollero Baturone A., “Robótica manipuladores y robots móviles”, Alfa omega Marcombo, 2001
[12] R.P., Paul “Robots Manipulators: Mathematics, programming”, MIT press 1981

[13] Yamaha Motor Company, “Programming Manual”, May 1997

[14] Yamaha Motor Company, “The Yamaha Robot Controller QRCH series”, May 1997

Referencias electrónicas

[15] Encarta ® 2005. © 1993-2004 Microsoft Corporation.

[16] García Martínez, S., “Introducción a los PLC”, disponible en Internet: lie.fie.umich.mx/maquinas3/intro.doc - 13 Jun 2005

[17] Gijón Yescas, N., “Estación robótica para ensamble”, disponible en Internet: nomelygi@hotmail.com

[18] Rodríguez H. “Lenguajes de lógica de estado para PLC”, disponible en Internet ,ns.fim.utp.ac.pa/Revista/vol1/plc.html

[19] http://www.ece.ubc.ca/danielac/constantinescumasc.pdf, “the university of british Columbia”

[20]http://www.isa.uma.es/personal/antonio/Robotica/Tema1%20-%20Introduccion.pdf.

[21]www.elprisma.com/apuntes/ingenieria_electrica_y_electronica/introduccionrobotica/default7.asp “cinemática ”