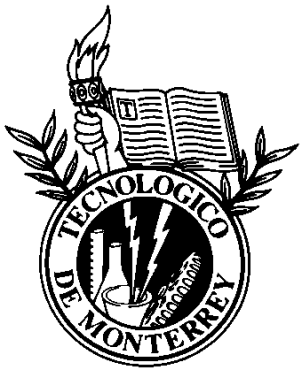
INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

CAMPUS MONTERREY

DIVISIÓN DE INGENIERÍA Y ARQUITECTURA
PROGRAMA DE GRADUADOS EN INGENIERÍA



INTERFAZ DE USUARIO PARA EL ROBOT PUMA 560 DE UNIMATION

TESIS

PRESENTADA COMO REQUISITO PARCIAL PARA OBTENER EL GRADO ACADÉMICO DE:

MAESTRO EN CIENCIAS
CON LA ESPECIALIDAD EN SISTEMAS DE MANUFACTURA

POR:

JULIO ARTURO OLIVARES FONG

MONTERREY, N. L. **MAYO 2002**

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

CAMPUS MONTERREY
DIVISIÓN DE INGENIERÍA Y ARQUITECTURA
PROGRAMA DE GRADUADOS EN INGENIERÍA

Los miembros del comité de tesis recomendamos que la presente tesis presentado por el Ing. Julio Arturo Olivares Fong sea aceptado como requisito parcial para obtener el grado académico de Maestro en Ciencias con la especialidad en

SISTEMAS DE MANUFACTURA

Comité de Tesis:

_____

Heriberto García Reyes, M. C.
ASESOR

_____ _____

Leopoldo Eduardo Cárdenas Barrón, M.C.
SINODAL

Rodolfo Villarreal Rodríguez, M.C.
SINODAL

Aprobado:

_____

Federico Viramontes Brown, Ph. D.
Director del Programa de Graduados en Ingeniería

Mayo 2002

DEDICATORIA

Dedico este trabajo de tesis a todas aquellas personas con las que he convivido en este tiempo que duraron mis estudios de maestría, muy especialmente a mis padres:

María de la Luz Fong de Olivares

Y

Arturo Olivares Pérez

de quienes he recibido, durante toda mi vida, su amor, apoyo y comprensión.

Y a la memoria de mi abuelo:

David Olivares García (✝ Q.E.P.D.)

AGRADECIMIENTOS

Quiero agradecer al **Ing. Heriberto García Reyes** por haber aceptado ser mi asesor de tesis, brindarme su apoyo y múltiples consejos durante todo el desarrollo del proyecto.

Al **Ing. Rodolfo Villarreal Rodríguez** y al **Ing. Leopoldo Eduardo Cárdenas Barrón** por haber aceptado ser parte del comité de tesis como sinodales.

Al **Ing. Eduardo García Dunna** por haberme dado la oportunidad de iniciar mis estudios de maestría, en el ITESM, dentro del programa de asistentes de docencia. Agradezco también al **Ing. Juan José Hinojosa Cavazos** por permitirme llevar a cabo la realización de este proyecto, como asistente, para terminar la maestría.

A **Gerardo Vallejo** y a **Marx Arellano** por su invaluable ayuda en los aspectos técnicos para la realización del programa, ya que sin ellos me hubiera tomado mucho más tiempo.

A **Carlos** por su gran ayuda en la elaboración de este documento, además de sus innumerables consejos y observaciones que me orientaron a "escribir" la tesis.

A mis hermanos: **David** y **Lucy, a todos mis amigos y compañeros** por haber confiado en mí y darme ánimos para salir adelante.

A los **Instructores** y **Alumnos del Laboratorio de Sistemas Integrados de Manufactura** que de alguna forma, tal vez sin tener la intención, colaboraron con sus comentarios e ideas para el desarrollo de este proyecto.

Un agradecimiento muy especial a **Claudia**, mi novia, por estar conmigo en todo momento y bajo cualquier condición, siendo mi mayor inspiración para seguir luchando.

Y doy muchas gracias a **Dios**, por todas las bendiciones que día a día me otorga.

ÍNDICE GENERAL

Capítulo 5 – Resultados, Conclusiones e Investigaciones Futuras

ÍNDICE DE FIGURAS

Capítulo 4 – Interfaz de Usuario para el Robot PUMA 560 de Unimation

Capítulo 5 – Resultados, Conclusiones e Investigaciones Futuras

ÍNDICE DE TABLAS

Capítulo 2 – Conceptos Básicos

Capítulo 4 – Interfaz de Usuario para el Robot PUMA 560 de Unimation

Capítulo 5 – Resultados, Conclusiones e Investigaciones Futuras

Capítulo 1
Introducción

1.1 Presentación

La interfaz de usuario para el robot PUMA 560 de Unimation es un programa que tiene como principal objetivo facilitar el aprendizaje y el manejo de este robot, para los alumnos de la carrera de Ingeniería Industrial y de Sistemas del ITESM, que cursen el Laboratorio de Sistemas Integrados de Manufactura. Actualmente este laboratorio tiene como apoyo fundamental una celda flexible de manufactura de AMATROL, en la cual se tienen los siguientes componentes:

1. Computadora Central
2. Almacén Automatizado AMATROL 862 – AS/RS
3. Banda Transportadora
4. Torno KRYLE KL200
5. Fresadora KRYLE VMC535
6. Robot PUMA 560 de Unimation
7. Robot Júpiter XL de Amatrol
8. Robot Mitsubishi (RV-M1)
9. Sistema de inspección por visión Amatrol 866 – VS

En la figura 1.1 se puede observar la ubicación de cada uno de estos equipos en la celda. El robot PUMA 560 se encuentra situado entre la banda transportadora, el torno KRYLE KL200 y la fresadora KRYLE VMC535.

La función principal del robot PUMA en la celda flexible de manufactura es alimentar de materia prima a los dos centros de maquinado (torno y fresadora), para que sea trabajada según se requiera. El material es recibido por medio de los dos paros de transferencia que se

encuentran frente al brazo del robot, uno destinado para el torno, denominado como estación número dos, y otro para la fresadora, identificado como estación número tres. El robot también puede realizar algunas operaciones de ensamble de componentes, por medio de estas mismas dos estaciones, en las que se pueden sujetar los contenedores ("pallets") que transportan la materia prima o los componentes a ensamblar.

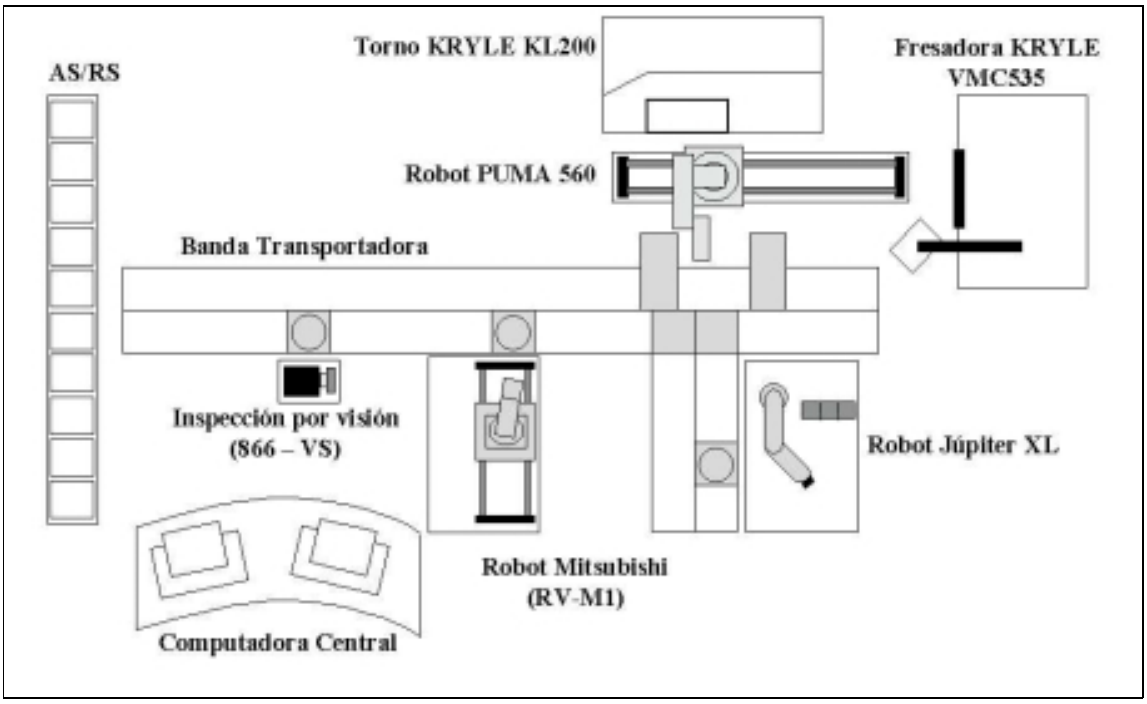


Figura 1.1 Celda Flexible de Manufactura del ITESM

Con el fin de dar un panorama general, en este primer capítulo se presenta una breve semblanza histórica de los dos temas principales que están involucrados en esta tesis: la manufactura y la robótica, desde sus inicios, cómo han ido evolucionando, para posteriormente mostrar como se relacionan en la actualidad y la importancia que tienen ambos en los nuevos medios de producción.

En las siguientes secciones del capítulo se establece la definición del problema, el objetivo de la tesis, la justificación, hipótesis, el alcance, la metodología de la investigación y los recursos disponibles, en los cuales se basa el desarrollo de este proyecto.

1.2 Antecedentes

1.2.1 Historia de la Manufactura

El término “manufactura” proviene del latín, significa “hecho a mano”, y aunque éste surge en 1683, el concepto de la manufactura tiene sus orígenes desde la edad de piedra, cuando el hombre empezó a fabricar utensilios que le ayudaban en sus actividades cotidianas. El primer gran paso fue cuando creó y aprendió a utilizar herramientas, ya que se convirtieron en extensiones de sus manos, y le facilitaron el manejo y la manipulación de los materiales. Es precisamente la habilidad para el uso de herramientas de mano uno de los factores que distingue al hombre de otros animales, éstas permiten hacer cosas simples en lugar de esperar a que la naturaleza lo provea.

El siguiente gran paso no fue hace mucho tiempo: La Revolución Industrial trajo consigo otro salto en los estándares de vida, y el desarrollo de máquinas herramientas. Las máquinas herramientas son uno de los productos principales de la Revolución Industrial, agregaron poder y precisión a las tareas humanas, con ellas era posible producir bienes más rápido y con menor variabilidad. La productividad se incrementó drásticamente, y los bienes industriales reemplazaron a los productos hechos a mano, ya que eran mas baratos y de mayor calidad.

Los europeos, con sus máquinas recientemente inventadas expandieron su influencia en todo el mundo. En los tiempos de la Revolución Industrial, los productos se fabricaban a petición del cliente con máquinas herramientas que se operaban manualmente. Mas adelante se dio otro de los grandes pasos en la manufactura, con el desarrollo de partes intercambiables y el concepto de líneas de producción / ensamble, la producción en masa se volvió una realidad en el siglo XX. Nunca antes en la historia de la humanidad, el hombre ha disfrutado tantas mejoras en lo estándares de vida, como en este siglo. La producción en masa y la administración científica de la manufactura ayudaron a producir más bienes, mejores y a un menor costo. Máquinas y sistemas automatizados (controlados mecánicamente) producían el equivalente a centenas de trabajadores. Los productos idénticos de la producción en masa eran

muchos y baratos. Sin embargo, la variedad estaba limitada por los altos costos de los cambios en los sistemas de manufactura.

Poco después de la Segunda Guerra Mundial, con la creciente demanda de partes más complejas, se inventaron las máquinas de control numérico. Éstas sustituyeron la necesidad de contar con operadores hábiles en el manejo de las máquinas. Desde el año de 1950 a la fecha han ocurrido más desarrollos científicos y tecnológicos que en toda la historia del hombre. Uno de los más importantes es la computadora digital. En la manufactura de productos discretos, las computadoras son esenciales para el desarrollo de los controles numéricos, robótica, diseño auxiliado por computadora (CAD), manufactura auxiliada por computadora (CAM), y los sistemas flexibles de manufactura (FMS). Estas tecnologías nuevas, basadas en computadoras, nos permiten producir productos en lotes pequeños a bajos costos. Muchas funciones del hombre en toma de decisiones han sido reemplazadas o son auxiliadas con computadoras [1].

Desde el punto de vista de la demanda de los productos y los tamaños de lote, también se ve una tendencia. Antes de la Revolución Industrial, la demanda era baja y todos los productos se hacían al gusto del cliente. Había pocos productos disponibles y los precios eran altos. En los talleres se producían lotes pequeños. La mayoría de los productos eran fabricados por artesanos. La calidad dependía grandemente de su habilidad, a este tipo de producción se le conoce como producción manual [2].

Con el cambio que se dio en 1950, el mundo cambió de una sociedad con pocos productos industriales para poca gente a una gran abundancia de productos para todos. La demanda por estos productos creció, por lo que, para responder a esta necesidad, se desarrollaron las técnicas de producción en masa [1]. En este tipo de ambiente, la mayoría de los procesos de manufactura se realizan con máquinas especializadas, se reducen los costos, haciendo que los productos se vuelvan accesibles para un mayor número de personas. Sin embargo, ya se cada máquina está diseñada para un cierto tipo de tarea, cada que se introducía un modelo nuevo, se tenía que parar la línea completa y cambiar de herramientas, esta operación para cada cambio

de modelo puede ser muy costoso. A este tipo de producción se le conoce como automatización dura [2].

El deseo de tener más variedad de productos hizo que la producción cambiara de altos volúmenes a medianos y pequeños nuevamente. En la década de 1980's, la competencia internacional mandaba que los productos fueran hechos rápidamente y que los inventarios se conservaran al mínimo. En la industria de manufactura moderna, la supervivencia se predica en la automatización, mientras mantiene su flexibilidad. La automatización debe proveer de calidad y bajos costos, y la flexibilidad necesaria para adaptarse a los cambios de productos y a sus demandas [1].

La inflexibilidad y relativamente, altos costos de la automatización dura han guiado a la manufactura hacia otro enfoque. Recientemente, se han introducido robots manipuladores en las industrias para realizar ciertas tareas productivas, como son el manejo de materiales, soldadura de puntos, pintura en aerosol y ensamble. Debido a que estos robots son controlados por medio de computadoras o micro procesadores, se pueden programar fácilmente para realizar diferentes tareas. Debido a que no es necesario reemplazar estas máquinas durante el cambio de un modelo de un producto, se le conoce como automatización flexible [2].

1.2.2 Historia de la Robótica

El concepto de robótica, aunque no había sido llamado así hasta hace relativamente poco, ha capturado la imaginación del hombre por siglos. Uno de los primeros animales automáticos, un pájaro de madera capaz de volar, fue construido por un amigo de Platón, Arquitas de Tarento, entre los años de 400 y 350 antes de Cristo. En el segundo siglo, antes de Cristo, Hero de Alejandría describe en su libro "De Automatis", un teatro mecánico con figuras robóticas que bailaban y marchaban en las ceremonias del templo. Fue hasta 1921 de nuestra era, cuando el escritor checoslovaco Karel Capek, escribe su drama satírico R.U.R. (Rossum's Universal Robots), e introduce la palabra robot del checo robota, que significa siervo o que está al servicio. Las máquinas en esta obra, semejaban gente, pero trabajaban lo doble [1].

Para colocar a las máquinas actuales y el interés en ellas desde una perspectiva histórica, se presenta un breve resumen del desarrollo de los robots. En la lista de fechas, se resalta el crecimiento de las máquinas autómatas que guiaron al desarrollo de los robots industriales disponibles en la actualidad.

1801 Joseph Jacqard inventa una máquina textil operada por tarjetas perforadas.

1892 En los Estados Unidos, Seward Babbitt diseña un grúa motorizada con un efector para retirar lingotes de un horno.

1921 La primer referencia a la palabra *robot* aparece en la apertura de una obra en Londres.

1938 Los norteamericanos Willard Pollard y Harold Roselund diseñan un mecanismo programable de pintura en aerosol para la compañía DeVilbiss

1939 Las obras de ciencia ficción de Isaac Asimov, introducen robots diseñados a ayudar a la humanidad y trabajar de forma segura.

1946 Geroge Devol patenta un dispositivo de propósito general para el control de máquinas. Este dispositivo registraba procesos magnéticamente. Surge la computadora

1948 El profesor Norbert Wiener, del MIT, publica *Cybernetics*, un libro en el que describe las comunicaciones y el control en sistemas electrónicos, mecánicos y biológicos.

1951 Raymond Goertz diseña un brazo articulado de control remoto para la Comisión de Energía Atómica

1954 El primer robot programable es diseñado por Geroge C. Devol, quien acuña el término *Universal Automation*. La patente identificada como Transferencia de Artículos Programada fue emitida en 1961. Joseph F. Engelberger se une a Devol en 1956, y forman *Unimation*, la primer compañía manufacturera de robots exitosa.

1959 Planet Corporation pone a la venta el primer robot comercialmente disponible.

1960 Condec Corporation compra Unimation y empieza el desarrollo de Unimate Robot Systems. La American Machine and Foundry (AMF Corporation) comercializa el robot Versatran, diseñado por Harry Johnson y Veljko Milenkovic.

1961 Se instala el primer robot Unimate para descargar una máquina de fundición.

1962 Motores Generales (GM) instala el primer robot industrial en una línea de producción. El robot seleccionado es un Unimate

1968 El instituto de investigaciones de Stanford construye y prueba un robot movible con capacidades de visión (Shakey).

1970 En la Universidad de Stanford se desarrolla un brazo robot que se convierte en estándar para proyectos de investigación. El brazo utilizaba energía eléctrica y se le conoció como el brazo Stanford

1971 La Asociación Japonesa de Robots Industriales (JIRA) empieza a promover el uso de los robots en las industrias japonesas.

1973 El primer robot industrial, controlado por computadora, comercialmente disponible, es desarrollado por Richard Hohn de Cincinnati Milacron Corporation. El robot recibe el nombre de T3 (The Tomorrow Tool)

1974 El profesor Scheinman, quien desarrolló el brazo Stanford, forma Vicarm Inc. para comercializar una versión del brazo para aplicaciones industriales. Éste ya era controlado por computadora.

1975 Se forma el Instituto de Robótica de América para ayudar a las industrias de los Estados Unidos a implementar robots para automatizar sus plantas.

1977 ASEA Brown Boveri Robotics Inc. ofrece dos tamaños de robots eléctricos controlados con micro computadoras.

1978 Con el apoyo de GM, Unimation desarrolla la Máquina Universal Programable para Ensamble (*PUMA*, por sus siglas en inglés *Programmable Universal Machine for Assembly*) usando tecnología de Vicarm Inc.

1980 La industria de robots comienza a crecer rápidamente, con un robot nuevo o una compañía que entraba al mercado cada mes.

1981 GM invita a la compañía japonesa manufacturera de robots Fanuc Robotics, al mercado estadounidense, formando GM-Fanuc Robotics.

1983 La industria de robots entra a un período de maduración, reconoce que los robots y los otros equipos de automatización deben integrarse a un sistema unificado.

1984 Adept Corporation introduce robots con motores eléctricos conectados directamente en los brazos, eliminando la necesidad de engranes intermedios o cadenas.

1986 Las aplicaciones e instalaciones de robots siguen creciendo, pero haciendo énfasis en la integración del robot a la celda de trabajo, sistemas flexibles de manufactura y sistemas de manufactura integrada por computadora.

1990 ASEA Brown Boveri Robotics Inc. compra la división de robótica de Cincinnati Milacron, y todos los robots futuros serán máquinas ASEA.

1991 El surtimiento de la economía global, el énfasis de fabricar productos competitivos, y el uso de tecnologías nuevas en los robots, renovó el interés de la robótica en los sistemas integrados de automatización.

1992 Después de ayudar a convertir en líder de ventas a Fanuc Robotics, en los Estados Unidos, se separan GM y Fanuc Robotics.

1994 La venta de robots en Estados Unidos, en los primeros seis meses, establece una nueva marca de unidades vendidas (4,355). Las dos compañías mas grandes son Fanuc Robotics Corporation y ASEA Brown Boveri (ABB).

1998 El uso de robots industriales sigue creciendo. El número de compañías integrando robots y equipo de automatización a celdas y sistemas se ha triplicado en los últimos dos años.

En un inicio, la industria de los robots era dominada por compañías manufactureras de los Estados Unidos: Unimation, Cincinnatti Milacron, Westinghouse, IBM, General Electric, Prab, Brinks, etc. En la actualidad, los más grandes fabricantes de robots son Fanuc Robotics Corporation, ABB, Motoman, Seiko, Panasonic Factory Automation, Kawasaki Robotics, Reis Robotics, Sony Component Products y Nachi Robotic System, todas son japonesas con excepción de ABB, la cual es un conglomerado industrial sueco, y Reis, que tiene su sede en Alemania [3].

1.2.3 Situación Actual de la Robótica en la Manufactura

El uso de los robots en los sistemas actuales de manufactura, cada vez es más común, y no solamente como unidades independientes, sino como elementos integrales en fábricas automatizadas o sistemas flexibles de manufactura. El éxito en la aplicación de un robot, incluso en las tareas más simples, requiere una interfaz del robot con los otros equipos como las máquinas productoras, alimentadores de partes, bandas transportadoras y dispositivos de inspección [4].

El rango de las posibles aplicaciones es muy amplio, se utilizan en la industria nuclear, ingeniería civil, trabajos marinos, misiones espaciales, construcción de naves, agricultura, trabajo doméstico, supervisión y seguridad, etc. En capítulo 2 se mencionan algunas aplicaciones concretas mas detalladamente. Normalmente se ven robots trabajando en conjunto con sistemas de flujo de material, máquinas herramientas de control numérico, estaciones de trabajo fijas y máquinas que procesan materiales. Los conceptos actuales de automatización, especialmente la automatización flexible, se basan fuertemente en el uso de robots. La generación que sigue de robots serán sistemas autónomos capaces de realizar su tarea en ambientes desconocidos, y se convertirán en la base de las fábricas del futuro.

Frecuentemente se piensa que los robots sustituirán al hombre, esta idea ha afectado de alguna manera su uso y aceptación. Existen diferentes opiniones de las capacidades de los robots entre ingenieros, editores, administradores y trabajadores. Tal vez puedan verse a los robots como réplicas pobres del hombre, capaces de hacer algunos trabajos simples y a veces sofisticados, de naturaleza repetitiva [5].

Los robots industriales son una herramienta que se utiliza en los ambientes de manufactura para incrementar la productividad. Se puede utilizar para trabajos rutinarios y tediosos en líneas de ensamble, o también en tareas que representen un peligro para trabajadores humanos. Por ejemplo, uno de los primeros robots fue utilizado para reemplazar trozos de combustible nuclear en plantas nucleares, lo cual pudiera representar, para un hombre, una gran exposición a la radiación. De igual manera, los robots pueden operar en una línea de ensamble, colocando componentes pequeños, como componentes electrónicos en una tarjeta de circuito impreso, liberando así al operador de esta rutinaria y tediosa operación [6].

Puede decirse entonces, que los robots cada vez van adquiriendo más popularidad para ser integrados en los nuevos sistemas integrados de manufactura, sin embargo, esta industria no fue inmune a la recesión económica del 2001. En septiembre de 2001, las empresas manufactureras de Estados Unidos, miembros de la Asociación de Industrias Robóticas recibieron órdenes por 6,763 robots, valuados en $520 millones de dólares, lo cual representa una caída del 31% en unidades y 37% en dólares con respecto a los primeros nueve meses de

2000. Pero sin embargo, la industria sigue fuerte, las debilidades que se tenían en la década de los 80's se han superado por los avances en ingeniería, tecnología, software, hardware y principalmente el saber como se hacen las cosas. Ahora los robots y los sistemas robóticos son más robustos, confiables y precisos; las mejoras en software han incrementado sus capacidades, y componentes como guía por visión, han incrementado sus posibles aplicaciones. [7].

1.3 Definición del Problema

El robot PUMA 560 que se encuentra en la celda flexible de manufactura de Amatrol, es el encargado, principalmente, de alimentar materia prima a los centros de maquinado de la celda, en él se presentan las siguientes situaciones durante su operación:

- La terminal de manejo actual del robot no cuenta con opciones que faciliten al usuario la operación del robot, por lo que a los alumnos se les dificulta saber todos los comandos necesarios para la enseñanza de puntos al brazo, el movimiento manual a estos puntos, así como el manejo de señales de salida.

- El editor de programas en la terminal es muy poco amigable, principalmente en lo que se refiere a la facilidad que le da al usuario para cambiarse de línea del programa, tanto a líneas anteriores a la posición actual, como a líneas posteriores, lo que hace que el tiempo destinado a la realización de un programa sea mayor al que se requiere en un editor de textos común.

- El respaldo de información que se genera por la enseñanza de puntos al brazo del robot, así como de los programas generados en la terminal, en la situación actual, resulta muy problemático, ya que el controlador solamente cuenta con entrada para disco de 5.25", los cuales en nuestros días son muy difíciles de conseguir debido a su obsolescencia. Cambiar este dispositivo por uno para disco de 3.5", resulta muy costoso.

El modo de operación actual de la terminal del robot PUMA 560, hace que la realización de actividades en este componente de la celda consuma mucho tiempo, principalmente el período de conocimiento del equipo y de sus funciones principales. De igual manera, la falta de un medio de respaldo de información, tanto de programas como de puntos, se vuelve todo un tema de discusión cuando se tienen varios equipos de distintos grupos que usan el robot, y muchas veces entre ellos, sin querer, pierden información de los demás, ya sea por fallas del controlador del robot o por duplicar identificadores (nombres de puntos o de programas).

1.4 Objetivo

El objetivo de esta tesis es diseñar y desarrollar un programa que funcione como una interfaz de usuario, para facilitar la enseñanza de tareas, la programación y la ejecución de programas del robot PUMA 560 de Unimation, además de brindar la opción de guardar la información generada en el disco duro de una computadora o en un disco de 3.5".

1.5 Justificación

El manejo de información, así como la optimización de los procesos de producción, son dos temas que han tomado importancia en los medios actuales de producción, sobre todo cuando se habla de automatización de procesos e integración en las empresas.

La optimización de los procesos de ensamble se vuelve crítica a medida que los volúmenes de producción crecen. La informática juega un papel determinante en este aspecto, cada vez se tienen avances que sorprenden a cualquiera, continuamente se duplica la capacidad de los equipos a la mitad del costo que tenía la tecnología anterior, y a su vez se va despertando más la expectativa por ver que será lo que nos depara para la siguiente generación de máquinas. En la actualidad, el uso del correo electrónico y del internet hacen cada vez más necesarias la información y la automatización en tiempo real, desde la comercialización a medida hasta la fabricación, para ofrecer al cliente productos que se ajusten exactamente a sus necesidades. El

tiempo necesario para comercializar un producto, su vida útil y el tiempo de amortización de los proyectos, han dejado de medirse en años para hacerlo en semanas o meses.

Se puede decir que la competitividad de las empresas depende más de los resultados que se obtengan del proceso de manufactura, lo cual implica, que un factor importante de éxito es la excelencia operativa [8].

La actualización de los equipos de producción se vuelve necesaria para obtener mejores resultados utilizando una misma tecnología. El manejo de robots en las empresas manufactureras cada vez es más común, y por lo mismo es muy conveniente contar con software que sea amigable con el usuario, y que incluso siga algunos patrones que se siguen en otras aplicaciones, permitiendo así que una mayor cantidad de operadores tengan la capacidad para operar estos equipos.

Este proyecto de tesis resalta la necesidad de crear una interfaz de usuario específicamente para el robot PUMA 560 de Unimation, que ayude a la realización de prácticas dentro y fuera del laboratorio. Además es importante que sirva como herramienta docente para un mejor aprovechamiento del equipo por parte de los alumnos.

1.6 Hipótesis

La creación de una interfaz de usuario para el robot PUMA 560, enfocado a las necesidades del Laboratorio de Sistemas Integrados de Manufactura, facilitará la enseñanza de puntos y la elaboración de programas, reduciendo el tiempo requerido para realizar una tarea determinada en comparación con el método actual. Además de aumentar la capacidad de almacenamiento de información al brindar la opción de cargar la información de puntos y programas del controlador hacia el disco duro de una computadora o un disco flexible de 3.5”, así como recuperar esa información y enviarla del sistema de almacenamiento deseado al controlador del robot.

1.7 Alcance

La interfaz de usuario para el robot PUMA 560 de Unimation esta enfocada al desarrollo de prácticas dentro del Laboratorio de Sistemas Integrados de Manufactura. Se contemplan dos modos de operación: en línea y fuera de línea. Para la operación en línea se tendrá acceso a todas las funciones del programa, siendo éstas las más comunes para la enseñanza y ejecución de tareas. Cuando se opere fuera de línea, solamente se tendrá acceso al Editor de Programas, el cual, de igual manera, presenta las opciones más comunes para la escritura de un programa, sin embargo, también permite la introducción de texto libre.

1.8 Metodología de la Investigación

Para el desarrollo de esta tesis se realizaron las siguientes actividades:

1. Investigación bibliográfica de los conceptos básicos de los temas que se involucran en este estudio: manufactura y robótica, así como la relación que se ha dado entre ellos y que han ido integrando a los robots a las actividades manufactureras, como lo es la manufactura integrada computadora, sistemas flexibles de manufactura y automatización.

2. Determinar las funciones más comunes que se requieren para la operación del robot en el desarrollo de una práctica normal del laboratorio, tanto en la operación directa con el controlador (en línea), como cuando no se esté físicamente conectado (fuera de línea), principalmente en lo que se refiere a la escritura de programas.

3. Investigación bibliográfica del lenguaje que utiliza el controlador del robot PUMA 560 (VAL-II), para conocer a detalle los comandos que serán necesarios incluir en la interfaz y saber qué parámetros necesitan y las limitaciones que tiene cada uno de ellos.

4. Analizar el tipo de protocolo con el cual se comunica la terminal del robot PUMA 560 con el controlador, para emular este protocolo con el lenguaje de programación seleccionado y

poder enviar y recibir información, principalmente: señales de salida, órdenes, puntos y programas.

5. Diseñar las pantallas de operación para la interfaz, de manera que se agrupen las funciones que estén relacionadas y se facilite la operación del robot para el usuario.

6. Escritura del código del programa que permita llevar a cabo la comunicación con el controlador del robot, de las funciones determinadas para la operación en línea, así como también la edición de programas en el lenguaje del robot para que sea posible llevarlo a cabo fuera de línea.

7. Realización de pruebas para verificar que el programa funcione completamente, teniendo especial énfasis en la enseñanza de puntos, la elaboración de programas, la ejecución de programas, y la carga y descarga de puntos y programas que se generen.

8. Realización de pruebas para comprobar la hipótesis planteada en la sección 1.6 de este capítulo.

9. Elaboración de resultados, conclusiones y propuestas para investigaciones futuras.

1.9 Recursos

El equipo con el que se cuenta para el desarrollo de la interfaz de usuario para el robot PUMA 560, es el siguiente:

1. Robot PUMA 560 de Unimation, que se encuentra en la celda flexible de manufactura del ITESM, Campus Monterrey, el cual es el centro de estudio de la presente tesis y se tienen los siguientes componentes:
   a) Brazo manipulador, controlador y terminal del sistema.
   b) Manuales de programación del robot PUMA 560.

2. Analizador de Protocolos “Monitor 232” de “Black Box Corporation”. Éste se utilizará para saber la forma como se realiza la comunicación entre la terminal del sistema y el controlador del robot.

3. Computadora personal IBM 486 o compatible, que cuente con Windows 95 como requerimientos computacionales mínimos para poder ejecutar las aplicaciones hechas en el lenguaje de programación seleccionado.

4. Compilador para el lenguaje de programación Visual Basic 6.0 de Microsoft, para la realización de la interfaz.

A continuación es importante definir claramente los conceptos involucrados de los temas principales de esta investigación: manufactura y robótica, así como la relación que existe entre ambos en la actualidad, originando una gran variedad de aplicaciones en las cuales podemos ver robots efectuando diversas operaciones productivas.

En el siguiente Capítulo 2 – Conceptos Básicos, se presenta un panorama general de la teoría que fundamenta la operación del Laboratorio de Sistemas Integrados de Manufactura, y por ende, a las ideas que dieron origen a la creación de este proyecto de tesis. También se describen las principales características y componentes del robot PUMA 560.

Capítulo 2
Conceptos Básicos

2.1 Introducción

En nuestros días, la creciente demanda de productos y el deseo de las empresas por satisfacer esta demanda, además de la necesidad de supervivencia por la competencia entre ellas, ha hecho que frecuentemente surjan nuevas teorías y métodos de producción para mejorar su productividad y eficiencia, trayendo consigo términos nuevos que con el tiempo se han hecho cotidianos en las industrias.

Sin embargo, muchos de estos términos no son familiares para los que no han tenido un contacto directo con los medios de producción, y tomando en cuenta que la presente tesis tiene fines docentes, dirigida para los alumnos de la carrera de Ingeniería Industrial y de Sistemas, se desarrolla el presente capítulo con el fin de dar un panorama general a los lectores que no tengan experiencia en el tema, así como unificar criterios y definiciones con aquellos que tengan conocimientos previos de manufactura y robótica, ya que algunos conceptos varían entre autores.

En la sección 2.2 se habla de los conceptos de manufactura, su definición, clasificación y la generación de los sistemas flexibles de manufactura, así como el concepto de manufactura integrada por computadora de una manera general. En la sección 2.3 se tratan los conceptos de robótica, definiciones de términos y componentes comunes, y sus clasificaciones de acuerdo a varios criterios. En la sección 2.4 se presentan algunas de las aplicaciones donde generalmente se utilizan robots en las industrias con el fin de mejorar la calidad ya sea de sus productos o de sus procesos en sí. Finalmente en la sección 2.5 se presentan las características y los componentes del robot PUMA 560 de Unimation para ubicarlo dentro de los conceptos mencionados en las secciones anteriores.

2.2 Manufactura

2.2.1 Definiciones Básicas para los Sistemas de Manufactura

En la actualidad se utilizan con bastante frecuencia los términos manufactura, ingeniería de manufactura y sistemas de manufactura, tanto en las escuelas como en la industria. A continuación se definen algunos términos importantes en el área manufacturera.

1. ***Manufactura*** es un conjunto de operaciones y actividades correlacionadas, en las que se incluye el diseño del producto, selección de materiales, planeación, producción, inspección, administración y la comercialización de los productos, para las industrias manufactureras.

2. ***Producción de Manufactura*** se refiere a la serie de procesos necesarios para la fabricación de un producto, en estas actividades se excluye el diseño, la planeación y el control de la producción.

3. ***Procesos de Manufactura,*** son las actividades de bajo nivel para hacer productos. En esta clasificación tenemos a los tradicionales procesos de maquinado, como el torneado, fresado y rectificado, y las mas avanzadas, que no generan viruta, como el maquinado electroquímico (ECM) y el maquinado por electrodescarga (EDM).

4. ***Ingeniería de la Manufactura,*** involucra el diseño, la operación y el control de los procesos de manufactura. Además requiere del conocimiento de otras disciplinas, como ingeniería eléctrica, ingeniería mecánica, ingeniería de materiales, ingeniería química e ingeniería de sistemas o de información.

5. ***Sistema de Manufactura,*** es una organización que reúne varios subconjuntos de manufactura interrelacionados. Su objetivo es ser la interfaz con las funciones externas de la producción para optimizar la productividad total del sistema, como el tiempo de producción, costos y utilización de las máquinas. Las actividades de los subconjuntos

incluye el diseño, la planeación, la manufactura y el control. Estos también se conectan con las funciones productivas fuera del sistema, como contabilidad, mercadotecnia, finanzas y personal.

La actividad en un sistema de manufactura inicia desde el diseño del producto, posteriormente le sigue la planeación del proceso de manufactura, y se desarrollan estrategias de control para el sistema. Una vez verificadas las operaciones de planeación y control, se puede procesar la materia prima y se hace un producto. Las salidas de los sistemas de manufactura se pueden clasificar en dos categorías: (1) salidas físicas o productos, y (2) información de la eficiencia de los procesos que puede utilizarse como retroalimentación para el sistema para realizar ajustes a las máquinas y a los mecanismos de control [1].

2.2.2 Clasificación de los Sistemas de Manufactura

La clasificación de los sistemas de manufactura, se basa en la manera como los productos son fabricados. La clasificación de los procesos divide todas las operaciones de producción en cinco grupos: proyecto, lotes, repetitivo, línea y continuo. No se puede evitar que algunas categorías se empalmen, y muchas compañías recurren a dos o más sistemas de manufactura en la producción de una línea de productos. Para clasificar a las empresas en estas categorías requiere de un análisis detallado y una evaluación de las operaciones productivas. Las características de cada una de las categorías se explican a continuación.

- **Proyecto**

La característica más importante de esta categoría es que los productos son complejos, y muchas veces las cantidades de producción son solamente una unidad. Algunos ejemplos de los productos que pertenecen a este tipo de sistema de manufactura son las refinerías, edificios grandes, cruceros y aviones grandes. En cada uno de los casos pueden haber productos similares, pero no idénticos. La distribución del equipo para este sistema se le conoce como de posición fija. Debido al peso y tamaño de los productos, éstos permanecen un lugar y el

equipo de producción y las partes se mueven hacia ellos. En este tipo de manufactura por proyecto se ven pocas aplicaciones de robots porque los equipos deben ser movibles.

- **Lotes**

Las cantidades de producción que maneja esta categoría son pequeñas, así como el tamaño y peso de los productos, y se les denomina lotes. Las partes son trasladadas entre celdas de trabajo fijas para ser procesadas. Los clásicos talleres con tornos, fresadoras, rectificadoras y taladros verticales son los ejemplos más utilizados para este tipo de sistema de manufactura. La distribución del equipo se le conoce como "job shop layout" o "layout" de proceso. Otras características que lo distinguen es que la repetición de la producción de una misma parte es menor del 20%, son productos no complejos y se presenta mucho movimiento de los productos entre las máquinas. Existen algunas oportunidades para la aplicación de robots, pero son limitadas por la alta variedad de partes y productos.

- **Repetitivo**

El sistema de manufactura del tipo repetitivo tiene como característica que las órdenes para repetir negocios están cerca del 100%, existen contratos con clientes por muchos años, el volumen de productos es moderadamente alto con cantidades de producción que pueden variar en un rango muy amplio, y poca variación en la ruta que siguen las partes entre las máquinas productivas. Éstas normalmente son máquinas dedicadas, y es muy común encontrar sistemas automatizados con robots integrados a los procesos. Por ejemplo están los proveedores de los grandes OEM's, quienes pueden tener contratos por varios años para fabricar algún componente en grandes cantidades.

- **Línea**

El sistema de manufactura de tipo línea tiene varias características que lo distinguen: el tiempo de entrega al cliente, frecuentemente es menor que el tiempo total que requiere construir todas las partes individuales del producto, el producto tiene varias opciones o modelos, y

normalmente se tienen inventario de subensambles. En ensamble de carros o camiones es un ejemplo de este sistema de manufactura. Los robots son utilizados frecuentemente para realizar tareas de ensamble mientras el producto se va moviendo por la línea.

- **Continuo**

En este sistema se presentan las siguientes características: el tiempo requerido para manufacturar el producto es mayor que el tiempo que el cliente está dispuesto a esperar para la entrega del producto, la demanda del producto es predecible, se tiene inventario del producto, los volúmenes de producción son altos y no hay muchas opciones de productos. Por ejemplo están algunos componentes eléctricos, como interruptores para la industria automotriz. Los robots que se utilizan en estas aplicaciones son capaces de operar a altas velocidades y producir altos volúmenes de piezas.

En la tabla 2.1 se puede ver una comparación de las características mencionadas de los cinco sistemas de manufactura [3].

	Proyecto	Lote	Repetitivo	Línea	Continuo
Velocidad del proceso	Varía	Lento	Moderado	Rápido	Muy Rápido
Nivel de complejidad	Alto	Alto	Mediano	Bajo	Muy Bajo
Nivel de habilidades	Alto	Alto	Moderado	Bajo	Varía
Cantidad de órdenes	Muy Bajo	Bajo	Varía	Alto	Muy Alto
Costo unitario	Muy Alto	Alto	Moderado	Bajo	Muy Bajo
Variaciones de ruteo	Muy Alto	Alto	Ninguna	Bajo	Muy Bajo
Variedad de productos	Bajo	Bajo	Ninguna	Muy Alto	Muy Bajo
Diseño de componentes	Muy Alto	Grande	Muy Bajo	Moderado	Bajo

Tabla 2.1 Características de los Sistemas de Manufactura de acuerdo a su clasificación.

2.2.3 Sistemas Flexibles de Manufactura

Un sistema flexible de manufactura (FMS) en realidad se refiere a un conjunto de sistemas que tienen diferentes grados de mecanización, transferencia automática y control por computadora. Andrew Kusiak definió cinco sistemas y como se relacionan con la producción y al número de partes usadas. Los sistemas son módulos flexibles de manufactura, celdas, grupos, sistemas de producción y líneas.

- **Módulo Flexible de Manufactura (FMM)**

Un Módulo Flexible de Manufactura es una máquina controlada numéricamente (CNC) provista de un inventario de partes, un intercambiador de herramientas y un intercambiador de contenedores o dispositivos de sujeción o transporte de las partes.

- **Celda Flexible de Manufactura (FMC)**

Una Celda Flexible de Manufactura consiste de varios Módulos Flexibles de Manufactura (FMM's), organizados de acuerdo a los requerimientos particulares del producto.

- **Grupo Flexible de Manufactura (FMG)**

Un Grupo Flexible de Manufactura es una combinación de Módulos Flexibles de Manufactura (FMM's) y Celdas Flexibles de Manufactura (FMC's), unidos por un sistema de manejo de material, tal como puede ser una banda transportadora o un vehículo guiado automáticamente (AGV)

- **Sistema Flexible de Producción (FPS)**

Un Sistema Flexible de Producción está formado por varios Grupos Flexibles de Manufactura (FMG's) que conectan diferentes áreas de manufactura, tales como fabricación, maquinado y ensamble.

- **Línea Flexible de Manufactura (FML)**

Una Línea Flexible de Manufactura es una serie de máquinas especializadas o dedicadas conectados por vehículos guiados automáticamente (AGV's), bandas transportadoras, robots o algún otro tipo de dispositivo automático de transporte.

Convencionalmente se cree que los sistemas flexibles de manufactura se aprovechan mejor en producciones de bajo volumen y con poca variedad de partes, sin embargo, la compañía de motores Cummins ha tenido éxito y ahorro en costos utilizando FMS en volúmenes altos [9].

2.2.4 Manufactura Integrada por Computadora (CIM)

El concepto de manufactura integrada por computadora (CIM, por sus siglas en inglés, "Computer Integrated Manufacturing") fue definido por primera vez en 1973, por Joseph Harrington, Jr., . Esta forma de trabajar surgió como una técnica con la que las compañías manufactureras pudieran aprovechar la tecnología computacional para reorganizar la manera como se recababa la información, analizarla y usarla para hacer más eficientes sus operaciones de manufactura, debido a que las empresas enfrentaban una creciente demanda de los clientes, así como el riesgo de éstos se fueran con la competencia. Era necesario responder mejor y más rápido sus requerimientos, darles una mayor calidad en los productos, y a la vez, que la empresa tuviera la flexibilidad suficiente para introducir nuevos productos lo más rápido posible, de acuerdo a las demandas del mercado. Además se esperaba aumentar la productividad de la empresa, así como reducir costos para obtener mayores ganancias [10].

La Manufactura Integrada por Computadora, integra todos los aspectos productivos en un sistema automatizado. Desde el diseño, pruebas, fabricación, ensamble, inspección, y el manejo de materiales, todos pueden tener funciones automatizadas. En muchas compañías, sin embargo, la comunicación entre departamentos sigue siendo a través de papelería, en CIM, estas islas de automatización son integradas, con el fin de eliminar la necesidad del papeleo.

Los sectores de la compañía son conectados por medio de una computadora, aumentando la eficiencia, reduciendo el papeleo y los gastos de personal.

Con el término "islas de automatización" se refiere a la transición de la manufactura convencional a la automatizada, las islas típicas de automatización cuentan con máquinas herramientas de control numérico, robots, sistemas automatizados de almacén, y centros de maquinado [9].

La Asociación de Computadoras y Sistemas de Automatización (CASA), de la Sociedad de Ingenieros en Manufactura (SME) define a la manufactura integrada por computadora de la siguiente manera: "*La integración de toda la compañía manufacturera a través del uso de sistemas integrados y comunicación de información, junto con nuevas filosofías de administración que mejoran la eficiencia de la organización y del personal*". CIM describe un nuevo enfoque de la manufactura, administración y operación de la compañía. Aunque los sistemas de CIM pueden incluir muchas tecnologías modernas de manufactura como robots, control numérico por computadora, diseño asistido por computadora (CAD), manufactura asistida por computadora (CAM) y producción justo a tiempo (JIT), va más allá de estas tecnologías, es una nueva forma de hacer negocios que incluye un compromiso a la calidad total de la empresa, mejora continua, satisfacción del cliente, el uso de una sola base de datos computacional para toda la información de los productos con la participación de todos los departamentos, la eliminación de las barreras de comunicación entre departamentos y la integración de los recursos de la empresa [3].

El propósito del diseño de un sistema de manufactura es hacer una empresa que cumpla de la mejor manera sus objetivos de eficiencia. Un sistema basado en manufactura integrada por computadora (CIM) es el resultado de cuando el esfuerzo en el diseño incluye el uso de las computadoras para alcanzar un flujo integrado de actividades de manufactura, basándose en el flujo de información integrado que conjunta a todas las actividades de la organización. Más que un producto que pueda comprarse e instalarse, se trata de una manera de pensar para resolver problemas [11].

2.3 Robótica

2.3.1 Terminología

En esta sección se definen los términos más comunes que se emplean en la robótica, empezaremos por definir qué es un robot. En la actualidad se manejan muchas máquinas especializadas o dedicadas, consideradas dentro del término de automatización dura, con algunas características que las hacen ver como si fueran robots. Para diferenciar a los robots industriales de los millones de máquinas automatizadas, la Organización Internacional de Estándares (ISO), define a un ***robot industrial*** en el estándar ISO/TR/8373-2.3, de la siguiente manera:

"Un robot es una máquina manipuladora controlada automáticamente, reprogramable, de propósitos múltiples con varios ejes reprogramables, los cuales pueden estar fijos en un lugar o móviles para ser usados en aplicaciones de automatización industrial" [3].

Existen otras definiciones, aunque similares, como la del Instituto de Robótica de América, la cual define a un robot como:

"un manipulador multifuncional y reprogramable diseñado para mover partes, materiales, herramientas o dispositivos especiales a través de movimientos variables programados para la ejecución de diferentes tareas" [1].

En ambas definiciones las palabras clave son reprogramable y multifuncional o de propósito múltiple, porque la mayoría de las máquinas para un solo propósito (dedicadas) no cumplen estos dos requerimientos. Que sea reprogramable implica dos elementos: uno, que el movimiento del robot es controlado por un programa escrito, y el segundo, que el programa puede ser modificado para cambiar considerablemente el movimiento del brazo del robot. Multifuncional significa que el robot puede ser utilizado para llevar a cabo diferentes funciones, dependiendo de la programación y la herramienta que se utilice. Por ejemplo, un robot puede estar programado y equipado para realizar operaciones de soldadura en una

compañía, y en otra, el mismo modelo de robot, podría equiparse y programarse para el manejo de partes.

Otro término importante son las ***localizaciones preprogramadas***, conocidas también como ***"puntos"***. Éstas marcan la trayectoria que el robot debe seguir para cumplir con su trabajo. Alunas veces en estas localizaciones, el robot debe detenerse y hacer alguna operación, como el ensamble de partes, aplicación de pintura o soldadura. Estas localizaciones preprogramadas se guardan en la memoria del robot, y pueden ser llamadas posteriormente para realizar una operación. De igual manera pueden ser cambiadas después, si es que los requerimientos del trabajo así lo dictan. Por lo tanto, con respecto a esta característica de programación, los robots industriales son como computadoras en los que se puede guardar información y luego se puede recuperar y editar.

El ***manipulador*** es el brazo del robot, y permite que éste pueda flexionarse, alcanzar o girar. Estos movimientos se dan gracias a los ejes con los que cuenta el manipulador, o también conocidos como ***grados de libertad***. En general, el número de grados de libertad de un mecanismo es igual al número de parámetros independientes o de entradas necesarios para especificar la posición del mecanismo completamente. Exceptuando algunos casos especiales, se puede determinar el número de grados de libertad de un mecanismo contando el número de eslabones, el número de uniones y el tipo de uniones incorporados en el mecanismo [2]. Los ***eslabones*** son los cuerpos rígidos individuales que forman el mecanismo, por lo general, éstos se conectan en pares, la unión entre dos eslabones se le conoce como articulación o unión propiamente. Las articulaciones tienen algunas restricciones de movimiento entre los dos miembros que unen, se distinguen seis tipos básicos de articulaciones frecuentemente usadas en los mecanismos.

- La articulación rotativa, la cual permite la rotación o giro entre dos elementos unidos, proporciona un grado de libertad.

- La articulación prismática o lineal, permite el deslizamiento entre los dos elementos unidos, a lo largo del eje definido por la geometría de la unión. Proporciona un grado de libertad.

- La articulación cilíndrica, permite la rotación y el traslado independiente con respecto a un eje, definido por la geometría de la unión. Proporciona dos grados de libertad.

- La articulación helicoidal, permite la rotación y el traslado con respecto a un eje, definido por la geometría de la unión, pero el traslado se relaciona con la rotación por el paso de un tornillo (distancia entre hilos). Proporciona un grado de libertad.

- La articulación esférica, permite la rotación libre de un miembro con respecto al otro, con respecto al centro de una esfera en todas las direcciones posibles. No permite traslación. Proporciona tres grados de libertad.

- El par plano, permite dos grados de libertad de traslación a lo largo del plano de contacto, y un grado de libertad rotativo con respecto a un eje normal al plano de contacto. Proporciona tres grados de libertad.

Dependiendo de la configuración mecánica del sistema robótico, por medio de una cierta combinación entre eslabones y tipos de uniones, se restringe el ***volumen de trabajo*** de un robot, el cual se refiere al espacio en el cual el robot puede actuar, es decir, todos los puntos que a su alrededor que pueden ser alcanzados por su herramienta o efector final.

Para que un robot pueda realizar alguna operación, debe estar equipado con algún dispositivo, dependiendo de la tarea requerida, frecuentemente se les denomina efector o actuador final. Un efector final puede ser una herramienta o un dispositivo para asir, comúnmente llamado ***"gripper"***. El ***herramental*** y las manos ("grippers") no son parte del sistema del robot en sí, si no que son accesorios que se ajustan en un extremo del brazo del robot. En la industria se le conoce como "gripper" a cualquier dispositivo para asir material. Estos accesorios permiten

que el robot pueda levantar partes, soldar, pintar, perforar, entre otras tareas, dependiendo de qué sea lo que se espera de él.

El sistema del robot también puede controlar la celda de trabajo del brazo. La ***celda de trabajo*** es el ambiente donde el robot debe hacer su tarea. Dentro de la celda puede estar el controlador, el manipulador, una mesa de trabajo, dispositivos de seguridad, bandas transportadoras, etc. Es decir, todo el equipo que requiere el robot para hacer su trabajo debe incluirse en la celda. Además, puede recibir señales externas que le indiquen cuando entrar en acción, ya sea para recoger un componente, hacer un ensamble, descargar partes de la banda transportadora, entre otras.

2.3.2 Componentes Básicos

Los sistemas robóticos se componen básicamente tres componentes: el manipulador, el controlador y su fuente de energía. El actuador final puede considerarse como un cuarto componente, el cual esta presente en algunos sistemas robóticos. A continuación veremos cada uno de estos componentes.

- **Manipulador**

El manipulador es el que hace el trabajo físico del sistema robótico, éste consiste de dos secciones: la sección mecánica y los eslabones adjuntos. El manipulador también cuenta con una base sobre la que están unidos los eslabones. La base del manipulador normalmente está fija al suelo del área de trabajo, aunque algunas veces puede ser movible. En estos casos, la base se une a un riel que permite el movimiento del manipulador de un lado a otro. Como se mencionó anteriormente, los eslabones se extienden desde la base del robot, en sí los eslabones forman el brazo del robot. Pueden formar un brazo recto, movible o articulado. Los eslabones dan al manipulador sus ejes de movimiento, los cuales son los que le permiten realizar su trabajo en un área determinada.

El movimiento del manipulador es controlado por actuadores o sistemas de transmisión, éstos permiten el movimiento de los ejes, pueden utilizar energía eléctrica, hidráulica o neumática. Los sistemas de transmisión se conectan mediante eslabones mecánicos, los cuales transmiten el movimiento a los distintos ejes del robot. Éstos pueden estar conformados por cadenas, engranes y tornillos de bolas.

- **Controlador**

El controlador generalmente es un sistema basado en microprocesadores, y en los sistemas robóticos constituyen el corazón de la operación. El controlador almacena la información preprogramada para ser recuperada posteriormente, controla los movimientos del robot manipulador, los dispositivos periféricos y se comunica con las computadoras dentro de una planta para actualizar constantemente su producción. Los controladores en la mayoría de los robots son dispositivos complejos y representan el estado del arte en la electrónica. El controlador puede enviar señales eléctricas por sus líneas de comunicación que le permiten comunicarse con los ejes del manipulador. Esta comunicación de dos vías entre el robot manipulador y el controlador mantiene una actualización constante de la localización y la operación del sistema. El controlador también manipula la herramienta que esté colocada en la muñeca del brazo.

Debido a la característica que deben cumplir los robots, por la definición, de ser reprogramable, el controlador debe contener algún tipo de memoria. El sistema basado en microprocesadores opera en conjunto con dispositivos de memoria de estado sólido, como memorias RAM, discos flexibles o cintas magnéticas, para almacenar la información para ser utilizada o editada en ocasiones posteriores.

- **Fuente de Energía**

La fuente de energía es la unidad que abastece de poder al controlador y al manipulador. Se distinguen dos tipos de energía para ser utilizada por el sistema robótico. Un tipo es la corriente alterna (eléctrica) para la operación del controlador. El otro tipo de energía es la

utilizada para mover los ejes del manipulador. Cada sistema robótico requiere de energía para poder operar el manipulador, ésta puede ser desarrollada por una fuente de energía hidráulica, neumática o eléctrica, y forman parte de los componentes necesarios para colocar el manipulador en las posiciones deseadas.

Las fuentes hidráulicas y neumáticas producen fluidos bajo presión, y son enviados a los actuadores del manipulador, provocando el movimiento del brazo. Las fuentes eléctricas mueven motores eléctricos, ya sean de corriente alterna o de corriente directa. Las señales en forma de pulsos eléctricos se generan en el controlador y se envían al motor eléctrico en el manipulador. Estos pulsos le dan al motor la información suficiente para mover el brazo. Para cada uno de los tres tipos de fuentes de energía se utiliza un sistema de retroalimentación, este sistema informa constantemente la posición de los ejes al controlador.

- **Efector Final**

El efector final que se observa en la mayoría de las aplicaciones de robots, es un dispositivo conectado a la muñeca del brazo del manipulador. Es un componente crítico ya que es el que entra en contacto con el mundo y debe adaptarse a las distintas aplicaciones que se presentan en el área de producción; por ejemplo, para levantar partes, soldar o pintar. El efector final debe dar al robot la flexibilidad necesaria para su operación, generalmente son diseñados para satisfacer las necesidades del usuario, y pueden ser hechos por el mismo fabricante del robot, o por el dueño del robot. Éste es el único componente del robot que puede cambiar de un trabajo a otro, esto aunado con la característica de ser reprogramable, permite a estos sistemas ser muy flexibles [6].

2.3.3 Clasificación de los Robots

Los robots pueden clasificarse de acuerdo a varios criterios, entre ellos están: por grados de libertad, por su fuente de energía, por su volumen de trabajo o envolvente y por el tipo de

control. Aunque algunos autores identifican más clasificaciones, éstas son las más comunes y son las que se explican a continuación.

2.3.3.1 Clasificación por Grados de Libertad

Una clasificación, de alguna manera obvia, es por medio del número de grados de libertad. Idealmente un manipulador debe contar con 6 grados de libertad para poder manipular un objeto libremente en el espacio tridimensional. Desde este punto de vista, se les denomina ***robot de propósito general*** a los que precisamente cuentan con 6 ejes o grados de libertad, ***redundante*** si es que tiene más de seis grados de libertad y ***deficiente*** si posee menos de seis grados de libertad.

Los robots redundantes tienen más libertad para librar obstáculos en su movimiento y puede trabajar más fácilmente en espacios reducidos, pero por el otro lado, para algunas aplicaciones especiales, tales como ensamble de componentes en un solo plano, un robot con solamente 4 grados de libertad es suficiente [2].

2.3.3.2 Clasificación por su Volumen de Trabajo o Envolvente

Este tipo de clasificación es el más común para los robots industriales, generalmente se identifican cuatro configuraciones: Robots Cartesianos, Robots Cilíndricos, Robots Esféricos y Robots Articulados. Puede considerarse a los Robots SCARA como una quinta configuración, ya que son muy populares y no se contemplan en ninguna de las configuraciones antes mencionadas. Principalmente las clasificaciones se enfocan en los tres primeros ejes de movimiento, es decir, los ejes que forman el brazo, los cuales son los que le dan la capacidad al brazo de alcanzar cualquier punto dentro de la envolvente del robot. Los ejes subsecuentes, son los ejes de la muñeca, con los cuales se le da la orientación al actuador final. En caso de ser por lo menos otros tres ejes, entonces se puede dar cualquier orientación, si son menos, la orientación del actuador estará más limitada. En los siguientes párrafos se explica en que consiste cada una de las clasificaciones [1] y [3].

- **Robots Cartesianos**

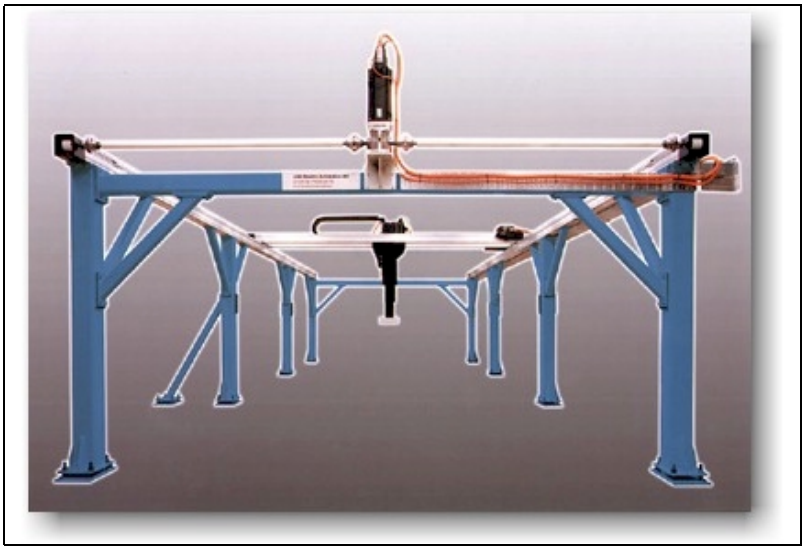


Figura 2.1 Ejemplo de un robot cartesiano

Se les denomina robots cartesianos a los que tienen su marco principal compuesto por tres ejes prismáticos o lineales, como el que se muestra en la Figura 2.1. Su nombre se deriva del sistema de coordenadas, ya que su movimiento normalmente es lineal en los tres ejes X, Y e Z, por lo que su volumen de trabajo o la envolvente que se genera en sus movimientos es un cubo o una caja rectangular. En la figura 2.2 se muestra el principio de la estructura de este tipo de robots, así como el volumen de trabajo en el cual pueden actuar. Generalmente son utilizados en aplicaciones de carga y descarga de material en maquinarias.

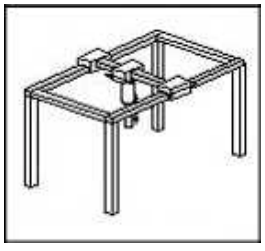
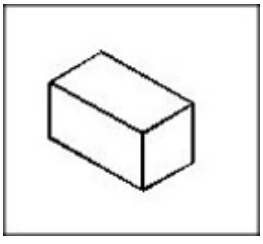
a)  b) 

Figura 2.2 a) Principio de la estructura que forma un robot de coordenadas cartesianas.
b) Volumen de trabajo o envolvente de un robot de coordenadas cartesianas.

- **Robots Cilíndricos**

Figura 2.3 Ejemplo de un robot cilíndrico

Si a un robot cartesiano, se le reemplaza la primera o la segunda articulación por una rotativa, se genera un robot cilíndrico, es decir, estos cuentan con dos ejes lineales y un eje rotativo, tal como el que se muestra en la Figura 2.3. El nombre se deriva del volumen de trabajo que se crea cuando se mueven los ejes de un límite al otro, el cual es confinado por dos cilindros concéntricos de longitud finita. En la figura 2.4 se muestra el principio de la estructura de este tipo de robots, así como el volumen de trabajo o envolvente en la cual el robot puede situar su efector final.

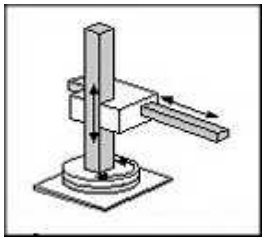
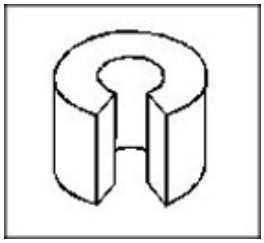
a)  b) 

Figura 2.4 a) Principio de la estructura que forma un robot de coordenadas cilíndricas.
b) Volumen de trabajo o envolvente de un robot de coordenadas cilíndricas.

- **Robots Esféricos**

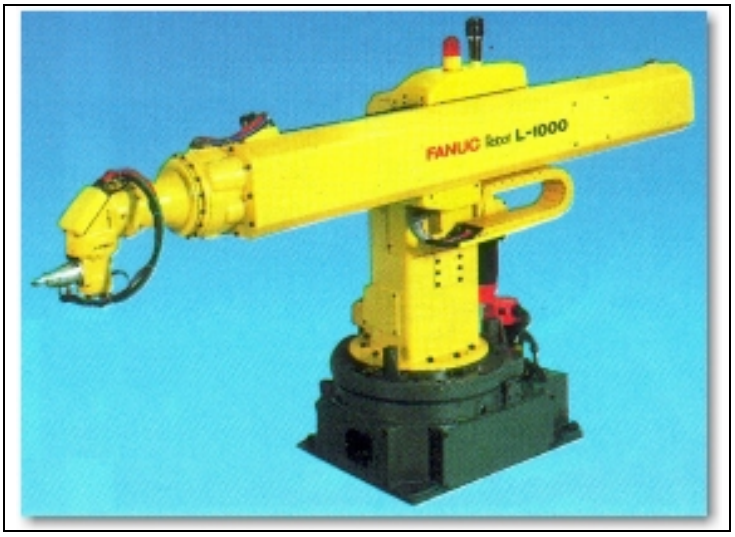


Figura 2.5 Ejemplo de un robot esférico

Un robot esférico o polar, tiene los dos primeros ejes rotativos y el tercero es una articulación prismática, normalmente la unión prismática no es paralela al eje de la segunda articulación. En la figura 2.5 se muestra un ejemplo de esta clasificación de robots. Éstos son usados en varias tareas industriales como soldadura y manejo de materiales. La posición del actuador final puede ser descrito por coordenadas esféricas, asociadas a las tres articulaciones, por lo que el volumen de trabajo que se genera se define con dos esferas concéntricas, tal como se muestra en la Figura 2.6.

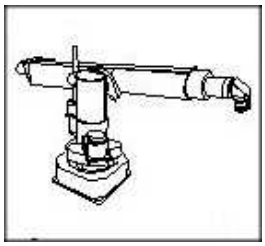
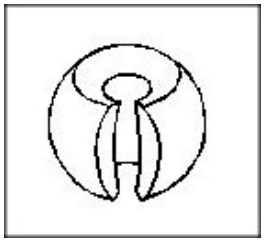
a)  b) 

Figura 2.6 a) Principio de la estructura que forma un robot de coordenadas esféricas.
b) Volumen de trabajo o envolvente de un robot de coordenadas esféricas.

- **Robots Articulados**

Figura 2.7 Ejemplo de un robot articulado

Los robots articulados tienen tres ejes rotativos, conectando tres eslabones rígidos y una base. Frecuentemente se les denomina como brazos antropomórficos por su semejanza con un brazo humano, como el ejemplo que se muestra en la Figura 2.7. La primer articulación sobre la base corresponde al hombro, la articulación del hombro se conecta al brazo superior, el cual a su vez está conectado a la articulación del codo. Este tipo de robots son apropiados para una gran variedad de tareas industriales, principalmente por la flexibilidad que presentan en sus articulaciones. El volumen de trabajo de un robot articulado es muy complejo para ser descrito por una figura simple, en la Figura 2.8 se muestra una aproximación de esta envolvente.

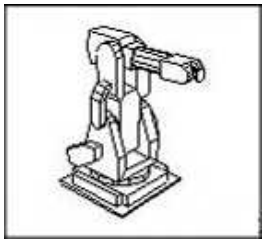
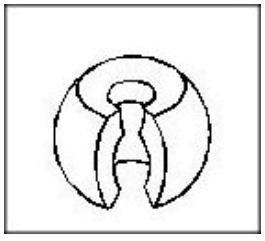
a)  b) 

Figura 2.8 a) Principio de la estructura que forma un robot articulado.
b) Volumen de trabajo o envolvente de un robot articulado.

- **Robots SCARA**

Figura 2.9 Ejemplo de un robot tipo SCARA

Este tipo de robots, usado ampliamente en el ensamble de componentes electrónicos, surgen de una combinación entre un brazo articulado y un robot cilíndrico. Su nombre corresponde a las iniciales de "*Selective Compliance Assembly Robot Arm*". Está conformado por dos articulaciones rotativas seguidas por una prismática, además los tres ejes son paralelos entre sí, y apuntan en el sentido de la gravedad, es decir, que están montados verticalmente, en lugar de horizontalmente. En la Figura 2.9 se muestra un ejemplo de este tipo de robots.

Esta configuración minimiza la deflexión del brazo mientras carga a un objeto a una velocidad determinada, debido a que los dos primeros actuadores no tienen que vencer a la fuerza de gravedad y a la carga que transporte en su movimiento. La muñeca posee un grado de libertad, por lo que en suma estos robots cuentan con cuatro grados de libertad, son muy rápidos pero pueden cargar solamente pesos ligeros. En la Figura 2.10 se muestra el principio de la estructura que forma a esta configuración de robots, así como el volumen de trabajo que se genera cuando el robot realiza sus movimientos.

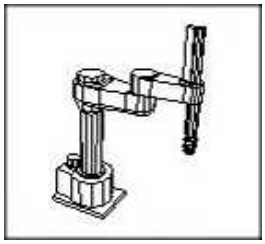
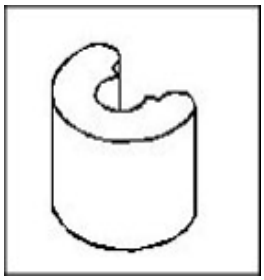
a)  b) 

Figura 2.10 a) Principio de la estructura que forma un robot SCARA.
b) Volumen de trabajo o envolvente de un robot SCARA.

2.3.3.3 Clasificación por su Fuente de Energía

Las fuentes de energía más comunes son la eléctrica, la hidráulica y la neumática. La mayoría de los manipuladores utilizan servomotores eléctricos de corriente directa o motores de pasos, por ser más limpios y relativamente fáciles de controlar, sin embargo, cuando se requiere de mayor velocidad o de mayor capacidad de carga, se prefiere la energía hidráulica o la neumática.

La desventaja de la energía hidráulica es la posible presencia de fugas, además que por sus propiedades físicas, presenta cierta elasticidad. Las fuentes neumáticas son mas limpias y rápidas, sin embargo es difícil de controlar por ser un fluido compresible [2].

2.3.3.4 Clasificación por su Tipo de Control

Con respecto al tipo de control que tiene un robot industrial, estos se clasifican en servo controlados (de lazo cerrado) y no servo controlados (de lazo abierto).

- **Robots Servo Controlados**

En los robots servo controlados, el servo control permite a los mecanismos del robot comunicarse con los dispositivos electrónicos del controlador. En estos robots el control de su

posición se debe a una constante retroalimentación de los ejes hacia el controlador. Esta constante actualización se lleva a cabo por dispositivos llamados sistemas servo o de lazo cerrado, con los cuales es posible medir y controlar la velocidad y posición de la herramienta en cada punto del movimiento del robot, dentro de su volumen de trabajo.

Los controladores de lazo cerrado son más complejos y estadísticamente tienen una mayor probabilidad de fallar que los sistemas de lazo abierto. Sin embargo las ventajas que tienen los robots servo controlados en ciertos procesos de manufactura superan a los problemas que trae consigo la complejidad inherente de estos sistemas, una ventaja, por ejemplo, tienen alta repetitividad para localizar puntos dentro de su envolvente.

- **Robots No Servo Controlados**

En los robots no servo controlados no se le da información al controlador acerca de la posición del brazo o su razón de cambio. El brazo solamente recibe la señal de movimiento y el controlador confía en la operación de cada actuador para llegar a la ubicación programada. La confiabilidad depende grandemente en los componentes que se usen en su implantación. La mayoría de estos sistemas son neumáticos, con controladores lógicos programables para el control, cilindros neumáticos como actuadores y activados por válvulas neumáticas.

Para controlar a este tipo de robots deben incluirse mecanismos internos para llegar a la posición exacta, como paros duros (fijos y ajustables), interruptores y motores de pasos; y dispositivos externos para asegurar la secuencia correcta de movimientos. También son conocidos como robots “bang-bang”, ya que el movimiento de sus ejes son controlados por mecanismos de paro “duros” que se colocan en su trayectoria [6].

2.4 Aplicaciones de Robótica en la Manufactura.

Muchas de las aplicaciones de los robots están en el área industrial. En un principio se utilizaban principalmente en aplicaciones donde existían riesgos altos para los humanos, el

primero de ellos, por ejemplo, fue en plantas de energía nuclear. Los brazos eran utilizados para cambiar rodillos de uranio en un reactor nuclear. En la actualidad, se pueden encontrar robots en muchas otras aplicaciones, como en trabajos que pueden ser muy repetitivos, como soldadura de puntos, manejo de materiales, etc. Principalmente los robots ahora son instalados para incrementar la productividad y la flexibilidad, realizar operaciones de alto riesgo o para mejorar la calidad de los productos.

Manejo de Materiales	**Manufactura**	**Ensamble**	**Pruebas**
- Carga - Descarga - Peletizado	- Corte - Formado - Recubrimientos - Pintura	- Acoplamientos - Con adhesivos - Soldadura - Atornillado - Remachado	- Calibrado - Verificación - Inspección

Tabla 2.2 Areas de aplicación de los Robots Industriales [5]

Independientemente del tipo de tarea que se vaya a realizar, siempre se debe contar con las herramientas adecuadas, y eso aplica también para los sistemas robóticos. Tradicionalmente, los robots cartesianos han sido la alternativa de bajo costo, los SCARA se han usado cuando se requiere de velocidad, además de ser más precisos que los brazos articulados. Sin embargo, éstos tienen más flexibilidad que los cartesianos y que los SCARA, lo cual los hace apropiados para aplicaciones en los que tengan que imitar el movimiento del brazo humano, como soldadura y pintura, además de que se pueden reprogramar más fácilmente, para manejar múltiples partes o trabajos, o incluso ser utilizados en otras aplicaciones completamente diferentes.

Por una parte, es cierto que cada robot tiene sus nichos de operación, y el tipo de aplicación va a determinar cuál robot es el adecuado, sin embargo existen algunas operaciones, como las de ensamble, en la que cualquiera de los tres tipos puede utilizarse. Otra herramienta que debe tomarse en cuenta al momento de seleccionar un robot, es el software de simulación, el cual permite determinar si el manipulador tiene el alcance necesario, que puede cumplir con el

tiempo de ciclo, que tiene la fuerza necesaria para soportar la carga útil, entre otras cuestiones importantes de su desempeño [12].

2.4.1 Manejo de Materiales

El manejo de materiales se refiere al proceso en el cual un robot transfiere partes de una operación a otra. Ésta fue una de las primera aplicaciones típicas de las primeras generaciones de robots. Realizaban operaciones de tomar y colocar, los ciclos podían ser de cargar materia prima en una máquina y luego retirar la parte terminada, o ambas. Dos operaciones típicas de robots en el manejo de materiales es el peletizado y el seguimiento en línea, además de la carga y descarga de las máquinas como de vaciado en moldes, de inyección en dados, de vacío, forjas, máquinas de corte de material, sistemas de tratamiento térmicos, y otras máquinas misceláneas como de inspección. La carga y descarga de partes frecuentemente es combinada también con otras actividades del proceso de manufactura.

Todas las tareas en una operación de manejo de materiales deben acoplarse al flujo del proceso, es decir, que el flujo de la operación debe ir de una estación a la siguiente sin interrupción. Por esto es importante que el manipulador tenga la fuerza suficiente para levantar las partes, que tenga el alcance necesario y que sea lo suficientemente rápido para cumplir con el ciclo de transferencia de la operación. A continuación se explican las dos operaciones típicas de esta aplicación: el peletizado y el seguimiento en línea.

- **Peletizado**

En la industria existen muchas aplicaciones de manejo de material que requieren que el robot acumule partes en algún contenedor, esto es, que las peletice. Esto implica dos procesos, el peletizado y la depaletización. El primero, por ejemplo, puede ser el proceso de tomar partes de la línea de ensamble y que las acumule en un contenedor. El segundo, puede ser el proceso de tomar las partes del contenedor y colocarlas en la línea de ensamble.

- **Seguimiento en Línea**

El seguimiento en línea se refiere a los procesos en los que el robot viaja junto con la parte en la línea de ensamble. Este seguimiento puede solamente ser por unos metros, pero mientras el robot sigue a la parte, puede hacerle alguna operación. Este tipo de aplicación es muy utilizada en el ensamble de autos, ya que se producen muchos carros por hora y no pueden detenerse en la línea para ser trabajados, por lo que el manipulador debe moverse junto con el ensamble para completar el proceso [6].

2.4.2 Inspección

Esta es una aplicación relativamente nueva para los robots, en la cual, puede tener un papel activo o pasivo. En el papel pasivo el robot toma la parte y la coloca en el dispositivo de medición el cual determina si la parte cumple con la especificación, mientras que el robot espera a que termine el proceso. En el papel activo, el robot es el responsable de determinar si la parte se acepta o se rechaza.

2.4.3 Ensamble de Productos

El ensamble de componentes discretos es un área de manufactura donde las eficiencias normalmente son menores a las que se tienen en maquinados o inspecciones, el incremento de productividad se logra por medio de máquinas especializadas y procedimientos de ensamble mejorados, muchas veces los productos deben ser rediseñados para el ensamble automático y entonces se crean nuevos sistemas de ensamble. La automatización puede ser fija o flexible. En la automatización fija, la operación se repite una y otra vez, por lo que se usa en lotes grandes. Un sistema de automatización fija no puede ser reprogramado fácilmente cuando se presenta un producto nuevo. La automatización flexible permite hacer cambios una vez que el ensamble actual se haya terminado, por lo que cuando se termina con un lote de bajo volumen, el robot puede reprogramarse para otra operación.

Otro aspecto importante a considerar es el diseño de las partes, para que puedan ser ensambladas por el robot. El actuador final debe ser capaz de tomar la pieza fácilmente y colocarla en su lugar, por ejemplo, se prefieren ensambles en los que las partes lleven una orientación similar, que estén diseñadas para la sujeción con la mano del robot, o que la variación de tornillos sea la mínima posible. También es importante minimizar el número de partes.

Las actividades de ensamble se pueden clasificar en dos tipos: el ensamble de partes pequeñas usando dispositivos de sujeción activos o pasivos; y ensambles usando operaciones de unión como la soldadura.

- **Soldadura**

Las tareas de soldadura es otra área donde los robots han sido de gran utilidad. Éstas se dividen en dos tipos: de arco y de puntos. Para este tipo de tareas el robot debe ser articulado, y muchas veces se requiere que cuente con seis ejes, la velocidad de sus ejes debe ser la suficiente para soldar según se requiera y el controlador debe cumplir con la capacidad de memoria para realizar la tarea.

Para aplicaciones de soldadura de arco, los robots requieren de dispositivos periféricos, mordazas y plantillas con movimientos de rotación para que tenga un buen acceso a la pieza a soldar. Muchas partes que se usan en la industria automotriz requieren de operaciones de soldadura de costura continua, la mayoría de estas son relativamente simples, sin embargo cuando se presentan perfiles difíciles de seguir y que pueden presentarse desviaciones, el movimiento se controla con sensores. Los sensores buscan la línea de trabajo y llevan a la herramienta de soldadura a lo largo de la costura deseada.

La soldadura de puntos requiere de robots precisos y con buena capacidad de repetitividad, además de tener la fuerza necesaria para cargar con la herramienta. En la industria automotriz, se utilizan mucho los robots para aplicar soldadura de puntos en las carrocerías y en sus partes,

frecuentemente se pueden ver varios robots soldando en serie dentro de una línea de producción, o en paralelo en una celda [13].

- **Ensamble de Tarjetas de Circuito Impreso**

La expansión de la robótica ha llegado a muchas operaciones de manufactura que requieren repetitividad y precisión durante el ensamble. En la actualidad muchos de los ensambles de tarjetas electrónicas de circuito impreso se realiza por medio de máquinas especializadas, sin embargo un problema común son los componentes de forma diferente. Las máquinas colocadoras de componentes no pueden contemplar todas las variaciones que existen de resistencias, diodos, transistores, microprocesadores, etc. por lo que para el ensamble de algunas formas diferentes que puedan tener estos elementos, en las tarjetas, se han empleado robots, los cuales se pueden programar para este objetivo.

2.4.4 Aplicación de Recubrimientos

La aplicación de recubrimientos fue de los primeros procesos de manufactura que se automatizaran por medio del uso de robots, y en la actualidad es su segunda aplicación más importante. La aplicación de recubrimientos incluye la pintura. Típicamente se utilizan robots de cinco o seis ejes rotativos, diseñados para operar en áreas grandes de trabajo y manejan cargas de 10 a 40 kilogramos. La mayoría de las aplicaciones están en la industria automotriz, como la aplicación de rellenos de PVC o de otros materiales y acabados de superficies. Sin embargo, al igual que otras aplicaciones de robótica que primero fueron utilizados en el área automotriz, así como por sus proveedores, la aplicación de recubrimientos han sido introducidos en otras aplicaciones industriales más generales, donde tienen un alto potencial de crecimiento. Algunas compañías ya empezaron a invertir en este aspecto, utilizando robots para aplicaciones como recubrimientos de muebles, de piezas sanitarias, recubrimiento de conductores de teléfonos celulares y otros dispositivos electrónicos de mano y pintura de los componentes de los teléfonos celulares. Esta industria de telefonía en particular ha adoptado a los robots para estas aplicaciones. [14]

- **Pintura**

Los robots se pueden utilizar para aplicaciones de pintura de muchos productos, como autos, productos de consumo y bienes domésticos. Y gracias a la capacidad de almacenamiento de información, se tiene flexibilidad para pintar distintos productos con solo llamar al programa correspondiente. Otro beneficio importante es la seguridad ambiental. La pintura, como la soldadura, se considera aplicación peligrosa, ya que son materiales volátiles, y aunque se trabaje según especificación, cualquier chispa puede causar una explosión. Tal vez haya algunas pinturas que no sean necesariamente explosivas, pero pueden ser tóxicas [14]. Gracias al empleo de robots en este tipo de tareas, los humanos no se exponen a estos riesgos.

Los robots que se requieren deben ser articulados, por lo menos de seis ejes, en algunas aplicaciones llegan a tener hasta nueve ejes. Se requieren así porque le dan al manipulador el movimiento necesario para pintar todas las áreas del producto [6].

2.4.5 Operaciones de Corte o Maquinado

Aunque no se espera que los robots sean, en un futuro próximo, sustitutos de las máquinas de control numérico, se pueden ver cada vez más en operaciones de maquinado.

En comparación con las máquinas herramientas convencionales, un robot puede abarcar volúmenes de trabajo mayores y tienen más flexibilidad, sin embargo, esta misma propiedad hace que pierdan precisión. Para que realicen verdaderas operaciones de maquinado, los robots deben ser más rígidos, y esperar a que se desarrollen sensores de fuerza y torque de tiempo real.

Entre las aplicaciones típicas están el fresado de metales, madera y plásticos con requerimientos de precisión bajos, perforado de hojas metálicas con ayuda de plantillas, corte con chorro de agua y rayo láser de contornos complicados de hojas planas y placas, operaciones de acabado como rectificado, cepillado y pulido [15].

2.5 El Robot PUMA 560 de Unimation

Figura 2.11 Sistema robótico PUMA 560 de Unimation

De acuerdo a las clasificaciones que se mencionan en las secciones anteriores, el robot PUMA 560 de Unimation, es un robot de propósito general por poseer seis grados de libertad. En cuanto a su clasificación por volumen de trabajo es un brazo articulado, en el cual los seis ejes son rotativos, los tres primeros conforman los movimientos del brazo superior, y los últimos tres corresponden al movimiento de la muñeca, por lo que puede alcanzar cualquier punto dentro de su envolvente en cualquier dirección.

Las fuentes de energía que utiliza son eléctrica y neumática. El controlador y los motores que mueven las articulaciones son eléctricos, y el mecanismo de activación del "gripper" es neumático. El controlador cuenta con sistemas de retroalimentación, por lo que por el tipo de control es un robot servo controlado.

2.5.1 Descripción del Sistema y del Software

El sistema PUMA esta diseñado para adaptarse a un amplio rango de aplicaciones. Las unidades básicas son: el control manual para enseñanza de puntos (teach pendant), software, el

controlador, equipos periféricos y el brazo manipulador. En la Figura 2.11 se muestra de forma general, los componentes básicos que forman al sistema, en las siguientes secciones se identifican cada una de ellas más detalladamente.

El software del sistema que controla el brazo del robot se llama VAL. El software VAL es un lenguaje de alto nivel y se guarda en la memoria de la computadora localizada en el controlador, donde también están los controles operacionales del sistema.

Para la enseñanza de puntos, se pueden seguir dos procedimientos: El control manual puede usarse para dirigir los movimientos del brazo manipulador a través de cada paso del trabajo a realizar. Estos pasos se guardan en la memoria de la computadora. El segundo método es introducir los datos de las posiciones, por medio del teclado de la terminal periférica, a la memoria de la computadora. En ambos casos, el controlador manda las instrucciones hacia el brazo del robot. Por medio de codificadores incrementales y potenciómetros, se obtiene información de la posición del brazo y se transmiten al controlador, dando un control de lazo cerrado para los movimientos del brazo. Además, el PUMA puede interactuar con otros equipos por medio de señales de entrada y de salida. Por medio de las señales de entrada la ejecución de un programa puede detenerse hasta que suceda alguna acción de otro equipo. Las señales de salida permiten al robot controlar otros equipos relacionados con su ambiente de trabajo.

Las posiciones que se le enseñan al brazo se pueden guardar de dos maneras: como transformaciones (con respecto a un sistema de coordenadas fijo, relativo a la base del robot) o como puntos de precisión (la posición se guarda con los ángulos de cada articulación) [16].

2.5.2 El Controlador

El controlador es el componente maestro del sistema eléctrico, todas las señales que van o que vienen del robot pasan por el controlador y se utilizan para realizar cálculos, en tiempo real,

para controlar el movimiento y posición del brazo. En la figura 2.12 se puede identificar este componente marcado con la letra “A”.

Las conexiones para el brazo, la terminal, la unidad de disco flexible, módulos de entradas y salidas, y accesorios se encuentran en el panel trasero de la unidad de control. El software VAL también se encuentra en la memoria de la computadora, localizada en la unidad de control. Éste software interpreta las instrucciones para el brazo del robot, y el controlador transmite estas instrucciones de la memoria de la computadora al brazo [16].

2.5.3 Periféricos

Los componentes periféricos del sistema del PUMA se utilizan para introducir o recibir información. Estos componentes están conformados por una terminal, un control manual para enseñanza de puntos (teach pendant), unidad de disco flexible, memoria adicional y un módulo de entradas y salidas.

- **Terminal**

Una terminal consta de un teclado y una unidad de vídeo, la cual se usa para la enseñanza de puntos, edición de programas y comunicación con el sistema de control, utiliza la interfaz estándar RS-232C. Cuando el robot se encuentra ejecutando un programa, la terminal puede ser desconectada. Ver en la Figura 2.12, el componente identificado con la letra “B”.

- **Control Manual para Enseñanza de Puntos (“Teach Pendant”)**

El “teach pendant” se utiliza para colocar el brazo del robot en alguna posición determinada, manipulando sus articulaciones. Existen varios modos de enseñanza disponibles para proveer de una flexibilidad de enseñanza completa. Los modos son: por articulaciones (“Joint”), modo libre (“Free”), ejes cartesianos absolutos (“World”) y ejes cartesianos relativos (“Tool”). La

velocidad del movimiento del brazo se controla por un interruptor de tres posiciones. En la Figura 2.12 se puede identificar a este componente, el cuál está señalado con la letra “C”.

- **Unidad de Disco Flexible**

La unidad de disco flexible es utilizada para almacenar programas e información de las posiciones grabadas. La transmisión de información hacia el disco flexible y su recuperación es controlado por su propio microprocesador a través de un puerto serial [16]. Este componente se puede ver en la Figura 2.12, identificado con la letra “D”.

2.5.4 El Brazo Manipulador

Éste es el componente mecánico del sistema, cuenta con 6 grados de libertad y cada uno es controlado por un servomotor de corriente directa. En la Figura 2.12 se identifica este componente con la letra “E”. Cada uno de sus miembros se conecta al siguiente por una articulación, semejante a un brazo humano. Cada articulación permite la rotación de los miembros. Haciendo una semejanza con el hombre, este robot cuenta con un tronco, un hombro, brazo, antebrazo, muñeca y mano. En la tabla 2.3 se explica cada una de estas articulaciones y su semejanza con el brazo humano. Cada miembro contiene servomotores y trenes de engranajes.

Cada miembro del brazo es movido por un servomotor de corriente directa de imán permanente, el cual mueve a su tren de engranaje asociado. Cada motor en el brazo tiene un codificador incremental y un potenciómetro. Para que el robot PUMA funcione adecuadamente se requiere controlar la posición y la velocidad de cada articulación del brazo. Para un sistema robótico de servo control, la posición debe ser medida con respecto a una posición inicial absoluta. Los potenciómetros incorporados en los motores se utilizan para inicializar el robot, es decir, establecer su posición absoluta. Esto debe hacerse cada que el sistema es encendido.

Articulación	Descripción
Cintura – Articulación #1	El eje de la articulación #1 es perpendicular al plano de montaje del PUMA y coincide con la línea central del tronco.
Hombro – Articulación #2	El eje de la articulación #2 es perpendicular e intercepta al eje de la articulación #1, coincide con la línea central del hombro.
Codo – Articulación #3	El eje de la articulación #3 es paralelo al eje de la articulación #2
Muñeca – Articulación #4	El eje de la articulación #4 es perpendicular e intercepta al eje de la articulación #5
Muñeca – Articulación #5	El eje de la articulación #5 es paralelo a los ejes de las articulaciones #2 y #3
Muñeca – Articulación #6	El eje de la articulación #6 es perpendicular e intercepta al eje de la articulación #5; coincide con la línea central de la montura para el "gripper".

Tabla 2.3 Ejes del brazo del robot PUMA 560

Los codificadores incrementales están montados en la flecha de cada motor, estos notifican del cambio de posición y velocidad al servo sistema. Las señales de posición se leen directamente de los codificadores, y las de velocidad son calculadas. Los servomotores para los ejes mayores (articulaciones #1, #2 y #3) cuentan con un freno electromagnético. Estos frenos se activan cuando se apagan los motores, y aseguran el brazo en una posición fija.

La energía para los motores se provee por medio de un cable que conecta al brazo y el controlador, las señales de retroalimentación de los codificadores incrementales y potenciómetros también son llevadas por este cable [16].

Figura 2.12 Componentes principales del sistema robótico PUMA 560

A) Controlador

B) Terminal

C) Teach Pendant

D) Unidad de Disco Flexible

E) Brazo Manipulador

2.5.5 Especificaciones

En la tabla 2.4, secciones a, b y c, se resumen la especificaciones del sistema robótico PUMA de la serie 500 [16].

Componente	Característica	Especificación
Brazo del robot	Ejes	Seis ejes rotativos
	Espacio requerido	Volumen esférico con el hombro al centro, de 0.92m de radio.
	Peso	534N (120 lb)
	Motores	Servomotores eléctricos de corriente directa
	Fuerza estática en la herramienta	58N ((13.0 lb) máximo
	Repetitividad	±0.1mm dentro de la envolvente primaria
	Aceleración de la herramienta	1 g máximo (con carga máxima)
	Velocidad de la herramienta	1.0 m/s máximo (con carga máxima dentro de la envolvente primaria
Valores máximos de Operación de Parámetros	Cintura – Articulación #1	5.59 r (320°)
	Hombro – Articulación #2	4.36 r (250°)
	Codo – Articulación #3	4.72 r (270°)
	Muñeca – Articulación #4	5.24 r (300°)
	Muñeca – Articulación #5	3.49 r (200°)
	Muñeca – Articulación #6	9.29 r (532°)

Tabla 2.4 – a. Especificaciones del sistema del robot PUMA

Controlador	Instrucciones	Por medio del control manual o de la computadora de la terminal
	Lenguaje de programación	VAL de Unimation
	Requerimientos de energía	110 – 130 VAC, 50 – 60 Hz, 1200 W
	Temperatura Ambiente	46° C (115°F) máximo
	Humedad Relativa	90% máximo sin condensación
Control Manual Para la Enseñanza de Puntos (Teach Pendand)	Tipo de Transmisión	Asíncrona
	Baud Rate	9600
	Número de bits de información	8
	Número de bits de paro	1
	Paridad	Ninguna
	Comunicación	Microprocesador 8748 con reloj de 6 MHz, línea serial RS232 dual completa
	Temperatura de operación	10°C a 50°C, a una humedad relativa de 10% a 90% sin condensación

Tabla 2.4 – b. Especificaciones del sistema del robot PUMA (continuación)

Terminal CRT	Tipo de Transmisión	Asíncrona
	Baud Rate	9600
	Modo	Un medio dual
	Paridad de Bits de Paro	Un Bit de paro
	Consumo de Energía	85 wats
	Requerimientos de Energía	0.70 amperes, 110 VAC, 60 Hz
	Temperatura de Operación	10°C a 40°C a una humedad relativa de 10% a 90% sin condensación

Tabla 2.4 – c. Especificaciones del sistema del robot PUMA (continuación)

A continuación, en el capítulo tres, se encuentran varios softwares que se han desarrollado para el manejo del robot PUMA, para varios de sus modelos. Este robot resulta de interés para las universidades ya que, por su tipo de configuración, resulta muy didáctico para las clases de robótica, por lo que muchos de los programas fueron desarrollados por estudiantes y profesores de distintas universidades.

En el ITESM, también se han creado varias aplicaciones en el área de robótica, sin embargo nos centraremos principalmente en aquellas que se utilizan actualmente, como herramientas, para el manejo de los equipos de la celda flexible de manufactura del laboratorio, tanto las que se hicieron para los robots como para las máquinas de control numérico.

Capítulo 3
Softwares Para el Manejo del Robot Industrial PUMA Y Softwares para la Celda de Amatrol Desarrollados en el ITESM

3.1 Introducción

Los robots, concebidos en un principio como producto de la ciencia ficción, en la actualidad se han convertido prácticamente en equipo de trabajo. Las ventajas que se obtienen al emplear robots en distintas tareas los han hecho cada vez más populares, entre las principales ventajas se pueden mencionar: el evitar que el hombre se exponga a trabajos peligrosos o que pongan en riesgo su salud, la capacidad de repetir movimientos con gran precisión, con lo cual se pueden lograr productos con menos variabilidad en su calidad, además de no presentar cansancio o aburrimiento durante la jornada de trabajo, pudiendo realizar tareas monótonas y repetitivas durante mucho tiempo, por lo que cada vez es más común verlos en distintas aplicaciones.

Este crecimiento en las áreas de utilización de los robots industriales, ha hecho imperiosa la necesidad de crear paquetes computacionales que faciliten la programación de actividades para los robots, además de poder predecir el comportamiento del brazo antes de que éste entre en acción. Por medio de estos paquetes los programas de los robots se pueden probar y depurar en un modelo de simulación antes de ser ejecutados en el equipo mismo. La ventaja radica en el hecho que algún daño posible al robot o a los objetos alrededor de él por algún problema que surja por errores no detectados en el programa, se pueden evitar. Además contar con sistemas que permitan la programación del robot fuera de línea, es decir, sin la necesidad de estar conectado físicamente al sistema robótico, y la simulación, también tiene un gran sentido económico en fábricas automatizadas, se puede mantener el nivel de producción ya que el robot no debe ser parado cuando surge la necesidad de programarlo para otra tarea o de

hacer pruebas, no hay necesidad de interrumpir el trabajo del robot. Las trayectorias también pueden ser revisadas para detectar posibles colisiones y evitar daños severos al equipo, así como aumentar la seguridad en el trabajo. Por otra parte, el uso de los sistemas de simulación de robots también tiene aplicaciones importante en la investigación y en la educación. Los algoritmos de control nuevos pueden probarse antes de que el equipo esté disponible, y los estudiantes pueden aprender principios de robótica en sistemas de simulación menos caros y más seguros. Es por esto que la simulación de robots, junto con las celdas de trabajo de los mismos, son de los principales aspectos en el campo de la investigación robótica [17].

Las empresas dentro del giro de la robótica industrial, se han dado a la tarea de crear softwares nuevos que ayuden a las compañías manufactureras a simplificar la preparación de la estación de trabajo, reducir los tiempos muertos del robot y mejorar la exactitud para cumplir con tolerancias de manufactura reducidas. Existe un software ofrecido por la compañía Fanuc Robotics el cual presenta características directamente aplicables a la carga de maquinaria. Una opción de este paquete es el "Chuck Center Finder", con el cual se puede determinar el centro de una figura en dos dimensiones, como un círculo, un hexágono o un cuadrado. El centro localizado se puede utilizar como punto de referencia, para grabar la posición de carga en la prensa. Sin este módulo, muchas operaciones de carga y descarga requieren de preparaciones repetitivas para operar confiablemente. Otra opción es el "Collision Guard", éste reduce el daño que pueda causarse en un robot por colisión, sin importar la dirección del impacto. Utiliza el sistema servo del robot para predecir y darle seguimiento a el torque en los motores para detectar colisiones rápidamente. El módulo de "SoftFloat" compensa las variaciones en las piezas de trabajo en aplicaciones donde el robot se usa para cargar y descargar piezas directamente en adaptaciones fijas de la máquina, con esto se elimina la necesidad de dispositivos mecánicos complejos, su costo asociado y durabilidad limitada. El cuarto módulo es el sistema de visión robótica "visTRAC", con el cual se le da al robot una mayor flexibilidad en la alimentación de piezas a maquinaria, reduciendo el costo de las adaptaciones fijas y de los costos operativos cuando hay cambios de productos, además el robot puede hacer inspección durante el proceso [18].

En este capítulo se presentan una serie de programadores que se han realizado con el objetivo de facilitar el uso de los brazos manipuladores, algunos de ellos principalmente para actividades docentes y otros que se ofrecen para aplicaciones industriales. En la sección 3.2, se muestran ejemplos de los que se han hecho en universidades extranjeras, que nacieron como proyectos de un laboratorio, y algunos de ellos que ahora se ofrecen comercialmente para fines didácticos. En la sección 3.3 se mencionan algunas de las soluciones comerciales que, las compañías dedicadas a la robótica industrial, han puesto en el mercado para la industria manufacturera principalmente. Finalmente en la sección 3.4, se describen algunos de los softwares que se han desarrollado en el ITESM, tanto en el campus Monterrey como en otros campus del sistema, algunos de ellos con fines de investigación y otros con fines docentes, los cuales son utilizados actualmente en el laboratorio de Sistemas Integrados de Manufactura.

3.2 Softwares Realizados en Universidades Extranjeras

En muchas de las universidades de los Estados Unidos diversos departamentos imparten la materia de robótica, entre ellos están los de Ingeniería Industrial, Ingeniería Mecánica, Eléctrica, de Control, etc. por lo que no es extraño que gran parte de los trabajos que se realizan con los robots, y que además se encuentran documentados en la red mundial sean de universidades norteamericanas. En los siguientes apartados de esta sección se presentan algunos de ellos, los cuales están enfocados principalmente al robot PUMA 560 de Unimation, objeto de estudio de la presente tesis.

3.2.1 Simulador Gráfico Fuera de Línea del Robot PUMA 560 de la Universidad Estatal de Mississippi

Este es un sistema experimental de programación gráfica de un robot fuera de línea. El ambiente integrado de programación fue diseñado para el robot PUMA 560, para mostrar una simulación gráfica de las condiciones de trabajo, de manera que se pueda evaluar el desempeño del robot estando fuera de línea. El objetivo principal de este proyecto es diseñar un ambiente de simulación para programación fuera de línea y una interfaz gráfica para el

usuario en tres dimensiones que hiciera posible obtener los beneficios de la programación fuera de línea para un robot industrial PUMA 560. Este robot industrial fue elegido para este proyecto ya que es uno de los más utilizados en el mercado, pueden encontrarse en casi cualquier aplicación donde se requiera agilidad, precisión y fuerza en los movimientos de trabajo, además de ser muy utilizado con propósitos experimentales en distintas universidades y centros de investigación [19].

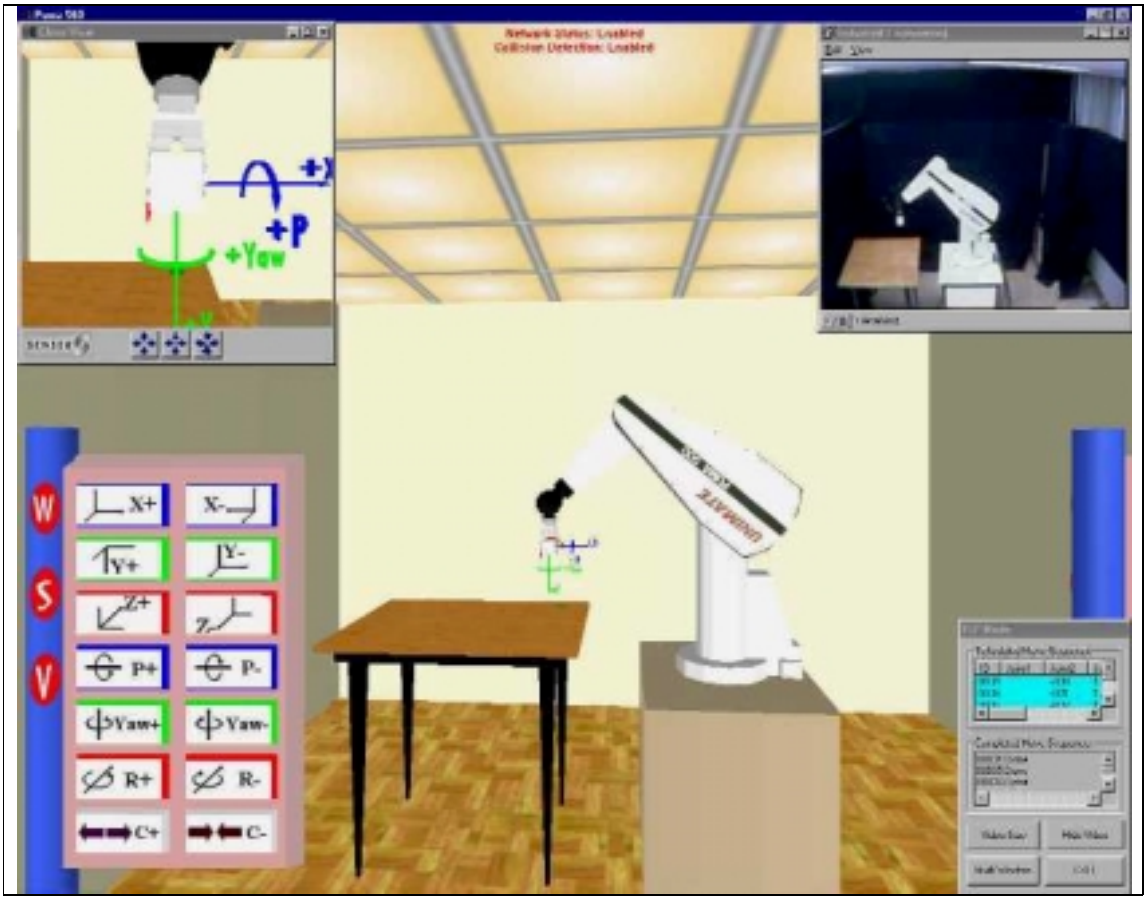


Figura 3.1 Pantalla Principal del Simulador Gráfico fuera de línea del robot PUMA 560 de la Universidad Estatal de Mississippi

Este simulador es capaz de calcular la cinemática directa del movimiento del robot, así como implementar la función de cinemática inversa. La pantalla gráfica e interfaz de usuario permiten al operador observar e interactuar efectivamente con la simulación del brazo. En la

Figura 3.1 se muestra una imagen de la pantalla principal de este simulador, donde se puede ver el modelo del robot actuando en una mesa de ensamble, como está en la fotografía de la parte superior derecha.

El sistema comprende las siguientes unidades:

1. Un módulo intérprete del lenguaje del robot, el cual interpreta la información de los programas fuente del robot.

2. Un módulo de simulación gráfico del ambiente virtual para simular los programas del robot y realizar gráficamente las operaciones en pantalla para pruebas y depuración.

3. Un módulo detector de colisiones, para detectar colisiones potenciales en el movimiento del brazo del robot.

4. Un módulo para depuración de programas, para brindar al programador del robot una herramienta para eliminación de errores cuando se esta programando.

5. Un módulo de comunicación entre el módulo intérprete del lenguaje del robot y el módulo de simulación gráfico del ambiente virtual.

Con estos módulos el sistema cuenta con las siguientes capacidades:

1. Capacidad de Visualización: El sistema puede mostrar los resultados de la ejecución de la secuencia de órdenes gráficamente. Además, en un futuro, se podrán utilizar lentes para tres dimensiones para enaltecer la representación gráfica del robot PUMA 560.

2. Capacidad de Edición de la Secuencia de Órdenes: El sistema presenta una interfaz amigable para la entrada, edición y eliminación de alguna orden.

3. Capacidad de Manejo de Archivos: El sistema permite al usuario abrir un archivo de secuencias de órdenes, guardado previamente.

4. Modo de Visualización Diferente: Por ejemplo, el usuario puede escoger entre ejecutar un programa de forma continua, se puede pedir al simulador que presente un cuadro a la vez.

5. Capacidad de Detección de Colisiones: El sistema advierte al usuario cuando se detecta una colisión. Posteriormente se presenta al usuario la identificación de la línea en la que ocurre dicha colisión.

Como se mencionó en la sección 2.5.1, lenguaje de programación del robot es el VAL, el sistema editor en el lenguaje del robot se implementó como un sistema editor- intérprete. Primero, el programa puede escribirse en la aplicación "Bloc de Notas" de Windows, el cual puede cargase dentro del simulador, y posteriormente el código es ejecutado por el intérprete línea por línea. Debido a que el número de palabras que acepta el software intérprete es fijo, no se requiere de estructuras de datos que permitan creaciones dinámicas, inserciones y recuperación de este tipo de información.

El módulo de simulación gráfica es el componente de visualización del sistema. Utiliza una herramienta de desarrollo rápido de prototipos virtuales llamada "WorldUp", el cual otorga un ambiente altamente orientado a objetos diseñado para acelerar el desarrollo de la aplicación. En su ambiente virtual del PUMA 560, se modelan cada una de las seis articulaciones del robot con gran precisión. El simulador permite que el área de trabajo sea vista desde distintos ángulos y escalas, para una mejor observación de la operación. Se tienen dos modos de operación: uno es el primitivo modo de control "Joint" y el otro es el modo de control de alto nivel "World". En el modo de control "Joint" el robot puede moverse en el sistema de coordenadas por articulación, mientras que en el "World" el efector final del robot se puede mover a lo largo de los ejes cartesianos X, Y, Z y su orientación se puede controlar con la manipulación de las variables O, A, T en el sistema. Un subsistema de detección de colisiones está acoplado al simulador para advertir de posibles errores en el programa simulado. Automáticamente se detiene la ejecución del programa, una vez que se detecta la colisión. La

pantalla de estado permite al usuario depurar el programa de manera interactiva con el simulador gráfico.

Los autores consideran que este sistema puede ser usado en un futuro por instituciones educativas para enseñar la programación de robots y para propósitos experimentales. Se le pueden agregar mas arreglos de celdas de trabajo al simulador para aumentar si aplicabilidad y flexibilidad [17].

3.2.2 Simulador de Propósito General para Robot Manipulador, de la Universidad de Illinois en Urbana- Champaign

El ambiente de este simulador esta pensado para pruebas de flexibilidad y la implantación de métodos de control de redes neuronales. El diseño permite que los distintos módulos del software sean ejecutados en paralelo en una red de estaciones de trabajo. Además el simulador permite al usuario definir arbitrariamente la posición de la cámara y visualizar el movimiento del robot en tres dimensiones.

La cinemática que emplea para la simulación está basado en el robot industrial PUMA 562, la cinemática directa se calcula por un programa llamado PEARL, el cual también maneja la comunicación con el controlador de la red neuronal. La visualización del brazo del robot se lleva a cabo con Geomview, un software de visualización en tres dimensiones de dominio público. El diseño modular del simulador permite a la cinemática del sistema, ser substituido por un robot real [20].

3.2.3 WinPuma, Controlador del robot Puma para MS- Windows, de la Universidad de Dakota del Norte

Este software fue desarrollado en la Universidad de Dakota del Norte en el verano de 1994. Es una aplicación para Windows que cuenta con un controlador en tiempo real así como un

simulador. Está construido completamente en C++. La versión de demostración se puede utilizar para brindar a los estudiantes la oportunidad de manejar un robot sin que corran el riesgo de daños personales o que maltraten el equipo [21]. En la Figura 3.2 se muestra una imagen de la pantalla de control principal, por medio de esta pantalla se pueden manipular las articulaciones con solamente arrastrar y soltar el ratón.

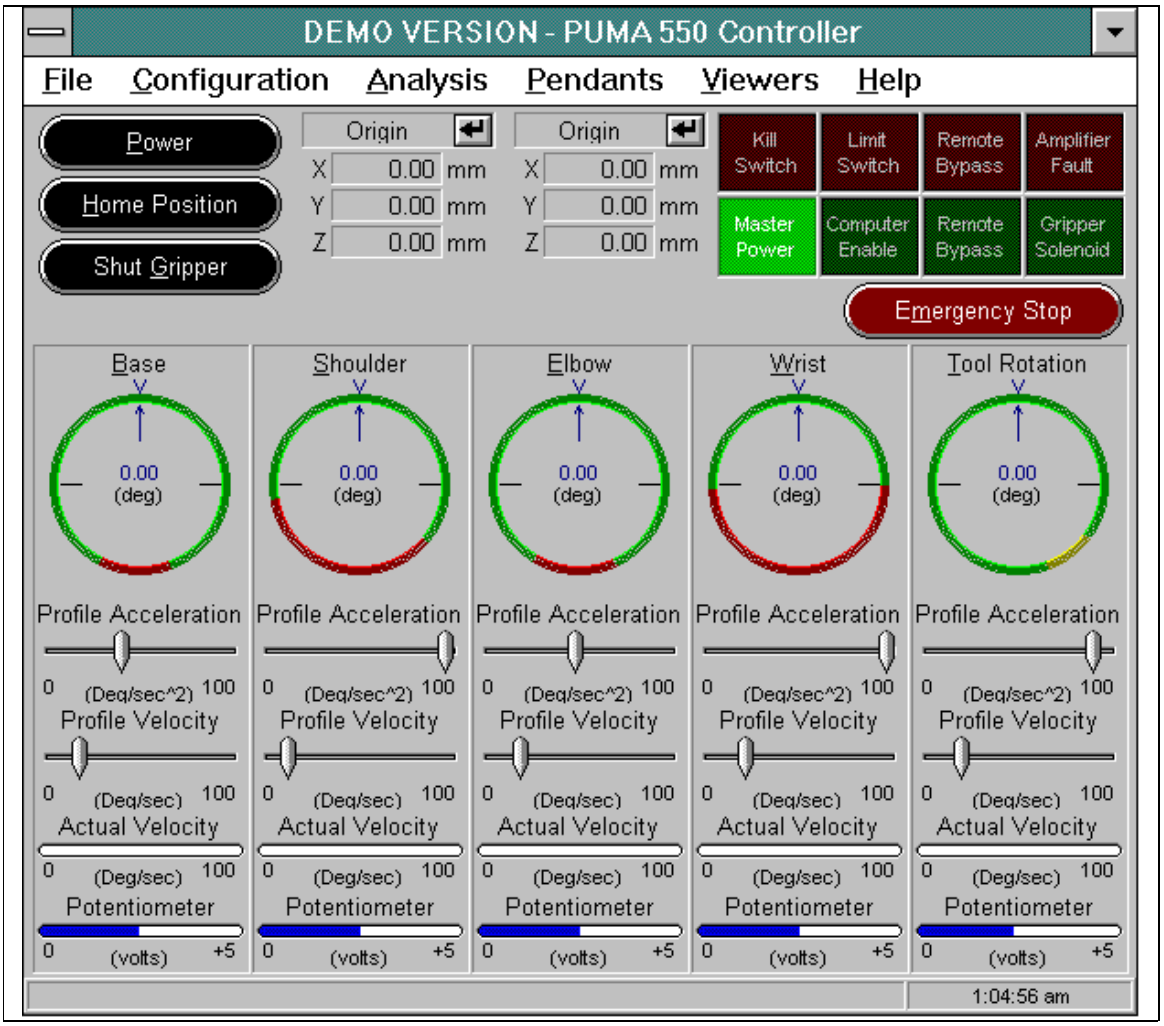


Figura 3.2 Pantalla de Control Principal de WinPuma

Esta interfaz está compuesta por controles en forma de brújula para modificar la posición de cada articulación, barras de desplazamiento horizontal para fijar la velocidad y aceleración del movimiento, medidores para indicar la velocidad de cada eje y el voltaje en los

potenciómetros, un botón de paro de emergencia, el cual funciona igual que el interruptor de paro de emergencia, y el usuario puede agregar campos para seguir las posiciones, fuerzas y torques. El robot en el que se basó el desarrollo de este programa es el robot PUMA 550, el cual cuenta con cinco grados de libertad, identificados en esta pantalla principal. Además cuenta con otras pantallas de utilidad que a continuación se mencionan:

- La Pantalla de Visión Cinemática en Tres Dimensiones, la cual se muestra en la Figura 3.3. Por medio de esta opción el usuario puede ver una representación en tres dimensiones del robot con sus movimientos reales, cuenta con dos controles para ajustar el ángulo de elevación y el ángulo de rotación sobre el cual se desea ver la simulación, además de cambiar el color y sombreado de los componentes.

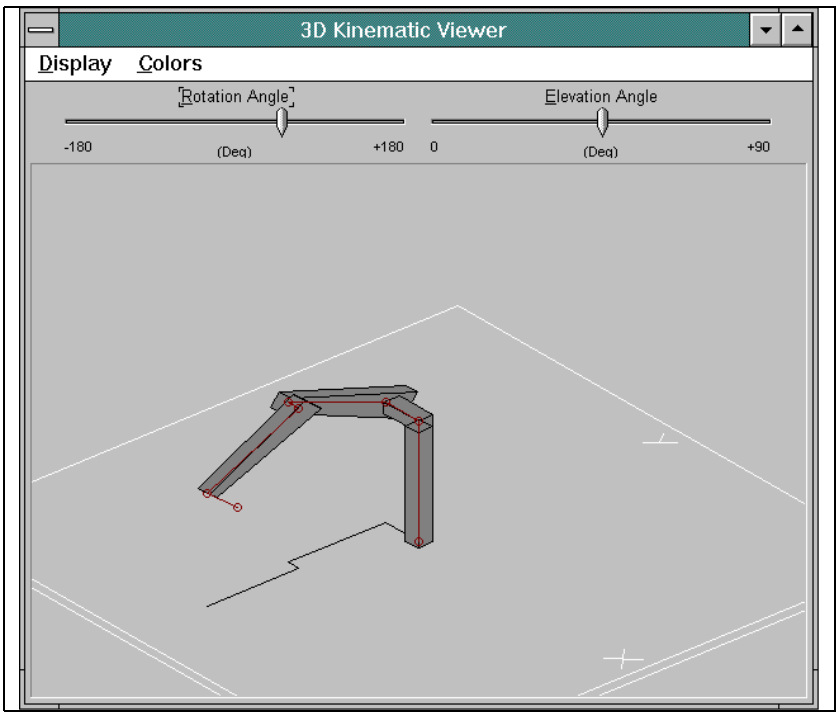


Figura 3.3 Pantalla de Visión Cinemática en tres dimensiones, de WinPuma

- La Pantalla de Perfiles de Posición y Velocidad. Esta pantalla permite al usuario ver una gráfica de las razones de cambio de la posición o velocidad de cada articulación con respecto al tiempo. Esta pantalla se puede ver en la Figura 3.4.

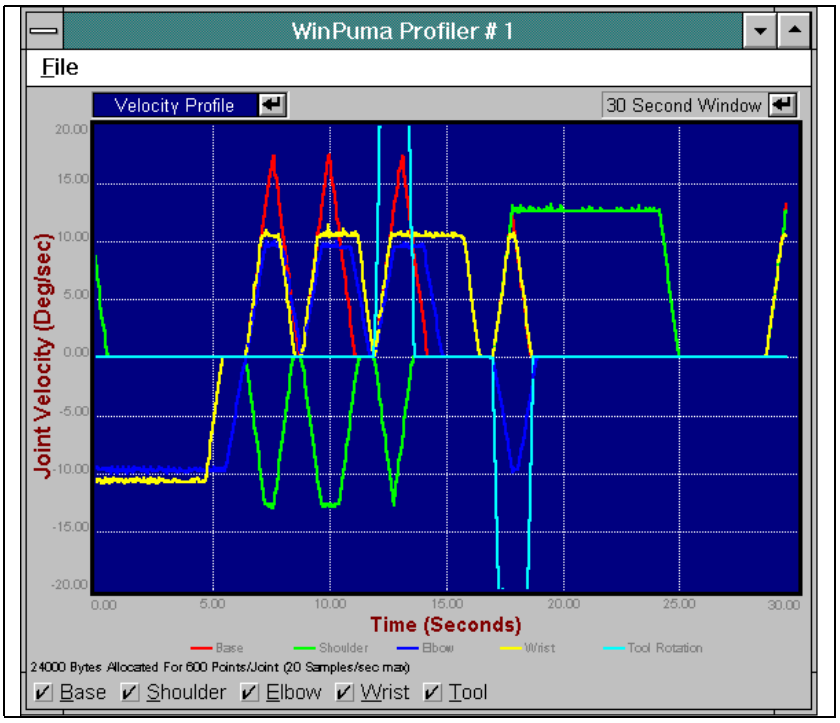


Figura 3.4 Pantalla de Perfiles de Posición y Velocidad, de WinPuma

- La Caja de Diálogo para Ajuste PID. Por medio de ésta se puede ajustar la ganancia de los controladores PID de cada articulación por separado.

- La Caja de Diálogo para Ejecución de Archivos. Se utiliza para correr archivos que contienen movimientos del brazo del robot o algún otro tipo de órdenes. Los archivos de pueden editar fácilmente porque se escriben en texto ASCII, los controles son semejantes a los de un reproductor de discos compactos y da la opción de realizar corridas múltiples e incluso infinitas. Esta caja de diálogo se muestra en la Figura 3.5

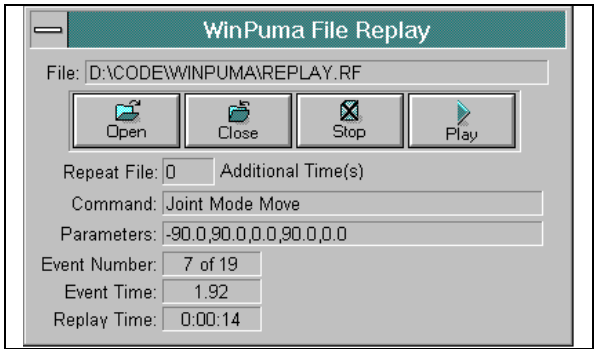


Figura 3.5 Caja de diálogo para Ejecución de Archivos

- La Caja de Diálogo para el Control Manual de las Articulaciones. Esta caja de diálogo, que se muestra en la Figura 3.6, realiza la misma función que el control manual alámbrico para el movimiento de las articulaciones. Permite controlar directamente la velocidad de los ejes y se puede controlar la magnitud de velocidad dependiendo de la posición del cursos sobre la flecha.

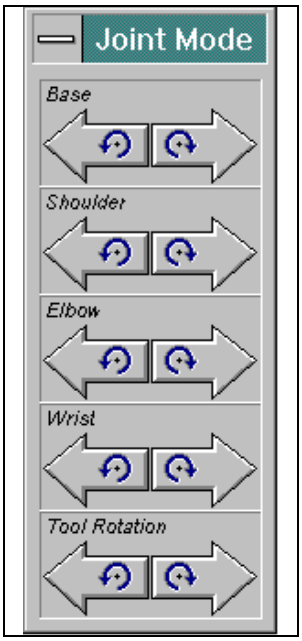


Figura 3.6 Caja de Diálogo para el Control Manual de las Articulaciones

3.2.4 Simulador para PUMA 560 de la Universidad de Bridgeport, en Connecticut

El propósito principal d este proyecto es brindar un software de simulación rápida a bajo costo para uno de los robots industriales más utilizados en el mercado. Considerando que el robot Puma 560 de Unimation puede encontrarse en casi cualquier ambiente de trabajo donde se requiere de agilidad, precisión y fuerza; así como también ser muy utilizado con propósitos experimentales, por sus altos estándares y servicio impecable, era el candidato idóneo para este desarrollo. Otra de las ideas principales, es la creación de un programa que, además de realizar la simulación y cálculos necesarios, pudiera correr en cualquier plataforma sin necesidad de compilar nuevamente el código. Por esta razón se escogió Java como la plataforma de desarrollo [22].

Este paquete, aunque funcional, aún no está completo, pues aún falta el módulo de cinemática inversa, además la generación de imágenes en tres dimensiones tiene sus defectos, por

ejemplo, en algunas posiciones el modelo se ve distorsionado. Por otra parte la simulación no fue posible hacerla cuadro por cuadro, ya que las estaciones de trabajo en la que se hicieron las pruebas no tienen la velocidad suficiente para la generación de las figuras, por lo que se cambió para solamente representar los cambios en las posiciones del robot, es decir, mostrar su posición inicial y luego la posición final. Tomando en cuenta que no todos los usuarios potenciales tengan intérpretes de Java en sus sistemas, en un futuro esperan poder distribuirlo por la red desarrollando un código mediante el cual se pueda correr en sistema operativo y que también pueda ejecutarse desde el Netscape o el Explorador de Internet.

3.2.5 Simulador Fuera de Línea para el Robot PUMA 760, de la Universidad de Rochester

En la Universidad de Rochester, en Nueva York, a cargo de Martin Jägersand, se desarrolló un simulador visual para brazos robóticos, e identificaron el modelo del motor visual para un robot PUMA 760. Los movimientos del robot pueden controlarse por articulaciones y también en los modos de coordenadas cartesianas absolutas ("World"), y relativas enclavadas en la herramienta del robot ("Tool"). El simulador planea las trayectorias del robot, las velocidades y aceleraciones de los movimientos. Además cuenta con una ayuda para que el usuario pueda calcular y graficar también las fuerzas requeridas en los motores [23].

3.2.6 El Simulador Robótico GRAS, de la Universidad de Portland

El simulador robótico GRAS fue desarrollado por Chris Lattner de la Universidad de Portland en febrero de 1998, y aún sigue estando vigente este proyecto. GRAS es un Simulador y Animador Gráfico para Robots, su nombre se deriva por sus iniciales en inglés: "Graphical Robotic Animator and Simulator". Sirve para hacer simulaciones de los robots PUMA y Pegaso para su manipulación, esto es programable y ajustable por el usuario por medio de archivos de datos. Realiza los cálculos de cinemática directa e inversa [24]. En la Figura 3.7 se muestra una imagen de la pantalla principal, con la animación del robot PUMA.

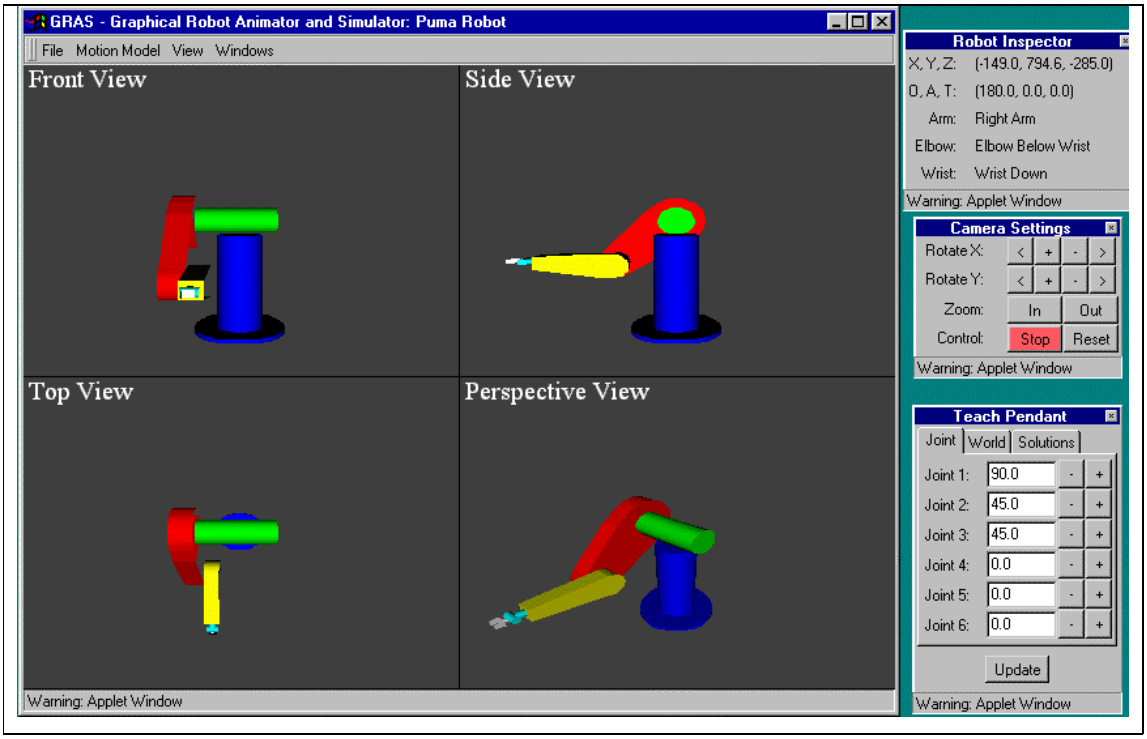


Figura 3.7 Pantalla principal del simulador robótico GRAS

Este simulador para brazos mecánicos está escrito en Java para tener una mejor portabilidad del programa y que sea fácil de usar. Fue diseñado como un marco de trabajo extensible para simulación de robots, y como tal, sigue creciendo y se le siguen intercalando nuevas ideas en su programación. En general presenta las siguientes características:

- La simulación del robot se presenta en tiempo real con un modelo en tres dimensiones.
- Se pueden especificar configuraciones robóticas y geometrías arbitrarias, por medio de un archivo donde se especifica la descripción del robot.
- Implantación completa de la cinemática directa en modelos de robots arbitrarios.
- Se puede disponer de la cinemática inversa si se cuenta con el archivo "class" correspondiente para ese robot específico.
- Es posible la visualización de modelos múltiples de movimiento, dando diferentes modos de interpolar los ángulos de las articulaciones.

- El lenguaje Java le da una completa portabilidad para todos los sistemas que cuenten con Java JDK [25].

Principalmente este software fue diseñado como herramienta para la docencia, muchas universidades ofrecen cursos de robótica, sin embargo, la complejidad de la geometría que pueden llegar a tener, hace que su enseñanza en un salón de clase no sea fácil. Además debido al alto costo de estos equipos y al mantenimiento que estos requieren, muchas escuelas enseñan robótica sin un robot real. Por lo que el uso de este simulador permite el desarrollo de una clase de robótica en un salón, y aún para las universidades que tienen robots resulta conveniente ya que se pueden realizar experimentos antes de estar físicamente manipulando y programando el equipo, y también permite que varios estudiantes trabajen con un robot, virtual, al mismo tiempo. [25] y [26].

En la simulación, se tiene una representación de un control manual alámbrico, el cual se puede apreciar en la Figura 3.7 en la parte inferior izquierda de la pantalla, con éste se puede interactuar de dos maneras, una es por medio del ratón al hacer clic en alguna de las articulaciones, por medio de las teclas marcadas con los signos "+" y "-", moviendo cada una por separado en un incremento. Otra es especificando todos los valores de las articulaciones, y el resultado sería la simulación de un movimiento del brazo de un punto a otro ya programado.

3.3 Softwares Comerciales

3.3.1 Equipo de Herramientas Robóticas "Simulink" para el Puma 560, de QRTS ("Quality Real Time Systems, Simulink Robotic Toolkit for the Puma 560")

El SRTK para el Puma 560, por sus siglas en inglés, es un conjunto de archivos .m de MATLAB, cajas de herramientas de MATLAB, y diagramas de bloque de Simulink que pueden utilizarse para simular en tiempo real, controlar, y registrar información para este robot manipulador. La compañía que lo vende se llama "Quality Real Time Systems", y fue fundada en 1999 con el propósito de crear soluciones de bajo costo y en tiempo real para

aplicaciones de control de alto rendimiento. Actualmente se enfocan al desarrollo de ambientes de software y dispositivos para la adquisición de datos y el control [27].

La filosofía del RTK es que con un ambiente amigable, basado en MATLAB y Simulink, habilitado por una interfaz gráfica de usuario (GUI) basada en MATLAB permite a usuarios con distintos antecedentes, explotar la funcionalidad de estas aplicaciones para realizar investigaciones de robótica en el Puma 560. Esto es, los esfuerzos de muchos tipos de investigadores se ven favorecidos por el hecho que el RTK explota la naturaleza modular de los diagramas de bloque de Simulink y la funcionalidad del amplio rango que ofrecen las cajas de herramientas de MATLAB. Por ejemplo, un investigador con experiencia en teoría de control, puede simplemente quitar el bloque de control del RTK de Simulink y sustituirlo por uno desarrollado por el usuario para tener un nuevo algoritmo de control. De igual manera, los investigadores relacionados con problemas de visión del robot o de generación de trayectorias pueden modificar fácilmente el diagrama de bloques de Simulink para comprobar sus conceptos de investigación sin tener que dedicar tiempo a aspectos que no están relacionados con su trabajo principal, como crear la interfaz al hardware del Puma, entre otros.

La estructura del RTK se basa en un enfoque por capas, en el cual, la capa superior es la Interfaz Gráfica para el Usuario (GUI, por sus siglas en inglés). La GUI permite que usuario aproveche la funcionalidad del RTK sin que tenga que trabajar con el diagrama de bloques de Simulink. Basándose en esta estructura por capas, el usuario puede integrar fácilmente equipos adicionales, incorporar funcionalidad adicional o modificar el algoritmo de alguna función existente. El algunos casos, el MATLAB GUI requiere ser modificado para reflejar la funcionalidad agregada por el usuario. Debido a que la GUI esta hecha por archivos script de MATLAB, el usuario puede crear su propia GUI o modificar la del RTK. En el manual de usuario se describe el diagrama de bloques del Simulink y se explica la manera de como agregar o quitar diferentes modos de operación o componentes auxiliares, principalmente se divide en tres secciones: la sección de entradas, la sección de cálculos y la sección de salidas.

Para la instalación del RTK se requiere contar con Windows 98, MATLAB, Simulink, Real Time Workshop (RTW) y Real Time Windows Target (RTWT). También debe hacerse un

cambio en el controlador original de Unimate para conectar el Puma 560 a una de las tarjetas de entrada y salida para poder operarlo con el RTK, el diseño de esta conexión se describe en la página de QRTS o se puee comprar un controlador ya modificado a Tident Robotics. Para el control de la posición se debe tener instalado ya sea la tarjeta de ejes STGII-8, de Servo To Go, Inc. o la MultiQ2, de Quanser Consulting. Para el control de la fuerza se debe contar con la sensor de Fuerza y toque multiaxial de Automatización Industrial de ATI y la tarjeta interfaz de la misma empresa.

Los diferentes modos de operación que se incluyen en la sección de Cálculos del diagrama de bloques de Simulink incluyen:

- **Calibración.**

El modo de calibración es el más vital porque calibra el hardware del Puma 560. Si el robot no está calibrado correctamente los otros modos del RTK no funcionan bien, ya que la información que recibe de los sensores no es válida. El modo de calibración llama a una serie de operaciones para determinar el ángulo de cada articulación del brazo. Específicamente las articulaciones del Puma 560 tienen un potenciómetro que da una aproximación del ángulo por medio de un voltaje de 0 a 5 voltios al RTK dependiendo de la inclinación de la articulación. Basado en la información que envía el potenciómetro, el RTK permite que el controlador mueva cada articulación del brazo al indicador de pulso más cercano de los codificadores. Una vez que se llega a este indicador, se obtiene la posición exacta del Puma y el valor del codificador para cada articulación se establece en el RTK, por lo que, cada vez que se cierra esta aplicación, se debe realizar la operación de calibración.

- **Gravedad Cero.**

El modo de gravedad cero se utiliza para la enseñanza de diferentes configuraciones del manipulador al RTK, permitiendo al usuario mover fácilmente el manipulador en el espacio por su propia mano. Una vez que el Puma es colocado en la posición deseada, la posición de cada articulación puede guardarse para usarlo posteriormente.

- **Control por Articulación.**

Este modo es uno de los más básicos para controlar el Puma 560, en éste, el usuario especifica el ángulo que desea para cada articulación del manipulador y un parámetro de escala para la velocidad del movimiento del manipulador. Cuando se ejecuta, el RTK lee la posición actual del brazo y genera una trayectoria para permitir que todas las articulaciones lleguen a la posición deseada al mismo tiempo. Un controlador Proporcional Derivativo (PD) utiliza el error entre la trayectoria deseada y la trayectoria real y las derivadas respectivas del tiempo para mover el actuador a la configuración deseada. Una ventaja de este modo es que el control PD no presenta singularidades ya que el control se lleva a cabo a nivel de articulación. Sin embargo, las características de seguridad del RTK inhabilitan el control por articulación si el controlador intenta forzar al Puma a una posición en la que sobrepase el límite de alguna articulación. Al final del movimiento las articulaciones se mantienen en la posición establecida.

- **Modo externo.**

El modo externo, a diferencia de los modos de movimiento anteriores, da al usuario la posibilidad de generar una trayectoria usando archivos script de MATLAB. Un controlador proporcional derivativo de articulación se utiliza cuando el RTK opera en este modo para forzar al manipulador a seguir la trayectoria definida por el usuario.

- **Control Cartesiano.**

El modo de control cartesiano se utiliza para mover al Puma de cualquier posición en el espacio cartesiano a cualquier posición deseada en el mismo espacio. En este modo el RTK genera la trayectoria y es escalada por el parámetro definido por el usuario. Un control PD se utiliza para que el Puma siga la trayectoria. La cinemática directa se calcula en línea. Los tres últimos grados de libertar no se toman en cuenta, se consideran fijos. En este modo de operación sí pueden presentarse singularidades durante la trayectoria.

- **Control Cartesiano PD.**

El modo control cartesiano PD es similar al modo de control cartesiano, ambos mueven al Puma de una posición a otra dentro del espacio cartesiano. Una diferencia en el cartesiano PD es que la cinemática inversa se calcula en línea para relacionar la trayectoria deseada en el espacio cartesiano al espacio por articulación. Un control proporcional derivativo se calcula en base a la diferencia de la posición y velocidad en el espacio por articulación actual con la posición y velocidad en el espacio por articulación deseado. Otra diferencia es que en este modo el usuario sí puede seleccionar un valor deseado para los tres últimos grados de libertad.

- **Control por Impedancia.**

El modo de control por impedancia es muy similar al modo de control cartesiano, pero este modo brinda una capacidad adicional de retroalimentación de fuerza por un sensor auxiliar de fuerza y torque incorporado al Puma. Este modo puede usarse en aplicaciones donde es necesario que el brazo del robot siga una trayectoria deseada mientras se regula la relación de movimiento y fuerzas de contacto. Puede presentar algunas singularidades.

- **Modo de mezcladora.**

En este modo el usuario tiene la posibilidad de dar una lista, de hasta siete posiciones deseadas para el manipulador, en lugar de tener que dar uno por uno. Basándose en esta lista de posiciones deseadas, el RTK calcula una trayectoria deseada en el espacio por articulación permitiendo una transición suave entre los puntos.

El simulador del Puma brinda al usuario el medio para ver una representación gráfica en tiempo real el movimiento del brazo, basado en el control de entrada de un modelo dinámico no lineal. Ésta es una herramienta útil para determinar si la entrada de control del torque contiene singularidades, resulta en un movimiento articulado que excede el límite mecánico del manipulador o detectar movimientos impredecibles que puedan ocurrir. En este modo, se

requiere que el usuario especifique el ángulo deseado en cada articulación o la posición cartesiana del efector final, y el parámetro de escala que establece la velocidad de movimiento. Cuando es ejecutado, el RTK lee la posición actual de cada articulación y genera la trayectoria para que las articulaciones lleguen a la posición deseada al mismo tiempo. El movimiento resultante del robot se despliega en la GUI. [28].

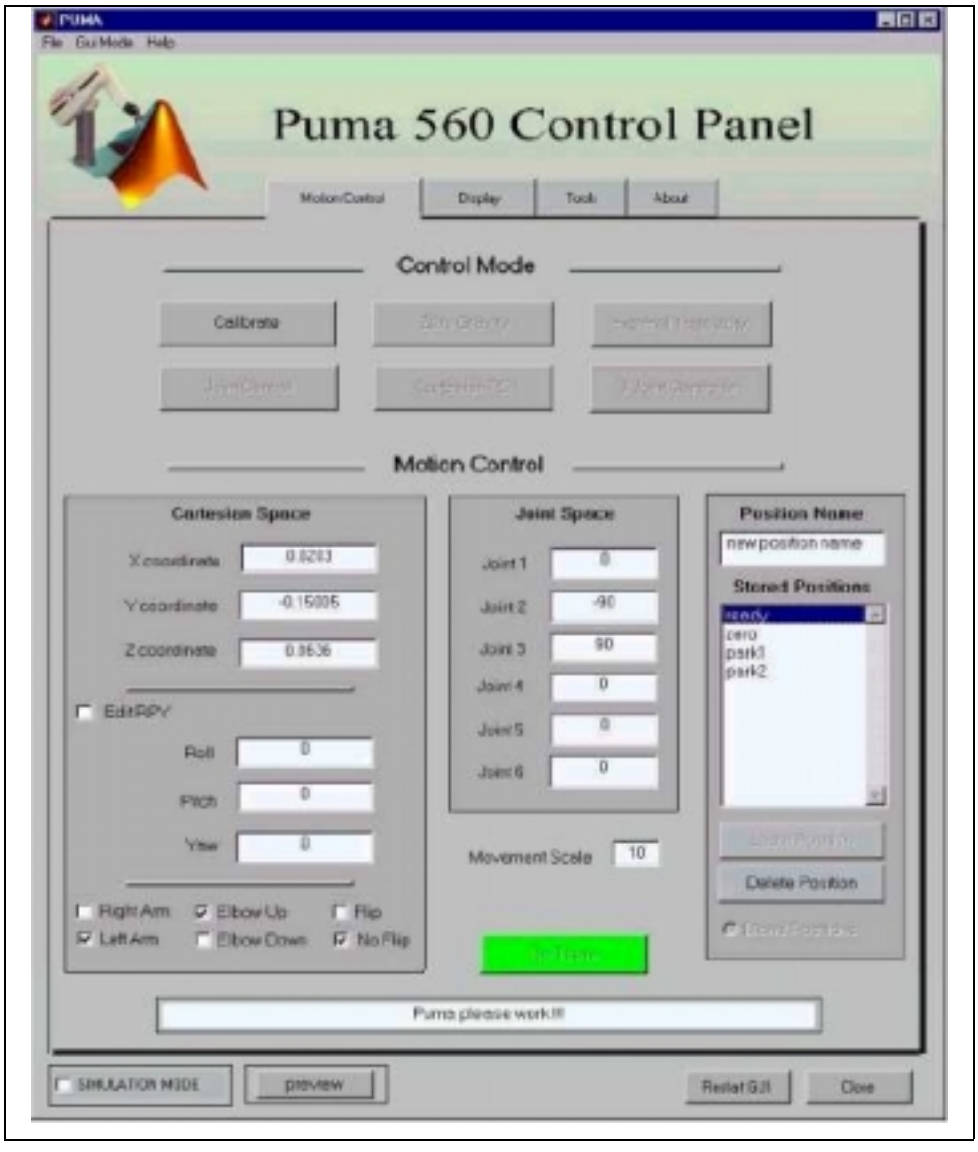


Figura 3.8 Pantalla “Control de Movimiento” del Panel de Control de RTK

La GUI está organizada de tal manera que el usuario puede ejecutar los modos de operación mencionados fácilmente. Esta interfaz, denominada como “Panel de Control del Puma 560”, está formada por cuatro pestañas: Control de Movimiento, Despliegue de Imagen, Herramientas e Información.

La pestaña de control de movimiento se muestra en la Figura 3.8. Esta sección, permite al usuario controlar los movimientos del manipulador. Los botones en la parte superior son para seleccionar el modo de operación deseado, los cuales se mencionaron anteriormente. Las cajas de texto, las de opción y los botones que se muestran en la sección de Control de Movimiento permiten al usuario recibir información acerca de la posición del Puma, la cual puede utilizarse para mover al robot a un punto específico.

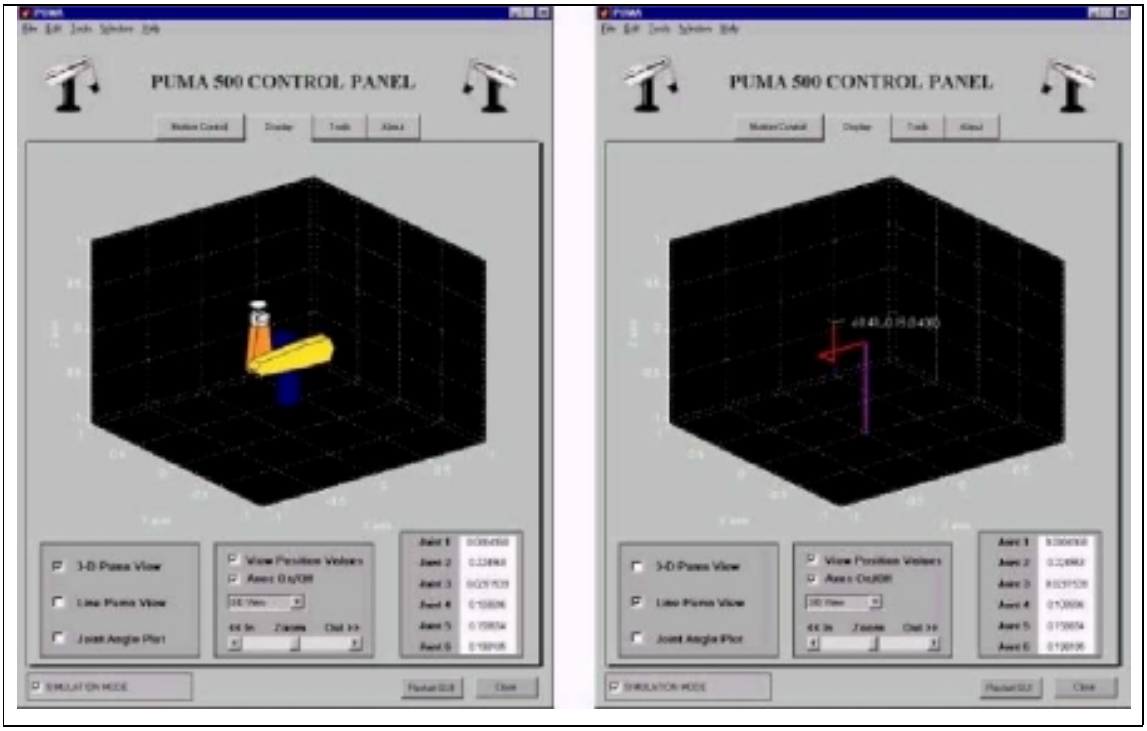


Figura 3.9 Pantallas de “Despliegue de Imágenes” del Panel de Control de RTK. A la izquierda la vista en tres dimensiones y a la derecha la vista unifilar

Sin importar en qué pestaña se encuentre el usuario, la casilla de verificación del Modo de Simulación, y los botones de Vista Preliminar, Re inicio y Cerrar, siempre quedan en la parte inferior de la pantalla.

La pestaña de Despliegue de Imágenes muestra una de las dos pantallas que se muestran en la Figura 3.9, éstas le dan al usuario una representación gráfica de la configuración del manipulador. La apariencia del manipulador en la imagen puede cambiarse seleccionando la vista en tres dimensiones, unifilar o la gráfica de los ángulos de las articulaciones. Al hacer clic en la casilla de verificación para ver los valores de posición, se despliegan las coordenadas en X, Y e Z del manipulador cuando el usuario está en la vista unifilar.

La pestaña de Control de Herramientas permite al usuario el uso de las líneas digitales de la tarjeta de entradas y salidas para controlar los accesorios que pueden estar instalados en el robot. Finalmente, la cuarta pestaña muestra la información acerca del RTK y de los creadores del software [28].

3.4 Softwares Realizados en el Instituto Tecnológico y de Estudios Superiores de Monterrey

En el ITESM también se han desarrollado algunas aplicaciones para manufactura y para robótica, tanto en el campus Monterrey como en otros campus. En esta sección se mencionan algunos de ellos, principalmente los que se utilizan en la Celda Flexible de Manufactura Amatrol que se encuentra en el edificio anexo de manufactura del CETEC, torre sur, como apoyo para el Laboratorio de Sistemas Integrados de Manufactura. Sin embargo, cabe mencionar que también se han hecho otros programas con otros propósitos o que no se utilizan actualmente en este laboratorio como es el caso del “Simulador y Controlador de Posición de un Manipulador tipo PUMA” y del “Simulador del Sistema Automático de Almacenamiento, de Carga y Descarga 862 de Amatrol”

El Simulador y Controlador de posición del PUMA fue un proyecto que se desarrollo en el Centro de Inteligencia Artificial, en 1998, junto con un proyecto similar para un motor sincrónico empleado para guiar un AGV (Autonomous Guided Vehicle). El objetivo de este trabajo fue diseñar y programar un simulador para un robot tipo PUMA, así como un controlador difuso, un adaptable y un PID para su control, además de un simulador para un motor sincrónico con un sistema de control. Todo se realizo en Java para poder ejecutarse con algún navegador de Internet [29].

El Simulador del Sistema Automático de Almacenamiento, de Carga y Descarga 862 de Amatrol fue diseñado como un proyecto de tesis en el departamento de Ingeniería Industrial, para simular la programación manual de este sistema. Éste consta básicamente de un traductor, programado en lenguaje C++, del código de programación manual del sistema AS/RS, al código requerido por el programa de animación Proof. A través de la ejecución del programa Proof, el usuario puede apreciar gráficamente la ejecución del programa escrito. El objetivo de este proyecto es la creación de una herramienta que permita capacitar a los usuarios del AS/RS, antes de que estos lo utilicen directamente [30].

3.4.1 Softwares Incorporados a la Celda de AMATROL Realizados en el Campus Monterrey

En el departamento de Ingeniería Industrial y de Sistemas del campus Monterrey, se han desarrollado dos aplicaciones para la Celda Flexible de Manufactura de Amatrol, y que se encuentran actualmente incorporados dentro del funcionamiento de la misma, estas son el Programador del Robot Júpiter XL [31] y el Programador del robot ASRS 862. Ambos equipos son de Amatrol, por lo que tienen varias características en común, como es el lenguaje de programación ACL y el manejo de archivos de códigos numéricos para la descarga de programas, entre otros. A continuación se presentan cada uno de ellos, así como el funcionamiento general de cada programador.

3.4.1.1 Programador del Robot Júpiter XL de Amatrol

El objetivo principal de este trabajo fue desarrollar un sistema de cómputo fuera de línea para programar el robot Júpiter XL. Este robot, parte de un módulo flexible de la celda de Amatrol, se presentó en la sección 1.1. De acuerdo a la clasificación de robots es de configuración SCARA, el manipulador no cuenta con un efector final fijo, sino que cuenta con tres herramientas: dos manos del tipo "gripper" y un destornillador. El lenguaje de programación recibe el nombre de Lenguaje Automatizado de Control (ACL, por sus siglas en inglés).

Este programador del robot Júpiter XL, realizado por el Ing. Leopoldo Eduardo Cárdenas Barrón, se hizo con la intención de reducir el tiempo de realización de un trabajo en la estación de ensamble y disminuir el número de paros por golpes accidentales. Anteriormente, los usuarios de esta estación de trabajo eran alumnos de la carrera de Ingeniería Industrial y de Sistemas y de Ingeniería Mecánica. Frecuentemente tenían problemas para manipular al robot, especialmente en cuanto al aprendizaje del lenguaje del robot y en el proceso de enseñanza de puntos. En este último aspecto se tenía que era difícil para ellos mover el robot en sus cuatro ejes para colocarlo en posición para tomar alguna herramienta. Los puntos para tomar y dejar las herramientas eran los más críticos porque frecuentemente se exponía al robot a colisiones.

Este programador fue hecho en lenguaje Turbo C++ 3.0 de Borland, y se basó en la teoría de redes de Petri Ordinarias para su funcionamiento. Actualmente es una herramienta computacional que apoya completamente el concepto de manufactura integrada por computadora generando beneficios al equipo y a los usuarios, además ya es un módulo más en el software AMNET, el cual es el que se utiliza para el manejo de la Celda Flexible de Manufactura de Amatrol [31].

En la Figura 3.10 se muestra la pantalla principal de este programador, el funcionamiento básico es bastante sencillo ya que en esta pantalla se muestra una lista de órdenes que se pueden programar con solamente oprimir la tecla de la función correspondiente a la función que se desea, posteriormente se introduce el parámetro correspondiente, si es que lo necesita, y por medio de archivos que cuentan con las instrucciones necesarias para realizar esa

operación, se va escribiendo el programa del robot, tanto en códigos ACL, como en códigos numéricos que son los que posteriormente se envían, por medio de una descarga de archivos, al controlador del robot.

Figura 3.10 Pantalla Principal del Programador del Robot Júpiter XL de Amatrol

En caso que se presente la necesidad de programar una operación que no se encuentra contemplada en este menú del programador, se tiene la opción identificada como programación individual (F8), mediante la cual se puede introducir uno por uno los códigos, en lenguaje ACL, que se requieran solo que es necesario que la persona que este haciendo el programa conozca los comandos. Normalmente, en una práctica del Laboratorio de Sistemas Integrados de Manufactura se tienen las opciones suficiente y necesarias para llevarla a cabo.

3.4.1.2 Programador del Robot AS/RS 862 de Amatrol

El programador del robot AS/RS 862 de Amatrol, también fue realizado por el Ing. Cárdenas tiempo después de haber hecho el programador del Júpiter. Su formato y modo de operación para el usuario es muy similar, sin embargo, fue necesario la actualización de las opciones en el menú, y de los archivos que contienen tanto los códigos en lenguaje ACL, como los archivos con los códigos numéricos, que también son los que se descargan en el controlador del Almacén. De igual manera se encuentra integrado como un módulo más al software de control de la celda AMNET y también fue programado en lenguaje C++. En la Figura 3.11 se muestra la pantalla principal de este programador.

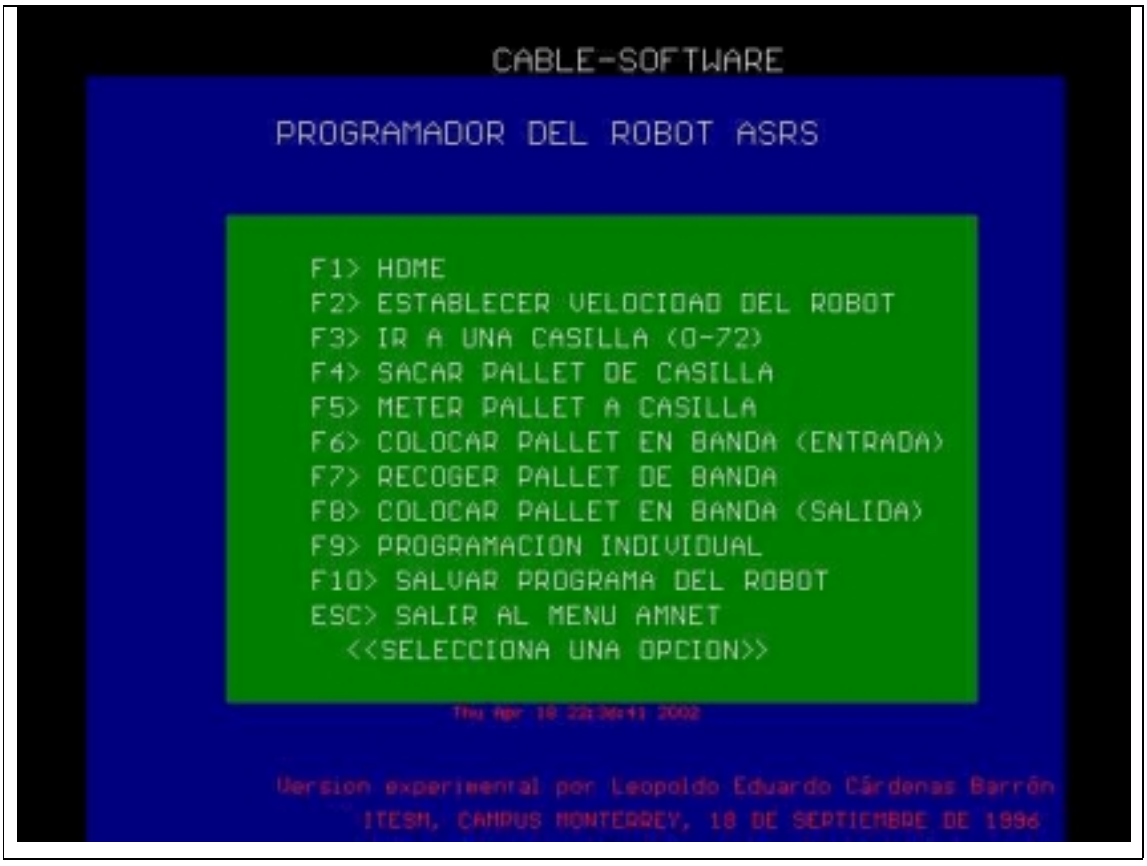


Figura 3.11 Pantalla Principal del Programador del Robot AS/RS 862 de Amatrol

El robot AS/RS, el cual fue presentado en la sección 1.1, es de configuración cartesiana, cuenta con cuatro grados de libertad, y un efector final en forma de "gripper" con el cual sujeta los contenedores de material que debe de meter y sacar de la banda transportadora de la celda de manufactura. Además tiene un armazón de acero con 72 casillas donde se colocan los contenedores antes y después de pasar por la banda.

Este programador reduce en gran medida el tiempo de programación de actividades del robot AS/RS, sobre todo cuando se trata de hacer programas de manejo de contenedores de material cuando no se está conectado con la computadora central de la celda. Las posiciones de las 72 casillas ya se encuentran grabadas previamente, por lo que los movimientos entre ellas resultan muy cómodas para el usuario, pues solamente se indica el desplazamiento hacia alguna de ellas y posteriormente se le ordena meter o sacar el contenedor, según se requiera. También cuenta con la opción de programación individual para algunos casos especiales en los que se necesite programar alguna operación que no se encuentra contemplada en el menú principal de opciones, sin embargo, para las prácticas de Laboratorio de Sistemas Integrados de Manufactura se tienen las opciones necesarias.

3.4.2 Softwares Incorporados a la Celda de AMATROL Realizados en Otros Campus

Para la realización de las prácticas del Laboratorio de Sistemas Integrados de Manufactura, también se cuentan con dos aplicaciones que fueron desarrolladas en otros campus del ITESM, estos son el Simulador de Máquinas de Control Numérico, del Campus Querétaro y el Programador para el Robot Mitsubishi, del Campus Estado de México. Ambos programas, aunque no están diseñados para correr dentro de el ambiente del AMNET, resultan de gran utilidad para el desarrollo de las prácticas.

3.4.2.1 Simulación de Máquinas de Control Numérico, del Campus Querétaro

El software para Simulación de Máquinas de Control Numérico se utiliza principalmente para simular los programas que se van a ejecutar en la fresadora del laboratorio. El programa fue

desarrollado como un proyecto de tesis, está escrito en lenguaje de programación Pascal y corre bajo sistema operativo DOS.

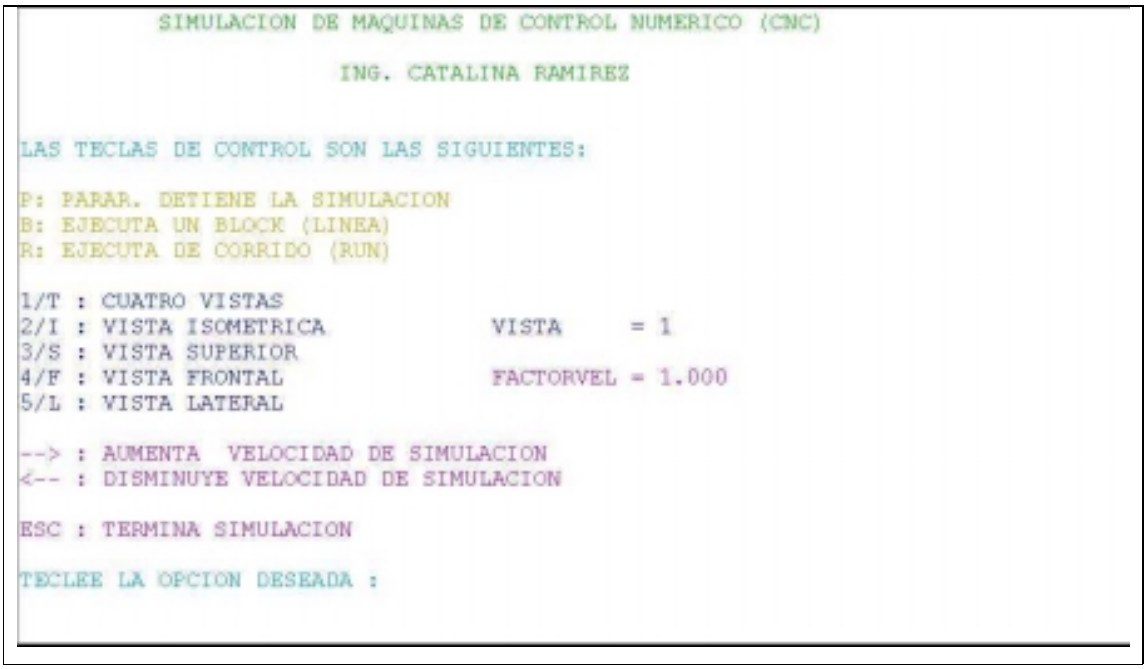


Figura 3.12 Pantalla Principal del Simulador de Máquinas de Control Numérico

El simulador realiza una compilación previa del programa para la fresadora, escrito en códigos G y M, para revisar posibles errores en la codificación del mismo, como códigos inexistentes, omisión de la declaración de la velocidad de avance o radios demasiado pequeños que no alcancen para trazar el arco que se solicitó en el programa. Posteriormente pide al usuario las medidas del bloque que se va a maquinar, en sus tres dimensiones X, Y e Z, y muestra la pantalla principal, la cual se muestra en la Figura 3.12. En esta pantalla se tienen las opciones para la ejecución de la simulación, por medio de las cuales se puede solicitar la ejecución del programa línea por línea, todo seguido o detener la simulación. Además también se puede escoger cuál de las cuatro vistas es la que se desea ver en pantalla: isométrica, superior, frontal o lateral, o las cuatro al mismo tiempo. El factor de velocidad hace que la simulación se corra en tiempo real, si éste es igual a uno, o en un tiempo menor a medida que el factor va aumentando, siendo en la mitad del tiempo si éste es igual a dos, y así sucesivamente hasta llegar a dieciséis.

En la Figura 3.13 se muestra la pantalla de ejecución de un programa para la fresa. En ésta pantalla están activas las cuatro vistas, y en cada una de ellas se puede ver la trayectoria que va siguiendo la herramienta de corte. En la parte inferior se tiene la información general de la corrida del programa, como es el tiempo total de la corrida, las coordenadas actuales del centro de la herramienta, la línea del programa que se encuentra ejecutando, la velocidad de avance, el número de herramienta que está utilizando, el modo de coordenadas, el estado del líquido de enfriamiento, el estado de la puerta y la mordaza, y finalmente, el factor de velocidad al cual realiza la simulación.

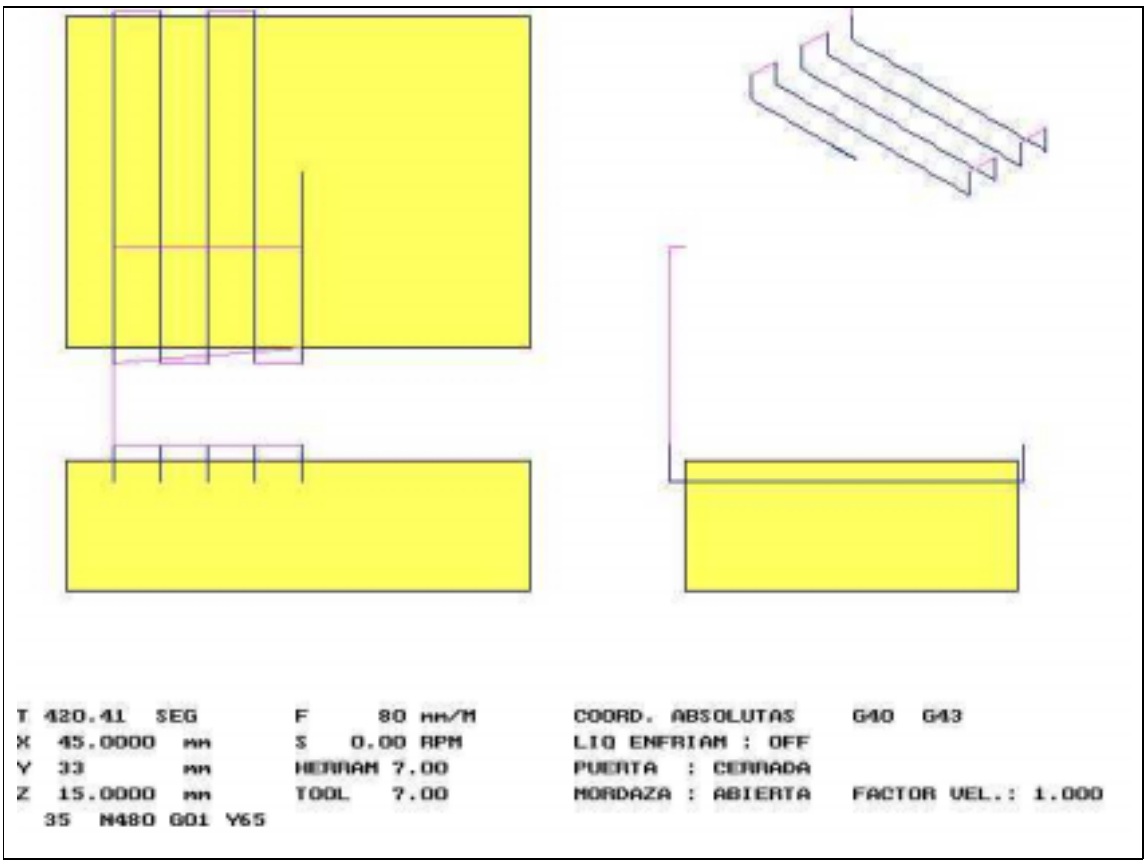


Figura 3.13 Pantalla de Ejecución del Simulador de Máquinas de Control Numérico.

Este programa resulta de gran utilidad para simular los programas de los alumnos antes de ser ejecutados en la fresadora, previniendo así accidentes que pueden detectarse de antemano,

como son maquinados en las mordazas, o movimientos de posicionamiento rápido cuando la herramienta aún se encuentra dentro del material que se quiere maquinar.

3.4.2.1 Programador para el Robot Mitsubishi, del Campus Estado de México

El Programador para el Robot Mitsubishi, desarrollado en el Campus Estado de México en 1994, es la primera aplicación que se utiliza en esta celda que está hecho en ambiente Windows. El lenguaje en el que se programó fue en Visual Basic. En la Figura 3.14 se muestra la pantalla principal de este programador.

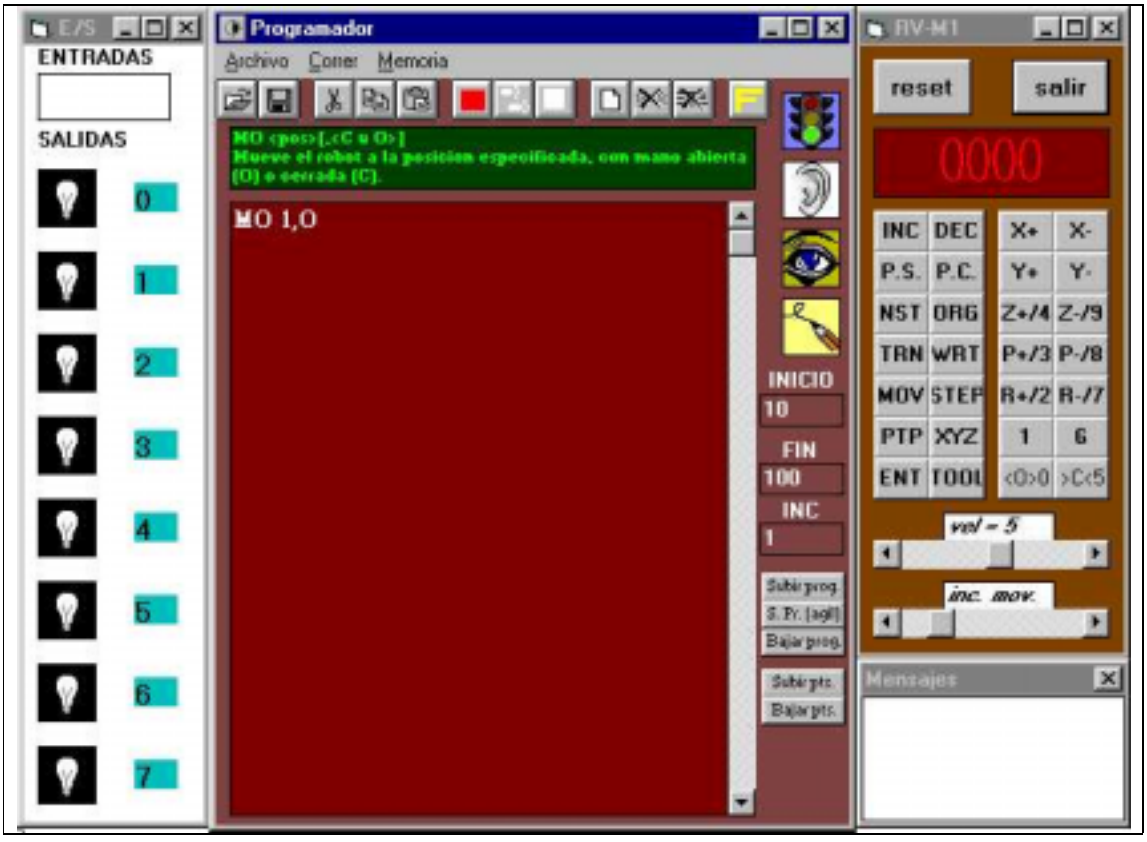


Figura 3.14 Pantalla Principal del Programador para el Robot Mitsubishi.

Este programador, consta de cuatro secciones principalmente: del lado izquierdo se tiene la pantalla para el manejo de señales de salida, al centro está la pantalla del programador, del lado superior derecho se tiene una réplica de la caja de enseñanza de este robot, y en la parte inferior derecha está la pantalla de mensajes para el usuario.

En la pantalla para el manejo de señales de salidas se tienen hasta siete señales, las cuales se pueden encender haciendo doble clic con el botón izquierdo del ratón sobre el foco que corresponda al número de señal que se encuentra del lado derecho del mismo. Para apagarla basta con hacer doble clic nuevamente sobre el foco encendido de la señal que se desee desactivar.

En la pantalla del programador se tiene un menú de programa el cual permite al usuario realizar las operaciones básicas para la carga y descarga de información, como es borrar la memoria del robot, abrir un archivo con un programa o con la información de una serie de puntos y enviarlos al controlador del robot, o viceversa, subir la el programa o los puntos que se encuentren en el robot para salvarlos en un archivo. Además en esta pantalla, en el área roja, se pueden teclear órdenes en el lenguaje del robot, tal como se escriben en un programa, para que éste las ejecute. En esta misma área, se puede utilizar como editor de textos para la escritura de un programa para el robot.

La réplica de la caja de enseñanza en la pantalla, permite al usuario la operación del brazo, tal como funciona la caja real, solamente que para activar algún botón es necesario hacerlo con el cursor del ratón y hacer clic en la opción deseada. La pantalla de mensajes es el medio por el cual el robot se comunica con el usuario para notificarle de una posible desconexión con el controlador, que se está ejecutando un programa o una instrucción, o la activación de algunos de los errores ya sea de software o de hardware.

Tomando en cuenta las ventajas que se obtienen de los softwares que se presentan en este capítulo y la experiencia de los instructores del laboratorio, se desarrolló la Interfaz de Usuario para el robot PUMA 560 de Unimation, la cual se presenta a detalle en el siguiente capítulo.

Capítulo 4
Interfaz de Usuario para el Robot PUMA 560 de Unimation

4.1 Introducción

En este capítulo se describe el desarrollo y funcionamiento de la Interfaz de Usuario para el Robot PUMA 560 de Unimation, el cual fue programado en el lenguaje Visual Basic 6.0 de Microsoft [32]. Para que corra adecuadamente se necesita contar con una computadora personal IBM 486 o compatible, con puerto serial identificado como "comm1", que cuente con Windows 95 o alguna otra versión más reciente y que la resolución del monitor sea de 1024 por 768 pixeles como mínimo.

Durante el desarrollo de la interfaz se realizaron varias pruebas de comunicación de la terminal con el controlador. En la sección 4.2, se explica como se realizaron las pruebas necesarias para encontrar la manera en la que se comunica normalmente el controlador del robot con su terminal. Una vez que se había descifrado la manera como se daba el flujo de información se procedió a determinar cuáles órdenes debían contemplarse para ser incluidas en la Interfaz de Usuario. En la siguiente sección se muestra a detalle esta interfaz, se explica cada uno de los botones de orden así como todas las pantallas que salen durante su operación. Se tiene un Panel Principal, para el manejo general del sistema, un Editor de Programas, que permite la escritura de programas nuevos y/o modificación de los ya existentes y un Modo Terminal, por medio del cual la surge una pantalla que emula el funcionamiento de la terminal original del sistema.

Finalmente en la sección 4.4 se mencionan las opciones disponibles cuando se usa el programa no estando en línea con el robot. La opción más atractiva es que se puede contar con el editor de programas, por lo que el usuario puede escribir un programa en cualquier computadora que tenga el software instalado, sin necesidad de estar en la terminal del sistema.

4.2 Comunicación con el Controlador del Robot PUMA 560

Como se mencionó en la sección 2.5.3, el protocolo de comunicación que utiliza el controlador para mandar información a la terminal es el estándar RS232, por lo que el Analizador de Protocolos "Monitor 232" de "Black Box Corporation" se empleo para descifrar qué información es la que se necesita enviar para que el controlador ejecute las instrucciones enviadas desde la terminal. Los parámetros que se ajustaron en el Analizador de Protocolos para hacer las lecturas son las que se especifican en la tabla 2.4.

La forma como se conecta el Analizador de Protocolos se muestra en la Figura 4.1, en ella se puede ver que el Analizador cuenta con un cable en forma de "T", el cual se coloca como un puente entre el emisor y el receptor de los equipos que se quieren analizar. En este caso este cable se colocó al final del cable que sale del controlador y antes de entrar a la terminal del sistema. De esta manera el Analizador muestra en su pantalla la información que fluye entre los dos equipos conectados, es decir, de la terminal al controlador, y también del controlador a la terminal [32].

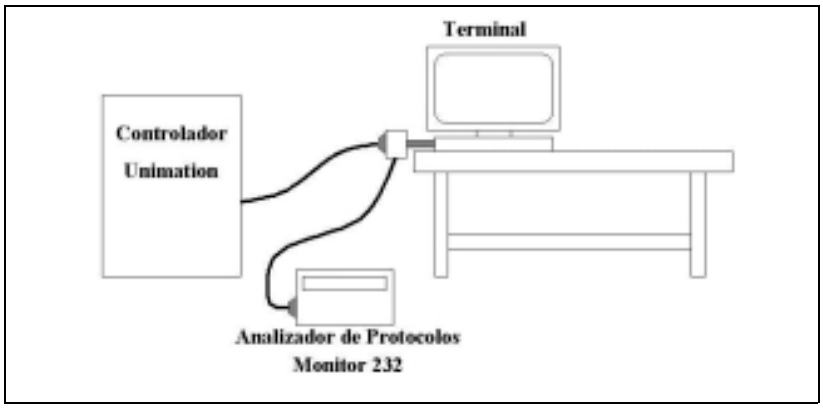


Figura 4.1 Conexión del Analizador de Protocolos entre la terminal y el controlador del sistema del robot PUMA

Gracias a este aparato se pudo determinar que las instrucciones tecleadas en la terminal son enviadas caracter por caracter hacia el controlador. Por cada uno de ellos (letra o número) que se teclea, el controlador responde con el mismo caracter, el cual es mostrado en la pantalla de la terminal. Cuando finalmente se oprime la tecla de entrada de datos "Return", se envía dicho

caracter (# 13 de la tabla ASCII, denominado como “retorno del carro” o “carriage return”, “Cr”), y el controlador responde con el mismo caracter seguido de un caracter de “avance de línea” o “line feed”, “Lf”. Éstos también se “imprimen” en la pantalla de la terminal y el robot inicia la ejecución de la instrucción deseada. Al terminar, el controlador envía el caracter “.” (punto) para indicar que ha terminado la realización de la orden y que esta listo para recibir la siguiente. En la Figura 4.2 se muestran dos ejemplos de la información que se puede ver en la pantalla del Analizador de Protocolos cuando se teclean instrucciones en la terminal y son enviadas al controlador para su ejecución. Los comandos son dos de los más comunes, uno es DO READY, el cual sirve para mandar al robot a su posición de inicio, y el otro es DO MOVE P1, mismo que se utiliza para enviar al brazo a una locación grabada previamente, en este caso a la posición identificada como P1.

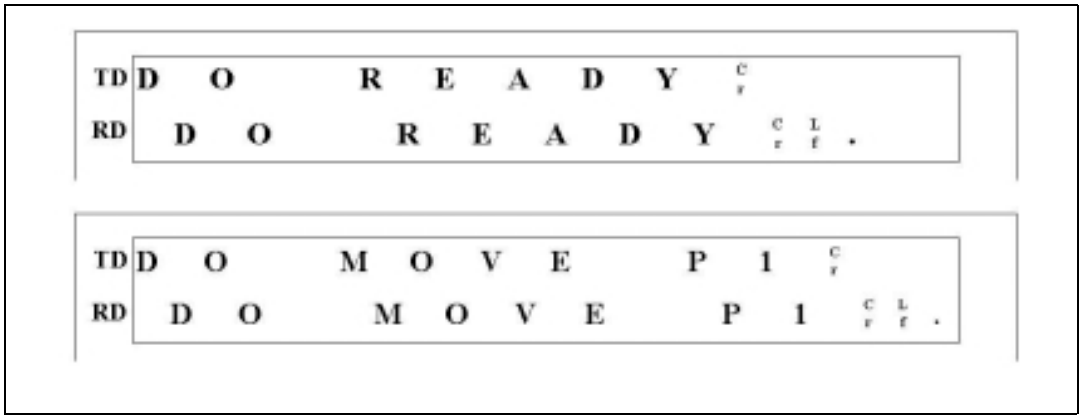


Figura 4.2 Pantalla del Analizador de Protocolos mostrando la información enviada de la terminal al controlador del robot PUMA. En la parte superior cuando se envía la instrucción DO READY, y en la parte inferior la orden DO MOVE con su parámetro P1.

Tomando como base esta forma de comunicación entre la terminal con el controlador, se realizó el código del programa para hacer que las instrucciones típicamente utilizadas en las prácticas de laboratorio, para manipulación del brazo, se pudieran enviar de una forma más automatizada, que no fuera necesario escribir todo el comando, sino que fuera una opción en un botón de orden y solamente se tuviera que hacer clic sobre dicho botón con el ratón y luego teclear solamente el parámetro correspondiente para las instrucciones que así lo requieren. Los comandos seleccionados se explican a detalle en la sección 4.3.1.

Por otra parte también hay otras instrucciones que nos dan información acerca del brazo o que se encuentra almacenada en la memoria del controlador, como son los comandos de "STATUS", "WHERE" y "LISTL", entre otros. En la sección 4.3.1 se explica la función de cada uno de ellos. Cuando éstos son tecleados en la terminal, el controlador responde con la información solicitada, la cual también es recibida e impresa en pantalla por la terminal, para finalizar, también se reciben los caracteres "Cr", "Lf" y ".", en este orden.

4.3 Elementos Principales de la Interfaz de Usuario para el Robot PUMA 560 de Unimation

La Interfaz de Usuario para el robot PUMA 560 tiene como objetivo principal, ser una herramienta de ayuda en el Laboratorio de Sistemas Integrados de Manufactura, por lo que para este fin docente se han incluido solamente las instrucciones típicas para la realización de una práctica. Esto con la intención de auxiliar a los alumnos de la carrera de Ingeniería Industrial y de Sistemas en el manejo del sistema del robot, tanto en la manipulación del brazo como en la escritura de programas. Los elementos principales con los que cuenta son:

- Panel de Control del Robot PUMA
- Editor de Programas
- Modo Terminal

En las siguientes secciones se explica el contenido y el funcionamiento de cada uno de estos elementos.

4.3.1 Panel de Control del Robot PUMA

El Panel de Control del Robot PUMA es la primer pantalla que el usuario ve una vez puesto a correr el programa, y es en ella dónde se encuentran las instrucciones típicamente utilizadas para la operación general del manipulador en una práctica de laboratorio.

Figura 4.3 Panel de Control de la Interfaz de Usuario para el Robot PUMA 560 de Unimation

Esta pantalla es la principal del software, en ella se pueden hacer operaciones como grabar un punto, mover el brazo a ese punto, enviar el robot a HOME, enviar un programa o la información de una lista de puntos de un disco al controlador o recibir esta misma información, del controlador para ser almacenado en un disco, ya sea duro o flexible, iniciar la ejecución de un programa por el robot, entre otras. Además aquí también es donde se habilita el acceso a las otras pantallas, como la del Editor de Programas y la del Modo Terminal. En la

Figura 4.3 se muestra esta pantalla, y se puede ver que los botones de orden están agrupados dependiendo de su función. A continuación se explica el sentido de cada uno de estos grupos, el funcionamiento de los botones que los conforman y las pantallas que surgen como respuesta del controlador. Los comandos que se ejecutan se definen de acuerdo a los manuales de programación del robot PUMA [34] y [35].

4.3.1.1 Información del Controlador

El grupo de botones que se encuentra en la parte superior izquierda del Panel de Control, con el título "Información del controlador", se refiere a instrucciones que muestran en pantalla, información del sistema robótico que se guarda en el controlador, o que permiten la eliminación de algún dato que ya no se desee tener almacenado. Los botones que conforman a este grupo son:

- **Status**

Al hacer clic en este botón, se envía la instrucción "STATUS" al controlador la cual muestra en pantalla la información de estado del sistema para el programa de usuario que vaya a ser ejecutado. En ella se incluye el modo de operación del robot (manual o por computadora), la velocidad establecida por el Comando referente a la velocidad de monitor, el número de ciclos completos ejecutados por un programa y los ciclos remanentes. En la Figura 4.4 se muestra un ejemplo del mensaje desplegado por este comando.

Figura 4.4 Mensaje que se obtiene al hacer clic en el botón "Status"

- **Posición Actual**

El botón de Posición Actual despliega las coordenadas de la localización de la herramienta del robot en coordenadas cartesianas y en grados de cada articulación, junto con un valor de 0.00 si el efector final se encuentra cerrado o de 25.41 si éste se encuentra abierto. Esta respuesta corresponde a la del comando "WHERE" del lenguaje VAL, el cual es enviado al controlador al hacer clic en el botón de orden. En la Figura 4.5 se muestra esta pantalla con la información de las coordenadas de la mano en su posición de HOME, con el gripper cerrado.

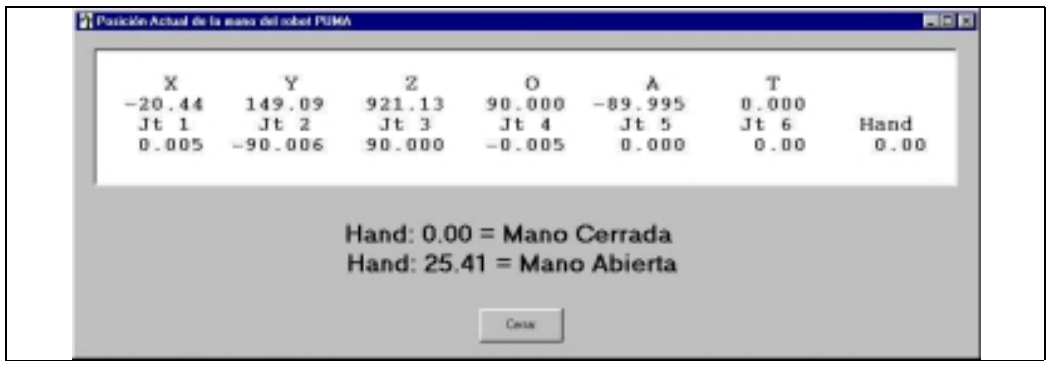


Figura 4.5 Ejemplo de la pantalla con la Posición Actual de la mano del robot PUMA

- **Lista de Puntos**

Este botón envía el comando "LISTL" al controlador del robot, recibiendo los valores de las posiciones grabadas en orden alfabético de acuerdo al nombre con el que se hayan grabado. Esta lista de posiciones con sus coordenadas se despliegan en una pantalla como la que se muestra en la Figura 4.6. La información que se presenta en la cuadro de texto representa las coordenadas cartesianas (X,Y,Z) que definen al punto almacenado, y los ángulos de orientación de la herramienta con respecto a los ejes cartesianos (O,A,T) o los ángulos de cada una de las articulaciones (Jt1, Jt2, Jt3, Jt4, Jt5, Jt6) si el punto fue grabado como de precisión. Esta información no puede ser editada desde el teclado de la computadora, sin embargo puede seleccionarse la información que deseada, copiarla y pegarla en cualquier editor de textos. En

caso que la cantidad de locaciones sobrepase la capacidad de renglones del área de texto, se pueden usar las barras de desplazamiento.

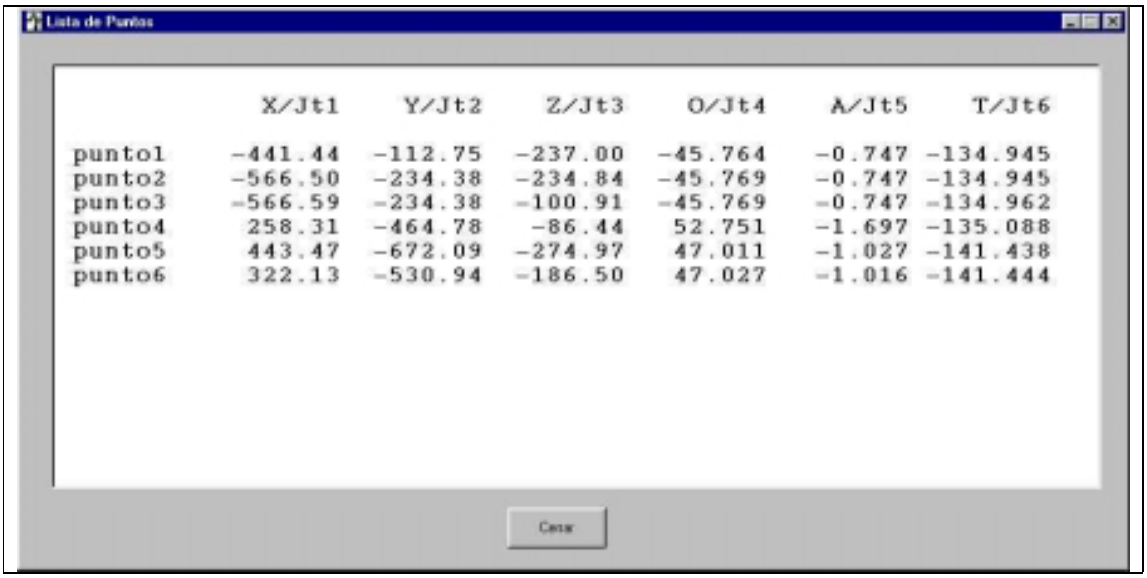


Figura 4.6 Pantalla que se despliega para mostrar la lista de puntos grabados en el controlador.

- **Lista de Programas**

El botón con esta leyenda despliega un cuadro de mensaje similar al de la Figura 4.4, pero muestra los nombres de los programas de usuario almacenados en la memoria del sistema que por lo menos tengan una línea definida. Esta información es el resultado de enviar el comando "DIR" al controlador.

- **Borrar un Punto**

Este botón se utiliza para eliminar locaciones grabadas en el controlador, por medio del comando de VAL "DELETEL". Al hacer clic sobre este botón se despliega una cuadro de entrada de datos para que el usuario escriba el nombre de la locación, o de las locaciones (separadas por comas) que desea borrar. Esta operación se puede usar para recuperar espacio en memoria ocupado por variables que ya no se necesitan. Una vez que una locación es eliminada ya no puede ser llamada por alguna otra función. Si se intenta mover el brazo a una

locación borrada se recibe un mensaje de error como si la variable nunca hubiera sido definida.

- **Borrar un Programa**

Al hacer clic en este botón, el programa busca en el controlador los nombres de los programas grabados en él, y los despliega en un cuadro de listado. El usuario escoge el programa que desea eliminar y por medio de la instrucción de VAL "DELETE" se borra completamente el programa de usuario. Esto es, borra el programa y todas las subrutinas, locaciones de puntos y variables reales que el programa elegido haga referencia, directamente o indirectamente a través de las subrutinas, siempre y cuando no sean llamadas por otro programa que se encuentre aún en el sistema.

4.3.1.2 Manipulación del Brazo

Este grupo de botones se encuentra en la parte media izquierda del Panel de Control, y los botones que lo conforman se refieren a comandos que de alguna forma afectan el movimiento físico del brazo manipulador, ya sea en la preparación del movimiento o que ordenen el movimiento mismo. La explicación de cada botón se presenta a continuación:

- **Calibrar**

El proceso de calibración es muy importante cada vez que se inicia un trabajo con el robot PUMA 560, ya que ajusta los sensores de posición de las articulaciones en el robot. Esta operación, en este modelo de robot, se puede hacer en cualquier posición en la que se encuentre el brazo, ya que solamente moverá un poco todas sus articulaciones.

Una vez que se da clic en el botón para calibrar, se envía el comando de VAL "CAL" al controlador, este responde con una cuadro de mensaje conteniendo una pregunta para el usuario, verificando que se encuentre seguro de querer realizar la operación. Para lograr que la

operación de calibración se lleve a cabo, se debe responder afirmativamente haciendo clic en el botón “Si” (“Yes”).

Esta instrucción, para ser ejecutada, requiere que el brazo cuente con energía eléctrica en sus motores y que esté activo el modo de “COMP” para que la terminal actúe como controlador del robot. En caso de que alguna de estas condiciones no se cumpla se desplegarán cuadros de mensajes señalando el error, como los que se muestran en la Figura 4.7, hasta que la energía del brazo sea activada y que el sistema se encuentre en el modo de computadora.

Figura 4.7 Mensajes de error desplegables por el controlador del robot. A la izquierda un mensaje notificando al usuario que el brazo no tiene energía. A la derecha notificando que la terminal no ha sido habilitada para control del robot.

- **HOME**

Este botón de orden envía al controlador la sentencia de VAL “DO READY”, mediante la cual se mueve el brazo a su posición de inicio, colocando el efector final en la parte superior de su volumen de trabajo. No importa donde se encuentre el robot, esta instrucción siempre se ejecuta, por lo que se debe tener cuidado que el brazo no encuentre obstáculos en su camino a esta posición.

Esta instrucción también requiere que el brazo cuente con energía eléctrica en sus motores y que esté activo el modo de “COMP”. En caso de que alguna de estas condiciones no se cumpla se desplegarán mensajes de error como los mostrados en la Figura 4.7, hasta que la energía del brazo sea activada y que el sistema se encuentre en el modo de computadora.

- **Velocidad Monitor**

La Velocidad Monitor es definida por el comando de VAL "SPEED" acompañado del valor deseado. Esta instrucción especifica la velocidad de todos los movimientos subsecuentes del robot bajo el control de un programa. El valor puede ser definido mediante una constante, una variable real o una expresión aritmética, y puede ir desde 0.39 la cual es muy lento, hasta 12800 que es extremadamente rápido, donde el valor de 100 es la velocidad "normal". La velocidad a la que se mueve el robot esta relacionada con el producto de la velocidad establecida por esta orden de monitor y la velocidad establecida por una instrucción de programación. Por ejemplo, si la velocidad monitor se fija en 50 y la velocidad establecida en un programa es de 60%, el robot se moverá aproximadamente al 30% de su velocidad normal.

Al hacer clic en este botón, el usuario escribe la velocidad que desea establecer por medio de un cuadro de entrada y posteriormente se envía la instrucción completa al controlador del robot. Automáticamente aparece en esta caja de entrada la velocidad de 30, la cual es la recomendada para hacer los primeros movimientos del robot, por lo que la instrucción VAL completa que se envía es: "SPEED 30"

- **Grabar Punto**

Este botón de orden se utiliza para definir el valor de la transformación o del punto de precisión igual a la localización actual del robot y ser guardado en el controlador bajo un nombre.

Al hacer clic en este botón, el usuario escribe el nombre que desea darle a esa locación por medio de un cuadro de entrada y posteriormente se envía la instrucción completa al controlador del robot, formada por el comando VAL "DO HERE", y el nombre del punto. En caso que el nombre tecleado no sea válido se reciben cuadros de mensaje señalando el error, semejantes a los que se muestran en la Figura 4.7

- **Mover a Punto**

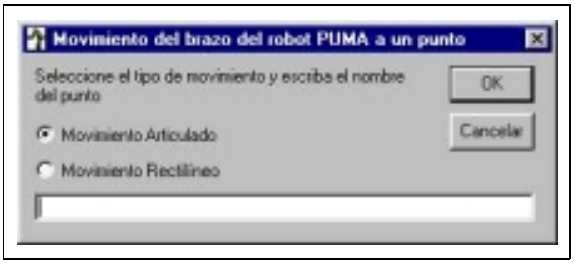


Figura 4.8 Cuadro de entrada de datos para el movimiento del brazo a un punto almacenado en el controlador

Por medio de esta instrucción el robot se mueve a una posición grabada en el controlador previamente, ver apartado anterior "Grabar Punto", de acuerdo a las coordenadas y a la orientación como se haya grabado. Si se selecciona el movimiento articulado, se calculan puntos intermedios entre la posición inicial del robot y la final, interpolando las posiciones inicial y final de las articulaciones, resultando este tipo de movimiento. Si se selecciona el movimiento rectilíneo, el efector final se desplaza a lo largo de una línea recta y va girando ligeramente hasta alcanzar la posición deseada.

Al hacer clic en esta opción, se despliega un cuadro de entrada de datos, como la que se muestra en la Figura 4.8, y el usuario debe seleccionar el tipo de movimiento que desea y teclear el nombre de la locación o punto al cual quiere mover el brazo del robot. Finalmente se envía al controlador la instrucción "MOVE <punto>", donde <punto> es el nombre de la locación, si se desea un movimiento articulado, o se envía la instrucción "MOVES <punto>", si desea un movimiento rectilíneo.

- **Abrir / Cerrar Clamp**

El botón que de inicio se muestra como "Cerrar Clamp", en realidad se trata de una casilla de verificación, de estilo gráfico. Una vez que el usuario hace clic en esta opción se envía al controlador la instrucción de VAL "CLOSEI", con la cual, el efector final o gripper, se manda cerrar instantáneamente, y la opción en la casilla cambia a "Abrir Clamp".

Al hacer clic sobre esta nueva opción, se envía al controlador la instrucción “OPENI”, con la cual el gripper se abre instantáneamente y la opción en la casilla cambia a “Cerrar Clamp”, nuevamente.

4.3.1.3 Señales de Salida

Este grupo está formado solamente por un botón, mediante el cual se despliega una nueva pantalla en la que se presentan ocho botones, cada uno de ellos corresponde a una señal de salida que se envía al controlador por medio de la instrucción de VAL “SIGNAL” acompañado del número de señal. Los números de señales positivos encienden la señal correspondiente, los números de señal negativos apagan esa señal.

Esta pantalla está compuesta por casillas de verificación de estilo gráfico, con apariencia de botón de orden, a la derecha se describe la función de la señal correspondiente y al colocar el cursor del ratón sobre alguno de los botones se despliega una imagen mostrando el efecto que tiene esa señal. Al hacer clic sobre la casilla para encender una señal, se envía el mensaje al controlador del robot y se cambia el texto para que, con el siguiente clic que de el usuario se efectúe la operación contraria, es decir, el apagado de esa señal.

- **Manejo de Señales**

De la Figura 4.9 a la Figura 4.16 se muestra la pantalla que despliega el programa, para encender o apagar las señales de salida con las cuales el robot se comunica con los equipos periféricos que complementan su operación.

En esta serie de Figuras, se puede ver que al situar el cursor del ratón sobre el botón con el número de señal, se despliega un pequeño mensaje explicando el funcionamiento de esa señal y en la parte derecha de la forma se ilustra con una imagen el efecto que produce la señal en cuestión. Las señales que se manejan están numeradas del 1 al 8, y en la tabla 4.1 se describe el funcionamiento de cada una de ellas.

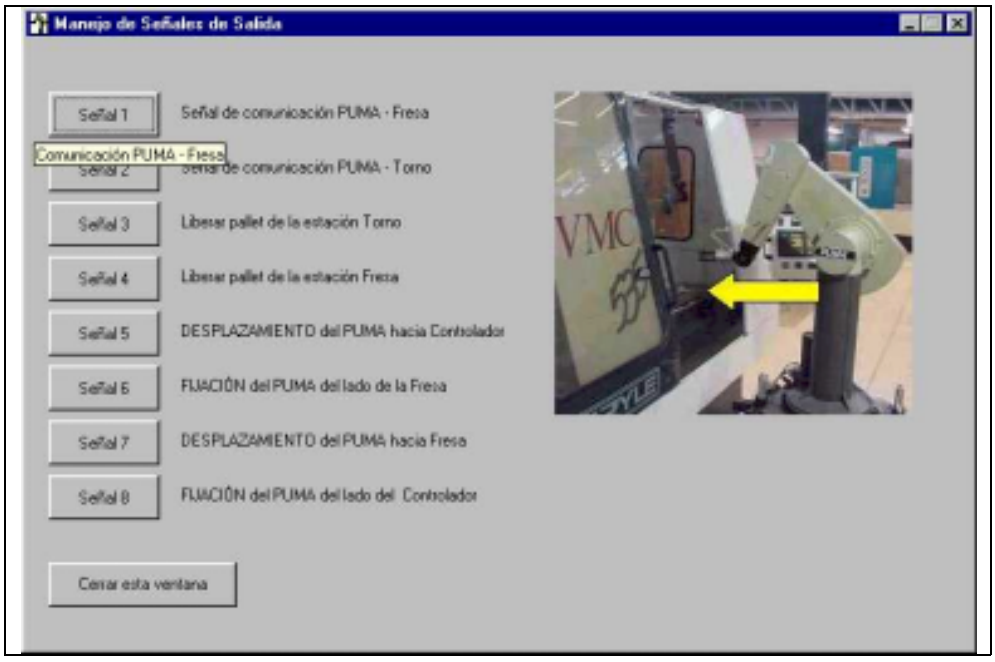


Figura 4.9 Pantalla indicadora del encendido o apagado de la señal #1.

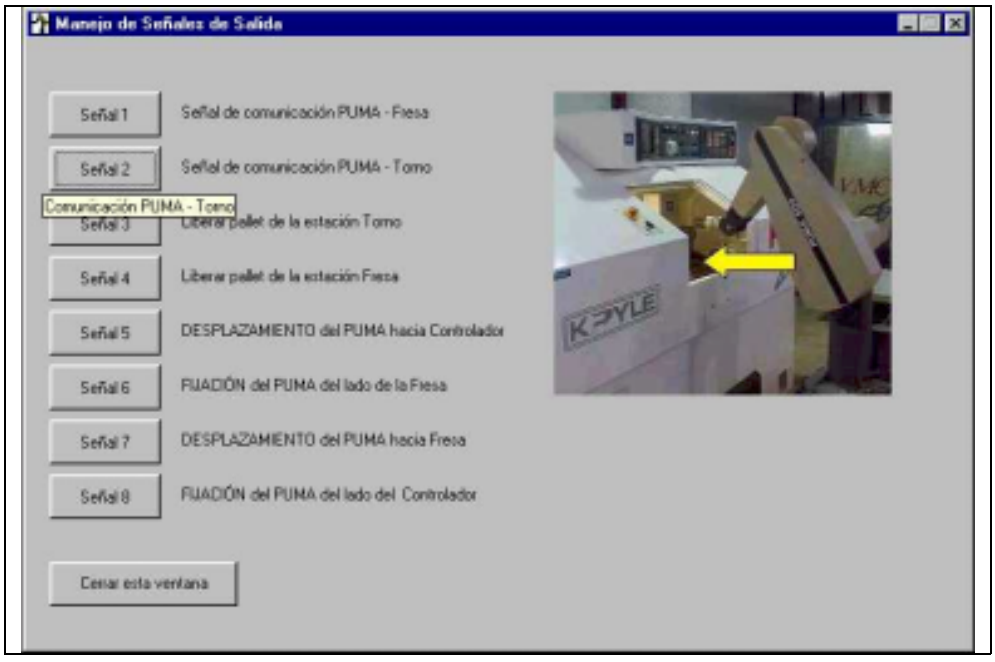


Figura 4.10 Pantalla indicadora del encendido o apagado de la señal #2.

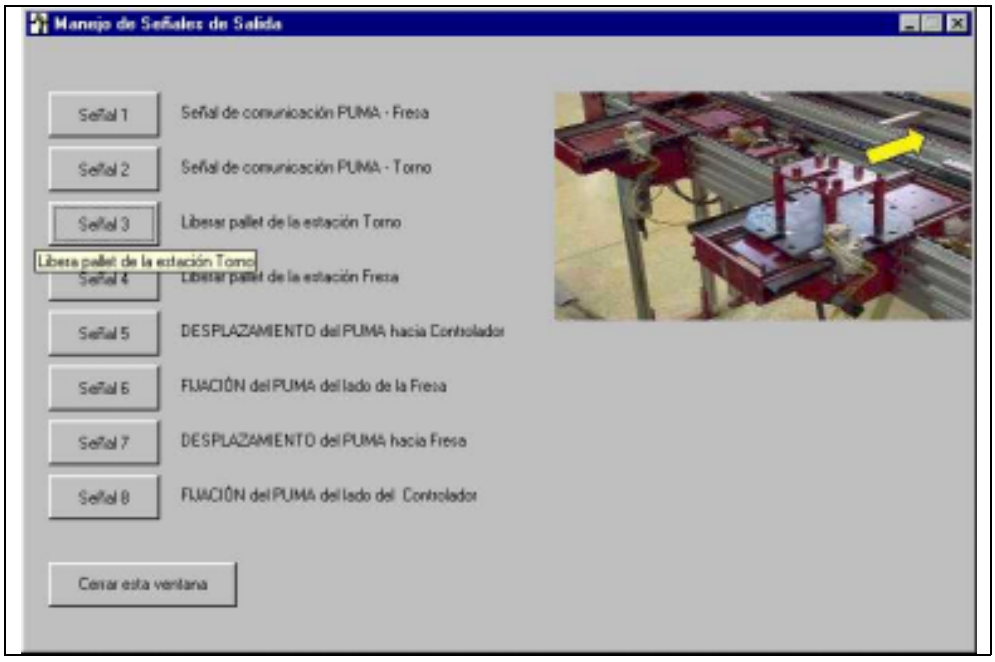


Figura 4.11 Pantalla indicadora del encendido o apagado de la señal #3.

Figura 4.12 Pantalla indicadora del encendido o apagado de la señal #4.

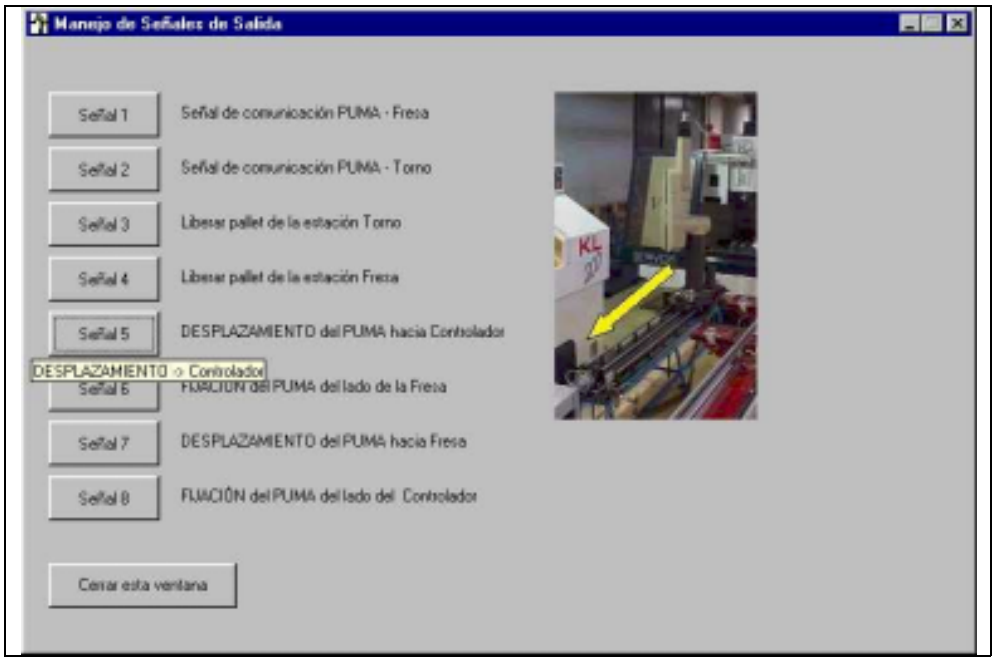


Figura 4.13 Pantalla indicadora del encendido o apagado de la señal #5.

Figura 4.14 Pantalla indicadora del encendido o apagado de la señal #6.

Figura 4.15 Pantalla indicadora del encendido o apagado de la señal #7.

Figura 4.16 Pantalla indicadora del encendido o apagado de la señal #8.

Número de Señal	Descripción
1	Señal de comunicación del robot PUMA con la Fresadora
2	Señal de comunicación del robot PUMA con el Torno
3	Señal de comunicación del robot PUMA con la banda transportadora para la liberación de un pallet en la estación #2
4	Señal de comunicación del robot PUMA con la banda transportadora para la liberación de un pallet en la estación #3
5	Señal de comunicación del robot PUMA con el riel transversal para activar el desplazamiento del brazo hacia el controlador.
6	Señal de comunicación del robot PUMA para activar el pistón neumático que fija al brazo del lado de la fresadora.
7	Señal de comunicación del robot PUMA con el riel transversal para activar el desplazamiento del brazo hacia la fresadora.
8	Señal de comunicación del robot PUMA para activar el pistón neumático que fija al brazo del lado del controlador.

Tabla 4.1 Señales de Salida del robot PUMA 560 en la celda de Amatrol del ITESM, campus Monterrey

4.3.1.4 Editor de Programas

Este grupo también esta formado por un solo botón, por medio del cual el usuario tiene acceso a la pantalla para la creación de un programa y/o edición de alguno ya existente.

- **Programador PUMA**

Al hacer clic en esta opción se despliega la pantalla del editor de programas, la cual se explica a detalle en la sección 4.3.2.

4.3.1.5 Ejecución

Este grupo contiene dos botones de orden, mediante los cuales se controla la ejecución de un programa que se encuentre grabado en la memoria del controlador. Estos botones son:

- **Correr Programa**

Cuando se hace clic sobre este botón, la interfaz lee la lista de programas grabados en el controlador y los muestra en una pantalla con un cuadro de listado. El usuario escoge el programa que desea ejecutar usando el cursor del ratón y posteriormente, al hacer clic en "Aceptar", se envía la instrucción VAL al controlador "EXEC" y el nombre del programa elegido, para que éste empiece a correr. Una vez que se termina la ejecución del programa, se despliega un cuadro de mensaje, como el que se muestra en la Figura 4.4, pero notificando al operador del equipo que el programa se ha terminado.

- **Cancelar Ejecución**

Durante la ejecución de un programa por el robot PUMA, puede presentarse la necesidad de interrumpir su actividad o simplemente ya no se desea que siga trabajando. En casos similares, el usuario puede hacer clic en este botón y la interfaz envía el caracter "a" al controlador y se suspende la corrida del programa. Cabe mencionar que este paro no es instantáneo, sino que el robot termina la instrucción que se encontraba realizando al momento de recibir la orden de cancelar. Si se desea detener el movimiento del robot, de forma instantánea, por algún imprevisto que haya ocurrido, es mejor presionar el paro de emergencia del control alámbrico o apagar la energía del brazo por medio del botón de paro (rojo) del controlador.

4.3.1.6 Cargas / Descargas

Este grupo, localizado en la parte derecha de la pantalla del Panel de Control, contiene cuatro de las funciones más representativas de esta Interfaz de Usuario. El procedimiento actual para

cargar o descargar la información de puntos o de un programa al controlador, es por medio de discos flexibles de 5.25” utilizando la unidad de disco instalada en el sistema, o mediante la computadora central. Ambos métodos resultan poco prácticos, ya que, por una parte, los discos flexibles de 5.25” ya no se encuentran disponibles en el mercado, además si un alumno lograra conseguir un disco de estos, no puede realizar ninguna modificación de los datos, sino hasta que vuelva a cargar todo en el controlador del sistema. Por otra parte, la computadora central cuenta con una opción para enviar y recibir programas, dando la opción de copiar la información a un disco de 3.5” y crea un archivo texto que puede ser editado sin ningún problema, sin embargo cuando se envía un programa de la computadora central al controlador, no se confirma la recepción correcta de cada línea del programa, por lo que si llega a existir un error en el código y no es reconocida esa línea del programa por el controlador, el usuario no es notificado de esta situación y el programa se envía incompleto, pudiendo tener consecuencias graves al momento de realizar la corrida. Además no cuenta con alguna opción para poder guardar la información de los puntos grabados en el controlador en un disco, o para enviar información de puntos grabados previamente hacia el controlador.

Esta interfaz cuenta con estas cuatro opciones, por medio de las cuales el usuario puede guardar información de locaciones y programas contenidos en el controlador, en un disco, y viceversa, también puede enviar su información de locaciones y programas de un disco hacia el controlador.

4.3.1.6.1 Programas

Un programa es una serie de instrucciones, escritas en el leguaje del robot VAL, que sigue el sistema para mover el robot, activar señales externas de salida o hacer algunos cálculos. En este sistema, los programas son identificados por nombres compuestos por cualquier cantidad de caracteres, deben empezar por una letra y luego pueden ser seguidos de cualquier combinación de letras, números o puntos. La cantidad de programa que se pueden grabar en el controlador, dependen de la memoria disponible en el mismo.

Los programas pueden ser escritos en el editor de la Interfaz de Usuario, en cualquier editor de textos o directamente en la terminal del sistema.

- **Enviar Programa**

Para hacer que el robot ejecute un programa, es necesario que éste se encuentre en el controlador del mismo. Una forma de hacerlo es tecleando directamente el código en la terminal, al salirnos del editor de este sistema el programa queda guardado en la memoria del controlador. Otra forma es escribir el programa en el editor de la Interfaz de Usuario o en cualquier editor de textos y enviarlo hacia el controlador por medio de este botón de orden. Al hacer clic en él, se abre un cuadro de diálogo para abrir un archivo. El usuario debe buscar el archivo en el que tiene guardado el programa y luego hacer clic en "Abrir". La interfaz busca en el controlador la lista de programas grabado y notifica al usuario si este programa ya existe en él. De ser así, se da la posibilidad de sobre escribirlo o de grabarlo con otro nombre. Si no encuentra un programa con el mismo nombre del archivo se inicia la secuencia para el envío del programa. La forma como se hace es enviando a la terminal la instrucción de VAL "EDIT" y el nombre del programa y se procede a enviar, una por una, las líneas del programa en el archivo. Al término de la operación se notifica al usuario.

- **Recibir Programa**

La recepción de programas es muy útil para respaldar información, además de incrementar la capacidad de almacenaje de programas al poder grabarlos en el disco duro o en discos flexibles. Cuando se activa esta operación, la Interfaz de Usuario busca en el controlador la lista de programas que están grabados en él y los muestra en un cuadro de listado. El usuario debe elegir el programa que desea recibir y posteriormente declarar el nombre bajo el que lo quiere grabar en la unidad de disco que determine.

Al final de ambas operaciones, tanto de enviar como de recibir un programa, se pregunta al usuario si desea crear un archivo con la lista de los puntos que fueron nombrados dentro del programa recién transferido. La utilidad de este archivo se explica en la sección 4.3.1.6.2

4.3.1.6.2 Puntos

Los puntos o locaciones, son las posiciones físicas que se guardan en el controlador ya se en forma de coordenadas cartesianas, con los grados de la herramienta con respecto a los ejes cartesianos o en forma de los ángulos de cada articulación, según se vio en la sección 4.3.1.2 en el apartado "Grabar Punto".

Los puntos normalmente se graban físicamente en el sistema robótico, colocando el brazo en la posición deseada y luego ordenando al controlador que registre las coordenadas o los ángulos de las articulaciones, a esta actividad se le conoce como "enseñanza de puntos". Posteriormente, en un programa se llaman estas posiciones para hacer que el robot realice una tarea.

- **Recibir Puntos**

Una vez que se han grabado en el controlador las posiciones necesarias para que el robot realice algún trabajo específico, es importante contar con una opción para almacenar esta información en otro dispositivo, como el disco duro de una computadora o un disco flexible. Al hacer clic en este botón de orden, el usuario va a recibir las coordenadas o los ángulos de las articulaciones que definen las posiciones que grabó en el controlador. Primero se le solicita un nombre para el archivo que se va a crear con esta información, posteriormente se le pregunta si cuenta con una lista de los puntos que desea recibir. Esta opción es útil para recibir solamente la información que necesita. En el controlador quedan grabados todas las locaciones que se hayan declarado, ya sea por otros usuarios o para otras tareas, por lo que si se desea solamente recibir ciertos puntos se debe crear un archivo que contenga una lista con los nombres bajo los que fueron grabados los puntos en el controlador. Este archivo puede crearse cuando se envía o cuando se recibe un programa. Cuando la Interfaz de Usuario termina la transferencia de un programa, pregunta al usuario si desea crear esta lista, a lo que el usuario debe responder afirmativamente si es que no cuenta con esta lista, y el archivo es creado automáticamente, o en su defecto, esta lista se puede escribir en cualquier editor de textos.

Una vez que se proporcionó toda la información, la Interfaz de Usuario envía al controlador la instrucción de VAL "POINT" junto con el nombre del punto que va a recibir. Éste responde con las coordenadas o con los ángulos de las articulaciones que definen a ese punto y esta información es escrita en el archivo de salida. Al final de la transferencia se le notifica al usuario acerca del número de puntos que contenía su archivo y del número de puntos que se encontraron en el controlador. Si existiera un error en el nombre de algún punto y éste no existiera en el controlador, se le avisa al usuario que se recibieron menos puntos de los solicitados, por lo que hay que revisar el contenido de la lista de puntos.

Si el usuario desea recibir todos los puntos contenidos en la memoria del controlador, éste debe responder negativamente a la pregunta que hace la Interfaz de Usuario con respecto al archivo con la lista de puntos. Esta opción envía al controlador la instrucción de VAL "LISTL" y éste responde con la lista e información completa de todos los puntos existentes, misma que se escribe en el archivo de salida.

- **Enviar Puntos**

Como se mencionó al inicio de esta sección, la forma más común de declaración de locaciones del robot, es por medio de la "enseñanza de puntos", sin embargo, también se puede hacer por medio de un archivo que contenga el nombre de los puntos que se desean grabar y las coordenadas o los ángulos de las articulaciones que los definen. Este archivo puede ser creado mediante un editor de textos, o mediante el proceso de recibir puntos que se explicó en los párrafos anteriores.

Cuando se activa esta opción, la Interfaz de Usuario pregunta el nombre del archivo en el cual se tiene la información de los puntos que se deseas enviar al controlador, por medio de una caja de diálogo. Una vez que se le proporciona, inicia la transferencia de información enviando el comando de VAL "POINT", acompañado del nombre del punto, el controlador responde con la información que tiene de ese punto, si no existía envía ceros en todos sus campos y entonces la interfaz envía las coordenadas del punto, contenidas en el archivo, para

declarar sus nuevas coordenadas. Si la respuesta es diferente a los ceros, significa que el punto ya existía en el sistema, se notifica al usuario y se le pregunta si desea sobre escribirlo.

4.3.1.7 Mensajes Operativos

Este grupo está compuesto solamente por un cuadro de texto, el cual presenta mensajes para el usuario con la intención de notificarle lo que esta sucediendo en el sistema, ya sea el movimiento del brazo a un punto, la ejecución de un programa, o si no se detecta la presencia del controlador del sistema robótico, se da el mensaje que no se encontró al robot y se procede a trabajar fuera de línea, entre otros.

4.3.1.8 Reiniciar

Este grupo compuesto por el botón de "Reconexión con el Robot", tiene la función de verificar que el controlador del sistema robótico esté conectado al puerto serie de la computadora en la cual está corriendo la interfaz de usuario.

- **Reconexión con el Robot**

Al hacer clic en este botón se envía una señal al controlador a través del puerto serial, y se da un tiempo de espera para recibir la respuesta de este y conseguir la operación en línea. Se utiliza principalmente si la Interfaz de Usuario se inició fuera de línea y posteriormente se hace la conexión física con el controlador, con esto, se evita tener que salir del programa y luego volver a entrar.

4.3.2 Editor de Programas

El Editor de Programas de esta Interfaz de Usuario, está diseñada para escribir programas en el lenguaje del robot PUMA 560, con los comandos de programación típicamente utilizados en el

laboratorio de Sistemas Integrados de Manufactura, suponiendo que el usuario se guía por un algoritmo de programación previamente establecido. El programa escrito se va desplegando en la caja de texto a la derecha de la pantalla, de esta manera, el usuario puede ver el código que corresponde a cada instrucción en la que hace clic, además si hubiera algún comando que desee utilizar y que no este contemplada en los botones de orden, esta caja de texto permite la edición del programa, ya sea para teclear directamente instrucciones diferentes a las contempladas o para hacer correcciones del código ya escrito por cualquiera de los dos métodos mencionados.

Los programas que se crean o modifican en el editor, es necesario posteriormente enviarlos al controlador del robot por medio de la opción de "Enviar Programa".

En la Figura 4.17 se muestra la pantalla del Editor de Programas, en ella se puede ver que también los botones de orden se encuentran en grupos de acuerdo con el tipo de comando que escribe en el programa. A continuación se explica el sentido de cada grupo, así como el funcionamiento de los botones que los componen.

4.3.2.1 Programación

En este grupo esta formado por cuatro subgrupos: Inicio, Configuración de Brazo, Control de Movimientos y Control del Gripper. Los botones de orden que se contemplan en estos cuatro subgrupos, se refieren a las instrucciones básicas para la programación de movimientos del robot PUMA 560, es decir, los comandos mínimos necesarios para lograr que el robot se mueva siguiendo una serie de posiciones previamente establecidas.

4.3.2.1.1 Inicio

El subgrupo de inicio, formado por un solo botón de orden, contempla los comandos que deben escribirse siempre al empezar a programar una rutina.

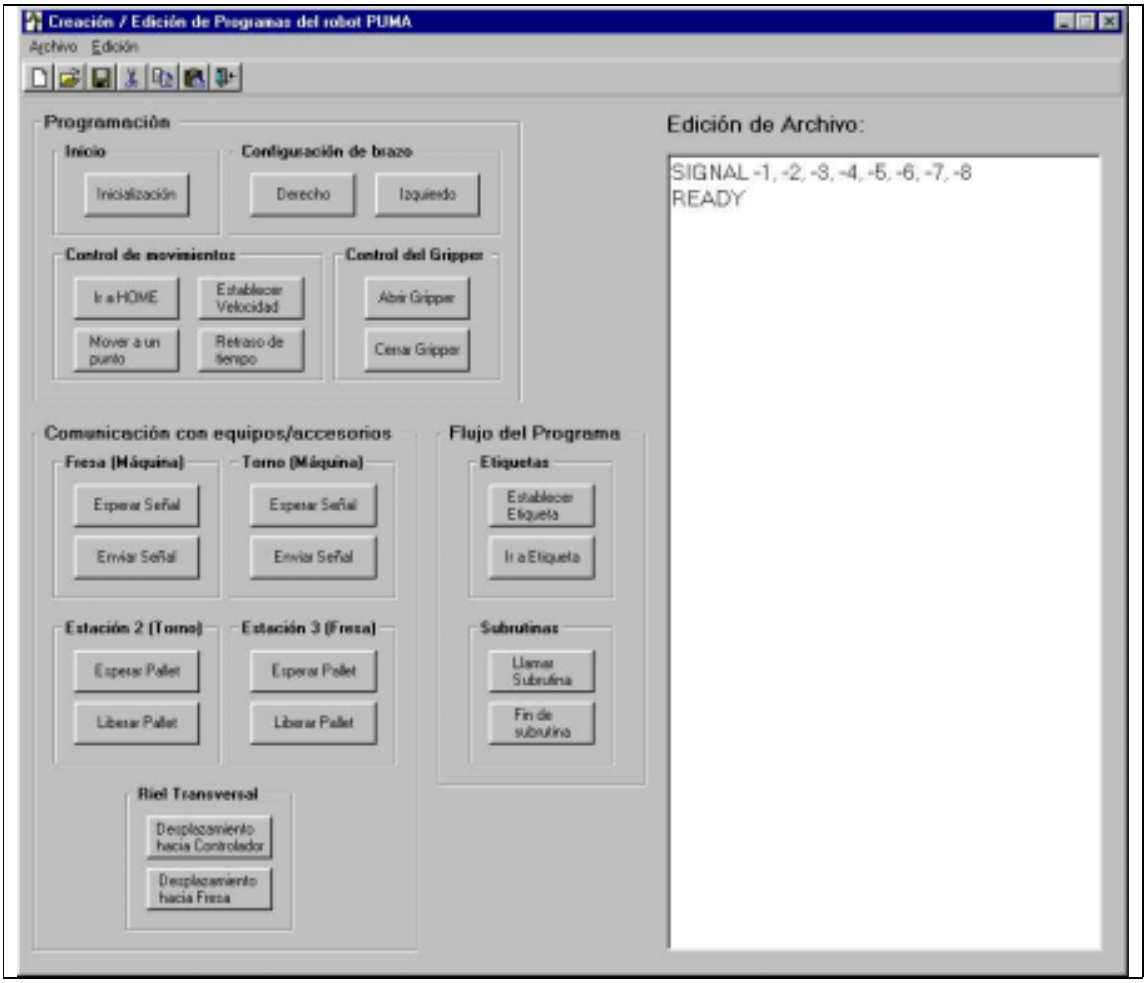


Figura 4.17 Pantalla del Editor de Programas para el Robot PUMA 560

- **Inicialización**

Básicamente, la inicialización del robot PUMA 560 para la ejecución de un programa, consiste en apagar todas las señales de salida, en este caso de la #1 a la #8, y posteriormente enviar el robot a su posición de inicio: "HOME".

Las líneas de código que inserta ese botón son:
SIGNAL –1, –2, –3, –4, –5, –6, –7, –8
READY

El significado de las señales de salida se mencionan en la tabla 4.1, y el del comando READY se explica en la sección 4.3.1.2 en el apartado “HOME”

4.3.2.1.2 Configuración de Brazo

Para un brazo antropomórfico, de seis articulaciones, la mayoría de los puntos dentro de su envolvente los puede alcanzar asumiendo alguna de sus configuraciones posibles, las dos más comunes en el laboratorio son de brazo derecho o de brazo izquierdo.

- **Derecho**

Al hacer clic sobre este botón, se inserta en el programa el código: RIGHTY

Este código cambia la configuración del robot de tal manera que las tres primeras articulaciones semejan un brazo humano derecho.

- **Izquierdo**

Al hacer clic sobre este botón, se inserta en el programa el código: LEFTY

Este código cambia la configuración del robot de tal manera que las tres primeras articulaciones semejan un brazo humano izquierdo.

4.3.2.1.3 Control de Movimientos

En este subgrupo, se tienen los comandos que tienen un efecto directo en el movimiento del robot, ya sea para enviarlo a una posición específica, para modificar la velocidad de su desplazamiento en el espacio o para provocar un retraso en sus movimientos.

- **Ir a HOME**

Por medio de este botón de orden, se inserta en el programa la instrucción: READY

La cual envía al robot a su posición de inicio, la cual se explicó en la sección 4.3.1.2 en el apartado “HOME”

- **Mover a un Punto**

Cuando el usuario hace clic en este botón, surge un cuadro de entrada de datos similar a la que se muestra en la Figura 4.8. En la que se pregunta el nombre del punto al que se quiere mover el brazo y se puede establecer el tipo de movimiento que deba describir en su trayectoria, ya sea articulado o lineal. Suponiendo que el robot deba moverse a un “Punto1”, la instrucción que se inserta en el programa para un movimiento articulado sería: MOVE Punto1

Mientras que para un movimiento lineal se inserta: MOVES Punto1

- **Establecer Velocidad**

El botón de orden para establecer la velocidad, despliega un cuadro de entrada de datos en el que se le pide al usuario dar el valor de la velocidad que desea para el movimiento del robot, y declarar si quiere que esa velocidad sea la que utilice el robot en el resto del programa. Suponiendo se requiere que el robot se mueva a una velocidad de 50, las dos opciones que se pueden insertar son:

SPEED 50 ALWAYS
o
SPEED 50

Con la primera, la velocidad declarada es la que va a seguir el brazo en todos sus movimiento subsecuentes, hasta que encuentre otra línea de código que la modifique nuevamente o hasta

terminar la ejecución del programa. A esta velocidad se le conoce como velocidad de programa. La segunda opción declara una velocidad temporal, la cual solamente será obedecida para el siguiente movimiento del brazo, recuperando su velocidad anterior en los movimientos subsecuentes.

- **Retraso de Tiempo**

Por medio de este botón se inserta la instrucción para hacer que el movimiento del brazo o la ejecución del programa en general, haga una pausa por un tiempo determinado. Suponiendo una pausa de 2 segundos, la instrucción que se inserta es: DELAY 2

4.3.2.1.4 Control del Gripper

En este subgrupo se tienen las opciones para activar o desactivar el efector final del robot, en este caso un gripper. Debido a que no se cuenta con un efector servocontrolado, los comandos factibles son el abrir o cerrar el gripper instantáneamente.

- **Abrir Gripper**

Por medio de este botón, se insertan las líneas:
DELAY 0.5
OPENI
DELAY 0.5

Con las cuales el gripper abre instantáneamente, los retrasos de tiempo se incluyen para que el efector alcance a abrir completamente, antes de que el robot empiece la ejecución de la siguiente instrucción en el programa.

- **Cerrar Gripper**

Por medio de este botón, se insertan las líneas:
DELAY 0.5
CLOSEI
DELAY 0.5

Con las cuales el gripper cierra instantáneamente, los retrasos de tiempo se incluyen para que el efector alcance a cerrar completamente, antes de que el robot siga con la siguiente instrucción en el programa.

4.3.2.2 Comunicación con Equipos / Accesorios

El robot PUMA 560, instalado en esta celda flexible de manufactura de AMATROL, es el encargado de alimentar materia prima a los dos centros de maquinado: torno y fresadora, por lo que, cuenta con señales eléctricas, de entrada y de salida con todos los equipos que tiene a su alrededor para poder realizar eficientemente su trabajo. Cuenta con señales eléctricas de comunicación con la fresadora, el torno, la banda transportadora y el riel transversal sobre el cual está montado el brazo del robot. Las señales de salida se envían mediante el comando "SIGNAL #" y las señales de entrada se leen con la instrucción "WAIT SIG (####)"

4.3.2.2.1 Fresa

El robot necesita comunicarse con la fresadora para coordinar sus movimientos y colocar la pieza que se va a maquinar en la prensa de ésta. La manera que se comunica es por medio de señales de entrada y de salida que se explican a continuación.

- **Esperar Señal**

Este botón inserta el código: WAIT SIG(1001)

Mediante el cual el robot hace una pausa en la ejecución del programa hasta que la señal de entrada #1001 sea activada por la fresadora.

- **Enviar Señal**

Este botón inserta las líneas de código:
SIGNAL 1
WAIT SIG(-1001)
SIGNAL -1

Principalmente, con la señal #1 es con la que la fresadora recibe la señal eléctrica para seguir la ejecución de su programa. Las siguientes dos líneas son necesarias para no dejar permanentemente encendida la señal #1, esto es, una vez que se sabe que la fresadora ha seguido con las siguientes líneas de su programa, la señal #1001 es apagada, y por lo tanto el robot puede apagar la señal #1 que había encendido previamente.

4.3.2.2.2 Torno

Al igual que con la fresadora, el robot necesita comunicarse con el torno para coordinar sus movimientos y colocar la pieza que se va a maquinar en las mordazas de éste. La manera que se comunica es por medio de señales de entrada y de salida que se explican a continuación.

- **Esperar Señal**

Al hacer clic en este botón, se inserta la línea de código: WAIT SIG(1002)

Mediante el cual el robot hace una pausa en la ejecución del programa hasta que la señal de entrada #1002 sea activada por el torno.

- **Enviar Señal**

Este botón inserta las siguientes líneas de código al programa:
SIGNAL 2
WAIT SIG(-1002)
SIGNAL -2

Principalmente, con la señal #2 es con la que la fresadora recibe la señal eléctrica para seguir la ejecución de su programa. Las siguientes dos líneas son necesarias para no dejar permanentemente encendida la señal #2, esto es, una vez que se sabe que la fresadora ha seguido con las siguientes líneas de su programa, la señal #1002 es apagada, y por lo tanto el robot puede apagar la señal #2 que había encendido previamente.

4.3.2.2.3 Estación #2

La estación designada como #2, es conocida también como la estación del torno. La comunicación del robot con esta estación, así como con la estación #3 es muy importante, ya que supuestamente el robot PUMA no puede realizar ninguna operación sino hasta que reciba un contenedor con materia prima para ser maquinada, y es por medio de estas señales que el robot es notificado cuando recibe un contenedor en la estación para que inicie su trabajo. De igual manera, cuando termina su operación y regresa el material al contenedor, debe ordenar la liberación del contenedor para que este siga su proceso en la celda.

- **Esperar Pallet**

Este botón inserta la línea de código: WAIT SIG(1003)

Mediante el cual el robot hace una pausa en la ejecución del programa hasta que la señal de entrada #1003 sea activada por un contenedor que entra a la estación del torno.

- **Liberar Pallet**

Mediante este botón se insertan las líneas de código:
SIGNAL 3
WAIT SIG(-1003)
SIGNAL -3

Principalmente, con la señal #3 es con la que la banda transportadora recibe la señal eléctrica para liberar el contenedor de la estación del torno y que siga su recorrido. Las siguientes dos líneas son necesarias para no dejar permanentemente encendida la señal #3, esto es, una vez que se sabe que el contenedor se ha retirado de la estación, la señal #1003 es apagada, y por lo tanto el robot puede apagar la señal #3 que había encendido previamente.

4.3.2.2.4 Estación #3

Al igual que con la estación #2, estas señales permiten comunicarle al robot la presencia de un contenedor en esta estación para que éste realice su tarea, y también permiten que el robot mande liberar el contenedor una vez terminado el trabajo.

- **Esperar Pallet**

Al hacer clic en este botón se inserta el código: WAIT SIG(1004)

Mediante el cual el robot hace una pausa en la ejecución del programa hasta que la señal de entrada #1004 sea activada por un contenedor que entra a la estación de la fresadora.

- **Liberar Pallet**

Este botón inserta las líneas de código:
SIGNAL 4

WAIT SIG(-1004)
SIGNAL -4

Principalmente, con la señal #4 es con la que la banda transportadora recibe la señal eléctrica para liberar el contenedor de la estación de la fresadora y que siga su recorrido. Las siguientes dos líneas son necesarias para no dejar permanentemente encendida la señal #4, esto es, una vez que se sabe que el contenedor se ha retirado de la estación, la señal #1004 es apagada, y por lo tanto el robot puede apagar la señal #4 que había encendido previamente.

4.3.2.2.5 Riel Transversal

El riel transversal sobre el cual se encuentra montado el brazo del robot PUMA 560, es un accesorio que le permite desplazarse del controlador hacia la fresadora, aumentando su volumen de trabajo en gran medida. Las opciones que se tienen son:

- **Desplazamiento hacia Controlador**

Al hacer clic en esta opción, se inserta en el programa las siguientes líneas de código:
SIGNAL -6
SIGNAL 5
WAIT SIG(1006)
DELAY 2
SIGNAL -5
SIGNAL 8

Mediante las cuales se realiza el desplazamiento del robot por el riel hacia el controlador y se activa el pistón que lo fija de este lado. Las señales de salida se explican en la tabla 4.1. La señal de entrada #1006 es activada una vez que el robot se encuentra en el extremo del riel del lado del controlador.

- **Desplazamiento hacia Fresa**

Al hacer clic en este botón de orden, se insertan en el programa las siguientes líneas de código:
SIGNAL -8
SIGNAL 7
WAIT SIG(1005)
DELAY 2
SIGNAL -7
SIGNAL 6

Mediante las cuales se realiza el desplazamiento del robot por el riel hacia la fresadora y se activa el pistón que lo fija de este lado. Las señales de salida se explican en la tabla 4.1. La señal de entrada #1005 es activada una vez que el robot se encuentra en el extremo del riel del lado de la fresadora.

4.3.2.3 Flujo del Programa

Las siguientes instrucciones controlan la secuencia en la cual, las instrucciones de un programas son ejecutadas. Por lo que se utilizan para controlar el flujo lógico dentro de un programa o de varios programas.

4.3.2.3.1 Etiquetas

Las etiquetas son números que se utilizan para identificar una instrucción en el programa con el propósito de tener una rama en la ejecución del programa a esta instrucción.

- **Establecer Etiqueta**

Las etiquetas que identifican una línea en un programa de usuario se establecen simplemente con un número entero positivo, el cual se coloca al inicio de la línea que se quiere identificar. Cuando el usuario hace clic sobre este botón, se abre un cuadro de entrada de datos en el que se debe teclear el número que se usará como etiqueta.

- **Ir a Etiqueta**

Al hacer clic sobre este botón, se despliega un cuadro de entrada de datos donde se pide al usuario que teclee en número de la etiqueta donde se encuentra la línea a la que desea cambiar el flujo del programa. Posteriormente se inserta la línea de código:
GOTO <número de etiqueta>
La cual realiza un salto incondicional del flujo del programa a la instrucción identificada por la etiqueta dada.

4.3.2.3.2 Subrutinas

Las subrutinas son programas que se invocan dentro de la ejecución de un programa principal. Puede decirse que un programa principal es el que pone a correr usando la instrucción de ejecución de VAL "EXECUTE", mientras que una subrutina es un programa que se ejecuta por la instrucción "CALL" dentro de otro programa.

- **Llamar Subrutina**

Para insertar el llamado de una subrutina en un programa principal, el usuario debe hacer clic en este botón y teclear el nombre de la subrutina en el cuadro de entrada que se despliega. La línea que se inserta es: CALL <nombre de la subrutina>

Con esta instrucción, la ejecución del programa actual se suspende temporalmente, y la ejecución continua en la primer línea de la subrutina señalada. La ejecución automáticamente regresa al programa principal al encontrar la instrucción “RETURN” en la subrutina.

- **Fin de subrutina**

Este botón inserta la instrucción: RETURN

La cual termina la ejecución de la subrutina en la que se encuentre y regresa la ejecución del programa principal en la línea siguiente a la instrucción que causó el llamado a la subrutina. Si se escribe un “RETURN” en el programa principal, termina la ejecución de dicho programa.

4.3.2.4 Área de Escritura del Código del Programa

El área de escritura del código del programa es el cuadro de texto que se encuentra del lado derecho de la pantalla de edición que se muestra en la Figura 4.17. Este cuadro funciona como un editor de textos, donde los programas pueden crearse de forma semiautomática o manual. En la forma semiautomática el usuario puede ir viendo los códigos del programa que se van insertando mediante el uso de los botones de orden y de los cuadros de entrada de datos para los parámetros de los comandos que así lo requieren. En caso que se necesite insertar una línea de código, antes de la última, basta con colocar el cursor de texto al inicio de la línea donde se desea hacer la inserción. Al hacer clic en el botón de la instrucción que se va a agregar, la línea donde se encuentra el cursor, así como todas las siguientes, son movidas un renglón hacia abajo para colocar la nueva instrucción.

También se puede hacer la escritura de un programa de forma manual, si es que existiera un código que no está contemplado en los botones de orden, el usuario puede introducir texto al programa mediante el uso normal del teclado, como cualquier editor de textos. Además también están disponibles los botones de cortar, copiar y pegar, para la edición de lo que se haya escrito, como normalmente se utilizan en las aplicaciones comerciales.

Al terminal la edición de un programa, el usuario puede grabarlo en disco, ya sea mediante la opción del menú Archivo, Guardar, o con el botón de atajo que tiene un disco flexible como icono. Esta opción también es activada automáticamente cuando el usuario decide salir del editor y no ha grabado los cambios que haya escrito, mediante un cuadro de mensaje el cual pregunta si se desea hacer esta operación.

4.3.3 Modo Terminal

El Modo Terminal es el tercer elemento que compone a esta Interfaz de Usuario. Esta opción se puede activar mediante el menú Terminal, Modo Terminal, o usando el botón de atajo que muestra una terminal como icono. En la Figura 4.18 se muestra la pantalla que se despliega cuando el usuario entra a este modo de operación, con la información que se imprime en el monitor de la terminal cuando se teclea la instrucción "status".

El objetivo de este modo de operación es darle al operador la libertad de enviar al controlador cualquier instrucción que desee por medio del teclado de la computadora, tal como lo haría si estuviera conectado al controlador con la terminal del sistema. Esta opción es especialmente útil para usuarios experimentados quienes tal vez no tengan en el Panel de Control todas las instrucciones necesarias para manipular el robot de acuerdo a sus habilidades.

4.4 Operación de la Interfaz en Línea y Fuera de Línea

La Interfaz de Usuario tiene dos modos de operación: en línea y fuera de línea.

- **Operación en Línea**

La operación en línea se entiende que es cuando la computadora en la que se está corriendo el programa de la interfaz, se encuentra conectada al controlador del sistema robótico PUMA 560 de Unimation mediante el puerto serial.

Cuando se encuentra operando el programa de este modo, el usuario tiene acceso a todas las pantallas de los elementos que componen la interfaz. Los botones de orden son desactivados solamente cuando el robot se encuentra ejecutando alguna instrucción, y no permite el envío de otro comando. Esta situación es confirmada al usuario mediante el cuadro de texto para Mensajes operativos (sección 4.3.1.7). Al terminar la ejecución automáticamente se activan nuevamente los botones de orden.

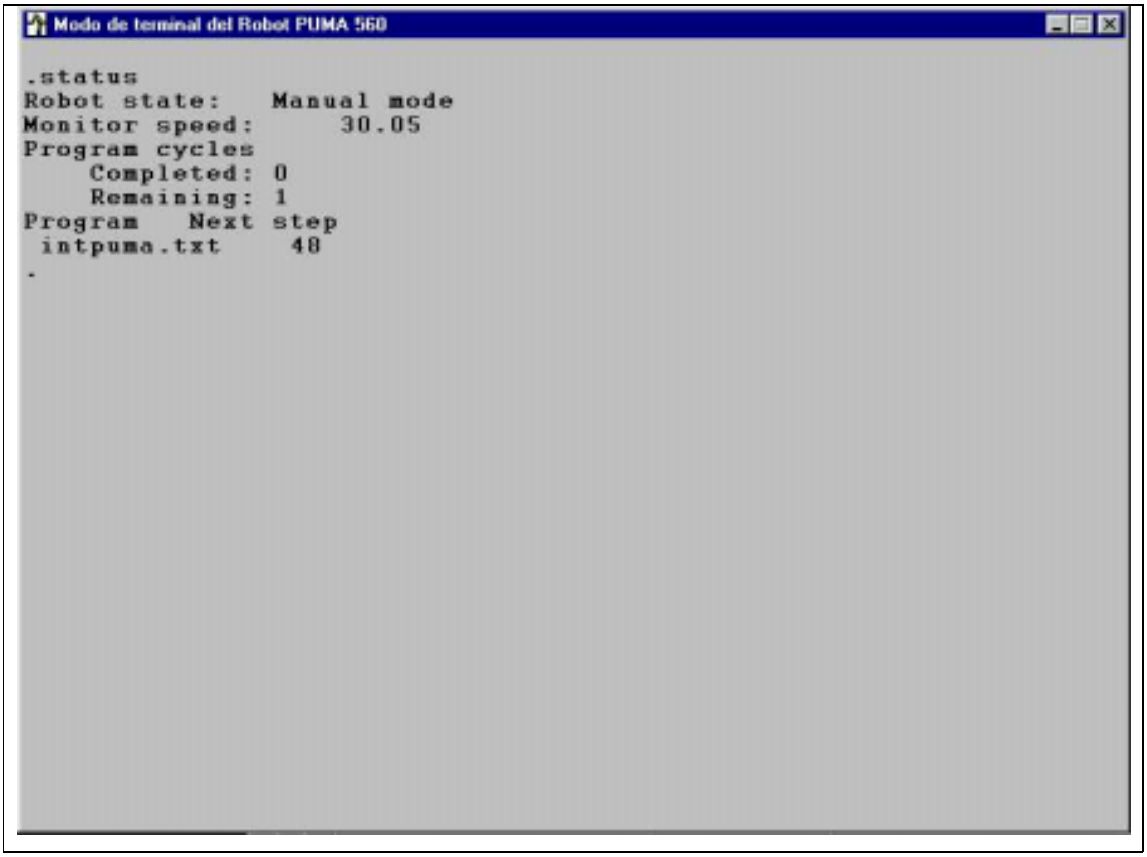


Figura 4.18 Pantalla Emuladora de la Terminal de Control del Robot PUMA 560

- **Operación Fuera de Línea**

La operación fuera de línea se entiende que es cuando la computadora en la que se está corriendo el programa de la interfaz, no se encuentra conectada al controlador del sistema robótico PUMA 560 de Unimation.

Cuando se encuentra operando el programa de este modo, en el Panel de Control se desactivan todos los botones de orden que envían señales hacia el controlador y se despliega, mediante el cuadro de texto para Mensajes Operativos, ver sección 4.3.1.7, el siguiente mensaje:

"Robot no disponible
Trabajando fuera de línea"

Los únicos botones que quedan disponibles bajo esta condición son: el botón que despliega la pantalla para el manejo de señales de salida (Figuras 4.9 a 4.16) y el que despliega la pantalla del editor de textos (Figura 4.17), es decir, aquellos que no se comunican directamente con el controlador, y por otra parte también está activo el botón para realizar la "Reconexión con el robot".

La pantalla para el manejo de señales de salida, estando fuera de línea, permite la operación de los botones de orden, sin embargo las señales no son enviadas al controlador. La pantalla del editor de textos tiene el mismo funcionamiento estando en línea que fuera de línea, por lo que el usuario puede realizar la escritura de un programa en cualquiera de las dos condiciones.

El botón de "Reconexión con el robot", queda disponible para activar los botones de orden del Panel de Control, si es que posteriormente se conecta la computadora personal al controlador del robot.

Una vez que se realizó la programación de la Interfaz de Usuario, se hicieron pruebas para asegurar su completo funcionamiento. Posteriormente se diseñó un experimento para evaluar el impacto real con los usuarios del robot. En el siguiente capítulo 5, se describe en qué consistió el experimento y los resultados que se obtuvieron, así como las conclusiones y las investigaciones futuras que se desprenden de este proyecto de tesis.

Capítulo 5
Resultados, Conclusiones e Investigaciones Futuras

5.1 Introducción

Para concluir con el presente trabajo de tesis, se desarrolló una prueba por medio de la cual fuera posible evaluar si la Interfaz de Usuario en verdad podía reducir el tiempo en la programación de una tarea para el robot. La prueba consistió en la elaboración de un programa para una tarea muy sencilla en el Laboratorio de Sistemas Integrados de Manufactura. Se tomó una muestra de 21 personas entre alumnos e instructores del Laboratorio. Se les pidió que escribieran un programa utilizando el editor de la Interfaz y posteriormente que escribieran el mismo programa utilizando la terminal del sistema.

En las hojas que se les proporcionaba a los participantes, se daba una breve descripción de la tarea a realizar, un diagrama de puntos y el algoritmo de programación. Además en una segunda hoja estaban todos los comandos de programación que se requerían para teclear el programa en la terminal, así como una relación de las señales de entrada, de salida, y las instrucciones de manejo del editor de programas de la terminal del sistema. En el Anexo A, se muestran esta información tal y como la recibieron los usuarios.

Los datos que se recolectaron fueron el tiempo que les llevaba hacer el programa en la interfaz y luego el tiempo necesario para hacer el programa en la terminal. Posteriormente se revisaban ambos programas y también se tomaba el dato del número de errores, tanto en la interfaz de usuario como en la terminal del sistema. Se consideró como error cualquier falta que pudiera provocar que el robot no trabajara de manera correcta, como la omisión de retrasos de tiempo al momento de accionar el efector final, o apagar la señal de liberar un pallet una vez que éste abandonó la estación en la que se recibió.

5.2 Resultados

Los resultados de las pruebas realizadas se resumen en la Tabla 5.1. En ella se puede ver que todas las personas que realizaron el ejercicio hicieron un tiempo menor usando la Interfaz de Usuario que en la Terminal del Sistema del robot PUMA. También se puede ver en las últimas dos columnas de la derecha la diferencia de tiempo entre un método y otro, así como el porcentaje que representa en la reducción del mismo. Cabe señalar que el tiempo tomado en la interfaz, incluye el envío del programa de la computadora personal al controlador del sistema, ya que el objetivo final era tener un programa que se pudiera ejecutar en el controlador.

En la parte inferior de la Tabla 5.1, puede verse que, en promedio, las personas se tardaban aproximadamente siete minutos y medio en hacer el programa en el editor de la terminal y 2:40 minutos menos en programar la misma tarea usando la interfaz, lo cual representa un 32% de reducción en el tiempo de programación. Por otra parte las desviaciones estándar nos indican que al trabajar en la terminal se tiene una variabilidad mucho mayor, casi del doble, que haciéndolo con la interfaz.

Para aceptar que la media de los tiempos obtenidos utilizando la interfaz es menor que la media de los tiempos obtenidos en la terminal del sistema se realizó una prueba T para dos muestras que siguen una distribución normal, asumiendo varianzas diferentes. Primero se verificó que la información de los tiempos siguieran una distribución normal, para poder aplicar la prueba mencionada. Por medio de MINITAB se aplicó la prueba de Ryan – Joiner, la cual dice que para:

H_0: la distribución poblacional es normal, versus
H_a: la distribución poblacional no es normal

Consiste en rechazar H_0 cuando $r \leq c_\alpha$. Donde r es el coeficiente de correlación muestral y el valor crítico de c_α se obtiene de la tabla para esta prueba para varios niveles de significancia α y tamaños muestrales *n* [36]. En el Anexo B está incluida esta tabla.

	Terminal		Interfaz		Diferencia	
Persona	Tiempo	Errores	Tiempo	Errores	Tiempo	Porcentaje
1	04:28.8	8	04:22.9	0	00:05.9	2
2	06:39.0	0	04:56.4	0	01:42.5	26
3	07:59.7	3	04:49.8	0	03:09.9	40
4	09:37.7	6	04:36.8	0	05:00.9	52
5	10:16.5	4	04:57.9	0	05:18.6	52
6	07:34.3	3	04:26.6	0	03:07.7	41
7	07:53.0	0	04:32.3	0	03:20.7	42
8	05:44.4	2	04:33.3	0	01:11.1	21
9	08:12.1	2	06:36.3	0	01:35.7	19
10	05:16.4	0	05:02.1	0	00:14.3	5
11	09:45.1	5	05:52.5	0	03:52.6	40
12	08:25.5	0	05:04.1	0	03:21.4	40
13	05:09.2	0	04:31.4	0	00:37.8	12
14	07:33.0	4	04:24.3	0	03:08.7	42
15	06:06.0	9	03:32.8	0	02:33.2	42
16	08:37.6	4	06:54.3	0	01:43.3	20
17	13:06.3	7	03:47.7	0	09:18.6	71
18	05:32.6	3	03:59.9	0	01:32.7	28
19	05:56.3	2	03:48.1	0	02:08.2	36
20	06:57.8	5	04:42.4	0	02:15.3	32
21	05:32.9	0	04:51.1	0	00:41.8	13
Promedio	07:26.9	3	04:46.8	0	02:40.1	32
Des. Est.	02:05.4	2.8	00:50.1	0.0	02:05.2	17.0

Tabla 5.1 Tiempos obtenidos en las pruebas realizadas con la Interfaz de Usuario y la terminal del sistema del robot PUMA.

El resultado de la prueba Ryan – Joiner para las lecturas obtenidas en la terminal, aparecen en la Figura 5.1. El valor estadístico de prueba es r = 0.9655 y de la tabla de valores críticos para

esta prueba del Anexo B se tiene que c_α = 0.9290. Como 0.9655 > 0.9290, no se puede rechazar la hipótesis nula de normalidad, para un nivel de significancia de 0.01

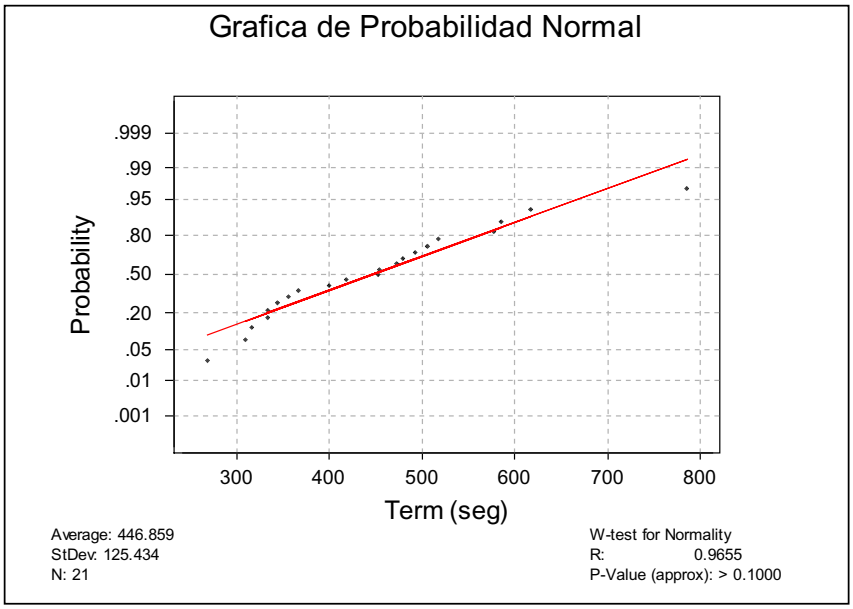


Figura 5.1 Resultado de la prueba de Ryan – Joiner para las lecturas en la terminal.

De igual manera, el resultado de la prueba Ryan – Joiner para las lecturas obtenidas en la interfaz, aparecen en la Figura 5.2. El valor estadístico de prueba es r = 0.9372 y utilizando el mismo valor crítico c_α = 0.9290, se tiene que 0.9372 > 0.9290, y por lo tanto no se puede rechazar la hipótesis nula de normalidad, para un nivel de significancia de 0.01

Una vez que se aceptó la normalidad de ambas lecturas, se procedió a verificar que las varianzas fueran diferentes, para lo cual se realizó una prueba F por medio de EXCEL obteniendo el resultado que se muestra en la Tabla 5.2. Gracias a la información que proporciona esta prueba se puede afirmar que las varianzas de las muestras son diferentes, ya que el valor F estadístico de las muestras es mayor que el valor F crítico, con un nivel de significancia de 0.01.

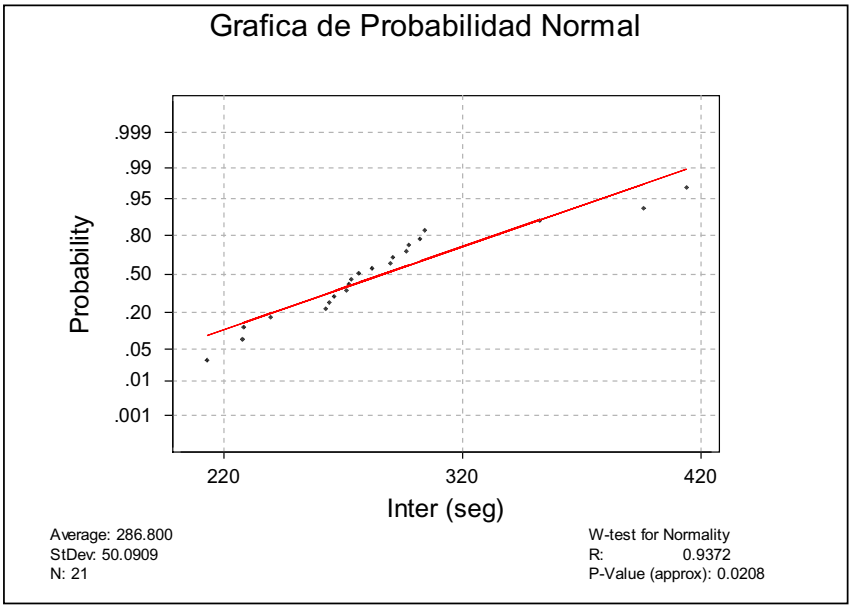


Figura 5.2 Resultado de la prueba de Ryan – Joiner para las lecturas en la interfaz.

	Terminal (seg.)	**Interfaz (seg.)**
Media	446.8585714	286.8004762
Varianza	15733.73079	2509.094065
Observaciones	21	21
Grados de libertad	20	20
F	6.270681922	
P(F<=f) una cola	6.76888E-05	
F Crítica para una cola	2.93772473	

Tabla 5.2 Prueba F de dos muestras para varianzas

Finalmente se realizó la prueba T para dos muestras normales asumiendo varianzas diferentes, por medio de EXCEL obteniendo el resultado que se muestra en la tabla 5.3. La prueba se planteó asumiendo una diferencia entre ellas de “0”, es decir, que eran iguales. Basándose en

los datos que ahí se presentan se puede aceptar que la media de los tiempos tomados en la terminal es diferente a la media de los tiempos tomados con la interfaz, con un nivel de significancia de 0.01, ya que el valor t estadístico es mayor a los valores t críticos. Necesariamente la media de los tiempos en la terminal es mayor a la media de los tiempos en la interfaz, y como se mencionó anteriormente la diferencia es aproximadamente de dos minutos y medio.

	Terminal (seg.)	**Interfaz (seg.)**
Media	446.8585714	286.8004762
Varianza	15733.73079	2509.094065
Observaciones	21	21
Diferencia Hipotética de Medias	0	
Grados de libertad	26	
t Estadística	5.430517872	
P(T<=t) una cola	5.41413E-06	
t Crítica para una cola	2.478627721	
P(T<=t) dos colas	1.08283E-05	
t Crítica para dos colas	2.778724593	

Tabla 5.3 Prueba T para dos muestras asumiendo varianzas diferentes.

El otro aspecto que se tomó en cuenta, fueron los errores en el código del programa. Principalmente los errores que se detectaron fueron omisiones de instrucciones necesarias para que el robot funcione adecuadamente. En la tabla 5.1 puede apreciarse que en la prueba con la interfaz, ninguna persona cometió un error de este tipo. Aunque esto no asegura que el robot vaya a realizar su tarea de forma correcta en cualquier otro programa que se elabore, ya que no se revisa la trayectoria que debe seguir el brazo en el espacio al pasar por los puntos que se le programan, como lo haría un simulador, sin embargo sí se asegura que la comunicación con los equipos que deben coordinarse con el robot para realizar alguna operación sea completa, lo cual puede evitar movimientos no coordinados del robot con algún otro equipo que potencialmente signifiquen un riesgo de que suceda una colisión.

5.3 Conclusiones

La necesidad de las empresas por mantenerse en el mercado ha hecho que éstas busquen la manera de ampliar sus horizontes y de llegar cada vez a más clientes, incluso a niveles mundiales. En épocas pasadas muchos de los productos que se hacían, sobre todo con fines tecnológicos, se elaboraban específicamente para un mercado en especial, como las computadoras, cámaras fotográficas y posteriormente los paquetes de computadora como los lenguajes de programación, por mencionar algunos ejemplos. Los productos eran poco amigables con usuarios, en especial con los que se iniciaban en esas áreas, por lo que para manejarlos adecuadamente se requería de algún conocimiento previo o de algún experto que pudiera ayudarlos.

La inminente competencia entre empresas provoca que cada una de ellas quiera incrementar su mercado, para lo cual es necesario hacer algunas adaptaciones a sus productos o algunos cambios de tal manera que cada vez más gente desee adquirir alguno de estos bienes. En el caso de los productos mencionados tenemos que las computadoras ya no pueden verse solamente como instrumentos de trabajo de los grandes centros de investigación, sino que una gran cantidad de personas adquieren una para su casa, ya sea con fines educativos o de entretenimiento. Las cámaras fotográficas, junto con los rollos de película, cada vez requieren de menos conocimientos en cuanto al manejo de luces, sombras, distancias, etc. para obtener una fotografía aceptable, casi profesional. De igual manera, los lenguajes de programación cuentan ahora con interfaces más amigables que permiten a muchos de sus clientes hacer aplicaciones que, aunque sencillas, antes podían esperarse sólo de un técnico programador o de un ingeniero en ciencias computacionales.

En el caso del mercado de robots sucede algo semejante, los fabricantes de estas máquinas cada vez desarrollan más aplicaciones en las cuales la interacción con el usuario se vuelve más amigable, desde programas para la manipulación directa del brazo, la programación fuera de línea hasta la simulación del movimiento del robot, con la posibilidad de detectar colisiones antes de que éstas sucedan. Todas estas ventajas resultan atractivas para cualquier empresario que desee incluir un robot en sus líneas de producción, más aún para los que ya lo tienen y

desean sacarle más provecho, y resulta aún mejor si se puede realizar de una manera sencilla, con algunos conocimientos previos pero sin la necesidad de traer al técnico de otra ciudad o más aún de otro país.

Las interfaces sirven como un vínculo entre el usuario y la máquina, que tienen como objetivo hacer que el usuario pueda solucionar su problema referente al manejo de la máquina, de una manera más fácil. En el caso de los paquetes computacionales, finalmente toda la información se convierte a código máquina para poder ser interpretado por la computadora, o las cámaras fotográficas, las cuales, a final de cuentas, logran la impresión de una imagen en un papel.

De igual manera, esta interfaz lo que hace es ayudar al usuario a resolver el problema del manejo del robot, de una manera sencilla, presentando por medio de botones de órdenes, las instrucciones más comunes para la manipulación del brazo, sin embargo, el comando que se envía al controlador es el mismo que si se tecleara en la terminal del sistema. Lo mismo sucede en el editor de programas, el resultado es un código que debe ser el mismo al que se tuviera que teclear en el editor de la terminal.

Con la elaboración de la Interfaz de Usuario para el robot PUMA 560 de Unimation, quedó demostrado que con una interfaz amigable y bajo un ambiente familiar para el que la opera, se puede programar al robot para realizar una tarea en un tiempo menor al necesario para hacerlo de la manera tradicional, además de reducir la probabilidad de cometer errores por omisión de comandos en la elaboración del programa. En este caso se trataba de un programa bastante sencillo, pues el tiempo que tomaba hacerlo en la terminal era apenas de siete u ocho minutos en promedio, y la reducción del tiempo fue de dos minutos y medio, es decir un 32% de ahorro en tiempo.

Sin embargo a medida que la programación de tareas se vuelve más compleja, la ayuda del programador puede verse más significativa, tanto en la disminución del tiempo de programación, en la reducción de errores por omisión de comandos y más aún, en la reducción de la variabilidad del proceso. Este último aspecto representa una ventaja muy importante, ya que el tener un mejor control de cualquier proceso, reduciendo su variabilidad, hace que sea

más confiable para la programación de actividades y el control de calidad de las mismas. En la Tabla 5.1 se presentan las desviaciones estándar de cada muestra de tiempos, estas nos indican que la variabilidad del proceso de programación usando la interfaz, resulta mucho menor que la variabilidad al programar en la terminal. Esta última se ve afectada por la habilidad de los usuarios, debido a que algunos sabían de memoria todos los comandos necesarios, otros tenían que consultar de vez en cuando la hoja que se les proporcionó, y otros la tuvieron que consultar para todos los comandos. Este factor se ve reducido con el uso de la interfaz al presentar los comandos en un lenguaje familiar para la persona que está haciendo el programa, haciendo que los tiempos de programación sean más constantes.

El editor de programas también tiene un impacto en la educación, ya que representa una ayuda didáctica para quien desea aprender el lenguaje de programación del robot. Al ir oprimiendo los botones de inserción de instrucciones, aparece en el área de edición del programa el comando correspondiente en el lenguaje del robot, permitiendo así que el alumno relacione la instrucción que está solicitando con el código insertado en el programa.

5.4 Investigaciones Futuras

El desarrollo de la Interfaz de Usuario para el robot PUMA 560 es apenas un inicio para otras aplicaciones que pueden desarrollarse con este robot. Una de ellas, y tal vez sea el paso que sigue, es la realización de un simulador de los movimientos del brazo.

Para la elaboración de este tipo de software se requiere calcular las matrices de la cinemática directa e inversa de este robot en particular, sin embargo, éstas pueden conseguirse de libros de robótica, de otras tesis que se han desarrollado enfocadas al PUMA [37] y [38] o incluso de páginas de Internet en las que se publican trabajos sobre este robot. Esto aunado con la facilidad que se tiene ahora con la realización de esta tesis para obtener las coordenadas de las posiciones grabadas en el controlador, se tiene la información necesaria para hacer una representación gráfica de las posiciones del robot en el espacio. Faltaría solamente ubicar las dos máquinas de control numérico, la banda transportadora con sus dos paros de transferencia,

así como el riel transversal para definir zonas de riesgo donde el robot pudiera golpear alguno de estos equipos con los que tiene que trabajar.

Otro proyecto que puede resultar interesante también con el robot PUMA es la sustitución de la unidad de disco flexible de 5.25", la cual es la que se encuentra instalada actualmente en el sistema, por una unidad de disco flexible de 3.5" o por una computadora personal. El protocolo de comunicación es el mismo que para la terminal, y es el estándar RS232. En los manuales de usuario del robot [35], se tiene la información de cómo se envían instrucciones del controlador a la unidad de disco y de la manera como éste responde, haciendo posible su emulación desde otra computadora. Durante el desarrollo de este proyecto se investigó el costo de una unidad de disco comercial de 3.5" especial para este robot y es aproximadamente de 1,000 dólares.

Finalmente propongo también la elaboración de un software que haga más fácil el manejo de la computadora central para programar una integración. Puede ser que al momento de ejecutarse envíe las mismas señales que el programa actual, sin embargo la amigabilidad del nuevo sistema, así como la reducción en la probabilidad de cometer errores, lo hace más atractivo para cualquier usuario.

RESUMEN

Las empresas manufactureras de hoy en día se ven en la necesidad de actualizar sus medios de producción constantemente, para poder competir con el resto de las compañías. Dos temas que tienen bastante peso en este tema son el manejo de información y la optimización de los procesos de producción, sobre todo cuando se trata de automatizar e integrar las actividades de la empresa. Uno de los recursos más utilizados con este propósito es el uso de los robots industriales, mediante los cuales se puede lograr la realización de las tareas productivas y la retroalimentación de lo que está sucediendo en la estación de trabajo, muchas veces en tiempo real.

Los robots pueden ser usados en cualquier industria que sea proveedora de bienes o servicios, estos pueden ser adaptados a trabajos de diferentes tipos y trabajar con gran habilidad y resistencia a lo largo de jornadas completas de trabajo. Los fabricantes de robots, además de apoyarse en esta funcionalidad de los mismos, ofrecen una variedad de softwares que permiten al usuario aprovechar al máximo su equipo, haciendo más atractiva la compra e implantación de un robot para estos clientes en sus sistemas productivos. Algunas de las ventajas que ofrecen son: la programación de estos equipos cada vez de una manera más sencilla, la simulación de los movimientos del robot previa a la corrida real para evitar accidentes y una mayor capacidad de almacenaje, entre otras.

En el Laboratorio de Sistemas Integrados de Manufactura, clase que se imparte a los alumnos de la carrera de Ingeniería Industrial y de Sistemas, se cuenta con una celda flexible de manufactura, la cual está formada por tres robots, dos máquinas de control numérico, una banda transportadora, un almacén automático, una estación de inspección por visión y una computadora central. Esta celda fue instalada en el ITESM en el año de 1992 y con el paso del tiempo se han desarrollado aplicaciones que facilitan la operación de algunos de sus equipos.

El presente trabajo de investigación trata del desarrollo de una interfaz de usuario para el robot PUMA 560 de Unimation, el cual forma parte de esta celda y es el encargado de abastecer

materia prima a los dos centros de maquinado. Actualmente la forma de programar a este robot es por medio de la terminal del sistema, en ella, el usuario debe teclear el comando que requiera para que esta instrucción sea enviada al controlador del robot y que posteriormente se ejecute por el brazo manipulador. La intención de la interfaz de usuario es sustituir el uso de la terminal del sistema, por medio de una pantalla, en ambiente Windows, en la que se tienen botones de órdenes con los comandos típicamente utilizados en una práctica del laboratorio. Las instrucciones de igual forma son enviadas al controlador del robot para posteriormente ordenar su ejecución, sin embargo, el alumno tiene en una pantalla una cantidad finita de opciones, haciendo mucho más fácil la operación del brazo que cuando tiene una terminal que le permite la entrada de texto libre.

La interfaz cuenta con tres elementos principales: un panel de control principal, un editor de programas y un modo emulador de la terminal del sistema. En el panel de control principal están las opciones que el usuario necesita para enviar instrucciones directamente al controlador del brazo, como es grabar una posición en el controlador o mover al brazo a una posición previamente grabada. El editor de programas cuenta con las opciones más comunes para la realización de un programa en el lenguaje del robot, para su utilización no es necesario estar conectado con el robot. El modo emulador de la terminal del sistema es una pantalla que permite al usuario la introducción de texto libre, si es que se presentara la necesidad de enviar algún comando al controlador que no se encuentra contemplado en el panel principal, por lo que es necesario estar conectado con el robot para poder utilizar este modo de operación.

Para concluir, se realizaron algunas pruebas con alumnos e instructores del laboratorio, para evaluar el beneficio real de la utilización de la interfaz. Principalmente se enfocó el estudio al uso del programador y los resultados mostraron que sí hay un beneficio palpable con la realización de la interfaz, ya que es posible escribir un programa, en el lenguaje del robot, utilizando el editor de programas, en un tiempo menor que haciéndolo directamente en la terminal del sistema. Además de reducir considerablemente la posibilidad de cometer errores por omisión de comandos necesarios para que las tareas sean realizadas por el brazo adecuadamente.

BIBLIOGRAFÍA

[1] Chang T., Wysk, R. y Wang H., *Computer-Aided Manufacturing*, Prentice Hall, 1998, Englewood Cliffs, N.J. USA

[2] Tsai, Lung-Wen, *Robot Analysis, The Mechanics of Serial and Parallel Manipulators*, Wiley-Interscience, 1999, New York, N. Y. USA

[3] Rehg, J., *Introduction to Robotics in CIM Systems*, Prentice Hall, 2000, Upper Saddle River, N.J. USA

[4] Fisher, Edward L. *Robotics and Industrial Engineering, Selected Readings*, Industrial Engineering and Management Press, Institute of Industrial Engineerings, 1983-86, Atlanta GA. USA

[5] Rembold Ulrich, *Robot Technology and Applications*, Dekker, 1990, New York, N. Y. USA

[6] Malcom, Douglas R., *Robotics: An Introduction*, PWS-Kent Publishing Company, 1988, Boston, MA. USA

[7] Spencer, Rob, *Despite tough times, robotics industry fundamentally strong*, Robotics World, Jan/Feb 2002, Atlanta, GA. USA

[8] ABB Corporate Management Services AG, *Revista AB.* ABB Ltd, 2000. Zurich, Suiza

[9] Chase, Richard B. y Aquilano, Nicholas J., *Production and Operations Management: A life cycle approach*, Irwin, 1992, Homewood, IL. USA

[10] Williams T., Rathwell G. & Li H., A Handbook on Master Planning and Implementation for Enterprise Integration Programs, Purdue University, 1996, USA

[11] Mitchell, F.H., *CIM Systems, An Introduction to Computer- Integrated Manufacturing*, Prentice Hall, 1991, Englewood Cliffs, NJ. USA

[12] Spencer, Rob, *Cartesian, SCARA or articulated arm: Choose the right robot for the job*, Robotics World, 2001,Atlanta, GA, USA

[13] Nof, Y. Shimon, *Handbook of Industrial Robotics*, John Wiley & Sons, 1985, USA

[14] Spencer , Rob, *Coating robots migrate: From auto industry into general industry*, Robotics World, 2001,Atlanta, GA, USA

[15] Spencer, Rob, *Robots add flexibility to machining, machine-tending applications*, Robotics World, 2001,Atlanta, GA, USA

[16] Unimate Industrial Robot, *PUMA MARK II ROBOT, 500 SERIES, Volume I – Equipment Manual*, Unimation, 1983, Danbury Connecticut. USA

[17] Zhou Rong & Sha Xin, *Development of An Off- Line Graphical PUMA 560 Robot Simulator*, Department of Industrial Engineering, Mississippi State University, USA (http://www.cs.msstate.edu/~rzhou/Simulator.htm)

[18] Anonymous, *Better Machine Loading Through Software*, American Machinist, 2001, Cleveland OH, USA

[19] Ovo Dafe, *Off- Line Robot Programming and Simulation*, University of Iowa, July 1995, USA (http://www.cs.uiowa.edu/reu/summer95/Ovo-Dafe/ovo/node1.html)

[20] Theoretical Biophysics Group, *Industrial Robot Simulator*, NIH Resource for Macromolecular Modeling and Bioinformatics, May 2000 (http://www.ks.uiuc.edu/Research/Neural/puma.html)

[21] http://www.winternet.com/~trostad/puma/

[22] Vamoser Damir P. & Sobh Tarek, *Puma 560 Simulator*, University of Bridgeport, April 1997, Connecticut, USA (http://www.bridgeport.edu/~sobh/html/proj/damir/po.html)

[23] Jägersand Martin, *An Off- Line PUMA 760 Robot Simulator*, University of Rochester, February 1999, Rochester, N.Y. USA

[24] http://nondot.org/sabre/Projects/#GRAS

[25] Lattner Chris, *The GRAS robotics simulator* (http:// nondot.org/sabre/java/GRAS/)

[26] Lattner Chis & Hsu Ming-Shu, *Developing a Graphical Robotics Simulator*, School of Engineering, University of Portland, 1998, Portland, OR, USA

[27] Quality Real Time Systems, *Simulink Robotic Toolkit for the Puma 560*, (http://qrts.com/products/pumartk)

[28] Quality Real Time Systems, *Simulink Based Robotic Tool Kit for the Puma 560 Robot Arm, Manual Version 1.0*, QRTS, (http://qrts.com/products/pumartk/)

[29] http://www-cia.mty.itesm.mx/~gordillo/REDII/Reporte/contenido.html

[30] Molina Falcón, Gustavo A., *Simulador del Sistema Automático de Almacenamiento, Carga y Descarga 862 de Amatrol*, Tesis para obtener el grado académico de Maestro en Ciencias con especialidad en Ingeniería Industrial, Diciembre 1994, Monterrey, N.L. México.

[31] Cárdenas Barrón, Leopoldo E., *Programador del Robot Júpiter XL de Amatrol*, Tesis para obtener el grado académico de Maestro en Ciencias con especialidad en Sistemas de Manufactura, Mayo 1995, ITESM, Monterrey, N.L. México

[32] Cornell Gary, Manual de Visual Basic 4 para Windows 95, Traducción de Carlos de la Fuente Chacón, McGraw Hill, 1996, España

[33] Black Box Corporation, *232 Monitor User's Guide*, Black Box Corporation, 1989, Pittsburgh, PA, USA

[34] Unimate Industrial Robot, *Programming Manual, User's Guide to VAL II, Part 1 – Control from the System Terminal*, Unimation, 1985, Danbury Connecticut. USA

[35] Unimate Industrial Robot, *Programming Manual, User's Guide to VAL II, Part 2 – Communication with a Supervisory System*, Unimation, 1985, Danbury Connecticut. USA

[36] Devore, Jay L., *Probabilidad y estadística para ingeniería y ciencias*, International Thompson Editores, 2001, México

[37] Cantú de la Garza, Adrián Epifanio, *Simulación y control de un manipulador tipo PUMA y un motor sincrónico para un AGV*, Tesis para obtener el grado académico de Maestro en Ciencias con especialidad en Ingeniería de Control, 1998, ITESM, Monterrey, N.L. México

[38] López Pérez, José Felipe, *Análisis y diseño de sistemas de control para el robot PUMA*, Tesis para obtener el grado académico de Maestro en Ciencias con especialidad en Ingeniería de Control, 1996, ITESM, Monterrey, N.L. México

[39] http://www.ifr.org/ (International Federation of Robotics)

[40] http://www.antenen.com/htdocs/frame.html (Antenen Research)

[41] http://www.ar2.com/ (Advance Research & Robotics)

Anexos

Anexo A

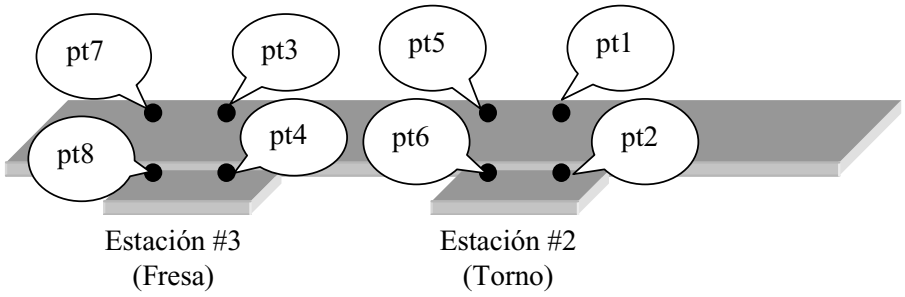
Hojas Proporcionadas a los Participantes para la Realización de las Pruebas

---

Descripción de la Tarea

Siguiendo el diagrama de puntos y el algoritmo de programación que se muestran en las siguientes secciones, escribir el código del programa necesario para que el robot PUMA 560 tome una por una, las piezas de aluminio que se llegan en un pallet a la estación #2 y las ensamble en los orificios correspondientes en la placa de acrílico que llegará en un pallet a la estación #3.

Diagrama de Puntos



Algoritmo de Programación

1. Inicializar el robot:
    - Apagar señales de salida (1 a 8)
    - Enviar al robot a su posición de inicio (HOME)
2. Declarar configuración de brazo derecho
3. Declarar velocidad de 90 para todo el programa
4. Abrir el gripper
5. Mover el brazo al punto 1, en movimiento articulado
6. Esperar la llegada de un pallet a la estación #2
7. Mover el brazo al punto 2, en movimiento rectilíneo
8. Cerrar el gripper
9. Mover el brazo al punto 1, en movimiento rectilíneo
10. Mover el brazo al punto 3, en movimiento articulado
11. Esperar la llegada de un pallet a la estación #3
12. Mover el brazo al punto 4, en movimiento rectilíneo
13. Abrir el gripper
14. Mover el brazo al punto 3, en movimiento rectilíneo
15. Mover el brazo al punto 5, en movimiento articulado
16. Mover el brazo al punto 6, en movimiento rectilíneo
17. Cerrar el gripper
18. Mover el brazo al punto 5, en movimiento rectilíneo
19. Mover el brazo al punto 7, en movimiento articulado
20. Mover el brazo al punto 8, en movimiento rectilíneo
21. Abrir el gripper
22. Mover el brazo al punto 7, en movimiento rectilíneo
23. Liberar el pallet de la estación #2
24. Liberar el pallet de la estación #3
25. Regresar el robot a HOME

Códigos de Programación

Descripción	Código
Encender una señal de salida	SIGNAL # (# = número de señal)
Apagar una señal de salida	SIGNAL -# (# = número de señal)
Esperar el encendido de una señal de entrada	WAIT SIG(#) (# = número de señal)
Esperar el apagado de una señal de entrada	WAIT SIG(-#) (# = número de señal)
Enviar el robot a HOME	READY
Configuración de brazo derecho	RIGHTY
Declaración de velocidad (para todo el programa)	SPEED # ALWAYS (# = velocidad)
Declaración de velocidad (para un movimiento)	SPEED # (# = velocidad)
Retraso de tiempo	DELAY # (# = tiempo en segundos)
Abrir el Gripper	OPENI
Cerrar el Gripper	CLOSEI
Mover el brazo a un punto en movimiento articulado	MOVE punto (punto = nombre del punto)
Mover el brazo a un punto en movimiento rectilíneo	MOVES punto

Señales de Salida

No.	Descripción
1	Comunicación PUMA – Fresa
2	Comunicación PUMA – Torno
3	Liberar pallet de estación #2
4	Liberar pallet de estación #3
5	Desplazamiento hacia el controlador
6	Fijación del lado de la fresa
7	Desplazamiento hacia la fresa
8	Fijación del lado del controlador

Señales de Entrada

Número	Descripción
1001	Comunicación Fresa – PUMA
1002	Comunicación Torno – PUMA
1003	Pallet presente en la estación #2
1004	Pallet presente en la estación #3
1005	Robot del lado de la fresa
1006	Robot del lado del controlador

Comandos para edición de programas en la terminal del robot PUMA 560

Comando	Descripción
Edit “Nombre”	Iniciar la edición de un programa (nombre = nombre del programa)
L	Regresar una línea dentro de la edición de un programa
D	Borrar una línea dentro de la edición de un programa
I	Insertar una línea dentro de la edición de un programa
E	Salir del editor de programas

Anexo B

Tabla de Valores Críticos

para la Prueba de Normalidad de Ryan - Joiner

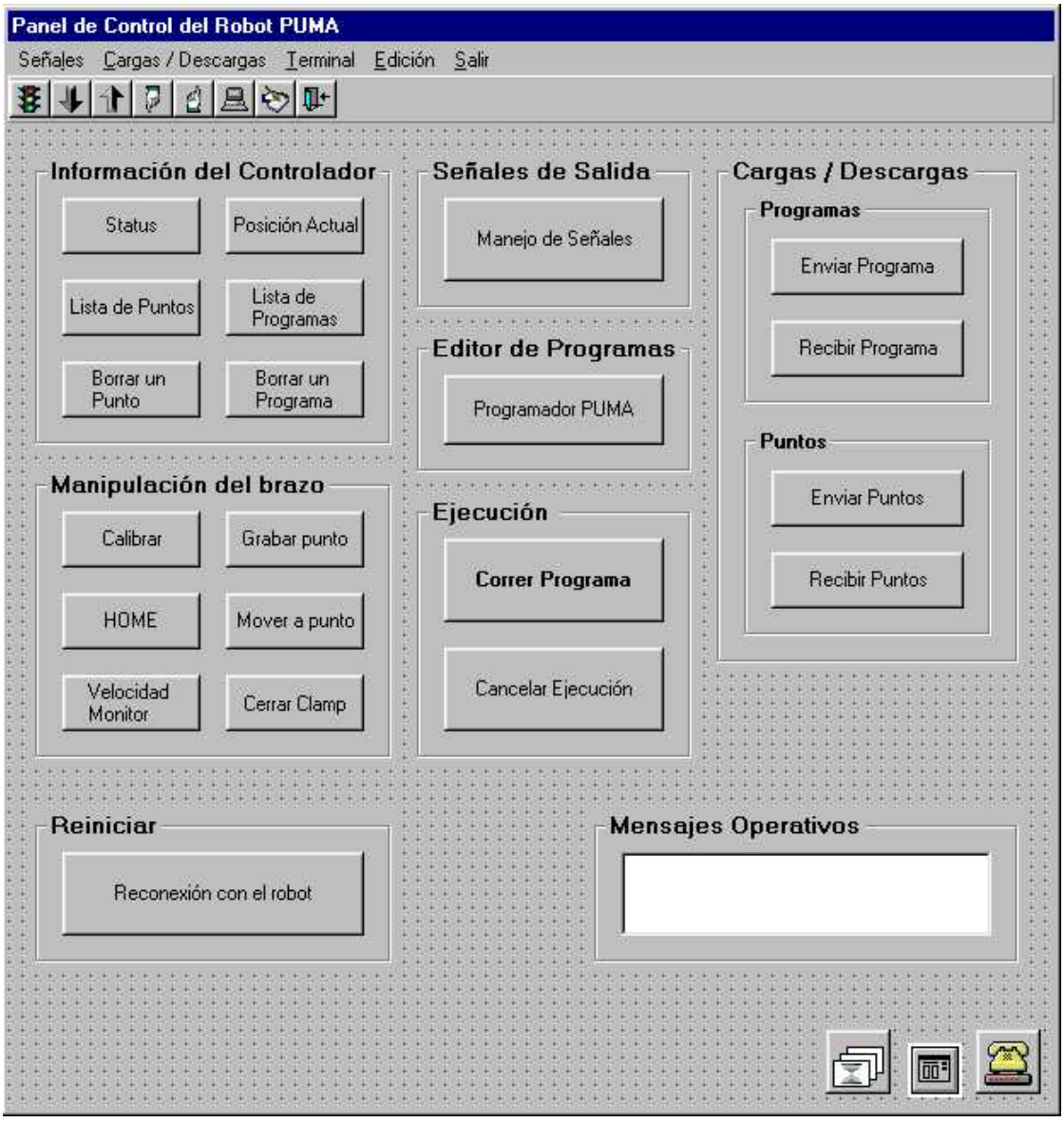
		α		
		0.10	0.05	0.01
n	5	0.9033	0.8804	0.8320
	10	0.9347	0.9180	0.8804
	15	0.9506	0.9383	0.9110
	20	0.9600	0.9503	0.9290
	25	0.9662	0.9582	0.9408
	30	0.9709	0.9639	0.9490
	40	0.9767	0.9715	0.9597
	50	0.9807	0.9764	0.9664
	60	0.9835	0.9799	0.9710
	75	0.9865	0.9835	0.9757

Anexo C

Forma y Código

del Panel de Control de la Interfaz de Usuario

Forma del Panel de Control del Robot PUMA



Código del Panel de Control del Robot PUMA

```
Public conRobot As Boolean
Public NombrePr As String
```

```
Private Sub Abortar_Click()
   MsgOper.Text = "Ejecución Cancelada..."
   Panel_Principal.Pfn_Mandar_Cadena ("a")
   Abortar.Enabled = False
   activar_botones
   Restart.Enabled = True
   MsgOper.Text = ""
End Sub
```

```
Private Sub Editor_Click()
   Edición.Show
End Sub
```

```
Private Sub Home_Click()
   Form_Load
   If conRobot Then
      MsgOper.Text = "Robot en movimiento a HOME"
      apagar_botones
      Restart.Enabled = False
      Pfn_Mandar_Cadena ("DO READY")
      MsgOper.Text = ""
      activar_botones
      Restart.Enabled = True
   End If
End Sub
```

```
Private Sub Listl_Click()
Dim mssge As String
   Form_Load
   If conRobot Then
      mssge = "Recuperando lista de puntos" + vbCrLf
      mssge = mssge + "Espere un momento, por favor..."
      MsgOper.Text = mssge
      Restart.Enabled = False
      apagar_botones
      ListLocations.Show
      activar_botones
      Restart.Enabled = True
   End If
End Sub
```

```
Private Sub Dir_Click()
   Form_Load
   If conRobot Then
      Pfn_Mandar_Cadena ("DIR")
   End If
End Sub
```

```
Private Sub Exec_Click()
   Dim mssge As String
   Dim orden As String

   Form_Load
   If conRobot Then
      MostrarProgs.Show 1
      mssge = "Robot no disponible" + vbCrLf
      mssge = mssge + "Ejecutando Programa:" + vbCrLf
      mssge = mssge + NombrePr
      MsgOper.Text = mssge
      apagar_botones
      Restart.Enabled = False
      If NombrePr <> "" Then
         Abortar.Enabled = True
         orden = "EXEC " + NombrePr
         Pfn_Mandar_Cadena (orden)
      End If
      Abortar.Enabled = False
      MsgOper.Text = ""
      activar_botones
      Restart.Enabled = True
   End If
End Sub
```

```
Private Sub mnuEditorPrg_Click()
   Edición.Show
End Sub
```

```
Private Sub mnuEnviarPrg_Click()
   Enviar_Prog_Click
End Sub
```

```
Private Sub mnuEnviarPts_Click()
   Enviar_Ptos_Click
End Sub
```

```
Private Sub mnuFinProg_Click()
   If MSComm1.PortOpen Then
      MSComm1.PortOpen = False
```

```
    End If
    End
End Sub
```

```
Private Sub mnuManSeñSal_Click()
    SeñalesSalidas.Show
End Sub
```

```
Private Sub mnuModoTerm_Click()
    TermMode.Show
End Sub
```

```
Private Sub mnuRecibirPrg_Click()
    Recibir_Prog_Click
End Sub
```

```
Private Sub mnuRecibirPts_Click()
    Recibir_Ptos_Click
End Sub
```

```
Private Sub Recibir_Prog_Click()
    Dim Archivo As String
    Dim contlinea As Integer
    Dim contcaran As Integer
    Dim contcarac As Integer
    Dim numlf As Integer
    Dim Ls_Out As String
    Dim Ls_In As String
    Dim Ls_In_Ant As String
    Dim As_cadena As String
    Dim mssge As String
    Dim aviso As String
    Dim primlet As Integer
    Dim I As Integer
    Dim J As Integer
    Dim numpts As Integer
    Dim Listapts(200) As String
    Dim nombrePt As String
    Dim ya As Boolean

    On Error GoTo ManejarErrores

    Form_Load
    If conRobot Then
    apagar_botones
    Restart.Enabled = False
    MostrarProgs.Show 1
```

```
If NombrePr <> "" Then
   CommonDialog1.DialogTitle = "Recepción de programa del robot PUMA a disco"
   CommonDialog1.DefaultExt = "prp"
   CommonDialog1.Filter = "Programas PUMA|*.prp"
   CommonDialog1.InitDir = "C:\PUMA"
   CommonDialog1.Flags = cdlOFNOverwritePrompt
   CommonDialog1.FileName = ""
   CommonDialog1.ShowSave

   As_cadena = "EDIT " + NombrePr

   Archivo = CommonDialog1.FileName
   Open Archivo For Output As #1

   mssge = "Recibiendo Programa en Disco" + vbCrLf
   mssge = mssge + "Por favor espere" + vbCrLf
   MsgOper.Text = mssge
   Mandar_Comando (As_cadena)

   contlinea = 0
   mssge = ""
   numpts = 1
   numlf = 0
   Do
      mssge = ""
      numlf = 0
      Do
         Do
            DoEvents
         Loop Until MSComm1.InBufferCount >= 1
         Ls_In = MSComm1.Input
         If Ls_In = vbLf Then
            numlf = numlf + 1
         End If
         mssge = mssge + Ls_In
      Loop Until Ls_In = "?"
      contcarac = Len(mssge)

      If contcarac > 7 Then
         contlinea = contlinea + 1
         aviso = "Recibiendo Programa en Disco" + vbCrLf
         aviso = aviso + "Por favor espere" + vbCrLf
         aviso = aviso + "Línea: " + CStr(contlinea)
         MsgOper.Text = aviso
         If numlf = 3 Then
            I = 10
```

```
            primlet = 0
            Do
               If Mid(mssge, I, 3) = " 1 " Then
                  primlet = I
               End If
               I = I + 1
            Loop Until primlet <> 0
            mssge = Mid(mssge, primlet - 4, Len(mssge) - (primlet - 5))
         End If
         I = 7
         primlet = 0
         Do
            If Mid(mssge, I, 1) <> " " Then
               primlet = I
            End If
            I = I + 1
         Loop Until primlet <> 0
         mssge = Mid(mssge, primlet, Len(mssge) - (primlet + 6))
         msgge = mssge + vbCr
         Print #1, mssge
         For I = 1 To Len(mssge) - 4
            If Mid(mssge, I, 4) = "MOVE" Then
               If Mid(mssge, I + 4, 1) = "S" Then
                  nombrePt = Mid(mssge, I + 6, Len(mssge) - (I + 5))
               Else
                  nombrePt = Mid(mssge, I + 5, Len(mssge) - (I + 4))
               End If
               ya = False
               For J = 1 To numpts
                  If nombrePt = Listapts(J) Then
                     ya = True
                  End If
               Next J
               If Not (ya) Then
                  Listapts(numpts) = nombrePt
                  numpts = numpts + 1
               End If
            End If
         Next I
      End If
      MSComm1.Output = vbCr
   Loop Until contcarac = 7

   MSComm1.Output = "E " + vbCr

   Do
      Do
```

```
            DoEvents
         Loop Until MSComm1.InBufferCount >= 1
         Ls_In = MSComm1.Input
      Loop Until Ls_In = "."
      Close #1

      mssge = "¿Desea crear un archivo con la lista de puntos?"
      I = MsgBox(mssge, vbQuestion + vbYesNo + vbDefaultButton1, "Recepción de
programa del Robot PUMA a diskette")

      mssge = "El Programa se ha recibido exitosamente" + vbCrLf
      mssge = mssge + "bajo el nombre: " + Archivo + vbCrLf + vbCrLf
      mssge = mssge + "Se recibieron: " + CStr(contlinea - 1) + " líneas." + vbCrLf + vbCrLf

      If I = 6 Then
         CommonDialog1.DialogTitle = "Creación de la lista de puntos de un programa"
         CommonDialog1.DefaultExt = "ptl"
         CommonDialog1.Filter = "Lista de puntos|*.ptl"
         CommonDialog1.InitDir = "C:\PUMA"
         CommonDialog1.Flags = cdlOFNOverwritePrompt
         CommonDialog1.FileName = ""
         CommonDialog1.ShowSave

         As_cadena = CommonDialog1.FileName
         Open As_cadena For Output As #1
         For J = 1 To numpts - 1
            Print #1, Listapts(J)
         Next J
         Close #1
         mssge = mssge + "Se creó el archivo: " + As_cadena + " con: "
         If numpts = 2 Then
            mssge = mssge + " 1 punto."
         Else
            mssge = mssge + CStr(numpts - 1) + " puntos."
         End If
      End If
      MsgBox mssge, vbInfo + vbOKOnly, "Recepción de programa del Robot PUMA a
diskette"
   End If
ManejarErrores:
   If Err.Number <> 0 Then
      If Err.Number <> cdlCancel Then
         MsgBox "Error " & Err.Number & vbCr & vbCr & Err.Description, vbCritical +
vbOKOnly, "Error en el proceso"
         Err.Clear
      End If
   End If
```

```
    activar_botones
    Restart.Enabled = True
    MsgOper.Text = ""
    End If
End Sub
```

```
Private Sub Enviar_Ptos_Click()
    Dim Archivo As String
    Dim li_cont As Integer
    Dim Ls_Out As String
    Dim Ls_In As String
    Dim Ls_In_Ant As String
    Dim As_cadena As String
    Dim mssge As String
    Dim nomPto As String
    Dim datopto As String
    Dim coords As String
    Dim ultlet As Integer
    Dim I As Integer
    Dim numpts As Integer

    On Error GoTo ManejarErrores
    Form_Load
    If conRobot Then
    CommonDialog1.DialogTitle = "Envío de puntos al robot PUMA"
    CommonDialog1.Filter = "Puntos PUMA|*.ptp|Todos los Archivos|*.*|"
    CommonDialog1.InitDir = App.Path
    CommonDialog1.FileName = ""
    CommonDialog1.ShowOpen
    Archivo = CommonDialog1.FileName
    apagar_botones
    Restart.Enabled = False
    Open Archivo For Input As #1
    mssge = "Enviando Puntos al Controlador" + vbCrLf
    mssge = mssge + "Por favor espere" + vbCrLf
    MsgOper.Text = mssge
    numpts = 0

    Do Until EOF(1)
       Line Input #1, mssge
       ultlet = 0
       I = 2
       Do
          If Mid(mssge, I, 1) = " " Then
             ultlet = I - 1
          End If
```

```
    I = I + 1
  Loop Until ultlet <> 0
  nomPto = Mid(mssge, 1, ultlet)
  mssge = Mid(mssge, ultlet + 1, Len(mssge) - ultlet)
  ultlet = 0
  I = 1
  Do
    If Mid(mssge, I, 1) <> " " Then
      ultlet = I
    End If
    I = I + 1
  Loop Until ultlet <> 0

  mssge = Mid(mssge, ultlet, Len(mssge) - (ultlet - 1))
  coords = ""
  For I = 1 To Len(mssge)
    If Mid(mssge, I, 1) <> " " Then
      coords = coords + Mid(mssge, I, 1)
    Else
      If Mid(mssge, I - 1, 1) <> " " Then
        coords = coords + ","
      End If
    End If
  Next I
  As_cadena = "POINT " + nomPto
  Mandar_Comando (As_cadena)

  Do
    Ls_In = MSComm1.Input
  Loop Until Ls_In = vbLf

  datopto = ""
  Do
    Do
      DoEvents
    Loop Until MSComm1.InBufferCount >= 1
    Ls_In = MSComm1.Input
    datopto = datopto + Ls_In
  Loop Until Ls_In = "?"
  datopto = Mid(datopto, 58, 49)
  If datopto = "0.00     0.00     0.00   90.000  -90.000    0.000" Then
    Mandar_Comando (coords)
    numpts = numpts + 1
    Do
      Do
        DoEvents
      Loop Until MSComm1.InBufferCount >= 1
```

```
            Ls_In = MSComm1.Input
          Loop Until Ls_In = "?"
        Else
          mssge = "El punto " + nomPto + " ya existe en el controlador, " + vbCrLf
          mssge = mssge + "¿Desea reemplazarlo? (Y/N)" + vbCrLf + vbCrLf
          opt = MsgBox(mssge, vbQuestion + vbYesNo + vbDefaultButton2, "Confirmación de
instrucción")
          If opt = 6 Then
            Mandar_Comando (coords)
            numpts = numpts + 1
            Do
              Do
                DoEvents
              Loop Until MSComm1.InBufferCount >= 1
              Ls_In = MSComm1.Input
            Loop Until Ls_In = "?"
          End If
        End If
        MSComm1.Output = vbCr
        Do
          Do
            DoEvents
          Loop Until MSComm1.InBufferCount >= 1
          Ls_In = MSComm1.Input
        Loop Until Ls_In = "."
    Loop

    Close #1

    mssge = "La información se ha enviado exitosamente" + vbCrLf + vbCrLf
    If numpts = 1 Then
      mssge = mssge + "Se envió 1 punto." + vbCrLf + vbCrLf
    Else
      mssge = mssge + "Se enviaron " + CStr(numpts) + " puntos." + vbCrLf + vbCrLf
    End If
    MsgBox mssge, vbOKOnly, "Envío de puntos al controlador del robot PUMA"
    MsgOper.Text = ""
    activar_botones
    Restart.Enabled = True

ManejarErrores:
    If Err.Number <> 0 Then
      If Err.Number <> cdlCancel Then
        MsgBox "Error " & Err.Number & vbCr & vbCr & Err.Description, vbCritical +
vbOKOnly, "Error en el proceso"
        Err.Clear
      End If
```

```
    End If
    End If
End Sub
```

```
Private Sub Recibir_Ptos_Click()
    Dim Archivo As String
    Dim Ar_Listapts As String
    Dim Listapts(200) As String
    Dim li_cont As Integer
    Dim contador As Integer
    Dim Ls_Out As String
    Dim Ls_In As String
    Dim Ls_In_Ant As String
    Dim As_cadena As String
    Dim mssge As String
    Dim opt As Integer
    Dim numpt As Integer
    Dim nombrePt As String
    Dim I As Integer
    Dim J As Integer

    On Error GoTo ManejarErrores

    Form_Load
    If conRobot Then
    CommonDialog1.DialogTitle = "Recepción de puntos del robot PUMA a disco"
    CommonDialog1.DefaultExt = "ptp"
    CommonDialog1.Filter = "Archivos de Puntos PUMA|*.ptp"
    CommonDialog1.InitDir = "C:\PUMA"
    CommonDialog1.Flags = cdlOFNOverwritePrompt
    CommonDialog1.FileName = ""
    CommonDialog1.ShowSave

    Archivo = CommonDialog1.FileName

    mssge = "¿Cuenta usted con un archivo con la lista de puntos que desea recibir?" + vbCrLf
+ vbCrLf + vbCrLf
    mssge = mssge + "Si cuenta con un archivo con la lista de puntos que desea pulse <Yes>."
+ vbCrLf + vbCrLf + vbCrLf
    mssge = mssge + "Si no cuenta con un archivo con la lista de puntos que desea pulse <No>"
+ vbCrLf
    mssge = mssge + "Esta opción implica que usted desea recibir todos los puntos grabados en
el controlador" + vbCrLf + vbCrLf + vbCrLf
    mssge = mssge + "Si ya no desea realizar ninguna operación pulse <Cancel>  " + vbCrLf +
vbCrLf + vbCrLf
    opt = MsgBox(mssge, vbYesNoCancel + vbQuestion + vbDefaultButton1, "Recepción de
puntos del robot PUMA a disco")
```

```
If opt = 6 Then
   CommonDialog1.DialogTitle = "Recepción de puntos del robot PUMA a disco"
   CommonDialog1.Filter = "Lista de puntos|*.ptl|Todos los Archivos|*.*|"
   CommonDialog1.InitDir = App.Path
   CommonDialog1.FileName = ""
   CommonDialog1.ShowOpen
   Ar_Listapts = CommonDialog1.FileName
   numpt = 0
   Open Ar_Listapts For Input As #1
   Do Until EOF(1)
      numpt = numpt + 1
      Line Input #1, Listapts(numpt)
   Loop
   Close #1
   For I = 1 To numpt
      For J = Len(Listapts(I)) + 1 To 10
         Listapts(I) = Listapts(I) + " "
      Next J
   Next I
   Open Archivo For Output As #1
   apagar_botones
   Restart.Enabled = False
   mssge = "Recibiendo Puntos... " + vbCrLf
   mssge = mssge + "Por favor espere..." + vbCrLf
   MsgOper.Text = mssge
   I = 0
   For J = 1 To numpt
      mssge = ""
      As_cadena = "POINT " + Listapts(J)
      Mandar_Comando (As_cadena)
      Do
         Ls_In_Ant = Ls_In
         Do
            DoEvents
         Loop Until MSComm1.InBufferCount >= 1
         Ls_In = MSComm1.Input
         mssge = mssge + Ls_In
      Loop Until Ls_In = " " And Ls_In_Ant = "?"
      mssge = Mid(mssge, 57, 52)
      If Mid(mssge, 4, 49) = "0.00     0.00     0.00   90.000  -90.000    0.000" Then
         MSComm1.Output = vbCr
         As_cadena = "DELETEL " + Listapts(J)
         Mandar_Comando (As_cadena)
         Do
            Ls_In = MSComm1.Input
         Loop Until Ls_In = "?"
```

```
            Mandar_Comando ("Y")
            Do
               Ls_In = MSComm1.Input
            Loop Until Ls_In = "."
         Else
            mssge = Listapts(J) + mssge
            Print #1, mssge
            I = I + 1
         End If
         MSComm1.Output = vbCr
         Do
            Do
               DoEvents
            Loop Until MSComm1.InBufferCount >= 1
            Ls_In = MSComm1.Input
         Loop Until Ls_In = "."

      Next J
      Close #1

      mssge = "Número de puntos solicitados: " + CStr(numpt) + vbCrLf
      mssge = mssge + "Número de puntos encontrados: " + CStr(I) + vbCrLf + vbCrLf

      If numpt > I Then
         If numpt - I > 1 Then
            mssge = mssge + "Se solicitaron " + CStr(numpt - I) + " puntos en el archivo: " +
Ar_Listapts + vbCrLf
            mssge = mssge + "que no existen en el controlador, por favor revise la lista de
puntos."
         Else
            mssge = mssge + "Se solicitó " + CStr(numpt - I) + " punto en el archivo: " +
Ar_Listapts + vbCrLf
            mssge = mssge + "que no existe en el controlador, por favor revise la lista de
puntos."
         End If
         MsgBox mssge, vbCritical + vbOKOnly, "Recepción de puntos del robot PUMA a
disco"
      Else
         mssge = mssge + "La información se ha recibido exitosamente" + vbCr
         MsgBox mssge, vbInformation + vbOKOnly, "Recepción de puntos del robot PUMA a
disco"
      End If
      MsgOper.Text = ""
      activar_botones
      Restart.Enabled = True
   End If
```

```
   If opt = 7 Then
      Open Archivo For Output As #1
      As_cadena = "LISTL"
      apagar_botones
      Restart.Enabled = False
      mssge = "Recibiendo Puntos... " + vbCrLf
      mssge = mssge + "Por favor espere..." + vbCrLf
      MsgOper.Text = mssge
      Mandar_Comando (As_cadena)
      contador = 0
      Do
         Ls_In_Ant = Ls_In
         Do
            DoEvents
         Loop Until MSComm1.InBufferCount >= 1
         Ls_In = MSComm1.Input
         If (Ls_In = vbLf) And (Ls_In_Ant = vbCr) Then
            contador = contador + 1
         End If
      Loop Until contador = 2
      Do
         mssge = ""
         Do
            Ls_In_Ant = Ls_In
            Do
               DoEvents
            Loop Until MSComm1.InBufferCount >= 1
            Ls_In = MSComm1.Input
            If (Ls_In <> vbLf) Then
               mssge = mssge + Ls_In
            End If
         Loop Until ((Ls_In = vbLf) And (Ls_In_Ant = vbCr)) Or ((Ls_In = ".") And
(Ls_In_Ant = vbLf))
         If Len(mssge) > 2 Then
            mssge = Mid(mssge, 2, Len(mssge) - 2)
            Print #1, mssge
         End If
      Loop Until (Ls_In = ".") And (Ls_In_Ant = vbLf)

      Close #1

      mssge = mssge + "La información de los puntos se ha recibido exitosamente."
      MsgBox mssge, vbInformation + vbOKOnly, "Recepción de puntos del robot PUMA a
disco"

      mssge = ""
      MsgOper.Text = mssge
```

```
      activar_botones
      Restart.Enabled = True
   End If

ManejarErrores:
   If Err.Number <> 0 Then
      If Err.Number <> cdlCancel Then
         MsgBox "Error " & Err.Number & vbCr & vbCr & Err.Description, vbCritical +
vbOKOnly, "Error en el proceso"
         Err.Clear
      End If
   End If
   End If
End Sub
```

```
Private Sub Borrar_pto_Click()
   Dim nombrePt As String
   Dim orden As String

   nombrePt = InputBox$("Escriba el nombre del punto que desea borrar", "Eliminación de
Puntos en el Controlador del PUMA")

   If nombrePt <> "" Then
      Form_Load
      If conRobot Then
         orden = "DELETEL " + nombrePt
         Pfn_Mandar_Cadena (orden)
      End If
   End If
End Sub
```

```
Private Sub Borrar_Prog_Click()
   Dim orden As String

   Form_Load
   If conRobot Then
      MostrarProgs.Show 1
      If NombrePr <> "" Then
         orden = "DELETE " + NombrePr
         Pfn_Mandar_Cadena (orden)
      End If
   End If
End Sub
```

```
Private Sub Restart_Click()
   Form_Load
End Sub
```

```
Private Sub Salidas_Click()
   Form_Load
   SeñalesSalidas.Show
End Sub
```

```
Private Sub Grabar_pto_Click()
   Dim nombrePt As String
   Dim orden As String

   nombrePt = InputBox$("Escriba el nombre del punto", "Registro de la posición actual del
brazo", "punto")

   If nombrePt <> "" Then
      Form_Load
      If conRobot Then
         orden = "DO HERE " + nombrePt
         Pfn_Mandar_Cadena (orden)
      End If
   End If
End Sub
```

```
Private Sub Move_pto_Click()
   Form_Load
   If conRobot Then
      MoveTerm.Show 1
   End If
End Sub
```

```
Private Sub Cal_Click()
   Form_Load
   If conRobot Then
      MsgOper.Text = "Robot calibrandose..."
      apagar_botones
      Restart.Enabled = False
      Pfn_Mandar_Cadena ("CAL")
      MsgOper.Text = ""
      activar_botones
      Restart.Enabled = True
   End If
End Sub
```

```
Private Sub Enviar_Prog_Click()
   Dim Archivo As String
   Dim linea As String
   Dim li_cont As Integer
   Dim contador As Integer
```

```
    Dim Ls_Out As String
    Dim Ls_In As String
    Dim Ls_In_Ant As String
    Dim As_cadena As String
    Dim mssge As String
    Dim aviso As String
    Dim opt As Integer
    Dim I As Integer
    Dim J As Integer
    Dim nombrePt As String
    Dim numpts As Integer
    Dim Listapts(200) As String
    Dim ya As Boolean
    Dim mal As Boolean

    On Error GoTo ManejarErrores
    Form_Load
    If conRobot Then
    CommonDialog1.DialogTitle = "Envío de programa al robot PUMA"
    CommonDialog1.Filter = "Programas PUMA|*.prp|Todos los Archivos|*.*|"
    CommonDialog1.InitDir = App.Path
    CommonDialog1.FileName = ""
    CommonDialog1.ShowOpen

    apagar_botones
    Restart.Enabled = False
    Archivo = CommonDialog1.FileName
    Open Archivo For Input As #1
    Archivo = CommonDialog1.FileTitle
    mssge = ""
    For contador = 1 To Len(Archivo)
       If Mid(Archivo, contador, 1) <> " " Then
          mssge = mssge + Mid(Archivo, contador, 1)
       End If
    Next contador
    Archivo = mssge
    contador = 0
    If Encontrar_Archivo_PUMA(Archivo) Then
       mssge = "El programa ya exsite en el controlador" + vbCrLf
       mssge = mssge + "¿Desea sobre-escribirlo? (Y/N)"
       mssge = mssge + vbCrLf
       opt = MsgBox(mssge, vbQuestion + vbYesNoCancel + vbDefaultButton2,
"Confirmación de instrucción")
       Select Case opt
       Case 6
          Borrar_Programa (Archivo)
       Case 7
```

```
        Do
           Archivo = InputBox$("Escriba el nombre del programa para el controlador",
"Cambio de nombre de programa para el Robot PUMA")
           If Archivo <> "" Then
              If Encontrar_Archivo_PUMA(Archivo) Then
                 mssge = "El archivo ya exsite en el controlador" + vbCrLf
                 mssge = mssge + "¿Desea sobre-escribirlo? (Y/N)"
                 mssge = mssge + vbCrLf

                 opt = MsgBox(mssge, vbQuestion + vbYesNoCancel + vbDefaultButton2,
"Confirmación de instrucción")
              End If
              If opt = 6 Then
                 Borrar_Programa (Archivo)
              End If
           End If
        Loop Until (opt <> 7) Or (Encontrar_Archivo_PUMA(Archivo) = False)
     End Select
  End If

  If (opt <> 2) And (Archivo <> "") And (Encontrar_Archivo_PUMA(Archivo) = False)
Then
     mssge = "Enviando Programa al Controlador" + vbCrLf
     mssge = mssge + "Por favor espere" + vbCrLf
     MsgOper.Text = mssge

     As_cadena = "EDIT " + Archivo
     Mandar_Comando (As_cadena)
     numpts = 1
     mal = False

     Do
        Do
           DoEvents
        Loop Until MSComm1.InBufferCount >= 1
        Ls_In = MSComm1.Input
     Loop Until Ls_In = "?"
     contador = 0
     Do Until EOF(1)
        Line Input #1, linea
        As_cadena = linea
        contador = contador + 1
        aviso = "Enviando Programa al Controlador" + vbCrLf
        aviso = aviso + "Por favor espere" + vbCrLf
        aviso = aviso + "Línea: " + CStr(contador)
        MsgOper.Text = aviso
```

```
      For I = 1 To Len(linea) - 4
         If Mid(linea, I, 4) = "MOVE" Then
            If Mid(linea, I + 4, 1) = "S" Then
               nombrePt = Mid(linea, I + 6, Len(linea) - (I + 5))
            Else
               nombrePt = Mid(linea, I + 5, Len(linea) - (I + 4))
            End If
            ya = False
            For J = 1 To numpts
               If nombrePt = Listapts(J) Then
                  ya = True
               End If
            Next J
            If Not (ya) Then
               Listapts(numpts) = nombrePt
               numpts = numpts + 1
            End If
         End If
      Next I

      Mandar_Comando (As_cadena)
      mssge = ""
      Do
         Do
            DoEvents
         Loop Until MSComm1.InBufferCount >= 1
         Ls_In = MSComm1.Input
         mssge = mssge + Ls_In
      Loop Until Ls_In = "?"
      If Len(mssge) > 7 Then
         mal = True
         mssge = Mid(mssge, 1, Len(mssge) - 4)
         mssge = "Error en la línea:  " + CStr(contador) + vbCrLf + linea + vbCrLf + mssge
         MsgBox mssge, vbCritical + vbOKOnly, "Envío de programa al robot PUMA"
      End If
   Loop
   MSComm1.Output = "E " + vbCr
   Do
      Do
         DoEvents
      Loop Until MSComm1.InBufferCount >= 1
      Ls_In = MSComm1.Input
   Loop Until Ls_In = "."

   mssge = "¿Desea crear un archivo con la lista de puntos?"
   I = MsgBox(mssge, vbQuestion + vbYesNo + vbDefaultButton1, "Envío de programa al
Robot PUMA")
```

```
      MsgOper.Text = ""
      mssge = "El programa " + Archivo + vbCrLf
      If mal Then
         mssge = mssge + "ha sido enviado incompleto por errores." + vbCrLf + vbCrLf
      Else
         mssge = mssge + "ha sido enviado exitosamente." + vbCrLf + vbCrLf
         mssge = mssge + "Se enviaron: " + CStr(contador) + " líneas." + vbCr + vbCr
      End If

      If I = 6 Then
         CommonDialog1.DialogTitle = "Creación de la lista de puntos de un programa"
         CommonDialog1.DefaultExt = "ptl"
         CommonDialog1.Filter = "Lista de puntos|*.ptl"
         CommonDialog1.InitDir = "C:\PUMA"
         CommonDialog1.Flags = cdlOFNOverwritePrompt
         CommonDialog1.FileName = ""
         CommonDialog1.ShowSave

         As_cadena = CommonDialog1.FileName
         Open As_cadena For Output As #2
         For J = 1 To numpts - 1
            Print #2, Listapts(J)
         Next J
         Close #2
         mssge = mssge + "Se creó el archivo: " + As_cadena + " con: "
         If numpts = 2 Then
            mssge = mssge + " 1 punto."
         Else
            mssge = mssge + CStr(numpts - 1) + " puntos."
         End If
      End If
      mssge = mssge + vbCr
      MsgBox mssge, vbInfo + vbOKOnly, "Envío de Programa al Robot PUMA"
   End If

ManejarErrores:
   If Err.Number <> 0 Then
      If Err.Number <> cdlCancel Then
         MsgBox "Error " & Err.Number & vbCr & vbCr & Err.Description, vbCritical +
vbOKOnly, "Error en el proceso"
         Err.Clear
      End If
   End If

   Close #1
   activar_botones
```

```
   Restart.Enabled = True
   End If
End Sub
```

```
Private Sub Status_Click()
   Form_Load
   If conRobot Then
      Pfn_Mandar_Cadena ("STATUS")
   End If
End Sub
```

```
Private Sub Speed_Click()
   Dim Velocidad As String
   Dim orden As String
   Dim verifica As Single

   Do
      Velocidad = InputBox$("Escriba la velocidad monitor que desea establecer. (0.39 -
12800)", "Declaración de la velocidad de monitor", "30")
      verifica = Val(Velocidad)
   Loop Until ((verifica > 0) And (CSng(Velocidad) >= 0.39 And CSng(Velocidad) <=
12800)) Or Velocidad = ""
   If Velocidad <> "" Then
      Form_Load
      If conRobot Then
         orden = "SPEED " + Velocidad
         Pfn_Mandar_Cadena (orden)
      End If
   End If
End Sub
```

```
Private Sub Toolbar1_ButtonClick(ByVal Button As ComctlLib.Button)
   Boton = Button.Key
   Select Case Boton

   Case "keySeñal"
       SeñalesSalidas.Show

   Case "keyDwnProg"
       Enviar_Prog_Click

   Case "keyUpProg"
       Recibir_Prog_Click

   Case "keyDwnPts"
       Enviar_Ptos_Click
```

```
   Case "keyUpPts"
      Recibir_Ptos_Click

   Case "keyEdit"
      Edición.Show

   Case "keyTerm"
      TermMode.Show

   Case "keyEnd"
     mnuFinProg_Click

   End Select
End Sub
```

```
Private Sub Where_Click()
Dim mssge As String

   Form_Load
   If conRobot Then
      mssge = "Recuperando posición actual" + vbCrLf
      mssge = mssge + "Espere un momento, por favor..."
      MsgOper.Text = mssge
      PosActual.Show
   End If
End Sub
```

```
Private Sub Clamp_Click()
   Form_Load
   If conRobot Then
      If Clamp.Value = 0 Then
         Call Pfn_Mandar_Cadena("DO OPENI")
         Clamp.Caption = "Cerrar Clamp"
      Else
         Call Pfn_Mandar_Cadena("DO CLOSEI")
         Clamp.Caption = "Abrir Clamp"
      End If
   End If
End Sub
```

```
Public Sub Borrar_Programa(nombre As String)
Dim li_cont As Integer
Dim Ls_Out As String
Dim Ls_In As String
Dim As_cadena As String

   As_cadena = "DELETEP " + nombre
```

```
    Mandar_Comando (As_cadena)

    Do
       Do
          DoEvents
       Loop Until MSComm1.InBufferCount >= 1
       Ls_In = MSComm1.Input
    Loop Until Ls_In = "?"

    MSComm1.Output = "Y" + vbCr

    Do
       Do
          DoEvents
       Loop Until MSComm1.InBufferCount >= 1
       Ls_In = MSComm1.Input
    Loop Until Ls_In = "."

End Sub
```

```
Public Function Encontrar_Archivo_PUMA(nombre As String) As Boolean
Dim As_cadena As String
Dim linea As String
Dim li_cont As Integer
Dim contador As Integer
Dim Ls_Out As String
Dim Ls_In As String
Dim Ls_In_Ant As String

Encontrar_Archivo_PUMA = False
As_cadena = "DIR"

Mandar_Comando (As_cadena)

contador = 0
Do
   Ls_In_Ant = Ls_In
   Do
      DoEvents
   Loop Until MSComm1.InBufferCount >= 1
   Ls_In = MSComm1.Input

   If ((Ls_In <> ".") Or (Ls_In_Ant <> vbLf)) And Ls_In <> " " Then
      mssge = mssge + Ls_In
      contador = contador + 1
      If (Ls_In = vbLf) And (Ls_In_Ant = vbCr) Then
```

```
            If contador > 2 Then
               mssge = Mid(mssge, 1, contador - 2)
            End If
            If nombre = mssge Then
               Encontrar_Archivo_PUMA = True
            End If
            mssge = ""
            contador = 0
         End If
      End If
Loop Until (Ls_In = ".") And (Ls_In_Ant = vbLf)

End Function
```

```
Public Sub apagar_botones()
      conRobot = False
      Status.Enabled = False
      Listl.Enabled = False
      Borrar_pto.Enabled = False
      Where.Enabled = False
      Dir.Enabled = False
      Borrar_Prog.Enabled = False
      Cal.Enabled = False
      Home.Enabled = False
      Speed.Enabled = False
      Grabar_pto.Enabled = False
      Move_pto.Enabled = False
      Clamp.Enabled = False
      Exec.Enabled = False
      Enviar_Prog.Enabled = False
      Recibir_Prog.Enabled = False
      Enviar_Ptos.Enabled = False
      Recibir_Ptos.Enabled = False
      Toolbar1.Buttons(2).Enabled = False
      Toolbar1.Buttons(3).Enabled = False
      Toolbar1.Buttons(4).Enabled = False
      Toolbar1.Buttons(5).Enabled = False
      Toolbar1.Buttons(6).Enabled = False
      mnuComunicaciones.Enabled = False
      mnuModoTerm.Enabled = False
End Sub
```

```
Public Sub activar_botones()
      conRobot = True
      Status.Enabled = True
      Listl.Enabled = True
      Borrar_pto.Enabled = True
```

```
        Where.Enabled = True
        Dir.Enabled = True
        Borrar_Prog.Enabled = True
        Cal.Enabled = True
        Home.Enabled = True
        Speed.Enabled = True
        Grabar_pto.Enabled = True
        Move_pto.Enabled = True
        Clamp.Enabled = True
        Exec.Enabled = True
        Enviar_Prog.Enabled = True
        Recibir_Prog.Enabled = True
        Enviar_Ptos.Enabled = True
        Recibir_Ptos.Enabled = True
        Toolbar1.Buttons(2).Enabled = True
        Toolbar1.Buttons(3).Enabled = True
        Toolbar1.Buttons(4).Enabled = True
        Toolbar1.Buttons(5).Enabled = True
        Toolbar1.Buttons(6).Enabled = True
        mnuComunicaciones.Enabled = True
        mnuModoTerm.Enabled = True
End Sub
```

```
Public Sub Mandar_Comando(As_cadena As String)
Dim li_cont As Integer
Dim Ls_Out As String
Dim Ls_In As String

    For li_cont = 1 To Len(As_cadena)
        Ls_Out = Mid(As_cadena, li_cont, 1)
        MSComm1.Output = Ls_Out
        Do
            DoEvents
        Loop Until MSComm1.InBufferCount >= 1
        Ls_In = MSComm1.Input
    Next li_cont
    MSComm1.Output = vbCr
End Sub
```

```
Public Sub Pfn_Mandar_Cadena(As_cadena As String)
Dim li_cont As Integer
Dim Ls_Out As String
Dim Ls_In As String
Dim Ls_In_Ant As String
Dim mssge As String
Dim opt As Integer
Dim letra As String
```

```
On Error GoTo ManejarErrores

Mandar_Comando (As_cadena)
mssge = ""
Do
   Ls_In_Ant = Ls_In
   Do
      DoEvents
   Loop Until MSComm1.InBufferCount >= 1
   Ls_In = MSComm1.Input

   If (Ls_In <> ".") Or (Ls_In_Ant <> vbLf) Then
      mssge = mssge + Ls_In
   End If
   If Ls_In = "?" Then
      opt = MsgBox(mssge, vbQuestion + vbYesNo + vbDefaultButton2, "Confirmación de
instrucción")
      If opt = 6 Then
         MSComm1.Output = "Y" + vbCr
      Else
         MSComm1.Output = "N" + vbCr
      End If
      mssge = ""
      Do
         Ls_In = MSComm1.Input
      Loop Until (Ls_In = "Y") Or (Ls_In = "N")
   End If
Loop Until (Ls_In = ".") And (Ls_In_Ant = vbLf)

If Len(mssge) > 4 Then
   If Mid(mssge, 3, 1) = "*" Then
      MsgBox mssge, vbCritical + vbOKOnly, "Mensaje del Robot PUMA"
   Else
      MsgBox mssge, vbInformation + vbOKOnly, "Respuesta del Robot PUMA"
   End If
End If

ManejarErrores:
If Err.Number <> 0 Then
   MsgBox "Error " & Err.Number & vbCr & vbCr & Err.Description, vbCritical +
vbOKOnly, "Error en el envío de información"
   Err.Clear
End If
End Sub
```

```
Private Sub Form_Load()
   Dim tiempo As Integer
   Dim Ls_In As String
   Dim Ls_In_Ant As String
   Dim mssge As String

   On Error GoTo ManejarErrores

   If Not MSComm1.PortOpen Then
      MSComm1.CommPort = 1
      MSComm1.Settings = "9600,n,8,1"
      MSComm1.InputLen = 1
      MSComm1.PortOpen = True
   End If

   tiempo = 0
   MSComm1.Output = vbCr
   Do
      tiempo = tiempo + 1
   Loop Until MSComm1.InBufferCount >= 1 Or tiempo = 3000

   If tiempo = 3000 Then
      mssge = "Robot no disponible" + vbCrLf
      mssge = mssge + "Trabajando fuera de línea"
      MsgOper.Text = mssge
      apagar_botones
   Else
      Do
         Ls_In = MSComm1.Input
      Loop Until Ls_In = vbLf
      MsgOper.Text = ""
      mssge = ""
      conRobot = True
      Do
         Ls_In_Ant = Ls_In
         Do
            DoEvents
         Loop Until MSComm1.InBufferCount >= 1
         Ls_In = MSComm1.Input
         If (Ls_In <> ".") Or (Ls_In_Ant <> vbLf) Then
            If (Ls_In >= Chr(40) And Ls_In <= Chr(122)) Or (Ls_In = Chr(32)) Then
               mssge = mssge + Ls_In
            Else
               If (Ls_In = vbLf) Then
                  mssge = mssge + vbCrLf
               End If
            End If
```

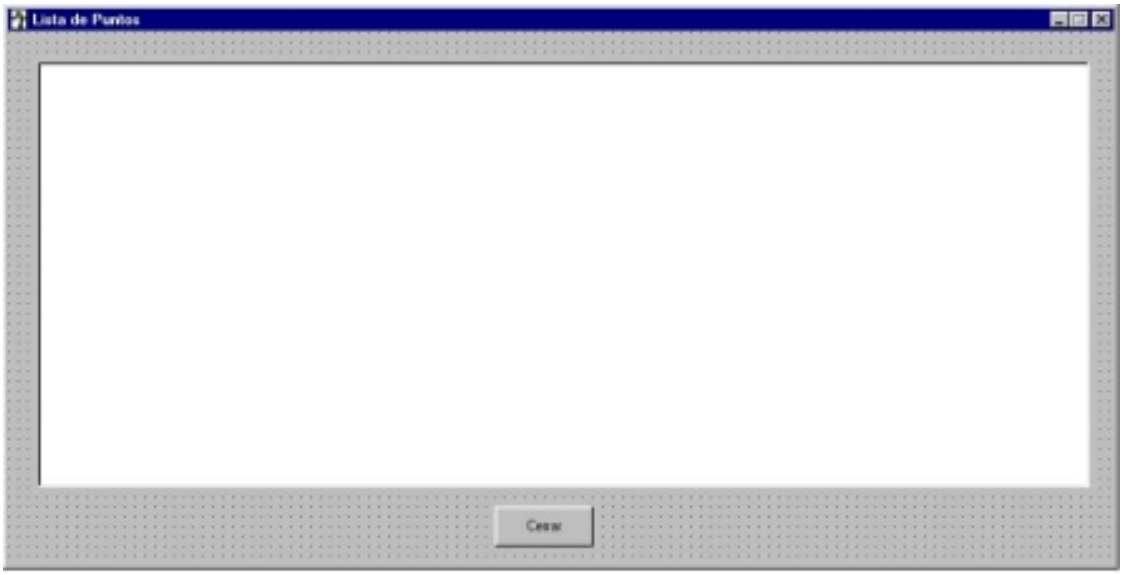
```
            End If
            If Ls_In = "?" Then
               If Len(mssge) <> 70 Then
                  If Len(mssge) < 70 Then
                     mssge = "* VAL-II Boot B560.2.0 24AUG85 *" + vbCrLf + vbCrLf + mssge
                  End If
                  opt = MsgBox(mssge, vbQuestion + vbYesNo + vbDefaultButton2,
"Confirmación de instrucción")
                  If opt = 6 Then
                     MSComm1.Output = "Y"
                  Else
                     MSComm1.Output = "N"
                  End If
                  MSComm1.Output = vbCr
                  Do
                     Ls_In = MSComm1.Input
                  Loop Until Ls_In = vbLf
               End If
               mssge = ""
            End If
         Loop Until (Ls_In = ".") And (Ls_In_Ant = vbLf)
         activar_botones
      End If

ManejarErrores:
      If Err.Number <> 0 Then
         MsgBox "Error " & Err.Number & vbCr & vbCr & Err.Description, vbCritical +
vbOKOnly, "Error en el proceso"
         Err.Clear
         mssge = "Robot no disponible" + vbCrLf
         mssge = mssge + "Trabajando fuera de línea"
         MsgOper.Text = mssge
         apagar_botones
      End If

End Sub
```

Anexo D

Forma y Código

de la Pantalla Desplegadora de la Lista de Puntos

Forma de la Pantalla Desplegadora la Lista de Puntos



Código de la Pantalla Desplegadora la Lista de Puntos

```
Private Sub Cerrar_Click()
   Unload ListLocations
End Sub
```

```
Private Sub Form_Load()
   Dim mssge As String
   Dim As_cadena As String
   Dim Ls_In As String
   Dim Ls_In_Ant As String

   As_cadena = "LISTL"
   Panel_Principal.MSComm1.InputLen = 1
   Panel_Principal.Mandar_Comando (As_cadena)

   mssge = ""

   Do
      Ls_In_Ant = Ls_In
      Do
         DoEvents
      Loop Until Panel_Principal.MSComm1.InBufferCount >= 1
      Ls_In = Panel_Principal.MSComm1.Input
      If (Ls_In <> ".") Or (Ls_In_Ant <> vbLf) Then
         mssge = mssge + Ls_In
      End If
   Loop Until (Ls_In = ".") And (Ls_In_Ant = vbLf)
   Panel_Principal.MsgOper.Text = ""
   RichTextBox1.Text = mssge
End Sub
```

Anexo E

Forma y Código

de la Pantalla que Muestra la Posición Actual

Forma de la Pantalla que Muestra la Posición Actual

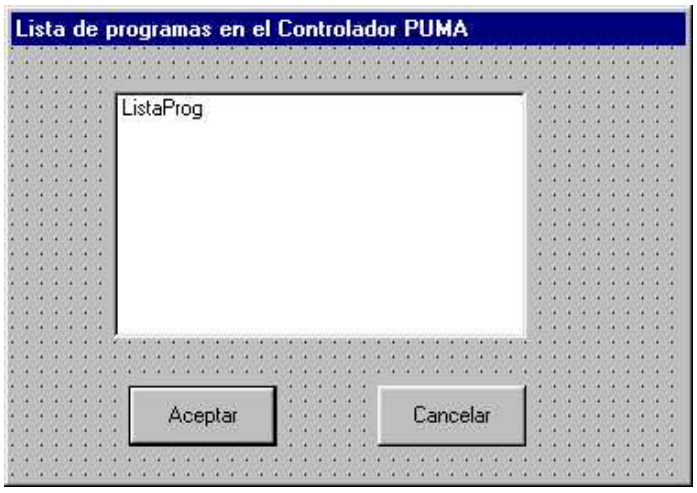
Código de la Pantalla que Muestra la Posición Actual

```
Private Sub Cerrar_Click()
   Unload PosActual
End Sub
```

```
Private Sub Form_Load()
   Dim mssge As String
   Dim As_cadena As String
   Dim Ls_In As String
   Dim Ls_In_Ant As String

   As_cadena = "WHERE"
   Panel_Principal.MSComm1.InputLen = 1
   Panel_Principal.Mandar_Comando (As_cadena)

   mssge = ""

   Do
      Ls_In_Ant = Ls_In
      Do
         DoEvents
      Loop Until Panel_Principal.MSComm1.InBufferCount >= 1
      Ls_In = Panel_Principal.MSComm1.Input
      If (Ls_In <> ".") Or (Ls_In_Ant <> vbLf) Then
         mssge = mssge + Ls_In
      End If
   Loop Until (Ls_In = ".") And (Ls_In_Ant = vbLf)
   Panel_Principal.MsgOper.Text = ""
   RichTextBox1.Text = mssge
   If Mid(mssge, Len(mssge) - 5, 4) = "0.00" Then
      Panel_Principal.Clamp.Value = 1
   Else
      Panel_Principal.Clamp.Value = 0
   End If
End Sub
```

Anexo F
Forma y Código
de la Pantalla que Muestra
los Programas en el Controlador

Forma de la Pantalla que Muestra los Programas en el Controlador



Código de la Pantalla que Muestra los Programas en el Controlador

```
Private Sub btnAceptar_Click()
   Panel_Principal.NombrePr = ListaProg.Text
   Unload MostrarProgs
End Sub
```

```
Private Sub btnCancel_Click()
   Panel_Principal.NombrePr = ""
   Unload MostrarProgs
End Sub
```

```
Private Sub Form_Load()
   Dim As_cadena As String
   Dim contador As Integer
   Dim linea As Integer
   Dim mssge As String
   Dim Ls_In As String
   Dim Ls_In_Ant As String

   As_cadena = "DIR"
   Panel_Principal.Mandar_Comando (As_cadena)

   contador = 0
   mssge = ""
   Do
      Ls_In_Ant = Ls_In
      Do
         DoEvents
      Loop Until Panel_Principal.MSComm1.InBufferCount >= 1
      Ls_In = Panel_Principal.MSComm1.Input
      If ((Ls_In <> ".") Or (Ls_In_Ant <> vbLf)) And Ls_In <> " " Then
         mssge = mssge + Ls_In
         contador = contador + 1
         If (Ls_In = vbLf) And (Ls_In_Ant = vbCr) Then
            If contador > 2 Then
               mssge = Mid(mssge, 1, contador - 2)
               ListaProg.AddItem mssge
            End If
            mssge = ""
            contador = 0
         End If
      End If
   Loop Until (Ls_In = ".") And (Ls_In_Ant = vbLf)
```

```
    mssge = ""
End Sub
```

```
Private Sub ListaProg_DblClick()
    Panel_Principal.NombrePr = ListaProg.Text
    Unload MostrarProgs
End Sub
```

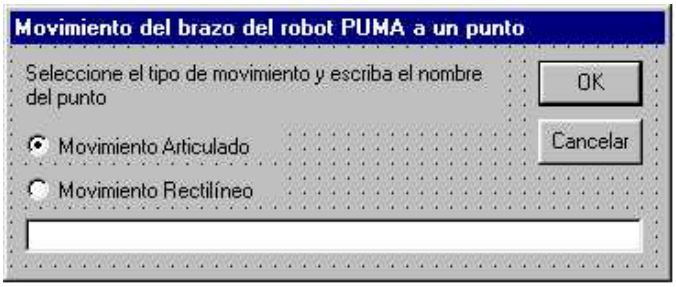
Anexo G

Forma y Código

de la Pantalla que Mueve al Robot a un Punto

desde la Terminal

Forma de la Pantalla que Mueve al Robot a un Punto desde la Terminal



Código de la Pantalla que Mueve al Robot a un Punto desde la Terminal

```
Private Sub Cancel_Click()
   Unload MoveTerm
End Sub
```

```
Private Sub Aceptar_Click()
   Dim mssge As String
   If OptArt Then
      mssge = "DO MOVE "
   Else
      mssge = "DO MOVES "
   End If
   If NombrePunto.Text <> "" Then
      mssge = mssge + NombrePunto.Text
      Panel_Principal.apagar_botones
      Panel_Principal.Restart.Enabled = False
      Panel_Principal.MsgOper.Text = "Robot en movimiento..."
      Panel_Principal.Pfn_Mandar_Cadena (mssge)
      Panel_Principal.activar_botones
      Panel_Principal.Restart.Enabled = True
      Panel_Principal.MsgOper.Text = ""
      Unload MoveTerm
   Else
      mssge = "Es necesario que escriba el nombre del punto"
      MsgBox mssge, vbCritical + vbOKOnly, "Panel de Control del robot PUMA"
   End If
End Sub
```

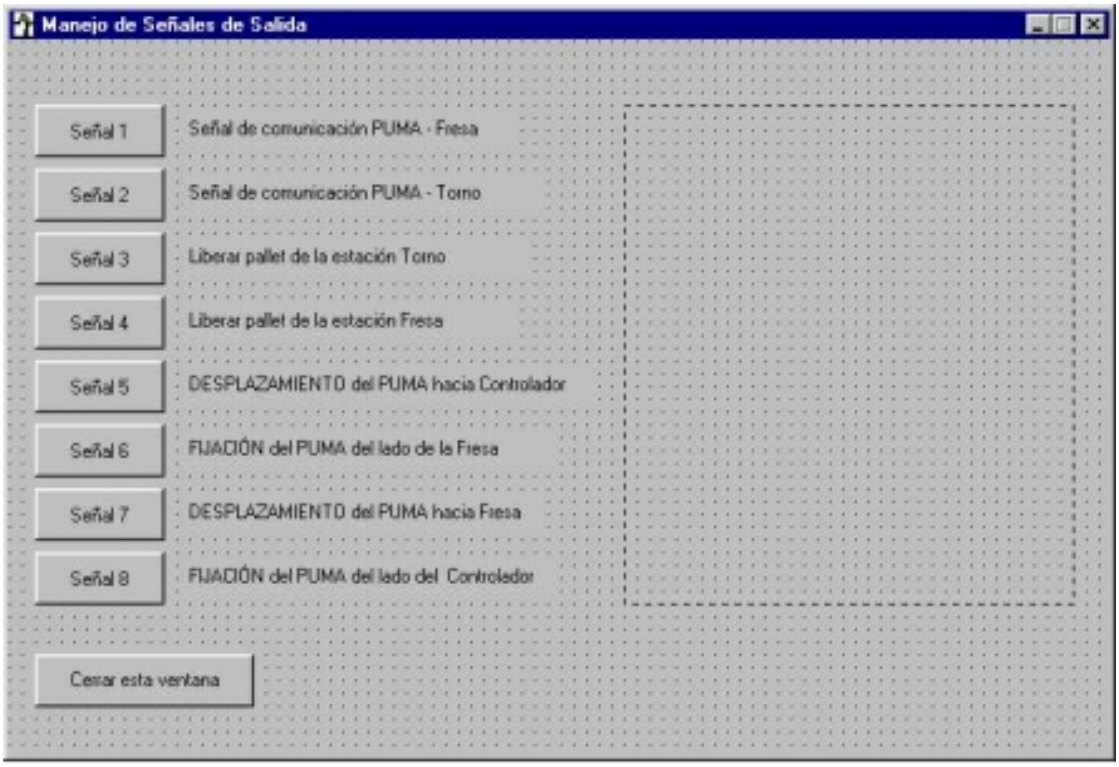
```
Private Sub Form_Activate()
   NombrePunto.SetFocus
End Sub
```

Anexo H

Forma y Código

de la Pantalla para el Manejo de las Señales de Salida

Forma de la Pantalla para el Manejo de las Señales de Salida



Código de la Pantalla para el Manejo de las Señales de Salida

```
Private Sub Cerrar_Click()
   Unload SeñalesSalidas
End Sub
```

```
Private Sub Sig1_Click()
   If Sig1.Value = 0 Then
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL -1")
      End If
      Sig1.Caption = "Señal 1"
   Else
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL 1")
      End If
      Sig1.Caption = "Señal -1"
   End If
End Sub
```

```
Private Sub Sig1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
   Image1.Picture = LoadPicture(App.Path & "\Signal1.jpg")
End Sub
```

```
Private Sub Sig2_Click()
   If Sig2.Value = 0 Then
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL -2")
      End If
      Sig2.Caption = "Señal 2"
   Else
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL 2")
      End If
      Sig2.Caption = "Señal -2"
   End If
End Sub
```

```
Private Sub Sig2_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
   Image1.Picture = LoadPicture(App.Path & "\Signal2.jpg")
End Sub
```

```
Private Sub Sig3_Click()
   If Sig3.Value = 0 Then
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL -3")
```

```
      End If
      Sig3.Caption = "Señal 3"
   Else
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL 3")
      End If
      Sig3.Caption = "Señal -3"
   End If
End Sub
```

```
Private Sub Sig3_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
   Image1.Picture = LoadPicture(App.Path & "\Signal3.jpg")
End Sub
```

```
Private Sub Sig4_Click()
   If Sig4.Value = 0 Then
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL -4")
      End If
      Sig4.Caption = "Señal 4"
   Else
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL 4")
      End If
      Sig4.Caption = "Señal -4"
   End If
End Sub
```

```
Private Sub Sig4_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
   Image1.Picture = LoadPicture(App.Path & "\Signal4.jpg")
End Sub
```

```
Private Sub Sig5_Click()
   If Sig5.Value = 0 Then
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL -5")
      End If
      Sig5.Caption = "Señal 5"
      Sig7.Enabled = True
   Else
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL 5")
      End If
      Sig5.Caption = "Señal -5"
      Sig7.Enabled = False
   End If
End Sub
```

```
Private Sub Sig5_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
   Image1.Picture = LoadPicture(App.Path & "\Signal5.jpg")
End Sub
```

```
Private Sub Sig6_Click()
   If Sig6.Value = 0 Then
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL -6")
      End If
      Sig6.Caption = "Señal 6"
   Else
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL 6")
      End If
      Sig6.Caption = "Señal -6"
   End If
End Sub
```

```
Private Sub Sig6_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
   Image1.Picture = LoadPicture(App.Path & "\Signal6.jpg")
End Sub
```

```
Private Sub Sig7_Click()
   If Sig7.Value = 0 Then
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL -7")
      End If
      Sig7.Caption = "Señal 7"
      Sig5.Enabled = True
   Else
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL 7")
      End If
      Sig7.Caption = "Señal -7"
      Sig5.Enabled = False
   End If
End Sub
```

```
Private Sub Sig7_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
   Image1.Picture = LoadPicture(App.Path & "\Signal7.jpg")
End Sub
```

```
Private Sub Sig8_Click()
   If Sig8.Value = 0 Then
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL -8")
```

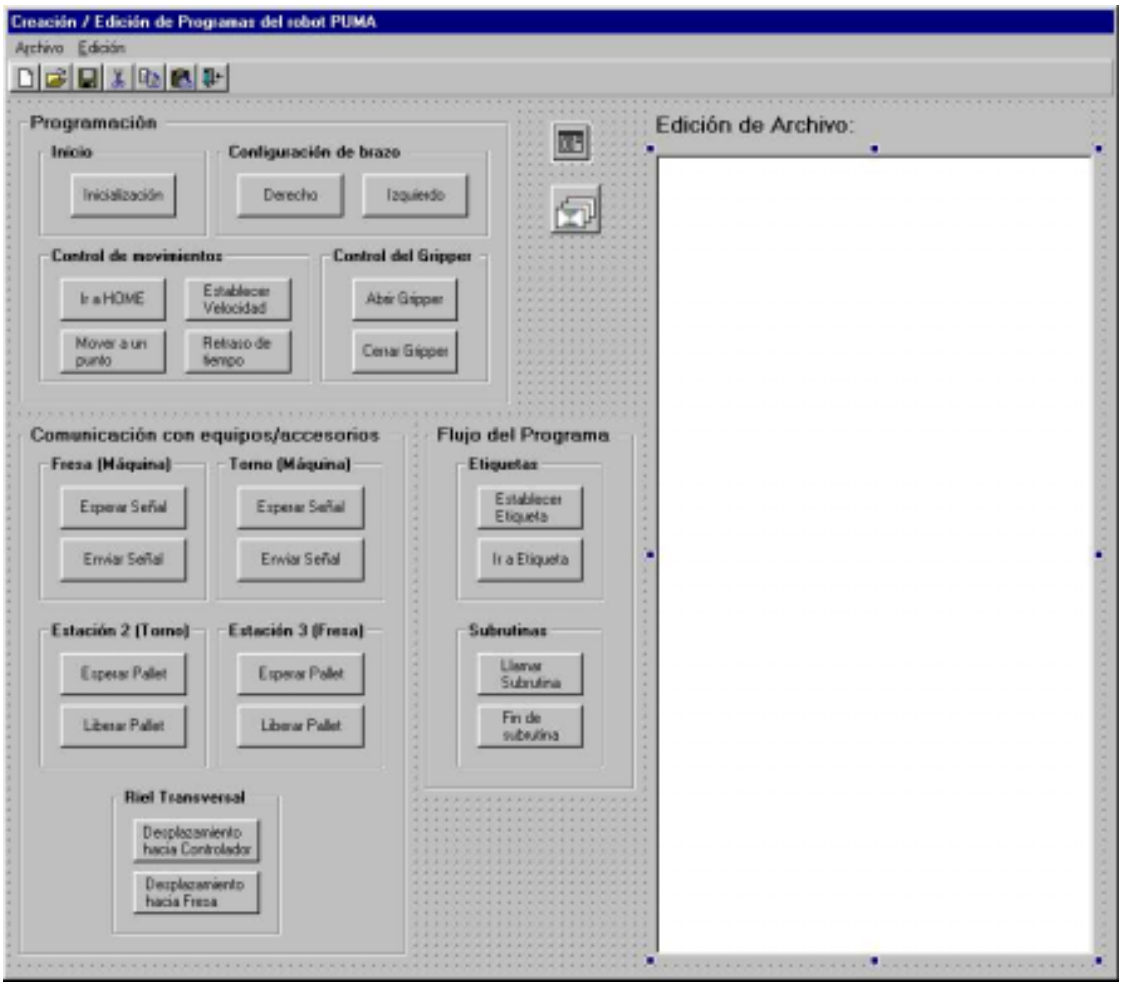
```
      End If
      Sig8.Caption = "Señal 8"
   Else
      If Panel_Principal.conRobot Then
         Panel_Principal.Pfn_Mandar_Cadena ("SIGNAL 8")
      End If
      Sig8.Caption = "Señal -8"
   End If
End Sub
```

```
Private Sub Sig8_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
   Image1.Picture = LoadPicture(App.Path & "\Signal8.jpg")
End Sub
```

Anexo I

Forma y Código de la Pantalla del Editor de Programas

Forma de la Pantalla del Editor de Programas



Código de la Pantalla del Editor de Programas

```
Dim cancelo As Boolean
Dim textoini As String
```

```
Private Sub AbrirG_Click()
   Dim mssge As String

   mssge = "DELAY 0.5" + vbCrLf
   mssge = mssge + "OPENI" + vbCrLf
   mssge = mssge + "DELAY 0.5" + vbCr
   insertar (mssge)
   RichTextBox1.SetFocus

End Sub
```

```
Private Sub CerrarG_Click()
   Dim mssge As String

   mssge = "DELAY 0.5" + vbCrLf
   mssge = mssge + "CLOSEI" + vbCrLf
   mssge = mssge + "DELAY 0.5" + vbCr
   insertar (mssge)
   RichTextBox1.SetFocus
End Sub
```

```
Private Sub Copy_Click()
   Clipboard.SetText RichTextBox1.SelText
   RichTextBox1.SetFocus
End Sub
```

```
Private Sub Cut_Click()
   Dim Inicio As Integer

   Clipboard.SetText RichTextBox1.SelText
   Inicio = RichTextBox1.SelStart
   If RichTextBox1.SelStart <> 0 Then
      RichTextBox1.Text = Mid(RichTextBox1.Text, 1, RichTextBox1.SelStart) +
Mid(RichTextBox1.Text, RichTextBox1.SelStart + RichTextBox1.SelLength,
Len(RichTextBox1.Text))
   Else
      RichTextBox1.Text = Mid(RichTextBox1.Text, RichTextBox1.SelLength + 1,
Len(RichTextBox1.Text))
   End If
   RichTextBox1.SelStart = Inicio
   RichTextBox1.SetFocus
```

```
End Sub
```

```
Private Sub Declarar_Click()
   Dim etiqueta As String
   Dim Inicio As Integer

   Do
      etiqueta = InputBox$("Escriba el número de etiqueta", "Edición de programas del robot
PUMA")
      Inicio = Val(etiqueta)
   Loop Until Inicio > 0 Or etiqueta = ""

   If etiqueta <> "" Then
      Inicio = RichTextBox1.SelStart
      etiqueta = etiqueta + " "
      If Inicio > 0 Then
         RichTextBox1.Text = Mid(RichTextBox1.Text, 1, RichTextBox1.SelStart) + etiqueta
+ Mid(RichTextBox1.Text, RichTextBox1.SelStart + 1, Len(RichTextBox1.Text) - Inicio)
      Else
         RichTextBox1.Text = etiqueta + Mid(RichTextBox1.Text, 1,
Len(RichTextBox1.Text))
      End If
      RichTextBox1.SelStart = Inicio + Len(etiqueta)
      RichTextBox1.SetFocus
   End If
End Sub
```

```
Private Sub Delays_Click()
   Dim mssge As String
   Dim tiempo As String
   Dim verifica As Single

   Do
      tiempo = InputBox$("Escriba el valor del retraso de tiempo (segundos)", "Edición de
programas del robot PUMA")
      verifica = Val(tiempo)
   Loop Until verifica > 0 Or tiempo = ""
   If tiempo <> "" Then
      mssge = "DELAY " + tiempo + vbCr
      insertar (mssge)
      RichTextBox1.SetFocus
   End If
End Sub
```

```
Private Sub DesplContrl_Click()
   Dim mssge As String
```

```
    mssge = "SIGNAL -6" + vbCrLf
    mssge = mssge + "SIGNAL 5" + vbCrLf
    mssge = mssge + "WAIT SIG(1006)" + vbCrLf
    mssge = mssge + "DELAY 2" + vbCrLf
    mssge = mssge + "SIGNAL -5" + vbCrLf
    mssge = mssge + "SIGNAL 8" + vbCr
    insertar (mssge)
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub DesplFresa_Click()
    Dim mssge As String

    mssge = "SIGNAL -8" + vbCrLf
    mssge = mssge + "SIGNAL 7" + vbCrLf
    mssge = mssge + "WAIT SIG(1005)" + vbCrLf
    mssge = mssge + "DELAY 2" + vbCrLf
    mssge = mssge + "SIGNAL -7" + vbCrLf
    mssge = mssge + "SIGNAL 6" + vbCr
    insertar (mssge)
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub Exit_Click()
    Dim textofin As String
    Dim opt As Integer

    textofin = RichTextBox1.Text
    If textoini <> textofin Then
        opt = MsgBox("¿Desea guardar los cambios?", vbQuestion + vbYesNoCancel, "Edición
de programas del robot PUMA")
        If opt = 6 Then
            Call Save_Click
            If cancelo Then
                opt = 2
            End If
        End If
    End If
    If opt <> 2 Then
        Unload Edición
    End If
End Sub
```

```
Private Sub Form_Load()
    Dim mssge As String

    mssge = "SIGNAL -1, -2, -3, -4, -5, -6, -7, -8" + vbCrLf
```

```
   mssge = mssge + "READY" + vbCr
   insertar (mssge)
   textoini = RichTextBox1.Text
End Sub
```

```
Private Sub gosub_Click()
   Dim mssge As String
   Dim nombre As String

   nombre = InputBox$("Escriba el nombre de la subrutina", "Edición de programas del robot
PUMA")

   If nombre <> "" Then
      mssge = "GOSUB " + nombre + vbCr
      insertar (mssge)
      RichTextBox1.SetFocus
   End If
End Sub
```

```
Private Sub goto_Click()
   Dim mssge As String
   Dim etiqueta As String
   Dim verifica As Integer

   Do
      etiqueta = InputBox$("Escriba el número de la etiqueta", "Edición de programas del
robot PUMA")
      verifica = Val(etiqueta)
   Loop Until verifica > 0 Or etiqueta = ""
   If etiqueta <> "" Then
      mssge = "GOTO " + etiqueta + vbCr
      insertar (mssge)
      RichTextBox1.SetFocus
   End If
End Sub
```

```
Private Sub Home_Click()
   Dim mssge As String

   mssge = "READY" + vbCr
   insertar (mssge)
   RichTextBox1.SetFocus
End Sub
```

```
Private Sub Inicializar_Click()
   Dim mssge As String
```

```
    mssge = "SIGNAL -1, -2, -3, -4, -5, -6, -7, -8" + vbCrLf
    mssge = mssge + "READY" + vbCr
    insertar (mssge)
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub Lefty_Click()
    Dim mssge As String

    mssge = "LEFTY" + vbCr
    insertar (mssge)
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub Mover_Click()
    MoveEdit.Show 1
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub New_Click()
    Dim textofin As String
    Dim opt As Integer

    textofin = RichTextBox1.Text
    If textoini <> textofin Then
        opt = MsgBox("¿Desea guardar los cambios?", vbQuestion + vbYesNoCancel, "Edición
de programas del robot PUMA")
        If opt = 6 Then
            Call Save_Click
            titulo.Caption = "Edición de Archivo: "
            If cancelo Then
                opt = 2
            End If
        End If
    End If
    If opt <> 2 Then
        RichTextBox1.Text = ""
        textoini = ""
        titulo.Caption = "Edición de Archivo: "
        RichTextBox1.SetFocus
    End If
End Sub
```

```
Private Sub Open_Click()
    Dim Archivo As String
    Dim linea As String
    Dim mssge As String
```

```
    Dim aviso As String
    Dim textofin As String

    On Error GoTo ManejarErrores

    textofin = RichTextBox1.Text

    If textoini <> textofin Then
        opt = MsgBox("¿Desea guardar los cambios?", vbQuestion + vbYesNoCancel, "Edición
de programas del robot PUMA")
        If opt = 6 Then
            Call Save_Click
            If cancelo Then
                opt = 2
            End If
        End If
    End If
    If opt <> 2 Then
        CommonDialog1.DialogTitle = "Edición de Programas del robot PUMA"
        CommonDialog1.Filter = "Programas PUMA|*.prp|Todos los Archivos|*.*|"
        CommonDialog1.InitDir = App.Path
        CommonDialog1.FileName = ""
        CommonDialog1.ShowOpen
        If Err.Number <> cdlCancel Then
            Archivo = CommonDialog1.FileName
            RichTextBox1.LoadFile Archivo, 1
            titulo.Caption = "Edición de Archivo: " + CommonDialog1.FileTitle
            textoini = RichTextBox1.Text
            RichTextBox1.SelStart = Len(RichTextBox1.Text)
        End If
    End If

ManejarErrores:
    If Err.Number <> 0 Then
        If Err.Number <> cdlCancel Then
            MsgBox "Error" & Err.Number & vbCr & vbCr & Err.Description, vbCritical +
vbOKOnly, "Error en el proceso"
            Err.Clear
        End If
    End If
End Sub
```

```
Private Sub Paste_Click()
    Dim Inicio As Integer
    Dim longlinea As Integer

    Inicio = RichTextBox1.SelStart
```

```
    If Inicio > 0 Then
        RichTextBox1.Text = Mid(RichTextBox1.Text, 1, RichTextBox1.SelStart) +
Clipboard.GetText() + Mid(RichTextBox1.Text, RichTextBox1.SelStart + 1,
Len(RichTextBox1.Text) - Inicio)
    Else
        RichTextBox1.Text = Clipboard.GetText() + Mid(RichTextBox1.Text, 1,
Len(RichTextBox1.Text))
    End If
    longlinea = Len(Clipboard.GetText)
    RichTextBox1.SelStart = Inicio + longlinea
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub return_Click()
    Dim mssge As String

    mssge = "RETURN " + vbCr
    insertar (mssge)
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub righty_Click()
    Dim mssge As String

    mssge = "RIGHTY" + vbCr
    insertar (mssge)
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub Save_Click()
    Dim Archivo As String
    Dim mssge As String
    Dim aviso As String

    On Error GoTo ManejarErrores

    cancelo = False
    CommonDialog1.DialogTitle = "Edición de Programas del robot PUMA"
    CommonDialog1.DefaultExt = "prp"
    CommonDialog1.Filter = "Programas PUMA|*.prp"
    CommonDialog1.InitDir = "C:\PUMA"
    CommonDialog1.Flags = cdlOFNOverwritePrompt
    CommonDialog1.FileName = ""
    CommonDialog1.ShowSave

    Archivo = CommonDialog1.FileName
    RichTextBox1.SaveFile Archivo, 1
```

```
    titulo.Caption = "Edición de Archivo: " + CommonDialog1.FileTitle
    If CommonDialog1.FileTitle = "" Then
       cancelo = True
    Else
       textoini = RichTextBox1.Text
    End If
    RichTextBox1.SetFocus

ManejarErrores:
    If Err.Number <> 0 Then
       If Err.Number <> cdlCancel Then
          MsgBox "Error" & Err.Number & vbCr & vbCr & Err.Description, vbCritical +
vbOKOnly, "Error en el proceso"
          Err.Clear
       End If
    End If
End Sub
```

```
Private Sub Signal1_Click()
    Dim mssge As String

    mssge = "SIGNAL 1" + vbCrLf
    mssge = mssge + "WAIT SIG(-1001)" + vbCrLf
    mssge = mssge + "SIGNAL -1" + vbCr
    insertar (mssge)
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub Signal2_Click()
    Dim mssge As String

    mssge = "SIGNAL 2" + vbCrLf
    mssge = mssge + "WAIT SIG(-1002)" + vbCrLf
    mssge = mssge + "SIGNAL -2" + vbCr
    insertar (mssge)
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub Signal3_Click()
    Dim mssge As String

    mssge = "SIGNAL 3" + vbCrLf
    mssge = mssge + "WAIT SIG(-1003)" + vbCrLf
    mssge = mssge + "SIGNAL -3" + vbCr
    insertar (mssge)
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub Signal4_Click()
   Dim mssge As String

   mssge = "SIGNAL 4" + vbCrLf
   mssge = mssge + "WAIT SIG(-1004)" + vbCrLf
   mssge = mssge + "SIGNAL -4" + vbCr
   insertar (mssge)
   RichTextBox1.SetFocus
End Sub
```

```
Private Sub Speed_Click()
   VelocEdit.Show 1
   RichTextBox1.SetFocus
End Sub
```

```
Public Sub insertar(linea As String)
   Dim Inicio As Integer
   Dim longlinea As Integer

   Inicio = RichTextBox1.SelStart
   longlinea = Len(linea) + 1
   If Inicio > 0 Then
      linea = linea + vbLf
      RichTextBox1.Text = Mid(RichTextBox1.Text, 1, RichTextBox1.SelStart) + linea +
Mid(RichTextBox1.Text, RichTextBox1.SelStart + 1, Len(RichTextBox1.Text) - Inicio)
   Else
      linea = linea + vbLf
      RichTextBox1.Text = linea + Mid(RichTextBox1.Text, 1, Len(RichTextBox1.Text))
   End If
   RichTextBox1.SelStart = Inicio + longlinea
End Sub
```

```
Private Sub Toolbar1_ButtonClick(ByVal Button As ComctlLib.Button)
   Boton = Button.Key
   Select Case Boton

   Case "keyNuevo"
       New_Click

   Case "keyAbrir"
       Open_Click

   Case "keyGuardar"
       Save_Click

   Case "keyCortar"
```

```
        Cut_Click

    Case "keyCopiar"
        Copy_Click

    Case "keyPegar"
        Paste_Click

    Case "keySalir"
        Exit_Click
    End Select
End Sub
```

```
Private Sub Wait11_Click()
    Dim mssge As String

    mssge = "WAIT SIG(1001)" + vbCr
    insertar (mssge)
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub Wait12_Click()
    Dim mssge As String

    mssge = "WAIT SIG(1002)" + vbCr
    insertar (mssge)
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub wait13_Click()
    Dim mssge As String

    mssge = "WAIT SIG(1003)" + vbCr
    insertar (mssge)
    RichTextBox1.SetFocus
End Sub
```

```
Private Sub wait14_Click()
    Dim mssge As String

    mssge = "WAIT SIG(1004)" + vbCr
    insertar (mssge)
    RichTextBox1.SetFocus
End Sub
```

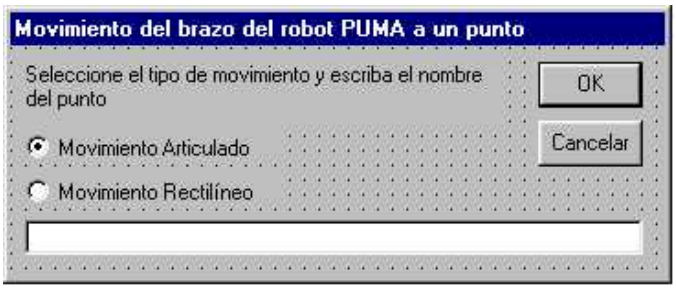
Anexo J

Forma y Código

de la Pantalla para Mover al Robot a un Punto

en el Editor

Forma de la Pantalla para Mover al Robot a un Punto en el Editor



Código de la Pantalla para Mover al Robot a un Punto en el Editor

```
Private Sub Cancel_Click()
   Unload MoveEdit
End Sub
```

```
Private Sub Aceptar_Click()
   Dim mssge As String
   If OptArt Then
      mssge = "MOVE "
   Else
      mssge = "MOVES "
   End If
   If NombrePunto.Text <> "" Then
      mssge = mssge + NombrePunto.Text + vbCr
      Edición.insertar (mssge)
      Unload MoveEdit
   Else
      mssge = "Es necesario que escriba el nombre del punto"
      MsgBox mssge, vbCritical + vbOKOnly, "Edición de Programas del robot PUMA"
   End If
End Sub
```

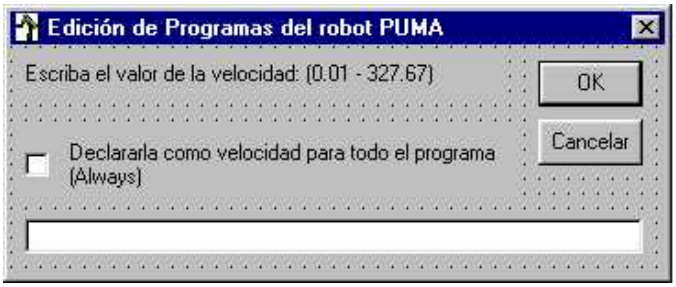
```
Private Sub Form_Activate()
   NombrePunto.SetFocus
End Sub
```

Anexo K

Forma y Código

de la Pantalla para declarar Velocidad en el Editor

Forma de la Pantalla para Declarar Velocidad en el Editor



Código de la Pantalla para Declarar Velocidad a un Punto en el Editor

```
Private Sub Aceptar_Click()
   Dim mssge As String

   If ValorVeloc.Text <> "" Then
      If CSng(ValorVeloc.Text) >= 0.01 And CSng(ValorVeloc.Text) <= 327.67 Then
         If ChkAlways Then
            mssge = "SPEED " + ValorVeloc.Text + " ALWAYS"
         Else
            mssge = "SPEED " + ValorVeloc.Text
         End If
         mssge = mssge + vbCr
         Edición.insertar (mssge)
         Unload VelocEdit
      Else
         mssge = "El valor de velocidad debe estar entre 0.01 y 327.67"
         MsgBox mssge, vbCritical + vbOKOnly, "Edición de Programas del robot PUMA"
      End If
   Else
      mssge = "Es necesario que escriba el valor de velocidad"
      MsgBox mssge, vbCritical + vbOKOnly, "Edición de Programas del robot PUMA"
   End If
End Sub
```

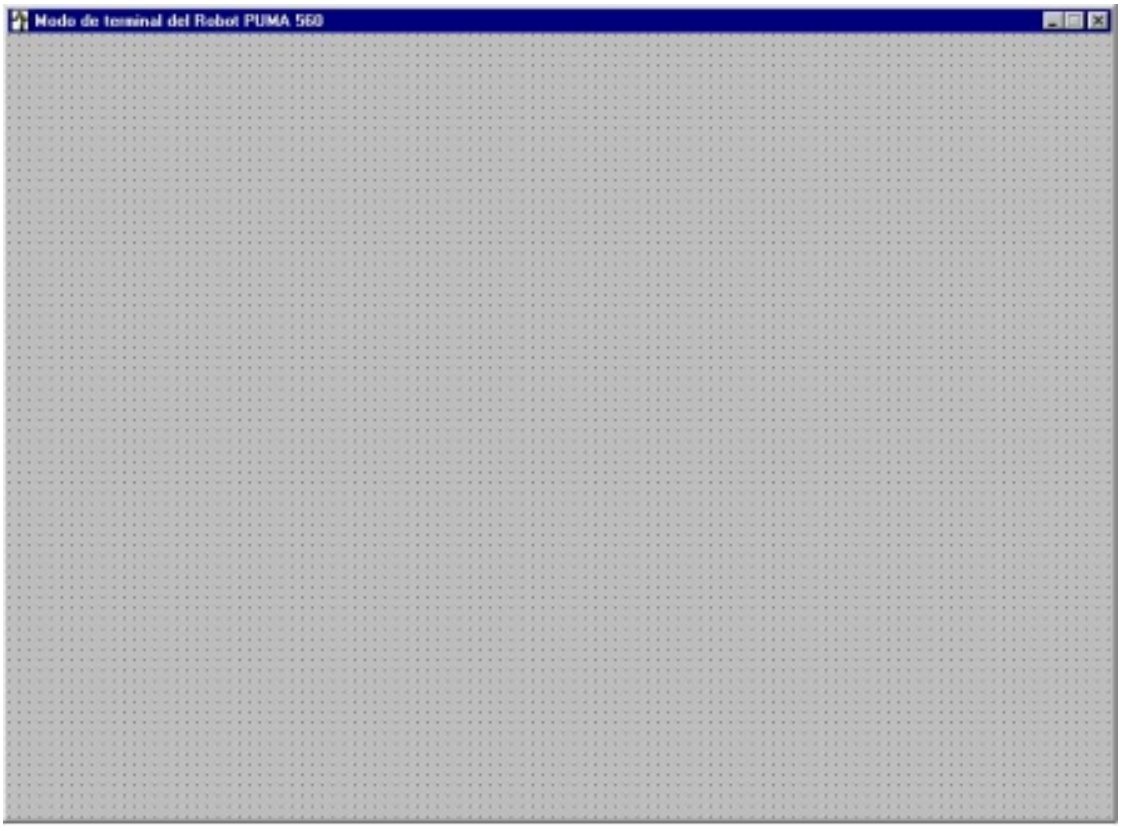
```
Private Sub Cancel_Click()
   Unload VelocEdit
End Sub
```

```
Private Sub Form_Activate()
   ValorVeloc.SetFocus
End Sub
```

Anexo L

Forma y Código de la Pantalla del Modo Terminal

Forma de la Pantalla del Modo Terminal



Código de la Pantalla del Modo Terminal

```
Dim Pantalla As String
Dim I As Integer
Dim J As Integer
Dim linea As Integer
```

```
Private Sub Form_KeyPress(KeyAscii As Integer)

Dim CharRes As String
Dim Char_Ant As String

   Char = Chr(KeyAscii)

   On Error GoTo ManejarErrores

   If KeyAscii <> 27 Then
      Panel_Principal.MSComm1.Output = Char
      If KeyAscii = 8 Then
         Do
            Do
               DoEvents
            Loop Until Panel_Principal.MSComm1.InBufferCount >= 1
            CharRes = Panel_Principal.MSComm1.Input
         Loop Until Panel_Principal.MSComm1.InBufferCount = 0
         If Len(Pantalla) > 0 Then
            If (Mid(Pantalla, Len(Pantalla), 1) <> ".") And (Mid(Pantalla, Len(Pantalla) - 1, 1)
<> vbCr) Then
               Pantalla = Mid(Pantalla, 1, Len(Pantalla) - 1)
               TermMode.Cls
               Print Pantalla;
            End If
         End If
      Else
         If KeyAscii = 13 Then
            Do
               Char_Ant = CharRes
               Do
                  DoEvents
               Loop Until Panel_Principal.MSComm1.InBufferCount >= 1
               CharRes = Panel_Principal.MSComm1.Input
               If CharRes <> vbLf Then
                  Print CharRes;
                  Pantalla = Pantalla + CharRes
               Else
```

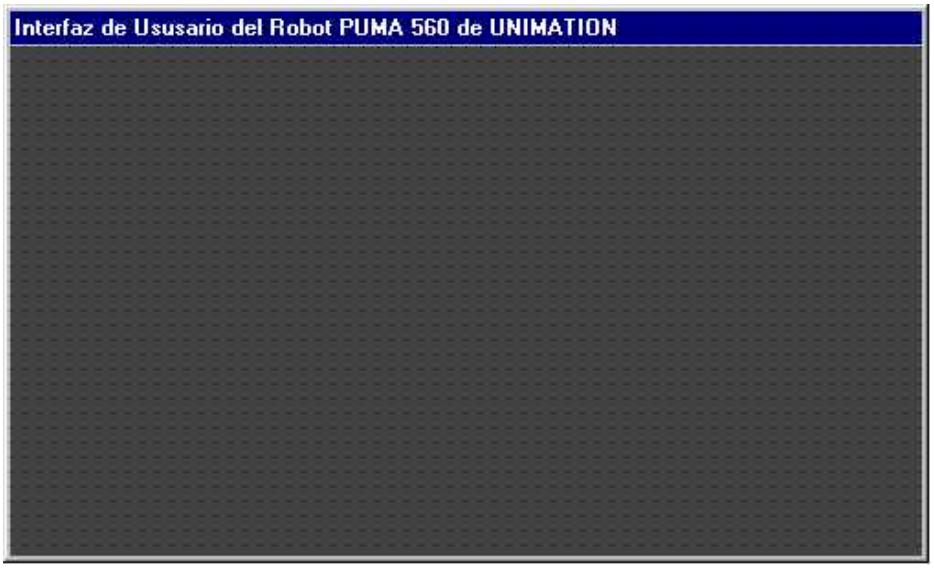
```
                linea = linea + 1
                If linea > 43 Then
                   I = 0
                   J = 0
                   linea = 44
                   Do
                      I = I + 1
                      If Mid(Pantalla, I, 1) = vbCr Then
                         Pantalla = Mid(Pantalla, I + 1, Len(Pantalla) - I)
                         TermMode.Cls
                         Print Pantalla;
                         J = 1
                      End If
                   Loop Until J <> 0
                End If
             End If
          Loop Until ((CharRes = "." And Char_Ant = vbLf) Or (CharRes = "?")) And
Panel_Principal.MSComm1.InBufferCount = 0
       Else
          Do
             Do
                DoEvents
             Loop Until Panel_Principal.MSComm1.InBufferCount >= 1
             CharRes = Panel_Principal.MSComm1.Input
             Print CharRes;
             Pantalla = Pantalla + CharRes
          Loop Until Panel_Principal.MSComm1.InBufferCount = 0
       End If
    End If
 Else
    Pantalla = ""
    linea = 0
    Unload TermMode
 End If


ManejarErrores:

 If Err.Number <> 0 Then
    MsgBox "Error " & Err.Number & vbCr & vbCr & Err.Description, vbCritical +
vbOKOnly, "Error en el proceso"
    Err.Clear
 End If

End Sub
```

Anexo M

Forma y Código

de la Pantalla de Fondo de la Interfaz de Usuario

Forma de la Pantalla de Fondo de la Interfaz de Usuario



Código de la Pantalla de Fondo de la Interfaz de Usuario

```
Private Sub Form_Activate()
   Panel_Principal.Show
End Sub
```