INSTITUTO TECNOLOGICO Y DE ESTUDIOS
SUPERIORES DE MONTERREY

CAMPUS MONTERREY
DIVISION DE INGENIERIA Y ARQUITECTURA
PROGRAMA DE GRADUADOS EN INGENIERIA

TECNOLÓGICO
DE MONTERREY®

INTERFAZ DE USUARIO PARA EL ROBOT MOTOMAN YASNAC K3

TESIS

PRESENTADA COMO REQUISITO PARCIAL PARA
OBTENER EL GRADO ACADEMICO DE:

MAESTRO EN CIENCIAS
ESPECIALIDAD EN SISTEMAS DE CALIDAD
Y PRODUCTIVIDAD

POR:
ENRIQUE GUAJARDO SANTOS

MONTERREY, N. L. MAYO DEL 2003

INSTITUTO TECNOLOGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

CAMPUS MONTERREY
DIVISION DE INGENIERIA Y ARQUITECTURA
PROGRAMA DE GRADUADOS EN INGENIERIA

TECNOLÓGICO DE MONTERREY®

INTERFAZ DE USUARIO PARA EL ROBOT MOTOMAN YASNAC K3

TESIS

PRESENTADA COMO REQUISITO PARCIAL PARA OBTENER EL GRADO ACADEMICO DE:

MAESTRO EN CIENCIAS
ESPECIALIDAD EN SISTEMAS DE CALIDAD
Y PRODUCTIVIDAD

POR:
ENRIQUE GUAJARDO SANTOS

MONTERREY, N. L. MAYO DEL 2003

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

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

INTERFAZ DE USUARIO PARA EL ROBOT MOTOMAN YASNAC K3

TESIS

PRESENTADA COMO REQUISITO PARCIAL PARA OBTENER EL GRADO ACADEMICO DE:

MAESTRO EN CIENCIAS
ESPECIALIDAD EN SISTEMAS DE CALIDAD Y PRODUCTIVIDAD

POR:

ENRIQUE GUAJARDO SANTOS

MONTERREY, N.L. MAYO DE 2003

DEDICATORIA

Dedico con todo mi corazón y mi amor cada minuto de trabajo de esta tesis a Dios dueño y Señor de mi vida de quién he recibido todo en Cristo. A El la gloria por los siglos de los siglos.

A mis padres Enrique Guajardo Martínez y Elva Aída Santos De Guajardo quienes con gran amor, comprensión, sacrificio incansable, apoyo incondicional y ejemplo me han dado una educación y una vida muy feliz construida sobre valores cristianos y morales. Gracias a Dios por mis padres.

A mi hermano Adrián a quien quiero entrañablemente.

A Los Siervos De LA Palabra que me han dado a conocer "La perla de gran valor". Con los que comparto la dicha de Servir Cristo con devoción indivisa, de amarle y que sea amado.

AGRADECIMIENTOS

Agradezco al **Ing. Heriberto García Reyes** asesor de esta tesis por todo su apoyo para este proyecto, por su interés y empeño que desde que le conozco, ha tenido en hacer de mi no solo un buen estudiante sino también una mejor persona. Muchas gracias.

Al ing. Rafael Bourguet y al **ing. Rodolfo Villarreal** por haber aceptado ser parte del comité de esta tesis, por su apoyo y aportaciones de gran valor para el desarrollo de este proyecto.

Al **ing. Eduardo García Dunna** quien me aceptó en el programa de asistentes de docencia dándome la oportunidad de hacer esta maestría.

Al ing. **Gerardo Vallejo** por su ayuda en aspectos técnicos y por facilitarme herramientas para esta investigación.

Al **equipo de asistentes del departamento de ingeniería industrial** por el animo que me transmitieron y sus aportaciones para este trabajo.

A los **alumnos del Laboratorio de Sistemas Integrados De manufactura** que me apoyaron con comentarios y con experimentos de este proyecto.

A **Luis Manuel Bravo, Pepe Vázquez, David Mijares, Israel Lizárraga y Javier Fernández,** por brindarme animo, comprensión, consejo y ejemplo durante estos años tan provechosos que hemos compartido. Gracias a Dios por sus vidas.

A **Helio García** por su amistad , por compartir conmigo su experiencia de haber hecho una maestría y por el buen ejemplo de su vida.

Al **Ing. Juan Francisco Gaytán** por su ayuda para iniciarme como asistente y como estudiante de maestría.

ÍNDICE GENERAL

Capítulo 1 INTRODUCCIÓN

1.1 PRESENTACIÓN

La robótica se ha convertido en un concepto crucial para la automatización y la integración de los sistemas flexibles de manufactura. El uso de manipuladores mecánicos ocurre en la industria en aplicaciones de ensamble, transporte, inspección, etc. Las marcas, los tipos y las características de los manipuladores son muy variadas, tanto, como la operación lo requiera.
La forma de programarlos depende del diseño hecho por el fabricante, generalmente es por medio de un controlador alámbrico. La interfaz entre el controlador alámbrico y el usuario es limitada, en ocasiones impráctica. Con el tiempo, controlar el robot por medio de esta interfase se ha hecho cada vez más sencillo. Sin embargo la vinculación usuario-máquina está incompleta si no existe una interfaz computacional con opción a trabajar fuera de línea.

Con esta tesis se pretende desarrollar una interfaz de programación y manipulación del robot MOTOMAN (Yasnac MRC K3) con opción a trabajar fuera de línea. La interfaz del usuario para el robot Motoman Yasnac MRC K3 es un programa que tiene por objetivo facilitar la operación de este robot, y a su vez, facilitar el aprendizaje del manejo del robot, para los alumnos del laboratorio de sistemas integrados de manufactura de la carrera de Ingeniería industrial y de Sistemas del ITESM.

1.2 Antecedentes

1.2.1 Automatización

Hay 3 tipos de automatización industrial: automatización fija, automatización programable y Automatización flexible.

La automatización fija se utiliza cuando el volumen de producción es muy alto, y por tanto se puede justificar económicamente el alto costo del diseño de equipo especializado para procesar el producto con un rendimiento alto y tasas de producción elevadas.
El inconveniente de la automatización fija es su ciclo de vida que va de acuerdo a la vigencia del producto en el mercado Un ejemplo de la automatización fija puede encontrarse en la industria del automóvil, en donde líneas de transferencia muy integradas constituidas por varias decenas de estaciones de trabajo se utilizan para operaciones de mecanizado en componentes de motores y transmisiones. [1]

La automatización programable se emplea cuando el volumen de producción es relativamente bajo y hay una diversidad de producción. En este caso el equipo de producción está diseñado para ser adaptable a variaciones en la configuración del producto. Esta característica de adaptabilidad se realiza haciendo funcionar el equipo bajo el control de un programa de instrucciones que van de acuerdo al producto. [2]

La automatización flexible: es una categoría entre automatización fija y automatización programable. Este tipo de automatización se ha visto que es más adecuado para el rango de producción medio. Una de las características que distingue a la automatización programable de la flexible, es que con la primera los productos se obtienen en lote. [2]

Cuando se completa un lote, el equipo sé reprograma para procesar el siguiente lote. Con la automatización flexible diferentes tipos de productos pueden obtenerse al mismo tiempo en el mismo sistema de fabricación. Esta característica permite un nivel de versatilidad que no está disponible en la automatización programable. [1]

Las celdas de manufactura son un conjunto de máquinas automáticas posicionadas en estaciones de trabajo controladas por una computadora principal. La finalidad de estas celdas es la manufactura de un producto. Las máquinas de control numérico son máquinas que se usan para dar forma a piezas por medio de desbaste de material. Un torno y una fresadora automáticos son ejemplo de máquinas de

control numérico. Los sistemas automatizados flexibles suelen estar constituidos por una serie de estaciones de trabajo que están interconectadas por un sistema de almacenamiento y manipulación de materiales. Una computadora central se utiliza para controlar las diversas actividades que se producen en el sistema, encaminando las diversas piezas a las estaciones adecuadas y controlando las operaciones programadas en las diferentes estaciones. [1]

De los tres tipos de automatización, la robótica coincide más estrechamente con la automatización programable. [1]

1.2.2 Robótica

Con el nacimiento de la Revolución Industrial muchas fábricas comenzaron a utilizar con gran aceptación la automatización de procesos repetitivos en la línea de ensamble. La mayoría de las industrias han sido automatizadas o utilizan tecnología para automatizar algunas labores, sin embargo no todas las industrias requieren el mismo grado de automatización, de hecho algunas son difíciles de automatizar como la industria de la agricultura que tienen mucha variabilidad en sus procesos.

Cuando comenzaron a darse a conocer los primeros robots se decía que sustituirían al hombre y que nuestra sociedad iba encaminada al desplazamiento de los seres humanos de sus tareas más cotidianas de trabajo y que esto desencadenaría la devaluación del hombre y un incremento en el desempleo. En la actualidad se puede demostrar que especulaciones que hablaban sobre el desplazamiento del hombre por la máquina han resultado erróneas. El motivo principal es que el criterio de implantación de un robot estriba en las capacidades del robot para trabajar en ambientes extremos y en poder llevar a cabo actividades repetitivas con calidad. No todas las tareas humanas pueden ser sustituidas por un robot. Los robots son utilizados en la actualidad para sustituir a los humanos en tareas peligrosas o nocivas para la salud física y mental del individuo y donde es económicamente justificable.

En la industria automotriz algunas de las aplicaciones más importantes son en la soldadura y el manejo de materiales como se puede observar en las figuras 1.1 y 1.2.

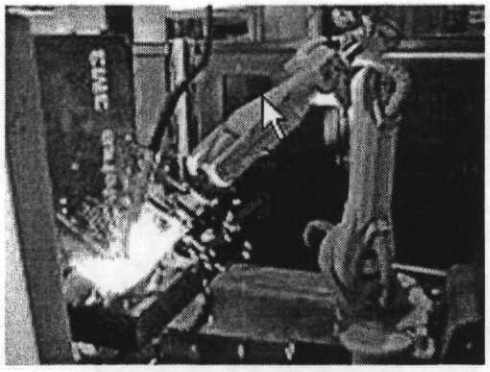


Fig. 1.1 Robot soldando en superficie

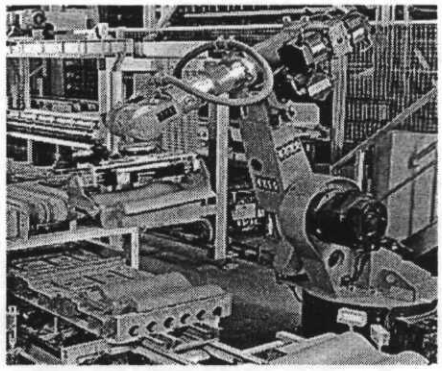


Fig. 1.2 Robot cargando una teja

Otra aplicación es en la industria electrónica para ensambles de mucha precisión y velocidad, que para un humano sería muy difícil repetir con calidad y rapidez. También se usan en la medicina para desempeñar trabajos en ambientes con alto riesgo de contagio o donde se necesita de precisión milimétrica, igualmente se usan en el manejo de materiales nucleares y químicos tóxicos.

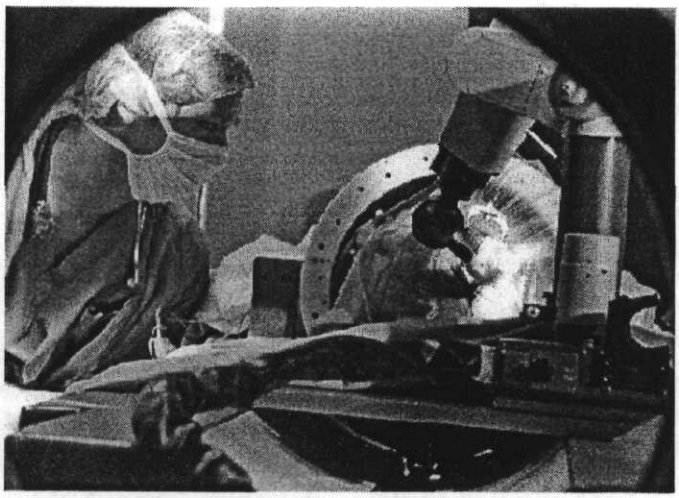


Fig. 1.3 Robot asistiendo en una cirugía

Recientemente se han usado los robots en la exploración de otros planetas como es el caso del robot Pathfinder enviado para explorar la superficie de Marte y controlado desde el planeta tierra.

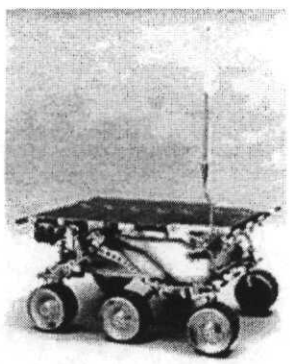


Fig. 1.4 Robot Pathfinder

1.2.3 Características importantes de los robots

Un robot se define como una máquina controlada automáticamente, reprogramable, multifuncional que puede ser de instalación fija o variable. Reprogramable y multifuncional son dos características que muchas de las máquinas no tienen. El hecho de ser reprogramable implica que el movimiento del robot es controlado por un programa que fue escrito anteriormente y que el programa del robot puede ser

modificado para así poder modificar sus rutinas o la secuencia de las mismas. Estos cambios se pueden hacer en tiempo real, por ejemplo, un brazo que debe de recoger partes de una posición variable debe de ser equipado con una cámara que le dé información al robot para que mientras se traslada a un lugar cercano modifique la estructura de su programa y se dirija al lugar correcto. [5]
Los Procesos automatizados se implementan generalmente por una necesidad de elevar la productividad o la calidad del producto. Las marcas, los tipos y las características de los manipuladores son muy variadas, tanto, como la operación lo requiera. La capacidad de modificar en vivo o fuera de línea sus programas dependen de factores como sensores, método de programación y medios de comunicación con el robot. [6]

1.2.4 Métodos de programación de robots

La programación de un robot se logra al establecer comunicación con su controlador. Establecer esta comunicación consiste en mandar comandos que el robot pueda reconocer y guardar en la memoria. Se supone que estos comandos fueron dados de alta desde la fábrica de manera que se pudiera establecer esta comunicación.

La programación de un robot tiene dos componentes que son:

1. Programar las posiciones por las que va describirse la trayectoria de trabajo.
2. Programar la secuencia del proceso incluyendo interacción con otros equipos o con su entorno.

Para mandar los comandos se usa una interfaz entre el robot y la máquina. A este dispositivo se le llama interfaz y está diseñada para que el que la opera se ajuste al ambiente de programación del robot, puede ser de 3 tipos: 1) por medio de una pantalla y un teclado conectados al robot usando un lenguaje propio del robot, 2) por medio de un control alámbrico y 3), una combinación de ambas.

La interfaz se utiliza para programar las posiciones, para elaborar el programa que mandará llamar dichas posiciones y algunas otras tareas en el orden estipulado por el usuario en el programa.
Estas 3 interfaces se pueden sintetizar en una sola por medio de una interfaz en ambiente Windows ajustada a las necesidades del usuario a través de una computadora personal. Con esto se logra que el robot se ajuste a la configuración del usuario para programarlo, en vez de que el usuario se ajuste al ambiente de programación del robot. Con esta interfaz los comandos especiales del robot para ejecutar tareas se pueden reducir a instrucciones de fácil uso y entendimiento para el usuario en un ambiente amigable y que la PC interpreta en el lenguaje del robot.

1.2.5 La programación de robots en la industria actual

La operación de los robots se optimiza cuando se les puede mantener en producción continua.
La velocidad y eficiencia de la programación es clave para lograrlo.
Los requerimientos de la industria en la actualidad para programación de robots incluyen:

- Programar los robots sin sacarlos de su línea de trabajo.
- Programar aún sin la presencia física de materia prima disponible
- Programar en cualquier parte que el programador se encuentre sin la necesidad de estar en la fábrica.
- Que se tenga la facilidad de que cualquier personal, especialmente los más talentosos, contribuyan a la programación.
- La disponibilidad de todas las herramientas que agilizan la labor de programación

Los sistemas de programación fuera de línea cumplen con los primeros tres requerimientos, es obvio que los métodos tradicionales de programación no ayudarán a cumplir el ideal de una operación robótica continua. En la actualidad se requiere prescindir de la programación que requiere presencia del programador en la planta. El mejor talento de la programación puede ser encontrado en la oficina, mientras que el mejor conocimiento del proceso está en planta.

Este Análisis lleva a concluir en estos requerimientos básicos para sistemas de programación dentro y fuera de línea:

- U so intuitivo y fácil de dominar
- Q ue agilice la tarea de la programación
- Q ue permita una revisión colaborativa de los programas a través de los medios de comunicación existente.

Un ejemplo que ayuda a entender la relevancia de la necesidad de programar fuera de línea se observa en esta situación ocurrida en una empresa automotriz en una de sus plantas de ensamble en la que se enfrentaban al problema de cambiar robots antiguos por robots nuevos de marca distinta. Se requería cambiar las unidades sin detener la línea de producción. Esta operación se vuelve complicada cuando se tienen en cuenta las cosas que tienen que estar listas para que el nuevo robot comience funcionar, por ejemplo: calibrar, cargar posiciones y programas. Dennis Archer del departamento de ingeniería de manufactura avanzada de esta empresa dijo en 1999 en la conferencia NASCAPE en Detroit: "Sabemos que las definiciones de fuera de línea varían en nuestra industria, nuestra definición es muy simple: No toques el control alámbrico del robot, debemos mantener las interrupciones de la producción al mínimo. Utilizaron software de simulación y de progrmación fuera de líena antes de llevar a cabo esta peligrosa operación de la que dependia en gran parte del éxito de la empresa ese año. [4]
Este ejemplo apunta a que de no haber sido por el trabajo previo no se hubiera logrado tener listos los robots nuevos para entrar en operación sin detener la producción. Este trabajo previo se hizo fuera de línea con paquetes de simulación y programación.

1.2.6 Interfaz del robot Motoman

Existen en el mercado muchos diseños de interfaz para los distintos tipos de robots incluyendo para MOTOMAN (Yasnac MRC K3), las capacidades son muy variadas y dependen de las necesidades del usuario o del proceso. El costo es muy elevado en algunos casos y es justificable si las necesidades lo ameritan. En el pasado para el laboratorio de sistemas integrados de manufactura se han hecho otras interfases para otros tipos de robots, recientemente la del robot Puma Unimate 500 la cuál ha facilitado la labor de enseñanza de este robot. y la del robot Mitsubishi hecha en el ITESM campus estado de México. La interfaz para el robot Motoman es del tipo interfaz hombre-máquina con fines didácticos.

El robot MOTOMAN (Yasnac MRC K3) opera dentro de una celda de manufactura. Esta celda de manufactura se encuentra en el ITESM campus MTY y es usada por alumnos de 7 y 8 vo semestre en el laboratorio de Sistemas Integrados De Manufactura. El objetivo de este laboratorio es llevar a los alumnos a tener un contacto muy cercano con los equipos de manufactura modernos. Estos equipos son vistos en clase de manera teórica. La celda cuenta con los siguientes componentes:

1. Computadora central: esta cuenta con un programa que integra todas las operaciones dela celda incluyendo las del robot Motoman.
2. Almacén.
3. Sistema de manejo de pallets.
4. Estación de maquinado.
5. Robot Motoman Yasnac MRC K3.
6. Estación de visión.
7. Estación de ensamble.
8. Computadora central

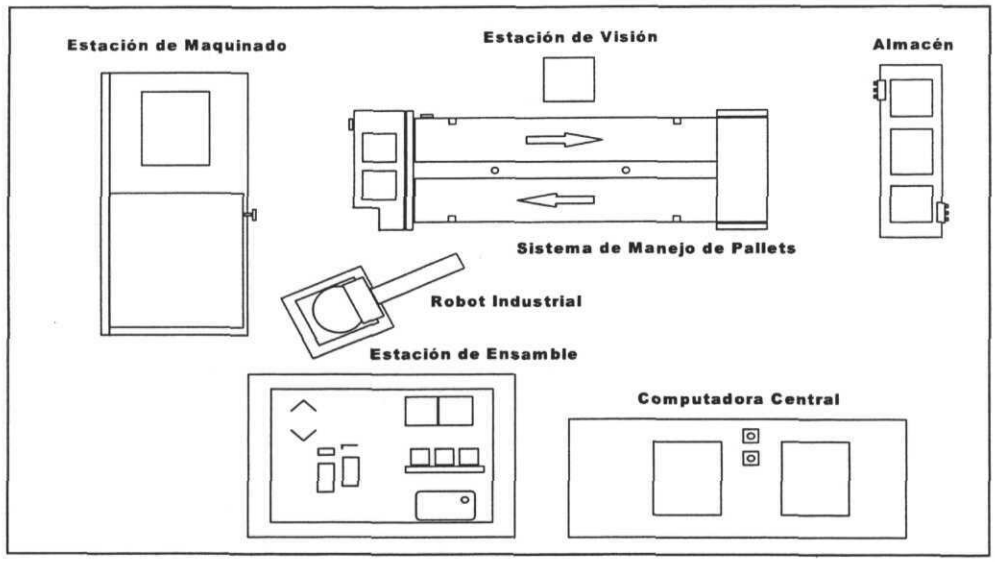


Fig. 1.5 Layout de la celda flexible de manufactura

En esta celda el robot Motoman se encarga de transferir las piezas para ensamble a los buffer de ensamble o a la estación del maquinado y/o a la mesa de ensamble. También participa en las labores de ensamble de productos. Para lo anterior el robot puede seleccionar una de tres herramientas ubicadas en el almacén de herramientas, la selección dependerá del material y la forma de la pieza a transportar o ensamblar. Algunos ejemplos son el gripper (manipulador), la ventosa y el desarmador.

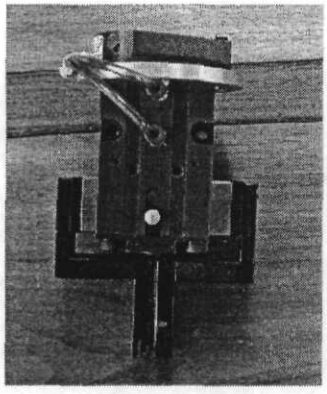


Fig. 1.6 Manipulador del robot

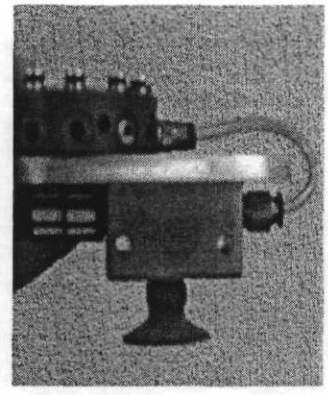


Fig. 1.7 Ventosa

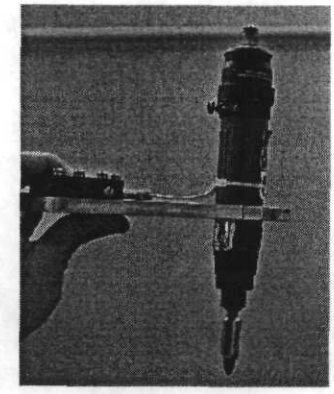


Fig. 1.8 Desarmador

1.3 La problemática

Una de las desventajas de la robótica son los largos tiempos que se invierten en la programación para la ejecución de tareas. Esta programación generalmente se lleva a cabo en el lugar en el que se encuentra el manipulador y por medio de un controlador alámbrico.

El robot Motoman cuenta con un control alámbrico de pantalla digital como única interfaz con el usuario. El ambiente de programación de este control es amigable para alguien que está en el contexto de la programación de estos robots y los conoce bien, por otro lado para el usuario que se topa por primera vez con este control constituye una enorme inversión de tiempo para comprender la lógica de las funciones de las teclas y saber a que menú de los mostrados en la pantalla se debe dirigir.

- El robot Motoman es utilizado constantemente por nuevos usuarios en proceso de capacitación. Generalmente se cuenta con poco tiempo para aprender a usar las operaciones básicas del robot y poder cumplir posteriormente con los requisitos de elaboración de un reporte de la operación realizada. La mayoría de los usuarios nunca ha tenido contacto con un robot de este tipo.

- El usuario tiene que navegar por una serie de menús y aprender una gran cantidad de comandos, esto debido a las limitaciones que presenta tener tanta información de una interfaz del tamaño de una calculadora gráfica.

- El manejo de errores de programación es complicado ya que requiere la desactivación de varias funciones antes de poder continuar programando. Los mensajes de error no son muy claros debido al tamaño de la pantalla del control.

- El respaldo de información se queda en el robot debido a la ausencia de un dispositivo de discos de 3 ½ o de una interfaz con una computadora personal.

- La pantalla de ensamble de programas es también confusa y su diseño hace que el nuevo usuario cometa errores de programación con mucha frecuencia.

- La interfaz actual del robot Motoman hace que el aprendizaje de los usuarios en cuanto a la manipulación del robot sea tardada y muchas veces incompleta. Muchas de las ventajas que si tiene la interfaz se podrían aprovechar mejor en un ambiente amigable. Es necesario que la nueva interfaz pueda ser utilizada con facilidad inclusive por alguien que no sabe nada de robótica.

- Dentro del enfoque de la celda se busca la optimización del uso del equipo por lo que programar el robot en línea con un controlador convencional como única interfase entorpece la labor de enseñanza, los tiempos de preparación se elevan y los programas no se pueden extraer del robot para su revisión posterior o para su optimización.

- Por estos factores la necesidad de desarrollar una interfaz que facilite estos procesos podría reducir el impacto de estos factores en el proceso.

1.4 OBJETIVO

El objetivo de esta tesis es el diseño de una interfaz para la programación y manipulación del robot Motoman con la opción de trabajar fuera de línea con el fin de facilitar la labor de capacitación en el uso del robot, facilitar la labor del alumno en el aprendizaje, aprovechando las ventajas de programas que se desarrollan en ambiente Windows y la familiaridad que la mayoría de los usuarios tiene con el manejo de una computadora personal.

1.5 JUSTIFICACIÓN

Cada vez es más necesario facilitar el uso de los equipos de trabajo en la industria para los medios de comunicación e informática existentes, tanto para facilitar su operación, como para el manejo de datos y la interacción de estos equipos con otros equipos, todo con el objetivo de optimizar los tiempos de preparación y operación.

La necesidad de crear una interfaz de usuario para el robot Motoman es evidente por todo lo anterior y para poder llevar a cabo una operación didáctica del robot que sea una herramienta práctica y eficaz dentro y fuera de la celda para llevar al usuario a un contacto guiado con el robot y de esta manera un mejor aprovechamiento del tiempo y los recursos

1.6 HIPÓTESIS

La creación de la interfaz del usuario del robot MOTOMAN (Yasnac MRC K3) con un enfoque a la docencia y a la didáctica facilitará la operación y la programación del robot a usuarios que nunca han tenido contacto con este equipo antes, reduciendo así el tiempo de requerido para aprender a usar el equipo en comparación con el que propone el fabricante. También aumentar la capacidad de almacenamiento de información así como explotar el uso de la misma con los medios actuales de manejo de información a los que los usuarios tienen acceso.

1.7 ALCANCE

Esta tesis cubrirá el Diseño de una interfaz con el usuario para el robot Motoman modelo Yasnac MRC K3 con el fin de aplicarla en el desarrollo de reportes prácticos propuestos en el Laboratorio de Sistemas Integrados de Manufactura. Los modos de operación serán en conexión con el robot y sin conexión con el robot para dar alternativa a trabajar fuera del laboratorio. Ambas con las mismas capacidades solo que la última con la posibilidad de que las tareas programadas sean observadas hasta que se lleve acabo la conexión con el robot.

1.8 METODOLOGÍA DE LA INVESTIGACIÓN

1. Investigación bibliográfica sobre conceptos básicos de automatización y de robótica.

2. Investigación bibliográfica sobre la aplicación de interfases en la robótica.

3. Investigación bibliográfica sobre el protocolo de comunicación externa que utiliza el robot Motoman modelo Yasnac MRC K3.

4. Analizar el tipo de protocolo por medio del analizador de señales black box monitor 232 para emular este protocolo con un lenguaje de programación seleccionado.

5. Investigación bibliográfica y aprendizaje del lenguaje seleccionado para la programación de la interfaz.

6. Analizar el protocolo de detección de errores de comunicación y desarrollo del algoritmo en el programa seleccionado.

7. Codificación del programa base para la comunicación inicial con el robot

8. Comprobar por medio de pruebas y resultados que el programa base de comunicación funciona de acuerdo a la teoría de la bibliografía y que el robot responde a los comandos ordenados de la manera esperada.

9. Recopilar las funciones de uso más frecuente en el laboratorio durante las prácticas para que aparezcan en el diseño de la interfaz.

10. Escritura del código completo y el diseño propio de la interfaz del Motoman con el programa seleccionado.

11. Pruebas técnicas de verificación para comprobar que el programa funciona totalmente y que cumple con las expectativas y requisitos proyectados.

12. Realización de pruebas para comprobar la hipótesis planteada anteriormente

13. Presentación de resultados, conclusiones y propuestas proyectos a futuro.

Toda esta información se irá cubriendo con detalle en los siguientes capítulos.

En el capítulo 2 se hablará más a detalle sobre las características técnicas del robot y sobre las formas con las que se cuenta para establecer comunicación. En el capítulo 3 se tratará la estructura del lenguaje de comunicación serial del robot. En el capítulo 4 se describirá la arquitectura de la interfaz, así como los criterios de diseño que fueron seguidos. En el capítulo 5 analizará la aplicación de la interfaz a un caso real y también se analizarán los resultado obtenidos. Por último en el capítulo 6 se muestran las conclusiones y el camino para investigaciones futuras

CAPÍTULO 2 EL ROBOT MOTOMAN

En este capítulo se explicarán brevemente conceptos importantes sobre la robótica. Estos ayudarán a que el lector pueda estar más familiarizado con ellos y lograr una mejor comprensión de la descripción técnica del robot Motoman.

2.1 CONCEPTOS BÁSICOS DE LA ROBÓTICA

La definición de un robot según la RIA(ROBOT INDUSTRIES ASOCIATION): "Un robot es un manipulador multifuncional reprogramable diseñado para mover materias, piezas, herramientas o dispositivos especiales según trayectorias variables, programadas para la realización de tareas diversas"

Otra definición es la proporcionada por la IFR(INDUSTRIAL FEDERATION OF ROBOTICS): "Máquina de manipulación automática reprogramable y multifuncional con tres o más ejes que pueden posicionar y orientar materias, piezas, herramientas o dispositivos especiales para la ejecución de trabajos diversos en las diferentes etapas de producción industrial, ya sea en posición fija o en movimiento".

2.1 CLASIFICACIÓN

Los robots se han clasificado de varias maneras, entre ellas se destacarán las más importantes. Se pueden clasificar según la generación a la que pertenecen, según el tipo de energía, según el método de control, según su nivel de inteligencia, según su nivel de lenguaje de programación, y según la función. [3] A continuación se detallará cada una de estas.

2.2.1 Generación

La generación de un robot se determina por el orden histórico del desarrollo de la robótica. [4]

1era generación: Son robots que repiten la tarea programada secuencialmente, no toman en cuenta las posibles alteraciones de su entorno. [5]
Son llamados robots PLAY-BACK se utilizan en operaciones de recubrimiento por spray o en soldadura por arco. [3]

2da generación: Adquiere información limitada de su entorno por medio de sensores y actúa en consecuencia. [5]

3era generación: Posee la capacidad para planificación automática de tareas. Están equipados por un sistema de Visión y manipulan los objetos con la información que obtienen. [5]

4ta generación: Son robots que pueden automáticamente reprogramar sus acciones sobre la base de los datos obtenidos por sus sensores. [5]

5ta generación: Son los que utilizan las técnicas de inteligencia artificial para hacer sus propias decisiones y resolver problemas. [5]

2.2.2 Clasificación de acuerdo al tipo de energía

Es la manera en que las diferentes partes del robot son energizadas. El controlador y todas sus interfases normalmente son eléctricos y trabajan con corriente alterna.
La base, cuerpo y brazo del robot generalmente utilizan energía hidráulica o eléctrica y el efector final, estos en muchos de los casos utiliza energía neumática.
La capacidad de un robot para mover su estructura y su herramienta de trabajo está definida por medio del sistema que lo energiza. Este determina la velocidad de los movimientos del brazo, la fuerza del robot y su desempeño dinámico, de alguna manera determina también las aplicaciones para las que está hecho el robot. [6]

Los tipos de energía más comunes entre los robots industriales son

1. Hidráulica
2. Eléctrica
3. Neumática

Hidráulica.
Generalmente se asocia con robots grandes y algunas de las ventajas son que le proporciona al robot más velocidad y fuerza. Algunas de las desventajas son el tipo de mantenimiento que se requiere y los goteos que se dan cuando las uniones del robot se deterioran.

Eléctrica.
Normalmente este tipo de energía no proporciona la velocidad y la fuerza que un sistema hidráulico, pero la precisión y la capacidad de repetir son mejores en este tipo de robots.
Tienden a ser más pequeños y el movimiento se hace posible por medio de motores de corriente directa en fases o por servomotores de corriente directa. El costo de un motor eléctrico es proporcional a su tamaño, lo cuál hace que la energía eléctrica se utilice en aplicaciones donde se requieran robots pequeños.

Neumática
Se reserva para robots pequeños que poseen pocos grados de libertad de ciclos rápidos, se usan para operaciones de recoger y colocar, este tipo de energía se puede combinar con las anteriores para dar movilidad a las herramientas del robot.

2.2.3 Clasificación de acuerdo al volumen de trabajo.

Para esta clasificación los tipos son: Cartesianos, cilíndricos, esféricos, SCARA. Las clasificaciones se enfocan en los grados de libertad del brazo principalmente y son los 3 primeros.

Cartesianos
Están formados por 3 correderas perpendiculares X, Y, Z. Por medio del movimiento de estos 3 ejes en forma relativa el robot es capaz de operar describiendo una trayectoria rectangular. [6]
Son utilizados para recoger material y posicionarlo, para soldadura de arco entre otras aplicaciones.

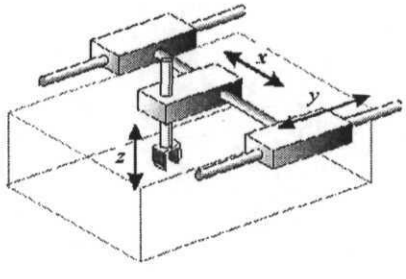


Fig. 2.1 trayectoria Cartesiano

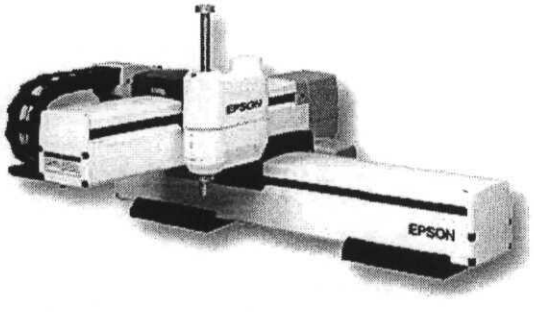


Fig. 2.2 Robot cartesiano modelo EPSON

Cilíndricos

Sus ejes forman un sistema de coordenadas cilíndricas. Usa una columna vertical (Z) en e cuál una columna horizontal (Y) se puede mover perpendicularmente. El brazo del robot está unido al eje horizontal para permitir un movimiento radial relativo a la columna, por medio de la rotación de la columna se logra que el brazo del robot describa esta trayectoria cilíndrica. [6]

Son usados para operaciones de ensamble, manejo de materiales en maquinas-herramientas, máquinas de fundición, soldadura de punto. [7]

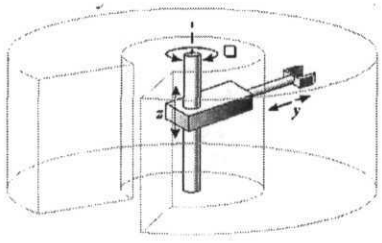


Fig. 2.3 Robot cilíndrico

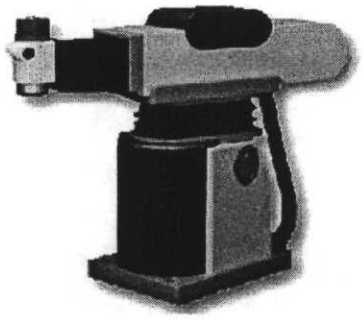


Fig. 2.4 Robot cilíndrico Modelo SEIKO

Polares o esféricos

Usa un brazo telescópico que puede ser alzado o bajado con respecto un pivote horizontal, el pivote está montado en una base rotatoria. Esta seria de uniones le permite al robot mover su brazo en un espacio esférico. Se utilizan en soldadura de gas y de arco y en casi todas las aplicaciones del robot cilíndrico.

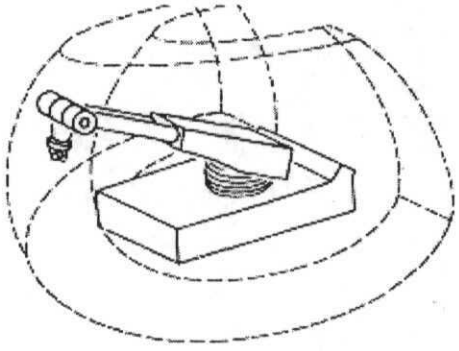


Fig. 2.5 Volumen de trabajo robot esférico

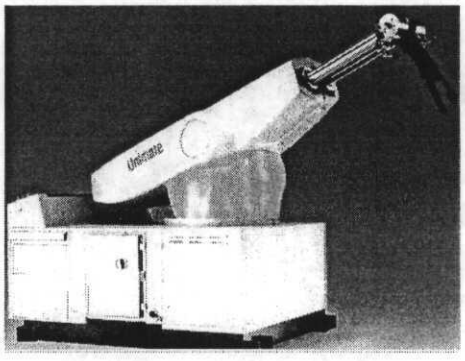


Fig. 2.6 Robot esférico de Unanimate

SCARA.
Las siglas significan Selective Apliance Arm Robot For Asembly. Este brazo puede realizar movimientos horizontales de mayor alcance debido a sus dos articulaciones rotacionales y también puede hacer un movimiento lineal por medio de su tercer articulación

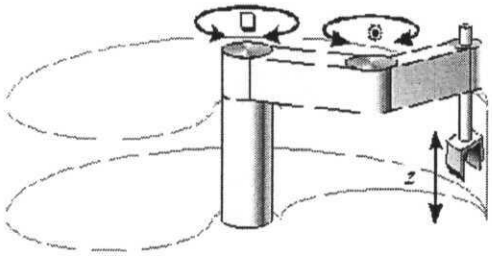


Fig. 2.7 Trayectoria SCARA.

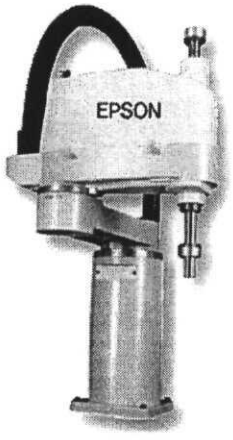


Fig. 2.8 Robot tipo SCARA modelo EPSON

Brazos articulados
Regularmente se le llama brazos articulados o máquinas antropomórficas, con un volumen de trabajo irregular. Tiene 2 variantes: Verticalmente articulado y horizontalmente articulado.
El articulado vertical es llamado también brazo articulado esférico, tiene 3 movimientos básicos que son la base, el hombro y el antebrazo. [5] Este es el más parecido a un brazo humano.

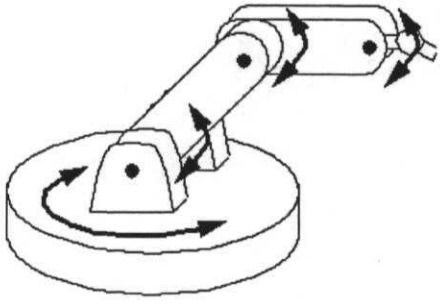


Fig. 2.9 trayectoria articulada

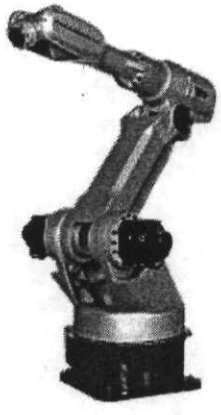


Fig. 2.10 Robot articulado

2.2.4 Calcificación de acuerdo al tipo de Control

También los robots se clasifican por su forma de control que puede ser servo-controlados y no-servo-controlados.

No-servo controlados: Cada articulación tiene un número fijo de posiciones con topes y solo se desplazan para quedar fijas en ellas. Suelen ser neumáticos y bastante precisos. [5]

Servo-controlados: En ellos cada articulación lleva un sensor de posición (linear o angular) que es leído y enviado al sistema de control que genera la potencia para el motor, se puede así parar en cualquier punto deseado. [5]

Servo-controlados punto a punto: Para controlarlos solo se les indican los puntos iniciales y finales de la trayectoria; El ordenador calcula el resto siguiendo algoritmos y normalmente pueden memorizar posiciones. [5]

2.2.5 Clasificación de a cuerdo a su nivel de inteligencia.

Clasificación obtenida de la JIRA (ASOCIACIÓN DE ROBOTS JAPONESA)

Manejo manual: Controlados por una persona.

Secuencia arreglada: Donde no se puede modificar fácilmente la secuencia.

Secuencia variable: Donde un operador puede modificar la secuencia fácilmente.

Robots regeneradores: Donde el operador humano conduce el robot a través de la tarea.

Robots de control numérico: Donde el operador alimenta la programación del movimiento, hasta que se enseñe manualmente la tarea.

Robots inteligentes: los cuales pueden entender e interactuar con cambios en el medio ambiente. [3]

2.2.6 Clasificación de acuerdo a su lenguaje de programación

Sistemas guiados: En el cual el usuario conduce el robot a través de los movimientos a ser realizados.

Sistemas de programación de nivel-robot: En los cuales el usuario escribe un programa de computadora al especificar el movimiento y el sensado.

Sistemas de programación de nivel-tarea: En el cual el usuario especifica la operación por sus acciones sobre los objetos que el robot manipula. [3]

2.2.7 Clasificación de acuerdo a su función su función

De producción: utilizados para la manufactura. Pueden ser a su vez de manipulación, de fabricación o de ensamblado.

De exploración: Utilizados para obtener datos acerca del terreno desconocido, pueden ser de exploración terrestre, minera, oceánica, espacial etc. [5]

De rehabilitación: Utilizados para ayudar a discapacitados. Pueden ser una prolongación de la anatomía, o sustituir completamente la función del órgano perdido. [5]

2.3 COMPONENTES BÁSICOS DE UN ROBOT

Los componentes básicos de un robot son: el manipulador, el controlador y el sistema de energía, estos se describen a continuación.

Manipulador
Este se encarga de realizar todo el trabajo físico del sistema. La base normalmente se coloca fija al piso del área de trabajo, algunas veces la base se puede mover y cuando es así la base está unida a un riel o a una corredera. El brazo del robot tiene al final una muñeca la cuál describe las trayectorias de movimiento del robot. Las articulaciones del robot son las que permiten que haya movimientos relativos entre los componentes del cuerpo, brazo y muñeca, a este conjunto le llama comúnmente manipulador. [8]

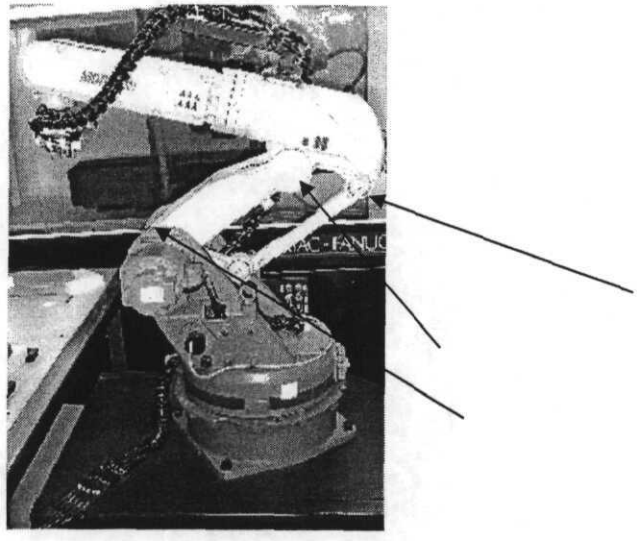


Fig. 2.11 Manipulador con algunos eslabones indicados

El brazo del robot está formado por eslabones y articulaciones. Cuando 2 eslabones se unen, al conjunto se le llama articulación. Las articulaciones rotatativas son las que permiten que un par de eslabones gire. Entre más articulaciones de este tipo tenga un robot se puede decir que tiene más flexibilidad en sus movimientos.

2.3.1 Controlador

Es el cerebro de la operación por medio de él se logra que el manipulador aprenda y ejecute las tareas ordenadas por el usuario. Al controlador entran y salen las señales de otros equipos que interactúan con el manipulador durante su operación, si es que el robot está en esta situación, tal es el caso de un robot que se encuentre en una celda de manufactura por ejemplo. [8]
El aprendizaje de las tareas se lleva a cabo en el controlador por medio de la enseñanza de puntos, son las trayectorias por las que él tiene que pasar el robot para cumplir un determinado trabajo. Estas locaciones programadas se guardan en la memoria y son llamadas más tarde para su operación continua. Es necesario manipular el robot, usando su interfaz, a la posición deseada y utilizarla en los procesos posteriores.

Fig. 2.12 El controlador

2.4 EL ROBOT MOTOMAN

2.4.1 Clasificación del robot Motoman

De acuerdo a las clasificaciones mencionadas el robot Motoman k3 se clasifica según su volumen de trabajo como tipo articulado vertical, de acuerdo a la forma de energía es eléctrico con adaptaciones neumáticas en el efector final, de acuerdo al tipo de control es servo controlado.
En general las aplicaciones de los modelos Motoman son el sellado, el corte con láser, corte plasma, corte a base de Gas, soldadura, entre otros. En la tabla 2.1 se resume este punto.

Generación	Volumen de trabajo	Forma de energía	Control	Inteligencia	Lenguaje
2da	articulado vertical	Eléctrica	Servo	Control numérico	Nivel tarea

Tabla 2.1 clasificación del robot Motoman

2.4.2 Descripción de componentes y partes

El robot Motoman está compuesto por el control manual llamado "Teach Pendant", el controlador, herramientas, el brazo articulado y equipos periféricos. El robot Motoman se encuentra frente a una mesa de trabajo donde forma parte de una celda.

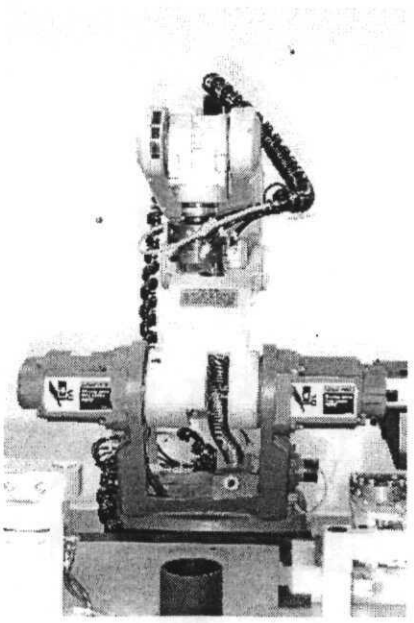


Fig. 2.13 Robot Motoman vista frontal

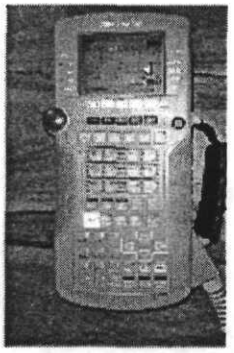


Fig. 2.14 Controlador manual

Fig. 2.15 controlador

La Mesa de Trabajo es el lugar donde se realizan las operaciones de ensamble, cambios de herramienta y carga / descarga de contenedores. Sobre la estructura de la mesa están ubicados los buffer de ensamble 1 y 2, los dosificadores de piezas y contenedores, una prensa, el almacén de herramientas y el Play back box del Controlador MRC. [9]
Sobre la mesa se coloca la referencia de las coordenadas de usuario (USER) que utiliza el Manipulador. Este sistema corresponde a coordenadas cartesianas X, Y, Z.

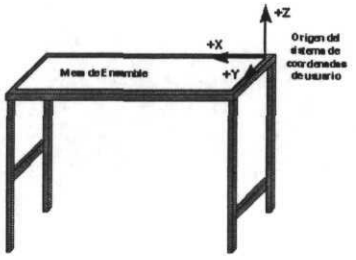


Fig. 2.16 Proyección de la mesa de trabajo

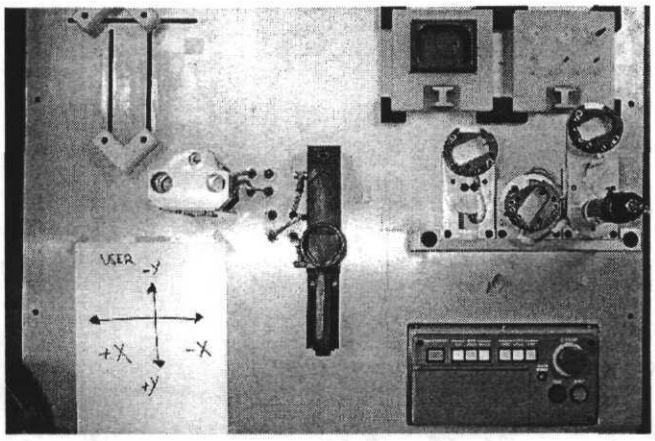


Fig.2.17 Vista superior de la mesa de trabajo

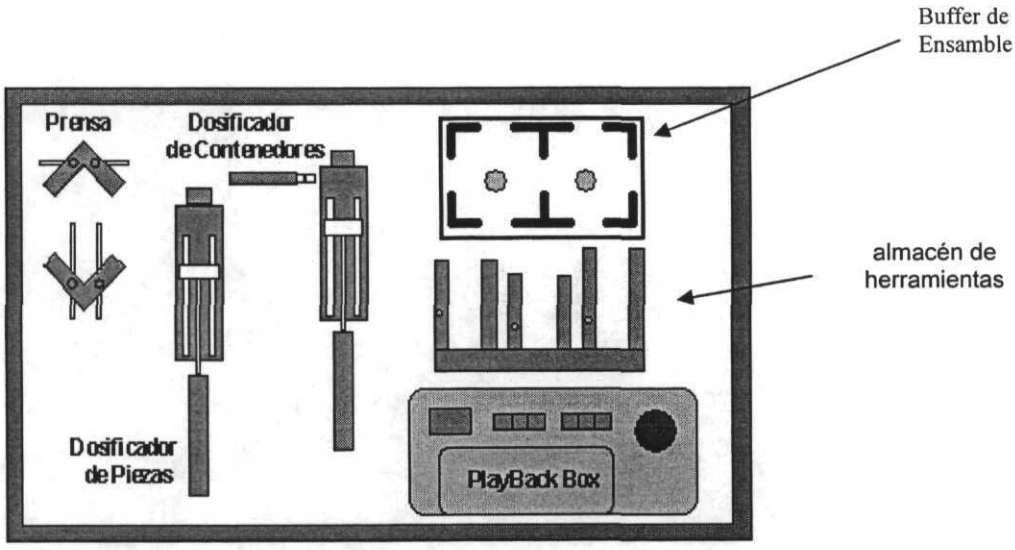


Fig. 2.18 Esquema de la vista superior de la mesa de trabajo

En el buffer de ensamble se coloca la pieza a trabajar junto con la materia prima.
El almacén de herramientas es una estructura de aluminio que se encuentra montada sobre la mesa de ensamble. El almacén de herramientas contiene las 3 herramientas que el robot Motoman utiliza, estas son el gripper, la ventosa y el desarmador. Todos son activados por energía neumática.
Al hacer los cambios de herramienta por software, el manipulador lleva la herramienta que va a desmontar a su puesto correspondiente y toma la nueva herramienta de otro puesto. Los puestos donde van colocadas las herramientas no son iguales, por lo que cada herramienta tiene una posición única en el almacén. Cada uno de los tres puestos tiene instalado un censor inductivo que indica al sistema de control la presencia de la respectiva herramienta.

La prensa se utiliza para sujetar y referenciar a la pieza que sirve de base para una tarea de ensamble. Está conformada principalmente por dos elementos: una mordaza móvil y otra fija. Esta última da la referencia final de la base y se puede ajustar a lo largo de un riel para posicionar adecuadamente a la pieza. Existen dos programas en el controlador del robot que sirven para abrir y cerrar la prensa. Las mordazas de la prensa están configuradas para sujetar piezas rectangulares. [9]
Para la enseñanza de puntos solo se cuenta con el control manual en el cuál se despliega la única interfaz existente con el robot Motoman.

2.4.3 El manipulador

Como ya se mencionó el robot tiene 6 ejes los cuales se mueven en combinaciones de movimiento vertical en su mayoría. La carga mínima con la que el robot puede trabajar es de 3Kg. y esto incluye el peso de la herramienta. Como se puede observar el robot se asemeja a un brazo humano y simula su movimiento. Lo que parecería los hombros son los servomotores que dan el movimiento principal.

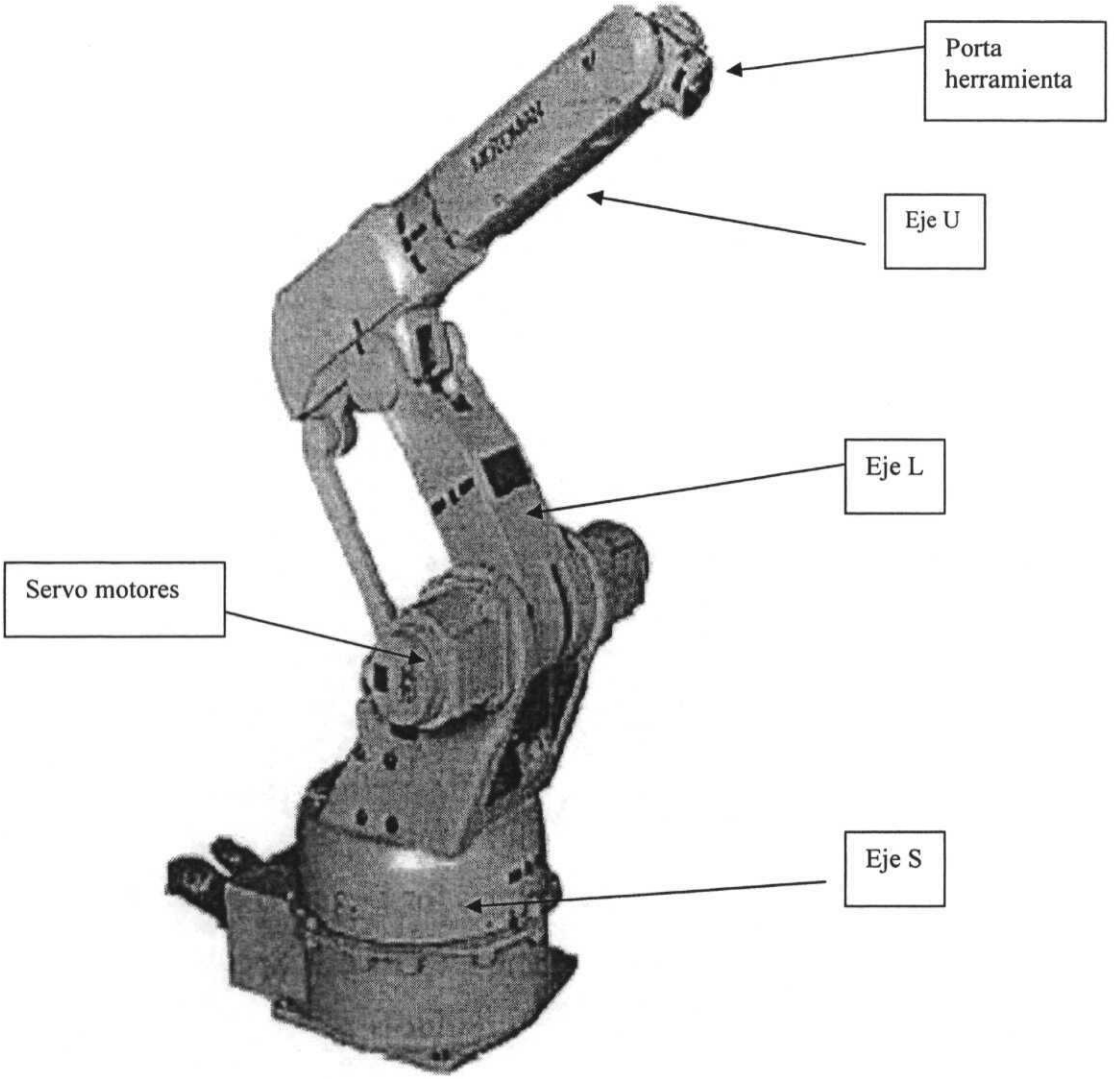


FIG 2.19 Manipulador

El manipulador como ya se mencionó también cuenta con 6 grados de libertad los cuales están nombrados con las siglas SLURBT que tienen su equivalente en X, Y, Z y RX, RY, RZ observar en la siguiente figura:

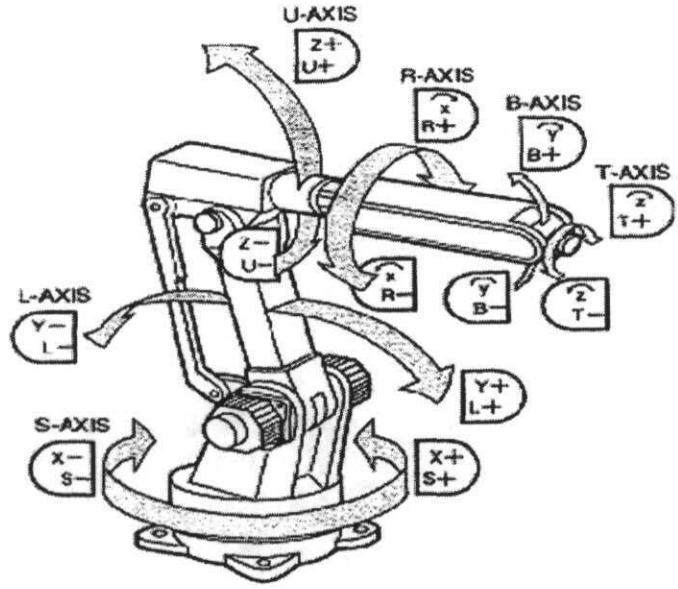


Fig. 2.20 Grados de libertad del manipulador

El eje que corresponde al movimiento principal del tronco es el eje L, el cuál también es llamado Y. El eje que corresponde al giro del robot corresponde al S (X) el movimiento principal del brazo está en el eje U (Z) el giro del brazo está en el eje R (Z. La flexibilidad de la Muñeca está en el eje B (Y) y el giro de la muñeca está en el eje T.

Los movimientos del robot se pueden hacer de 4 maneras:

JOINT: En esta forma de movimiento los ejes se mueven por separado de acuerdo a lo indicado en la figura 2.20

WORLD: Se toma en cuenta un punto de referencias fijo de coordenadas X, Y, Z las cuales estarán fijas en la base del robot. Este tipo de movimiento la herramienta del robot sigue la trayectoria estipulada por las posiciones de los ejes como se muestra en la figura 2.21.

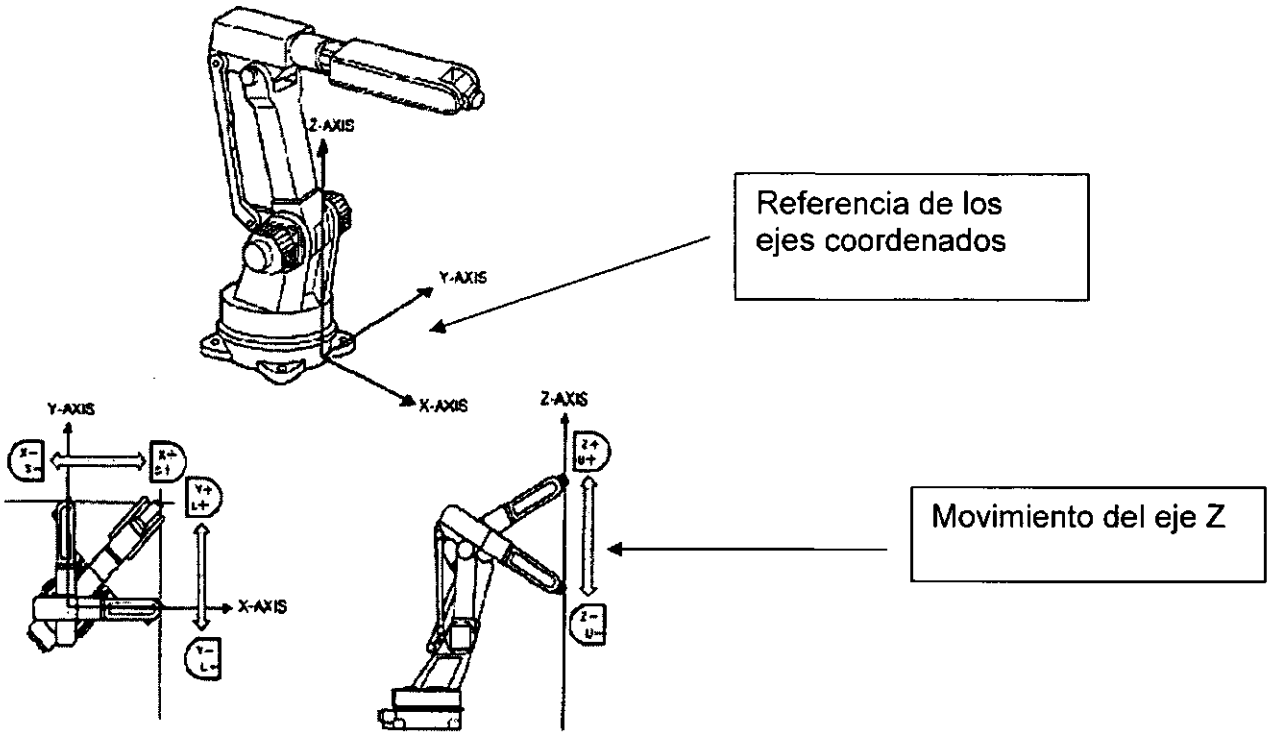


Fig 2.21 Movimiento World

Todos los ejes se combinan para que la herramienta siga las trayectorias mostradas.
Este tipo de movimiento es de gran ventaja cuando se desea mantener en todo momento una posición de la herramienta con respecto a una superficie de trabajo.
Cuando en esta forma de movimiento se activan los ejes T la herramienta gira con respecto al eje y en este momento los ejes T se llaman RX, RY y RZ respectivamente.

TOOL
El modo tool tiene el mismo propósito que World, combinar los ejes del robot para lograr una estabilidad en la herramienta de trabajo. En este caso los ejes coordenados se trasladan a la base de la muñeca del robot. Los ejes coordenados estarán de acuerdo a la posición de la herramienta. Esta forma de trabajo hace posibles los movimientos donde se requiere estar en constante observación de la herramienta y no buscar el sistema de coordenadas en otra referencia.

USER
El modo USER es una función de Motoman en la que el usuario declara donde quiere que esté la referencia de los ejes coordenados y está se puede ajustar a la superficie de trabajo.
Para el caso del robot en la celda de manufactura las coordenadas de usuario están colocadas como se indica en la figura 2.16 De manera que el movimiento del robot es de acuerdo a una mesa de trabajo.

2.4.4 Controlador

El controlador es el cerebro del robot, por medio de este se procesan todas las señales que salen o que entran al robot. El controlador se puede observar en la figura 2.22

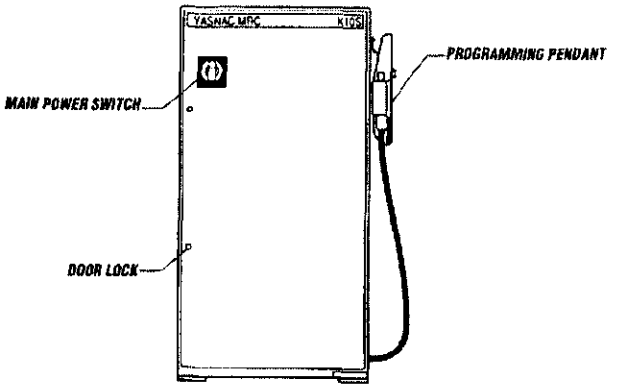


Fig. 2.22 Controlador Motoman

Modo Teach
Es el comando para que el robot esté en la forma de enseñanza. Estando de esta manera se activa el manejo manual remoto que dará movimiento al robot por medio de la pulsación de los botones de movimiento. Estando en esta forma de operación se programa el robot y se le enseñan posiciones que ejecutará en el futuro.

Modo play
El robot está en condiciones de trabajo automático ejecuta programas y está ejecución se puede hacer 1 sola vez (1CYCLE) o bien, indefinidamente (Auto) o por pasos (STEP).

Modo remote
Es la manera en la que el se pone el robot en forma de operación remota, en esta manera el robot está habilitado para mandar y recibir información de dispositivos externos como computadoras y discos flexibles. Remote se puede usar de manera combinada con play o con teach.

Botón E.STOP
Es un paro de emergencia para suspender totalmente la actividad del robot, esto incluye la desactivación de los servomotores.

Botón hold
Coloca al robot en pausa y no apaga los servomotores, la actividad se puede continuar en donde fue detenida.

En capítulos posteriores se profundizará más en estas funciones y en su aplicación con más detalle.

2.4.5 Play back box

En la siguiente figura se muestra el play back box y cada una de sus funciones.

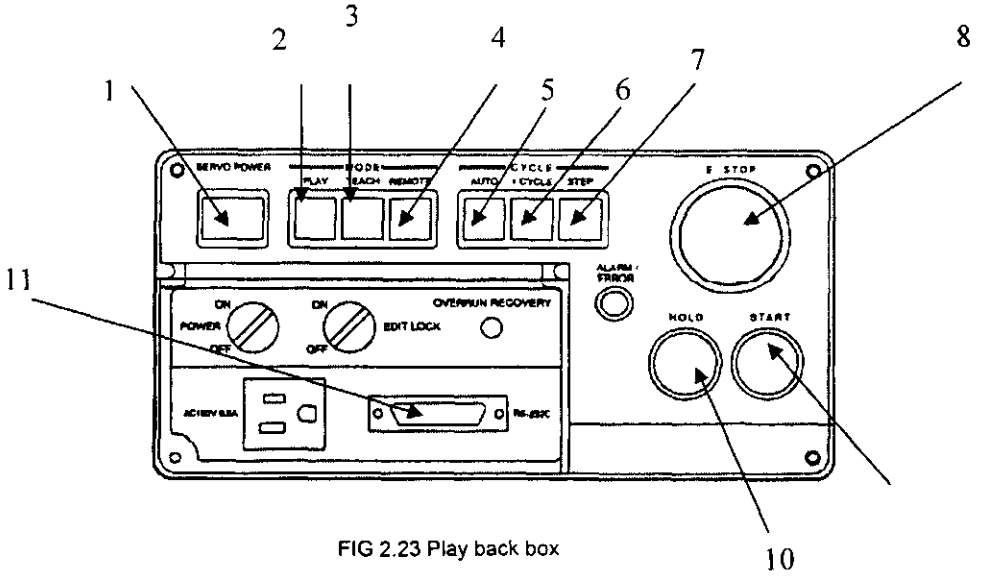


FIG 2.23 Play back box

El play back box tiene comandos básicos de ejecución:
1: Es el encendido de los servomotores
2,3,4: Son el modo de operación del robot Play, Teach y Remote respectivamente.
5,6,7: Ayudan a fijar la manera en que se ejecutará el programa.
8: Es uno de los paros de emergencia que suspende la actividad del robot y desactiva los servomotores.
9: Comienza la ejecución de una rutina cuando se está en modo play.
10: Detiene la subrutina momentáneamente.
11: Conexión serial para enviar comandos al robot cuando se está en el modo remote.
El robot se puede operar de 3 maneras con los botones 2,3 y4 que son play, teach y remote.

2.4.6 EL Programador manual

Es el dispositivo que sirve como interfaz entre el usuario y el resto de los componentes del sistema del robot Motoman. En el programador manual el usuario le indica al manipulador las trayectorias que debe seguir al ejecutar un programa. La inserción y edición de las instrucciones del programa solo se pueden efectuar en este dispositivo.
El desplegado de pantallas del robot es por menús a los cuales el usuario va entrando según las opciones que vaya seleccionado. Esta es la única forma de comunicación efectiva con el robot, aquí se controla también la memoria del robot y se guardan todas las rutinas del robot. A continuación se describen sus principales componentes:

1. Pantalla digital
2. Tipo de movimiento en el modo enseñanza
3. Velocidad del modo enseñanza
4. Teclas de selección de 2 y 3 y habilitador del programador manual.
5. Paro de emergencia
6. Controles de los ejes principales del manipulador
7. Controles de la muñeca del manipulador
8. Controles de edición y de ejecución a prueba.
9. Teclado alfanumérico.
10. Flechas de navegación.

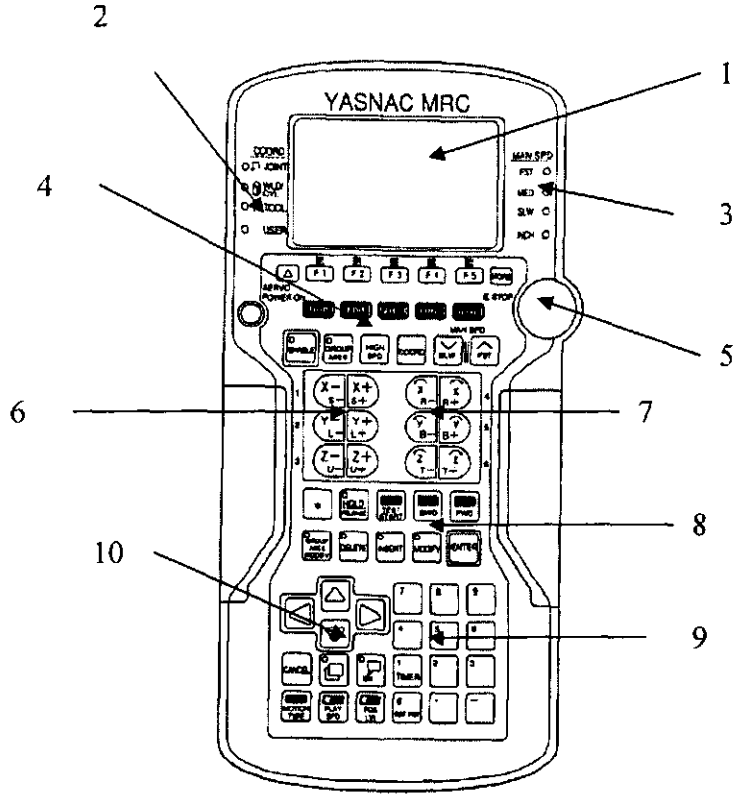


FIG 2.24 Programador

2.4.7 Especificaciones técnicas

Carga de trabajo: Es el peso del objeto más pesado que el robot puede cargar.

Preescisión de repetitividad: Se refiere a la desviación que presentará al repetir un movimiento a una posición previamente enseñada.

Rango de movimiento: se refiere a los grados máximos que puede girarse una articulación y la velocidad a la que pueden hacerlo.

Vibración: Si el robot está colocado próximo a una fuente de vibración, se debe de cuidar la intensidad de manera que no dañe los componentes.

Enseguida en la tabla 2.2 se muestran las especificaciones del robot Motoman K3

MODELO	Manipulador: YR-K3-C000		
clasificación	Articulado verticalmente		
Grados de libertad	6		
Carga de trabajo	3 KG		
Precisión de repetitividad	± 0.1mm		
Rango de movimiento	**Grados**	**Velocidad**	**MOMENTO**
S (Giro)	340°	2.62 Rad/Seg	
L (Brazo inferior)	240°	3.49 Rad/Seg	
U (Brazo superior)	260°	3.49 Rad/Seg	
R (Giro de brazo superior)	360°	4.89 Rad/Seg	5.9 N-M
B (Flexión de la muñeca)	270°	4.33 Rad/Seg	4.9 N-M
T (Giro de la muñeca)	400°	7.33 Rad/Seg	2.9 N-M
Masa	53 Kg		
Temperatura de trabajo	0-45°		
Humedad relativa permitida	20-80% RH		
Vibración	Menos de 0.5 G		
Ambiente permitido	No acercarse a lugares: Corrosivos Explosivos Ruidosos eléctricamente		
Potencia	Picos de 2.5 K-VA, promedio 1.5 K-Va		

Tabla 2.2 Especificaciones técnicas

2.5 EL MODO DE PROGRAMACIÓN (TEACH)

La programación del robot se lleva a cabo por medio del controlador, el robot se debe de colocar al modo "Teach" por medio del play back box. Después con el programador manual se lleva el efector final del robot hasta las posiciones a la que se quiere que el robot haga referencia en el futuro. En ese punto el robot por medio del programa que se está elaborando pedirá la velocidad del punto y la forma en que se desea ejecutar.

Se puede decir que el robot Motoman tiene 2 caras, una es la de programación y otra es la de enseñanza. Para llegar a un punto determinado puede seguirse un método y una forma de movimiento en especial como (WORLD, JOINT o TOOL) pero al robot se le puede pedir que llegue al punto de otra manera cuando opere de forma automática.
Las formas de movimientos de enseñanza son como ya se mencionó WORLD, TOOL, USER las formas de movimiento de ejecución son:

JOINT: Es exactamente igual que el Joint de enseñanza mencionado en la página 19, al utilizar este tipo de movimiento el robot solo moverá uno de sus ejes para llegar al punto deseado. El comando para mover el robot en forma JOINT es MOVJ y la velocidad de movimiento se declara en porcentaje de la velocidad nominal del robot.

LINEAL: Movimiento que hace que el robot llegue a los puntos por medio de interpolaciones lineales y lo hace siguiendo la trayectoria más corta a ese punto. Este movimiento solo se usa cuando se trabaja

en una mesa de trabajo con movimientos de preescisión. El comando para mover el robot en forma LINEAL es MOVL al cuál se le da de alta la velocidad en mm/seg.
CIRCLE: Movimiento que requiere darle al robot 4 coordenadas para que elabore una interpolación circular.
El comando de movimiento es MOVC.

SPLINE: Se requiere dar al robot 3 coordenadas para que elabore un movimiento parabólico.

Cada una de estas formas de movimiento tiene su tabla de selección de velocidades, las velocidades lineales se expresan en mm/seg y las velocidades joint se expresan en % de velocidad del eje que se está moviendo.

Estas son las velocidades que maneja el robot en el modo lineal y joint:

Lineal (MM/Seg)	Joint % de Vel. Nom.
11	.78
23	1.56
46	3.12
93	6.25
187	12.50
375	25
750	50
1500	100

Tabla 2.3 Velocidades seleccionables para rutinas lineal y joint

Dentro de un programa se pueden mandar llamar otros programas, lo que sucederá es que del programa principal se pasará a ejecutarlos y después saldrá al programa principal una línea después de donde se introdujo a la subrutina. La siguiente tabla muestra los programas que se encuentran disponibles en la memoria de Motoman. La primera tabla presenta comandos para las herramientas que como se mencionó en la página 19 son el gripper, la ventosa y el desarmador.

Nombre	**Operaciones**
RBHERRON	Activar la herramienta actual: • El gripper se cierra. • La ventosa succiona. • El desatornillador recibe alimentación de presión de aire.
RBHERROF	Desactivar la herramienta actual: • El gripper se abre. • La ventosa deja de succionar. • El desatornillador no recibe alimentación de presión de aire.
FIN_OPER	Indica que una operación del Manipulador finalizó.
HOME	Llevar al manipulador a la posición de referencia inicial. Inicializa también el código de la herramienta actual. Se debe correr manualmente con el PROGRAMMING PENDANT del Manipulador cada vez que sé inicializa el sistema de control.

Nombre	Operaciones
TGRIPPER	Tomar el gripper de su puesto de almacén.
DGRIPPER	Deja el gripper en su puesto de almacén.
TDESTORN	Tomar el destornillador de su puesto de almacén.
DDESTORN	Deja el destornillador en su puesto de almacén.
TVENTOSA	Tomar la ventosa de su puesto de almacén.
DVENTOSA	Deja la ventosa en su puesto de almacén.

Tabla 2.4 Subrutinas actuales del robot [20]b

Las subrutinas mostradas en las tablas fueron programas con puntos enseñados previamente, se usan con mucha frecuencia, por lo que es de gran ayuda tener estas subrutinas disponibles en la memoria.

Estas mismas posiciones también se pueden guardar como variables, definidas como
Llamadas variables tipo "P" también llamadas variables de posición. Los programas se pueden correr completos, por pasos. Dentro del menú de programación es permitida la copia de líneas del programa para pegarlas en otro punto del programa. El robot también utiliza variables donde almacena información digital en forma binaria como la de la disponibilidad de una herramienta en el almacén.

Un ejemplo típico de un programa de Motoman se muestra a continuación:

```
000 NOP
001 '#########################
002 'ROBOT CARGA TAPA EN VMC
003 '#########################
004 '
005 '# INICIALIZACION DE PROGRAMA #
006 '## HERRAMIENTA INICIAL
007 CALL JOB: TVENTOSA
008 '
009 '#########################
010 'ROBOT A □OSICIÓN DE HOME
011 MOVJ VJ=50.00
012 'ROBOT CARGA TAPA
013 JUMP *BUFF_GEN IF BO20=3
014 JUMP *BUFF_VMC IFBO20=4
015 '#########################
016 'ROBOT TOMA TAPA DE BUFF_GEN
017 *BUFF_GEN
018 MOVJ VJ=50.00
```

La línea 7 es un ejemplo del llamado de una subrutina que se encarga de tomar la herramienta de succión del almacén de herramientas. La línea 11 se trata de una posición asignada en forma joint con una velocidad del 50%

2.6 MODO DE EJECUCIÓN (PLAY)

Cuando el robot está en modo de ejecución la única manera de interactuar con él es por medio de los paros de emergencia y del botón HOLD del playbackbox. Antes de fijar el robot a la forma PLAY se selecciona el programa que será ejecutado automáticamente. Después de esto el robot comienza la ejecución de forma continua pasando por los puntos enseñados de la manera en que se le haya especificado y ejecuta las subrutinas llamadas por el programa principal.
Cuando se está en el modo TEACH hay una manera de correr el programa estando a prueba de fallos, es decir, se hace un simulacro de lo que va a pasar a velocidades disminuidas y con la posibilidad de detenerlo sin poner al robot en estado de emergencia.
En modo play se realiza la operación automática del programa.

2.7 MODO REMOTE

Es la manera en la que se debe de fijar el playbackbox para poder intercambiar información, se le llama también función de transmisión de rutinas con un dispositivo externo como podría ser un lector de discos flexibles o una computadora personal. Este modo se puede usar en combinación con los anteriores no solo para mandar archivos o rutinas, si no también para controlar el robot por comandos. Los comandos a los que responde el robot se interpretan por medio del código ASCII. El protocolo compilador utilizado para la comunicación es el BSC del cuál se hablará a detalle más adelante.

La función de transmisión de información está dividida en tres partes:

1. DCI (Comunicación de información por instrucciones)
2. Función de Aislamiento.
3. Función de control anfitrión.

2.8 FUNCIÓN DCI

La función DCI ejecuta instrucciones descritas en una rutina que sirven para llevar a cabo transmisión de datos con una computadora anfitriona.

La función DCI cubre dos tareas que son la trasmisión de trabajos y la trasmisión de variables

Transmisión de trabajos: Carga, descarga, almacenamiento y eliminación de archivos
Transmisión de variables: Carga y almacenamiento.

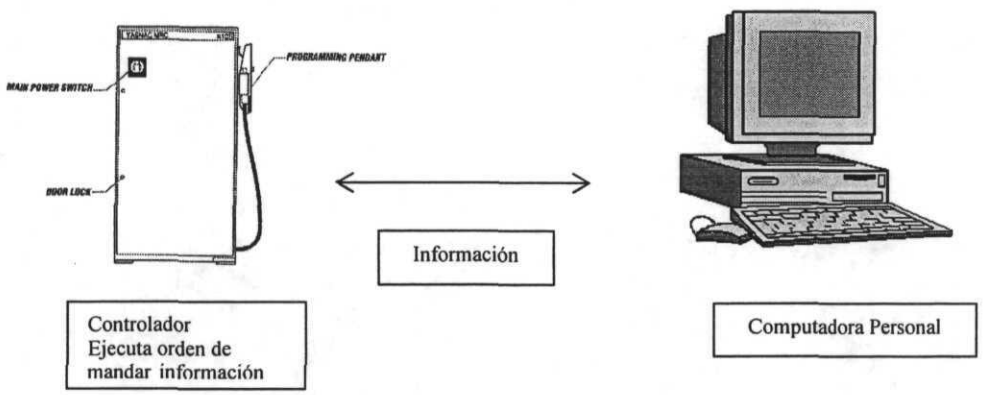


Fig. 2.25 Comunicación DCI

2.8.1 Función de trabajo aislado.

Esta es para transmitir información a un dispositivo externo por medio de la operación del programador manual. Las funciones a las que da soporte esta manera de funcionamiento son:

Transmisión de trabajos: Carga, descarga, almacenamiento, eliminación de archivos y verificación

Transmisión de datos de condición: Datos de herramientas, datos de coordenadas.

Transmisión de información del sistema: Datos del sistema, historia de alarmas.

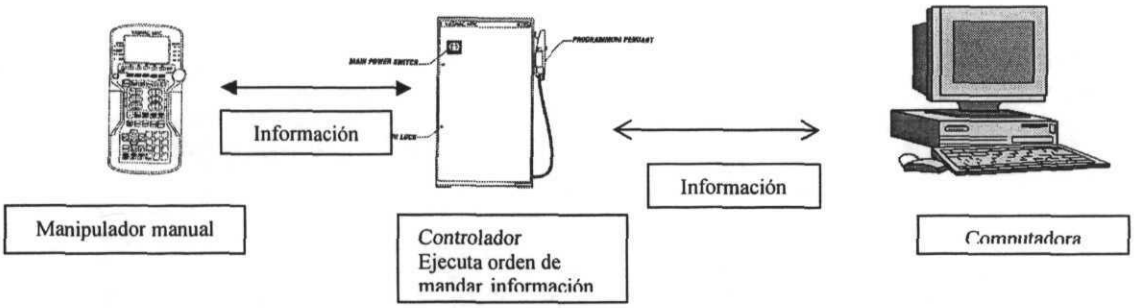


Fig. 2.26 Operaciones remotas por medio de manipulador

Las funciones a las que da soporte esta manera de funcionamiento son:

Transmisión de trabajos: Carga, descarga, almacenamiento, eliminación de archivos y verificación

Transmisión de datos de condición: Datos de herramientas, datos de coordenadas.

Transmisión de información del sistema: Datos del sistema, historia de alarmas.

2.8.2 Función de control modo anfitrión

Esta forma de operación es para guardar, archivar trabajos, leer el estado del robot y controlar el sistema por medio del envío de comandos

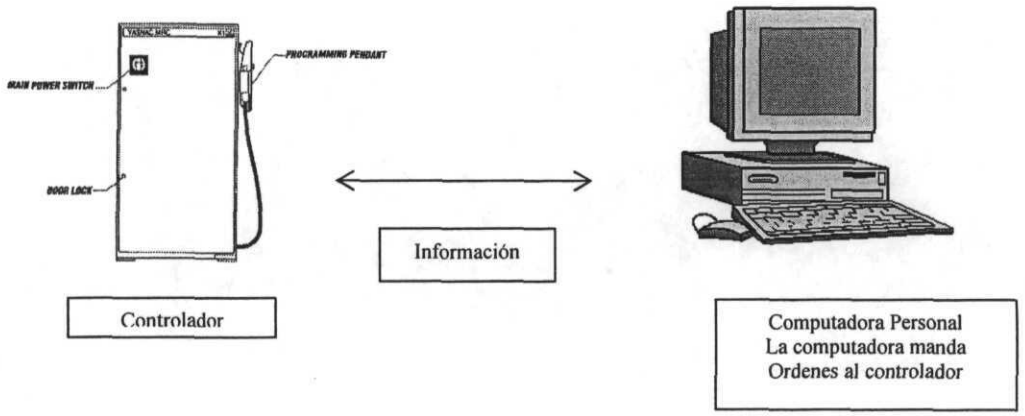


Fig.2.27 Flujo de información en el modo anfitrión

Las funciones a las que da soporte esta manera de funcionamiento son:

Transmisión de información: Trabajos, Datos de condición, Información del sistema, control del estatus y lectura de estatus.

Lectura de estatus: Códigos de alarma, posición actual en sistema cartesiano, posición actual en sistema joint, lectura del modo de operación, modo, ciclo, operación, alarma de error de estatus de los servomotores, lectura del trabajo actual, del número de línea y del número de posición y lectura de variables.

Control de estatus:
Comienzo de rutina, pausa de rutina, reinicio del robot o paro de actividades, eliminación de rutinas, fijar el nombre de un programa de la línea y el número de paso, Selección de modo y ciclo de operación, encendido de servomotores, desplegado de mensajes, movimiento del robot a coordenadas cartesianas o Joint, escritura de variables.
Para llevar a cabo todas las operaciones del modo remote se utiliza un puerto serial tipo (RS-232).

Hasta ahora se ha tratado sobre teoría básica en el tema de la robótica, después se trató sobre las características principales del robot Motoman. En el siguiente capítulo se presentará el tema de la comunicación entre el robot Motoman y dispositivos externos, también se hablará sobre el manejo del algoritmo que utiliza la interfaz para comunicarse con el Motoman.

CAPÍTULO 3 PRINCIPIOS DE COMUNICACIÓN DE MOTOMAN

En este capítulo se hablará de los principios de comunicación del robot. Con esto se pretende llevar al lector a conocer el protocolo de comunicación que se usó para construir la interfaz. Se explicarán los algoritmos usados por el robot para el envío y recepción de comandos y de información de variables.

3.1 ESPECIFICACIONES DE TRANSMISIÓN DEL ROBOT MOTOMAN K3

- **Conexión**: Serial tipo RS-232C (Ubicada en el play back box)
- **Velocidad de transmisión**: 9600 BPS
- **Modo de transmisión**: Sistema semiduplicado punto a punto
- **Sistema de sincronización**: Método asíncrono 1 bit de paro
- **Protocolo**: Tipo BSC
- **Código de transmisión**: ASCII, Shift JIS, bloques de 8 bits, par, sin transparencia.
- **Revisión de errores:** BCC
- **Forma de respuesta:** ACK

3.2 PROTOCOLO DE COMUNICACIÓN

El protocolo de comunicación serial del robot Motoman es el BSC estas siglas significan "Bynary Synchronous" lo cual significa literalmente binario sincronizado.
Este tipo de protocolo de comunicación es del tipo por bloque, con corrección de errores y fue introducido en 1964 por IBM junto con un producto llamado 270X transmisor de control. [10]

En los protocolos de orientación tipo carácter, cada carácter tiene un significado, cuando se recibe puede ser de dos tipos, un byte de datos o de control, los caracteres se transmiten por medio del uso del código de ASCII con valores entre 0 y 1F. [11]

Este tipo de comunicación comienza como con una especie de saludo entre el emisor y el receptor y se ejemplifica de la siguiente manera:

EMISOR (PC)	RECEPTOR (ROBOT)
Avisa que tiene un mensaje	
	Confirma que está listo para recepción
Envía carácter de presentación	
	Error el mensaje no se recibió
Reenvío de carácter de información	
	Caracter recibido
Se envía carácter de despedida	OK

Tabla 3.1 Ejemplo de dialogo PC-Robot

Como se observa en la tabla solo una de las partes puede enviar mensajes a la vez y por esto se le llama HALF DUPLEX, los mensajes largos se rompen en paquetes más cortos, cada paquete es recibido antes de esperar el siguiente. Si el paquete no es recibido el emisor lo intentará de nuevo, si el paquete es recibido pero contiene errores, el receptor mandará una aceptación negativa (NEGATIVE AKNOWLEDGEMENT. [11]

En este lenguaje se usan bites y estos contienen datos acordes con ASCII, los bites de control son los que determinan el comportamiento del enlace y tienen diversos propósitos.

Algunos de los ejemplos:

STX (START TEXT): Inicio de texto, se transmite antes de los primeros datos, significa que un bloque de datos viene en camino.

ETX(END OF TEXT): fin del texto, termina el bloque que comenzó con SOH o STX y termina una secuencia de bloques, el receptor enviará ACK o NACK

EOT(ENDOF TRANSMISIÓN): Fin de transmisión, concluye toda transmisión.

ACK (AKNOWLEDGEMENT): Reconocimiento positivo, este se envía por el receptor cuando sucede una recepción exitosa del bloque previo.

NAK (NEGATIVE AKNWLEDGMENT): Reconocimiento negativo y se envía para decir que hubo una recepción completa pero errónea.

SOH (START OF HEADER): Comienzo de un encabezado, se envía antes de los caracteres del encabezado.

ETB: Final de bloque de transmisión, indica el final de la transmisión de un bloque de texto que comenzó con un STX o SOH. El receptor enviará ACK o NACK dependiendo de la correcta recepción de los mensajes de bloque.

Cada bloque de información puede tener hasta 3 partes:

- Un encabezado
- Un Texto
- Un Trailer o nombre común

Los caracteres de control utilizados para identificar estar partes son:

- SOH: A continuación un encabezado
- STX: A continuación texto.
- ETX: Ha finalizado el texto.

El "Trailer" de cada bloque consiste es un bloque de verificación llamado BCC (BLOCK CHECK CARÁCTER) Cuando se genera una mensaje, ambos el emisor y el receptor generan este carácter, el emisor lo genera y lo manda, el receptor lo calcula con la información que se lo mandó el emisor y el receptor lo compara contra el del emisor, es como una palabra clave que corrobora el mensaje anterior. Mas adelante se explicará más de esto. [11]
Esta comparación es la que determinará si el receptor recibió el mensaje correcto, es decir, si la respuesta va a ser ACK o NACK, ambos casos indican que el mensaje se recibió completo pero no necesariamente correcto.

Estos caracteres tienen un equivalente numérico que se puede encontrar en tablas de códigos ASCII en los anexos. El equivalente numérico es de gran ayuda en la programación del software.

3.2.1 Detección de errores

La detección de errores es un método utilizado en los protocolos de comunicación serial. Normalmente es un carácter redundante. Pero algunas coincidencias pueden hacer que bloques de revisión sean aceptados aunque sean incorrectos. Hay diferentes tipos de métodos de detección de errores. [12]

Método de Paridad
También llamado revisión de redundancia vertical. Es un método antiguo de detección de errores. Los caracteres son codificados de manera que se les agrega un bit al final de cada caracter. El resultante es un número par o un número impar, siempre es 1 o 0.. Fue muy utilizado pero se ha dejado de usar en las comunicaciones actuales. [12]

En el siguiente ejemplo se colocan 7 números binarios para ser sumados, estos números corresponden a caracteres. Al final de la suma queda un número del cual solo se usa el primer número de derecha a izquierda. A este número se le llama el LRC y es corroborado para probar el error.

Ejemplo:

1 1 1 1 1 1 0

0 0 0 0 0 0 0

1 0 1 0 1 1 0

0 0 0 1 0 1 0

0 0 0 0 1 0 1

0 0 1 0 0 0 1

0 1 0 1 0 1 1

0 0 1 1 1 0 1 LRC

Revisión de bloque (BLOCK CHECK)
Es una suma de todos los caracteres de un bloque de información, este carácter es el llamado BLOCK CHECK SUM CARÁCTER. Es un carácter de la misma longitud que los caracteres de la suma. Por coincidencias matemáticas hay posibilidades de encontrar diferentes bloques que generen el mismo BCC. Es fácil de calcular y se usa generalmente en aplicaciones donde el cálculo de este carácter será hecho por un software. [12] Esta forma de revisión de errores es la que se aplica en esta tesis, por lo que posteriormente se profundizará más en el tema.
El ejemplo es sencillo se aplica sumando los equivalentes ASCII del mensaje y luego se divide entre el número 255 ya que este es el número de caracteres existentes en el código ASCII.

3.3 FORMATO DE TRANSMISIÓN REMOTA MOTOMAN

En lo siguientes esquemas se presentarán la secuencia de aparición de los caracteres. Estos se presentan de la manera en que pudiera aparecer según fuera el caso. EL robot Motoman aplica el protocolo BSC de las siguientes manera:

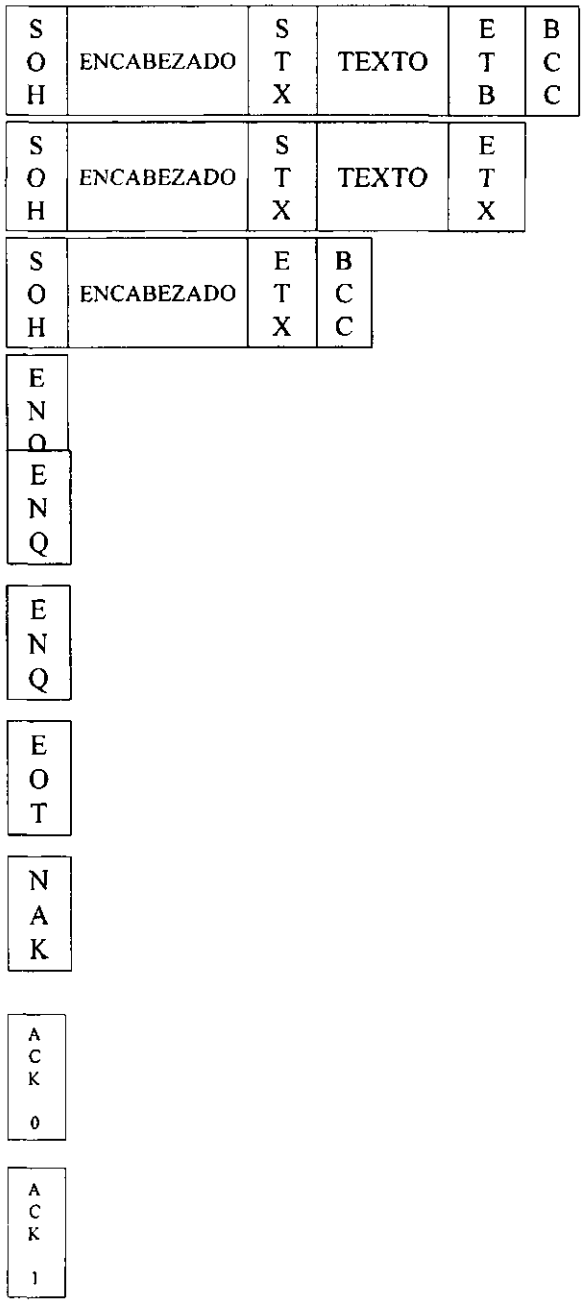


Fig. 3.1 Formatos comunes de comunicación [10]

Encabezados
Los encabezados son números que se colocan según como se indica en la Fig. 3.1 para que el robot identifique la operación que se va a realizar. Son de tamaño fijo, 6 caracteres en números binarios, a continuación presentamos los más utilizados:

Encabezado	Significado
01,000	Comando a computadora externa
02,0001	Datos de un trabajo único
02,002	Datos de un trabajo relacionado
02,200	Datos de herramienta
02,300	Petición de datos de herramienta
03,001	Variables tipo byte
03,0251	Petición de variables tipo byte
90,000	Comando o respuesta (normal / error)
90,001	Comando o respuesta (Datos)

Tabla 3.2 Encabezados comunes del protocolo[10]

Texto
Es un máximo de 256 caracteres y son generalmente comando preestablecidos que se verán más adelante con más detalle.

BCC (BLOCK CHECK SUM)
Se suman todos los caracteres desde el SOH hasta el STX o al ETB o ETX, un ejemplo propuesto de este proceso es el siguiente donde se sumará desde el texto hasta BCC.

Supongamos que se tienen letras y que cada letra equivale a un número binario de 8 caracteres
Como los que se muestran en la siguiente figura que es la l 3er secuencia de la Fig. 3.1.
Como lo que varía en la secuencia es el texto, esto es lo que hará que la suma desde la primer letra del texto hasta ETX se modifique. El valor de ETX es siempre el mismo. Entonces se tiene que cada X corresponde a un numero binario. Se procede a sumar los números.

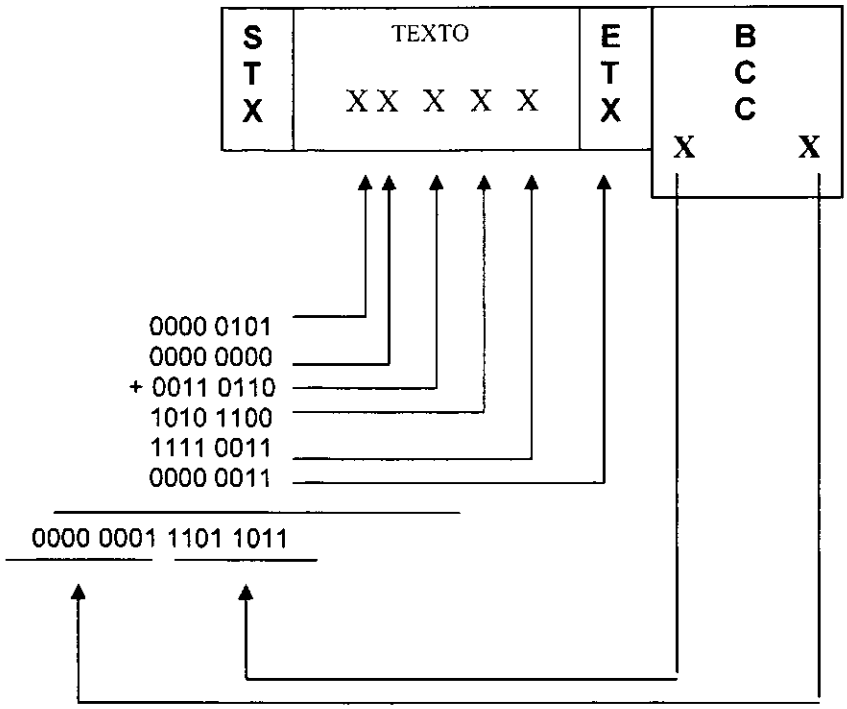


Fig. 3.2 Formatos comunes de comunicación [10]

Al final se generan 2 número de 8 bits que son los que forman el BCC.
El inicio del cálculo se hace cuando SOH o STX aparecen como bloques de comienzo en una secuencia. STX se incluye en la suma cuando SOH lo precede. El fin del cálculo se lleva a cabo cuando ETB o ETX se usan como el fin de la secuencia del bloque, Se incluyen ETB o ETX. En la práctica para lograr que el software calculara más rápido y con menos líneas de programación los 2 caracteres del BCC se buscó la manera de trabajar con decimales en lugar de usar números binarios. De acuerdo al método de detección de errores del Checksum se sacaron los equivalentes decimales de cada carácter y se sumaron, se calculó el residuo del total entre 256 y eso corresponde al primer carácter del BCC, el segundo carácter se calculó con el cociente de la misma división. A esta conclusión se llego por medio de cálculos y comparaciones de los resultados que se debían obtener en binario al enviar ciertos textos.

ENQ
Por medio de este se hace el contacto de una base remota al robot o viceversa, en cualquiera de los casos la respuesta debe de ser ACK0. Por parte del controlador del robot existen 2 cronómetros con los que el robot da tiempo a que se cumpla lo esperado. El cronómetro B es el que se activa después de que el robot responde ACK0 a ENQ, este cronómetro es para esperar la secuencia de Datos. Se vuelve activar cuando está a la espera de EOT. Cronómetro A se activa cuando el robot espera que se le responda al ENQ.

3.3.1 Transmisión de comandos.

En él capitulo anterior se describió que en el modo remoto del robot Motoman existen 3 maneras de operarlo; 1) función de trabajo aislado, 2) función de control modo anfitrión, 3) comunicaciones por instrucción. Esta última es la DCI(DATA COMMUNICATION BY INSTRUCTION) por medio del controlador y por medio de una computadora anfitriona. Toda la transmisión de comandos se hace de está ultima manera, es decir, el robot solo tiene que estar en espera de la orden. En las otras maneras el robot inicia él dialogo ya sea en el contexto de la ejecución de un programa o por la operación del control manual. Continuando con la función DCI; una vez activado el modo remoto el sistema está listo para recibir el código ENQ de la computadora, después de esto se envían los comandos los cuales

deben de ser enviados en un solo bloque por comando. Cuando la transmisión se ha completado la computadora espera recibir un ENQ y una vez establecido este último contacto el robot envía su respuesta.

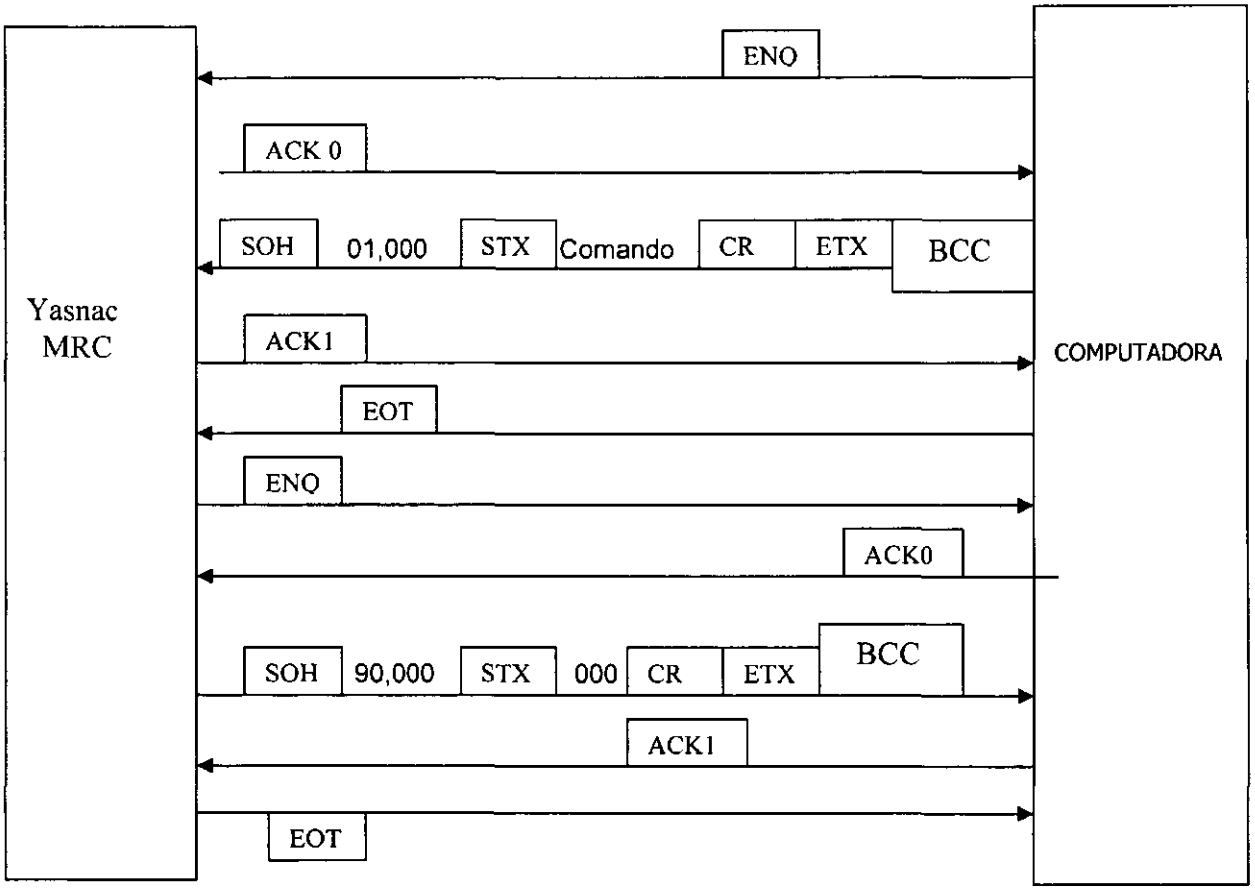


Fig. 3.3 Formato de transmisión de comandos[10]

Cuando ocurre algún error en la comunicación la respuesta del robot después del ACK no sería SOH 90,000 STX 000, si no que después de STX habría un número diferente de 000 que sería el código de error.

3.3.2 Los Comandos

Los comandos utilizados se dividen en 2 tipos: 1) Estado del sistema y Control del sistema.

Estado del sistema.

RALARM
Este comando lee el código de alarma y aunque el robot tiene el número del su código, no se puede leer con esta función. La respuesta de Motoman a este comando son 9 datos. El robot maneja los problemas de dos formas en errores y alarmas. El primer dígito corresponde a errores posibles, el resto de los números son código y datos del porque podría suceder la alarma.

RPOS
Este comando envía las posiciones de los ejes del robot en forma cartesiana.

RPOSJ
Este comando hace lo mismo que RPOS pero lo da en un sistema de coordenadas por ejes independientes llamada S.L.R.U.B.T. que están indicados en la Fig. 2.19 del capítulo 2, donde cada una de las siglas corresponde a cada uno de los ejes.

RSTATS
Lee el estado actual del play back box y por medio de números da información sobre los botones que se encuentran activados.

RJSEQ
Da lectura al trabajo actual y al número de línea

SAVEV
Manda variables del robot a la computadora.

Control del sistema.

Este se divide en comandos de operación, comandos de iniciación, comandos de edición y comandos de selección de trabajos

Comandos del sistema

HOLD
Activa o desactiva la pausa de operación. Cuando su valor es 1 está activado, cuando es 0 está desactivado.

RESET
Restablece al equipo cuando entra en estado de alarma.

CANCEL
Restablece al equipo cuando entre en estado de error.

MODE
Selecciona el modo de operación del robot 1 para TEACH y 2 para PLAY.

CYCLE
Selecciona el ciclo de trabajo 1 para STEP, 2 para 1Cycle y 3 AUTO.

Comandos de iniciación

START
Activa un trabajo desde la línea actual de operación. Un ejemplo de un trabajo es un programa creado por el usuario.
Ejemplo: START "Nombre del trabajo" Los nombres del trabajo tienen la variedad que el usuario desee, pueden ser numéricos o alfa numéricos.

PMOVJ
Mueve al manipulador con movimiento JOINT en coordenadas tipo pulso.
Requiere 14 datos de los cuales solo 6 se utilizan por ser los ejes con los que cuenta el robot. Los 8 restantes dan opción para otros ejes que se puedan instalar al robot.
El dato 1 corresponde a la velocidad en %.
Ejemplo: PMOVJ 20.0,100,200,300,400,500,0,0,0,0,0,0,0,0.
(S) (L) (U) (R) (B) (T) (PROTOCOLO)

Velocidad

PMOVL
Mueve al manipulador con movimiento LINEAL en coordenadas tipo pulso.
Requiere 15 datos de los cuales el primero corresponde al tipo de velocidad
(Se escoge entre 0 y 1) y el segundo al valor de la velocidad en mm/seg.
Ejemplo: PMOVL 0,123.0,10,200,300,400,500,0,0,0,0,0,0,0,0.
(S) (L) (U) (R) (B) (T) (PROTOCOLO)

Tipo Vel. Velocidad

Comandos de edición
DELETE
Borra un trabajo de la memoria.

LOADV
Se usa para guardar variables que provienen de la memoria del robot.

Comandos de selección de trabajos.
SETMJ
Hace que un trabajo se establezca como el trabajo maestro.

JSEQ
Fija en la pantalla del controlador manual un trabajo en una línea específico.

3.3.3 Construcción de un comando

A continuación se mostrará la construcción de un comando usando el protocolo BSC.
Uno de los comandos básicos es fijar el modo de operación.
Si se supone que el robot se encuentra en modo TEACH y se le quiere ordenar que pase al modo PLAY se tendrá que hacer lo siguiente:

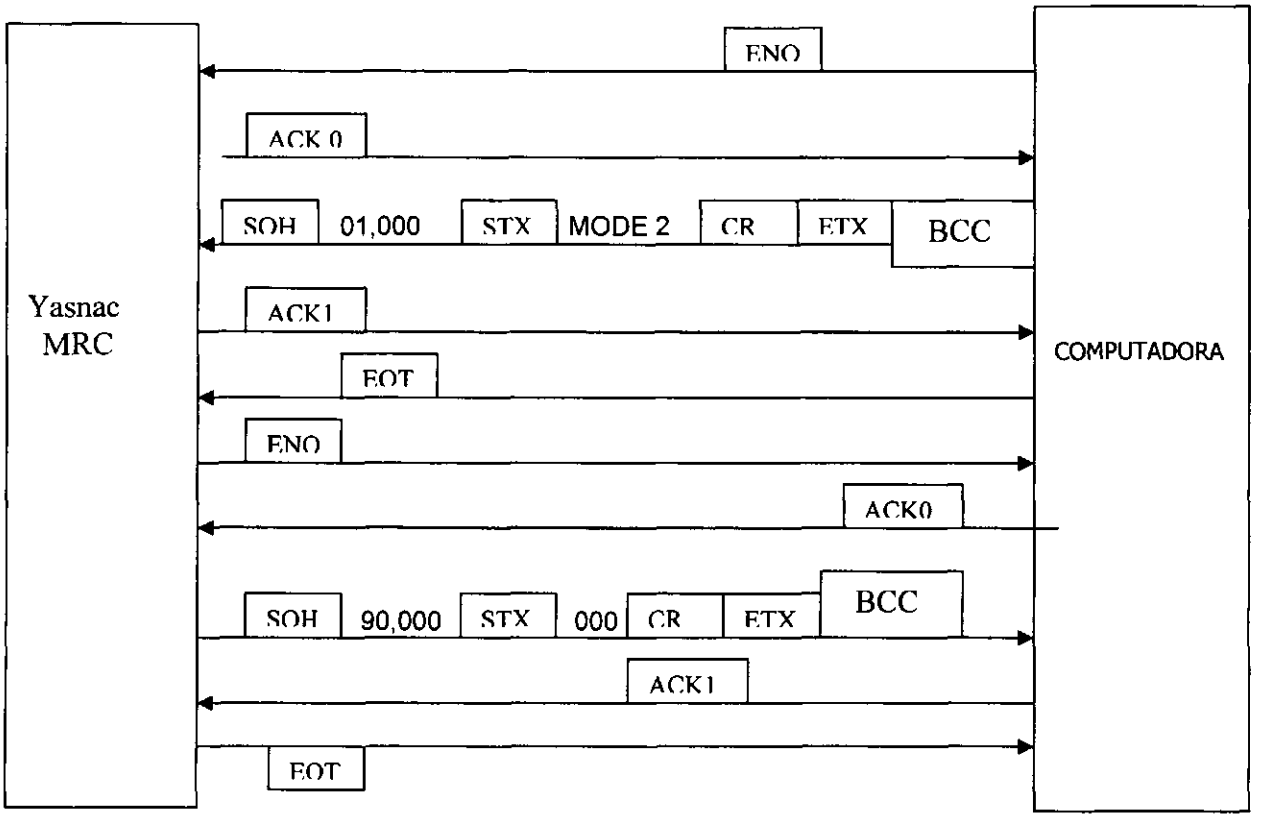


Fig. 3.4 Formato de transmisión de comandos [10]

El envío de comandos es la forma más sencilla de comunicación con el robot, ya que hay un número fijo de comandos. Las posibles respuestas a estos comandos y a los adicionales también son predeterminadas las variantes aparecen cuando se utiliza el comando STAR para ejecutar un trabajo creado por el usuario.
Con el siguiente ejemplo se observa él dialogo entre el robot y la computadora, se está haciendo el envío de un comando sencillo como la acción de tomar la herramienta llamada gripper del porta herramientas. El comando que se utiliza aquí es START. Este es el comando para ejecutar subrutinas. La subrutina es TGRIPPER, esta subrutina fue creada por el personal de mantenimiento para el propósito de tomar la herramienta, esto evita estar programando al robot cada vez que necesite tomar una herramienta y simplificar la operación. Sin embargo, antes de ejecutar una subrutina hay que asegurarse de que el robot está en el modo de operación correcto y en el modo de ejecución correcto. Para ejecutar una subrutina es necesario estar en modo PLAY, en este caso, se considera como un hecho que está activado ese modo. La operación se realizará una sola vez por lo tanto el modo de ejecutar única vez se debe de activar antes de mandar el comando de tomar herramienta, entonces se envía primero el comando que fija el modo de ejecución a CYCLE 2 y luego se envía el comando para comenzar la subrutina.

El renglón de arriba corresponde a la mitad de código enviado por la computadora y la de abajo a lo que corresponde al robot. Línea 1 corresponde a la fijación del ciclo en modo 2.

Liena 1

ENQ		SOH	01,000	STX	CYCLE	2 Cr	ETX	BCC	
	ACK 0								ACK1

EOT		ACK0							
	ENQ		SOH	90,000	STX	0000	Cr	ETX	BCC

Liena 2

ACK1		ENQ		SOH	01,000	STX	START	TGRIPPER	Cr
	EOT		ACK0						

ETX	BCC		EOT		ACK0				
		ACK1		ENQ		SOH	90,000	STX	0000

			ACK1						
Cr	ETX	BCC		EOT					

Fig. 3.5 Dialogo PC---Motoman

La respuesta normal del robot cuando el comando se ha recibido exitosamente es "0000"
Si hay alguna razón ajena al protocolo de comunicación por la que el robot no puede ejecutar el comando sé envía un código de respuesta, que generalmente corresponde a un error del procedimiento. Si hay algún error con el protocolo de comunicación el robot entrará en estado de alarma enviando un mensaje en la pantalla del controlador manual. Se hablará de estos códigos en el siguiente capítulo.

3.3.4 Envío y recepción de información.

La información se refiere a los programas creados por el usuario. Una de las dificultades con el uso de los programadores manuales es alimentar cantidades considerables de información.
Algunos de los controles tienen teclas de uso múltiple que hacen que el usuario ahorre pasos en el momento de estar alimentando un código de programación al robot. Sin embargo, muchas veces estas ventajas se vuelven desventajas cuando hay poco tiempo para aprender. El usuario pierde la noción de

la función en turno de la tecla y se cometen errores. Por lo tanto es necesario teclear la información en un ambiente amigable para después enviarla al controlador.
El envío de comandos se hace por bloque único de información, el envío y la recepción de información no necesariamente.

Del robot a la computadora (Recepción)
Cuando uno ha creado un programa o subrutina se almacena en la memoria del robot bajo la categoría de JOB. El título del programa es variable y puede ser cualquier palabra o número.

El algoritmo del dialogo es el siguiente.

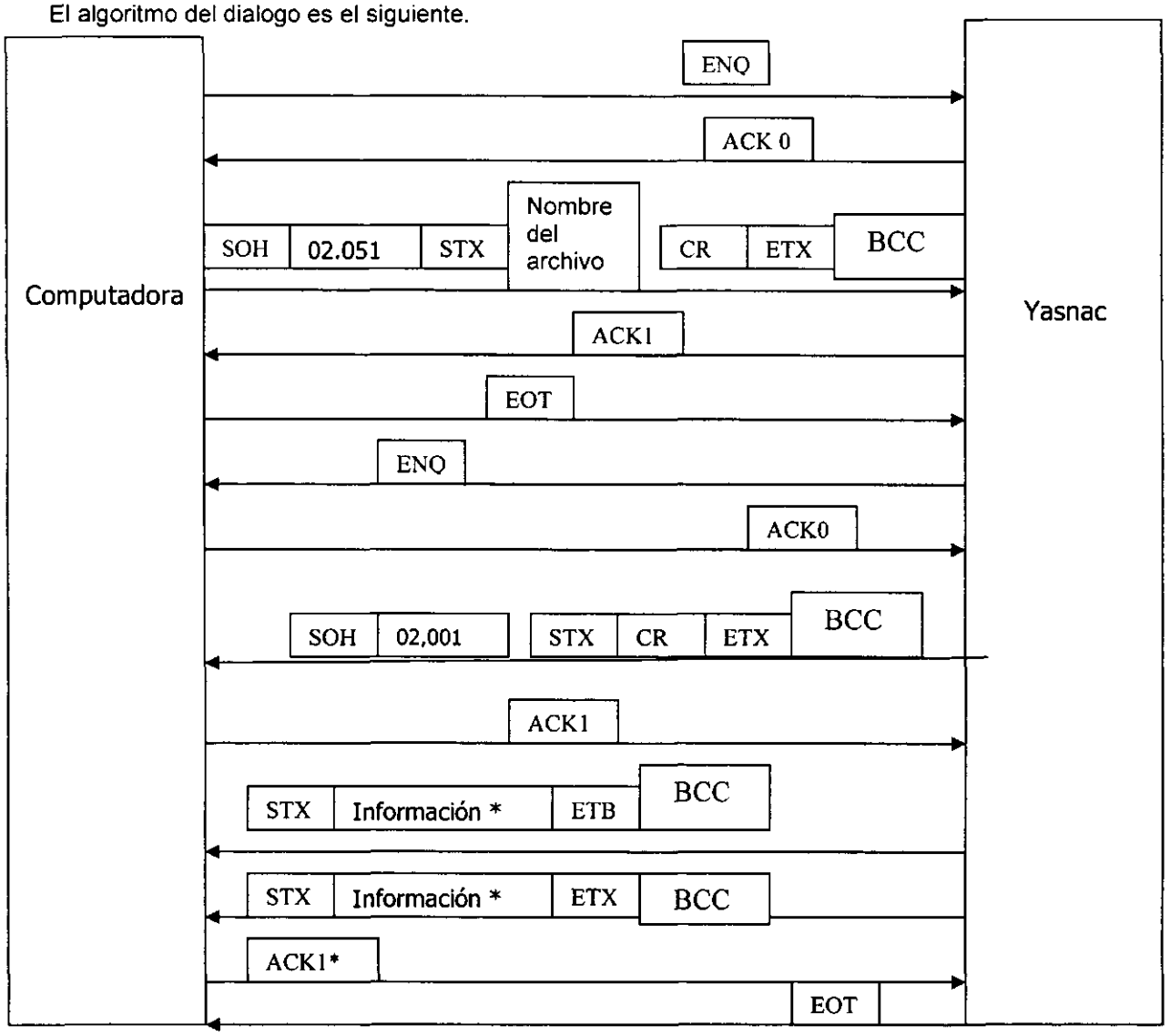


Fig. 3.6 Formato de trasmisión de información (Motoman-PC) [10]

El formato en el que se puede hacer la descarga de datos del controlador es de dos tipos: El simple y el relacionado. El simple es en el que se descarga un programa único y el relacionado es el que se baja el programa y también las subrutinas que se manden llamar dentro del programa principal. El encabezado 02,051 se usa en la petición de trabajos simples. El encabezado 02,001 lo usa el robot en respuesta a ese tipo de descarga. La información se manda por bloques máximos de 256 caracteres sin incluir los caracteres propios del protocolo. El número de bloques dependerá del número de caracteres totales del programa.

De la computadora al robot.

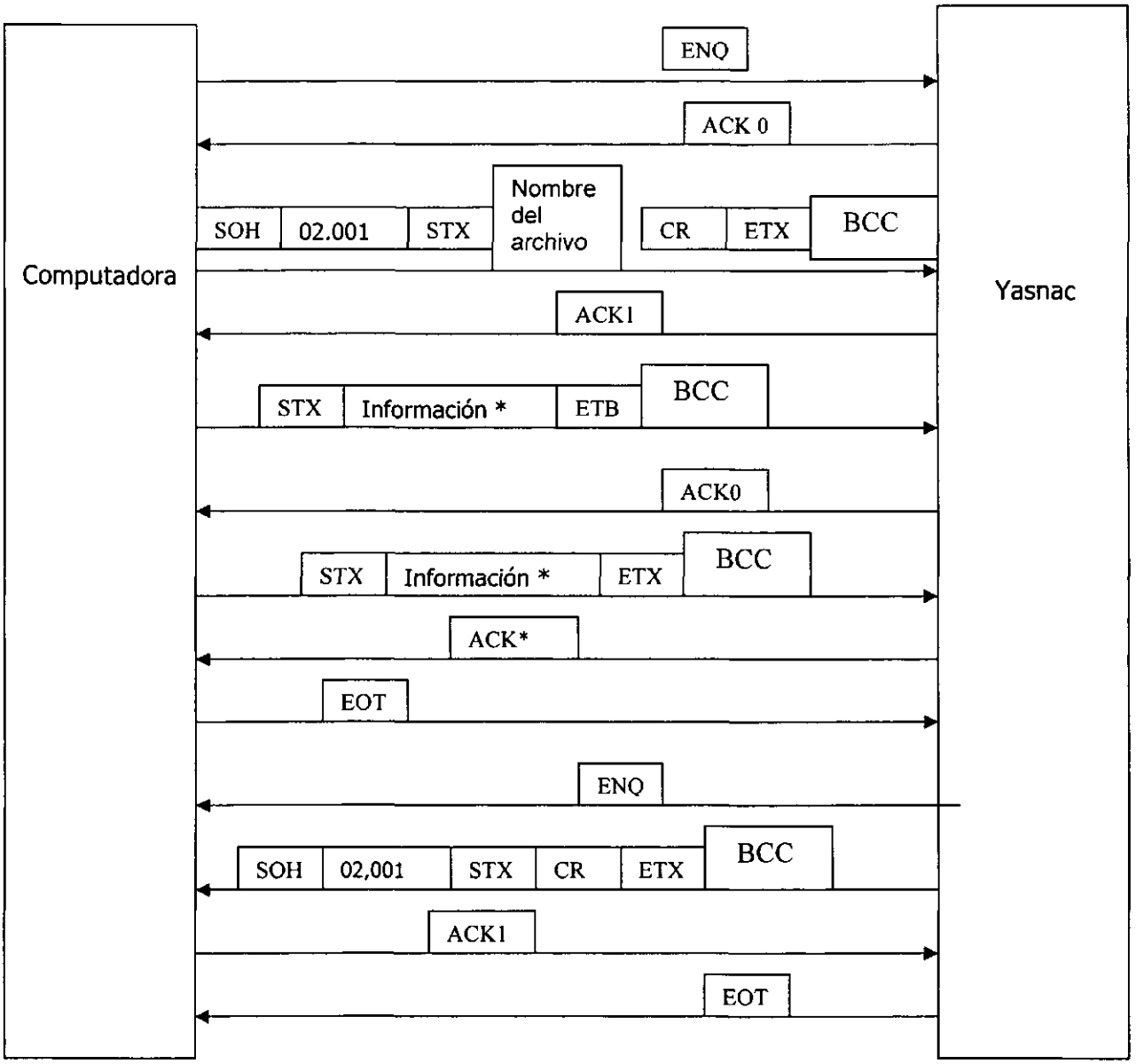


Fig. 3.7 Formato de transmisión de información (Motoman-PC) [10]

En este caso el algoritmo del protocolo tienen los mismo elementos que el anterior en diferentes puntos de la comunicación. Los paquetes de información se deben de mandar de 256 caracteres por paquete. El programa se almacena en la memoria del controlador y está listo para ejecutarse.

A continuación se muestra un ejemplo de un programa típico usado como subrutina. Este programa está almacenado en la memoria del controlador como TGRIPPER

```
NOP
JUMP *GRIPEROK IF
B001=1
MOVJ C0000 VJ=50.00
CALL JOB:DDESTORN IF
B001=2
CALL JOB:DVENTOSA IF
B001=3
WAIT IN#(11)=ON
DOUT OT#(3) ON
MOVL C0001 V=600.0
MOVL C0002 V=200.0
MOVL C0003 V=30.0 PL=0
DOUT OT#(3) OFF
HAND HNO: 1 ON
DOUT OT#(1) ON
DOUT OT#(2) OFF
TIMER T=1.00
WAIT IN#(11)=OFF
'ROBOT CON
HERRAMIENTA PUESTA
DOUT OT#(16) ON
DOUT OT#(18) OFF
SET B001 1
TIMER T=0.10
DOUT OT#(13) OFF
MOVL C0004 V=30.0
MOVL C0005 V=65.0
MOVL C0006 V=250.0
MOVJ C0007 VJ=50.00
HAND HNO: 1 OFF
DOUT OT#(1) OFF
DOUT OT#(2) ON
*GRIPEROK
DOUT OT#(1) OFF
DOUT OT#(2) ON
RET
END
```

El programa tiene mas de 256 caracteres por lo tanto se tiene que manipular antes de ser enviado. La recepción de un programa como este se hace también por partes.
Los detalles del manejo se hacen en el siguiente capítulo.

3.3.5 Pruebas de comunicación

La comunicación como ya se mencionó se lleva a cabo por medio del puerto serial. Él dialogo entre el robot y la computadora no se puede observar en ninguno de los dos. Para hacer las pruebas de comunicación y comprobar los protocolos se usó un intermediario entre la conexión de la computadora y el robot. El dispositivo es el Black Box modelo 232, es un equipo eléctrico con una pantalla digital que muestra en su pantalla la el intercambio de líneas de código entre el robot y la computadora.

Con esta figura se explica como debe de ser la conexión entre el "black box y la computadora"
Es necesario que el black box esté configurado para la velocidad de trasmisión y paridad

El equipo Black Box se inserta entre la conexión de la computadora y el robot.

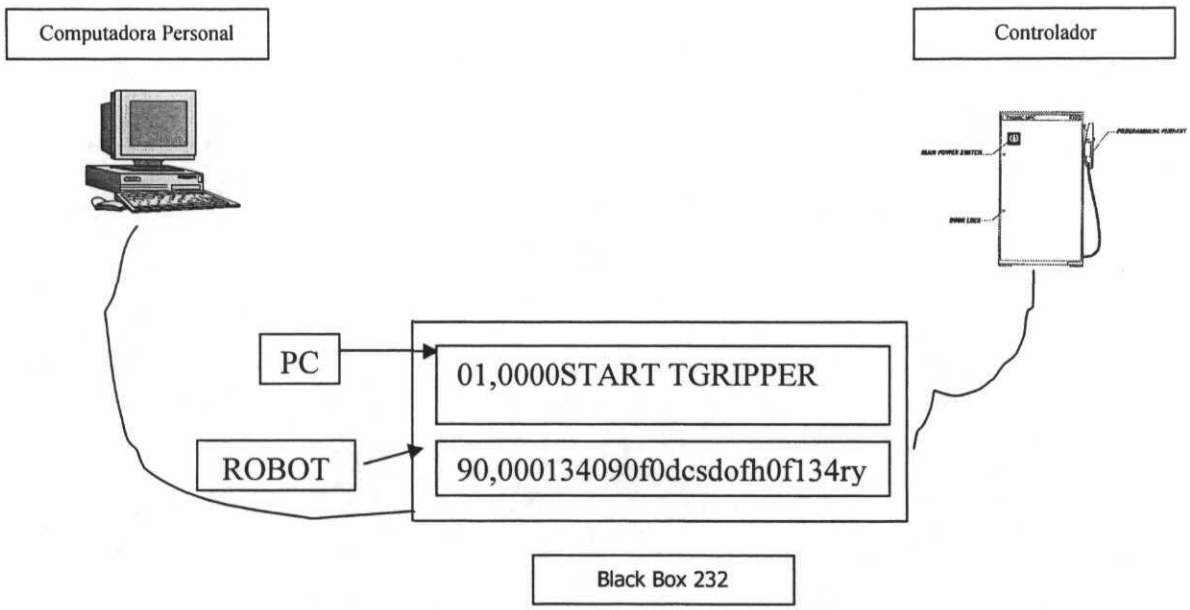


Fig.3.8 Ejemplo de conexión Black box

Para la construcción de la interfaz es básico conocer los algoritmos de comunicación que utiliza el robot ya que estos se deben emular para enviar y responder a comandos. En este capítulo se detallaron estos algoritmos, que son los más utilizados y recomendados por el manual del fabricante.

En el siguiente capítulo se presentará el programa interfaz, sus ventanas de dialogo y las instrucciones de uso. Se hablará sobre la conexión inicial, el envío de comandos, creación de programas, envió de programas, recepción de programas.

CAPÍTULO 4 EL PROGRAMA INTERFAZ

4.1 ELEMENTOS DE LA INTERFAZ

La interfaz de usuario para el robot Motoman está hecha con fines didácticos, por lo que en sus controles se encuentran los elementos necesarios para la realización de una práctica. Esta interfaz será manejada por alumnos que no están familiarizados con la operación de un robot. Los elementos principales son los siguientes:

- Panel de control Motoman
- Editor de programas

4.1.1 Panel de Control

Para el diseño del panel de control se utilizaron 2 criterios:

- Experiencia en el uso convencional del Robot
- Criterios de diseño de interfaces hombre-máquina

Experiencia en el uso convencional del robot:

El estar en contacto con el robot y haberlo programado usando la interfaz convencional por mucho tiempo, permitió que se tuvieran muy presentes las mejoras más significativas que requerían incluirse en el diseño de la interfaz hombre-máquina. Esta experiencia permitió decidir algunos de los siguientes criterios:

- Orden de aparición de los botones: Los botones están acomodados de tal manera que le facilite al usuario llevar la secuencia de actividades para las rutinas que se realizan con el robot en el Laboratorio De Sistemas Integrados De Manufactura.

- Los mensajes en los botones están escritos con la terminología que se usa durante la programación del robot para una práctica del laboratorio de Sistemas Integrados De manufactura.

- Se agruparon los botones por actividades en común para usarse como guía de continuidad y para evitar errores.

Criterios de diseño de interfaces hombre-máquina

Acomodo de objetos y letreros: Generalmente el usuario de captura la información de una pantalla de la misma manera que lo hace al observar una revista, esto en el oeste significa de la parte superior izquierda a la parte inferior derecha, a diferencia de un libro aquí no hay líneas así que el usuario generalmente da 2 o 3 vistazos. Las señales de alarmas principales se deben de colocar a lo largo de la parte superior de la página [13]

Teniendo esto en cuenta se colocaron los botones principales dentro de la trayectoria de observación más común

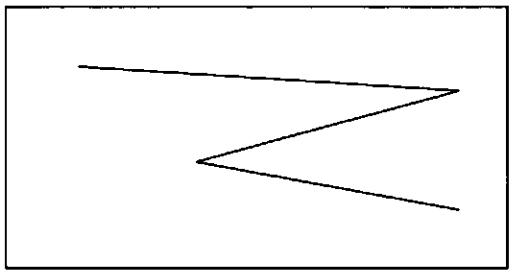


Fig. 4.1 trayectoria común de vistazo

Colores: En la selección de colores para una ventana, siguiendo el standard para las señales de alerta (BS5378):

Rojo: Detener, prohibir, peligro
Amarillo: Precaución, peligro
Verde: Condición segura
Azul: Acción imperativa

Para el contexto de la interfaz conviene usar colores disminuidos como café, gris y azules. Estos brindan un buen contraste para los colores que se usan en los botones principales. [13]

Selección del tipo de letra:
Debe de ser un tipo de letra común que esté disponible en la mayoría de las computadoras en las que el software pudiera ser utilizado. Debido a la resolución de la mayoría de los monitores es mejor usar letras del tipo san serif en lugar de serif. [13]

Es recomendable usar como máximo 3 tamaños de letras con frases cortas. Cuando se requiere profundizar más se usan las notas dinámicas que aparecen al posicionar el cursor. El estilo de letra se debe de mantener respetando siempre en mayúscula la primera letra. [13]

Control: Se recomienda que después de activar botones cuya repercusión es importante se corroborar la orden con una pregunta de rutina. Esto permitirá que el usuario pueda experimentar sin temor a dañar el equipo. [13]

Teniendo estos principios como criterios de diseño se procedió a crear la las ventanas de diálogo de la interfaz. El proceso de diseño de pantallas es iterativo ya que por varios motivos la interfaz cambia su estructura original como parte del proceso normal de diseño. La versión final de la pantalla principal es la siguiente:

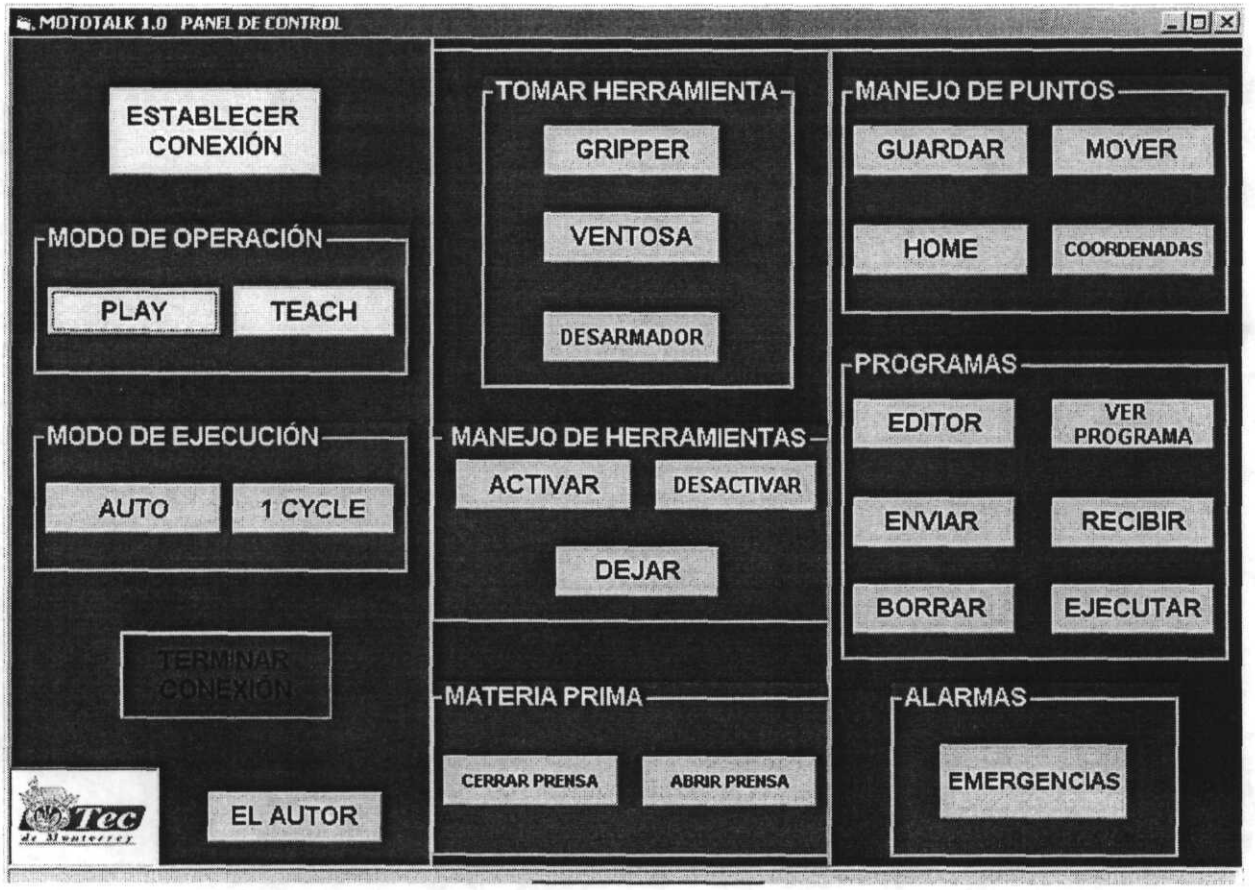


Fig. 4.2 Pantalla principal de diálogo

En esta primera pantalla con la que el usuario tendrá contacto se encuentran los comandos principales del robot, estos están agrupados según su función.

Las funcionès principales son:

- Básicas: conexión y posición
- Operación: PLAY y TEACH
- Modo: AUTO y CYCLE1
- Posiciones: guardar punto y mover punto.
- Herramientas: GRIPPER, VENTOSA y DESTORN.
- Programas: editor, fijar trabajo, borrar trabajo, enviar, recibir y ejecutar.
- Alarmas: emergencia

- CONEXIÓN: Al presionar este botón se envía al controlador de Motoman la instrucción RSTATS, el robot en forma de texto y en el formato BSC envía la información numérica correspondiente al estado de los indicadores principales.

  Tiene un menú de indicadores generales en los que se encienden los botones cuando estos se activan. Indica si los servomotores están activados. Indica si el manipulador está en estado de

alarma, error o detenido por un paro automático. Indica cual de las tres herramientas se encuentra en el manipulador y el modo de operación en el que está trabajando.

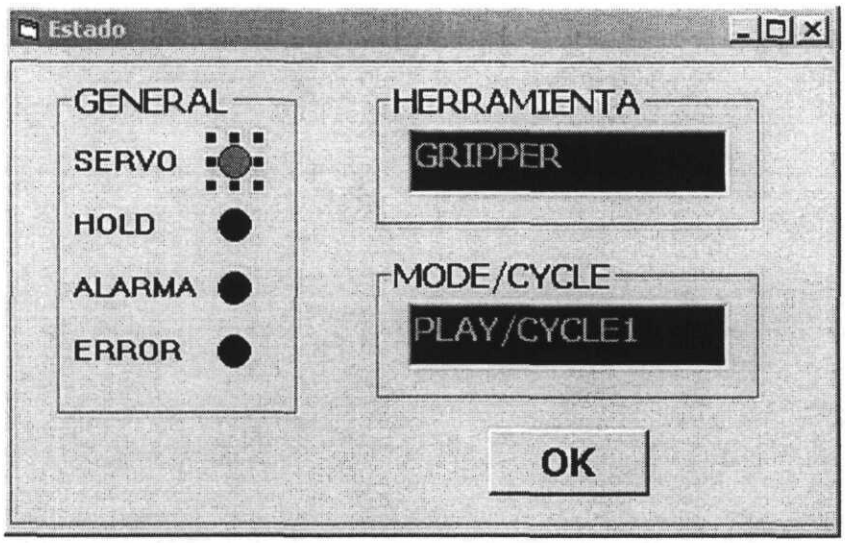


Fig. 4.3 Pantalla de status del robot

- POSICIÓN:
  En esta pantalla se muestran las posiciones de los ejes del robot Motoman.
  Los ejes se representan con letras, el comando que se envía al robot es RPOSJ, la respuesta del robot son 11 datos en los que se representan los siguientes ejes y espacios para ejes adicionales.

  S)Posición de giro de la base.

  L) Posición del tronco.

  U) Posición del Brazo

  R) Giro del brazo

  B) Posición de la muñeca

  T) Giro de la muñeca

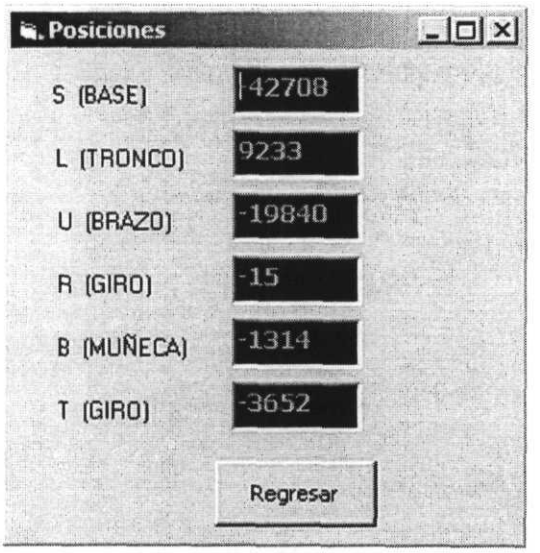


Fig. 4.4 Pantalla de posición actual

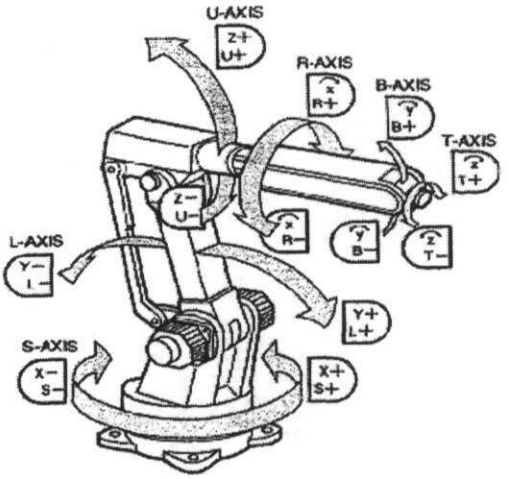


Fig. 4.4(A) Representación de ejes

- OPERACIÓN

  PLAY: Al activar este botón el robot está preparado para la ejecución automática de programas.

  TEACH: al activar este botón el robot está preparado para la enseñanza de puntos y rutinas.

- MODO
  AUTO: Al activar este botón se ejecutan los programas sin parar en el modo play.
  CYCLE 1: El robot solamente ejecutará una vez los programas en el modo play.

- POSICIONES: Guardar punto y mover a punto.

  GUARDAR PUNTO: Al activar este botón aparece una pantalla de dialogo donde se pide el número de punto a guardar. Las posiciones que se almacenan son las S.L.U.B.R.T. Estás se almacena en una variable global de posición. Este tipo de variable es diferente a la que se genera cuando se guarda una posición directamente desde el controlador remoto, las variables de posición del controlador remoto son locales, es decir, solamente se pueden usar en el programa en el que están siendo almacenadas. El robot no cuenta con un comando que permita almacenar posiciones en variables locales desde una computadora.

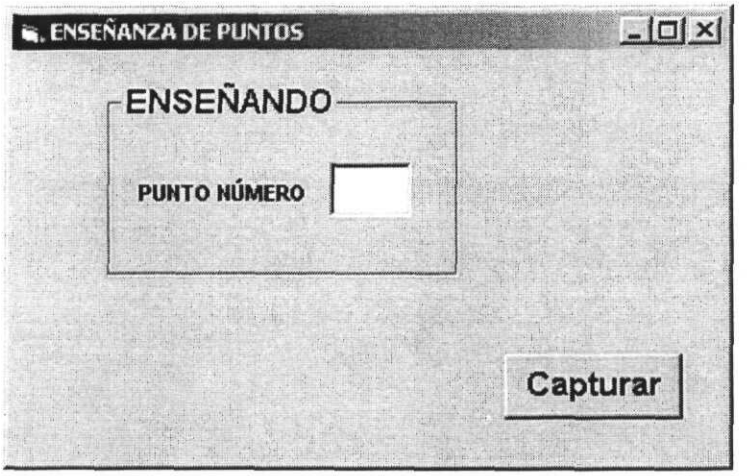


Fig. 4.5 Pantalla de enseñanza de puntos

MOVER A PUNTO

Al presionar este botón se activa la pantalla de dialogo para mandar al robot a una posición definida en variable global de posición únicamente. Fuera del contexto de un programa no se puede mover al robot a una posición que esté almacenada en una variable de posición local.

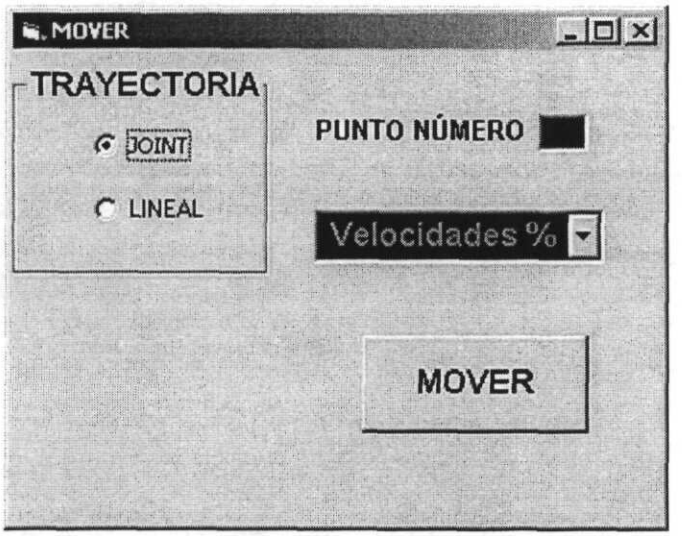


Fig. 4.6 Pantalla de movimiento a puntos

La forma de mover al robot de manera remota puede ser solo en 2 de los modos: En Joint y en lineal. En la parte derecha se observa el menú de velocidades que cambia según el tipo de movimiento deseado. También al lado de cada valor de velocidad se encuentra una referencia de la magnitud de la velocidad seleccionada.

- HERRAMIENTAS

Al presionar este botón se envía este comando "CYCLE 1"+ "START TGRIPPER"
Este ejemplo se explicó en el capítulo 3. Este comando hace que se ejecute una sola vez una rutina preprogramada en el controlador que sirve para que el robot tome la herramienta gripper, lo mismo sucede con las demás herramientas.

DEJAR HERR: Activa la subrutina preprogramada en el controlador de dejar la herramienta.

- PROGRAMAS

PROGRAMADOR: Este botón da acceso al editor de programas Motoman que se explicará con detalle posteriormente.

FIJAR TRABAJO: Activa un comando que permite seleccionar una rutina que se desee tener en la pantalla del control remoto para ser editada. Esto evita tener que usar el control manual para buscar el programa.

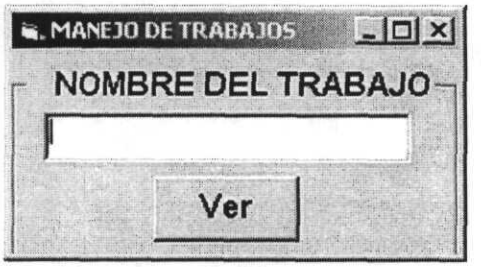


Fig. 4.7 Pantalla de búsqueda de programa

BORRAR TRABAJO: Activa el comando para borrar rápidamente un trabajo del controlador.

ENVIAR: al activar este comando se inicia el protocolo para enviar un archivo al robot Motoman. Los archivos que se mandan son programas que el usuario puede crear en el editor de textos. Los programas tienen extensión JBI.

RECIBIR: Este botón inicia el protocolo de recepción de archivos provenientes del robot Motoman. Los archivos que la computadora recibe son programas que se crearon o se modificaron en el controlador manual el robot. La extensión es la misma. JBI

EJECUTAR: Su función es semejante a la de los comandos de ejecución para tomar Herramientas, aquí se abre la misma ventana de dialogo de la figura 4.4 y se selecciona un programa para ser ejecutado por el robot Motoman. Este programa puede ser uno creado por el usuario o uno de las mismas rutinas de herramientas.

En la barra superior se colocaron botones de operación del robot que por seguridad deben estar en un lugar donde se dificulte presionarlas por error.

Todos los botones que se muestran en el panel de control activan comandos que el robot obedece cuando se trabaja en línea, a excepción del botón programador que es el que dá acceso al editor de programas.

- ALARMAS: Emergencia
  EMERGENCIAS: Por medio de este control se envía un comando que se llama "RALARM" Con el que el robot envía información correspondiente a errores o alarmas. Los estados de error generalmente suceden cuando el usuario presiona botones que van en contra de la lógica de uso del programador manual o en el play back box. Las alarmas se dan por situaciones de error en cuanto a la lógica de ejecución de programas. Este comando identifica ambas situaciones y manda un mensaje clasificatorio al usuario y automáticamente corrige el estado de alarma mandando el comando "CANCEL" para errores y "RESET" para alarmas. El robot volverá a su estado normal si es que la situación lo permite.

4.2 EL EDITOR DE PROGRAMAS

El programador funciona como emulador de la pantalla de control. Su función en este caso es reproducir el código para llamar estos comandos dentro de la estructura de un programa. Esta parte del programa es la que hace posible que se pueda programar fuera de línea, siempre y cuando se cuente con un diagrama de puntos y el conocimiento adecuado del área de trabajo.
En la ventanilla derecha aparecen los programas para editar.

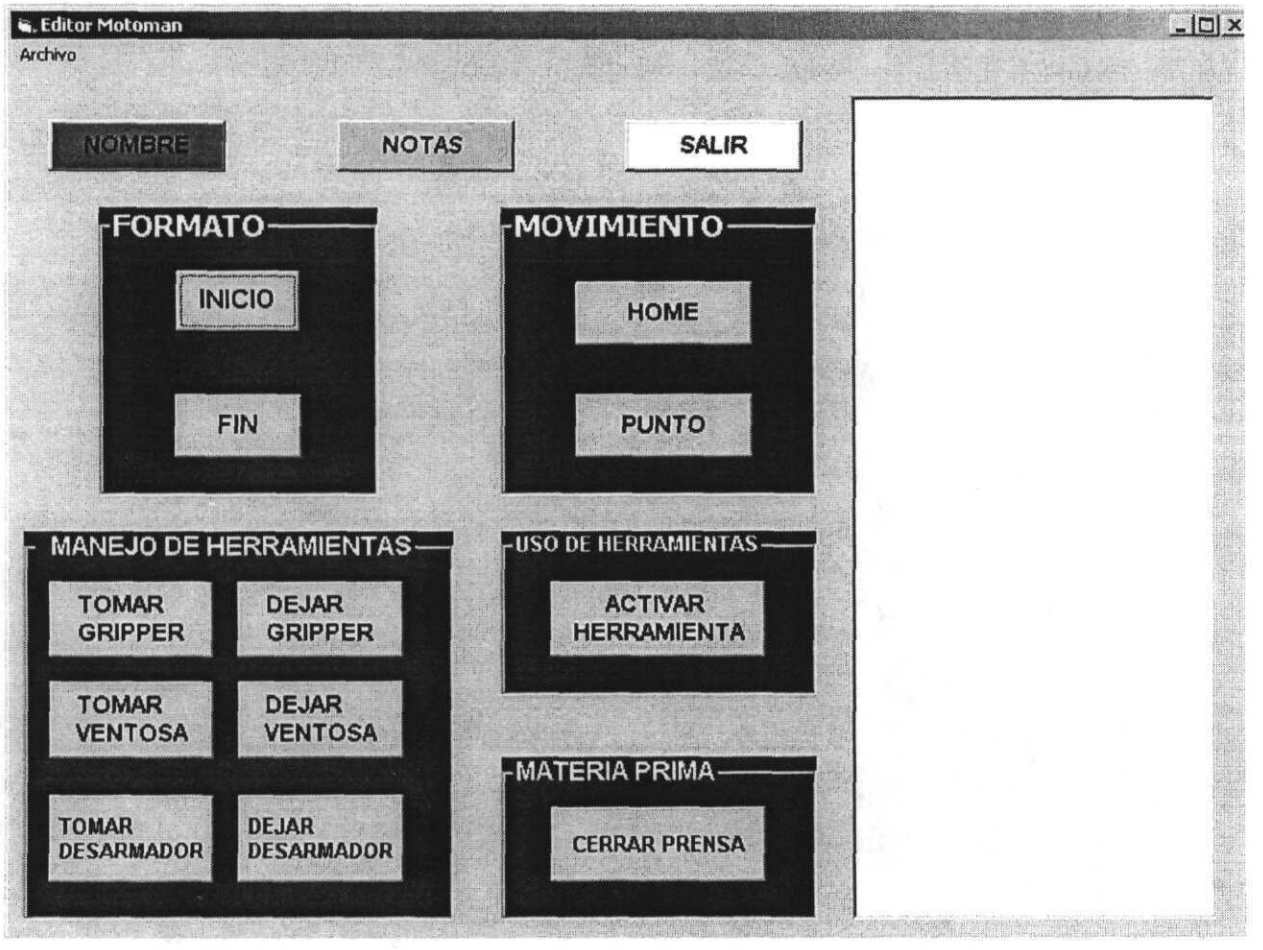


Fig. 4.8 Pantalla del programador

Los botones se han organizado de manera que esté a favor del flujo de la programación natural del robot Motoman.

- ENCABEZADO:
  Al Activar este botón se activa una pantalla en la que se dan de alta los requisitos necesarios para que el compilador del robot Motoman acepte un programa proveniente de un dispositivo externo, en este caso la computadora.

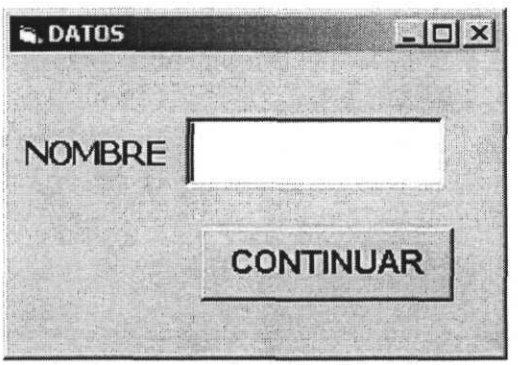
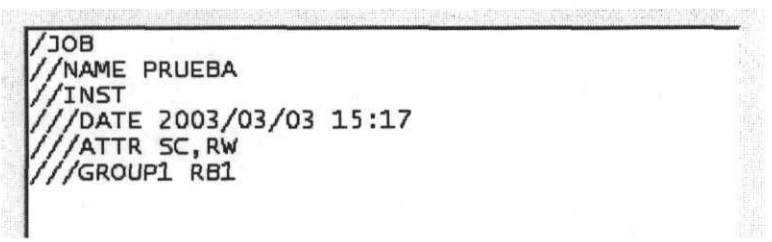


Fig. 4.9 Ventana de dialogo del encabezado

En esta pantalla el usuario da de alta el nombre del programa, el cuál debe de ser el mismo que el nombre del archivo, después se debe dar de alta la fecha y la hora en la que se está grabando el programa.

En la pantalla de programación se observará la siguiente información.



La palabra JOB es parte del código del robot, la palabra INST significa el inicio de instrucciones, RW es la propiedad de hacer el programa se puede editar.

- **FORMATO:** Inicio, fin.

INICIO: Al activar este botón se llevan a cabo se genera el código que indica le inicio del programa:

NOP
CALL JOB: HOME
La instrucción NOP indica al robot Motoman el inicio de operaciones dentro de un programa y el comando siguiente manda llamar la subrutina que envía al robot a su posición de referencia.

FIN: este comando coloca en el programa el comando END, el cuál indica el fin de un programa.

- **MOVIMIENTO**

  HOME: Al activar este botón aparecerá en pantalla el comando que manda llamar la subrutina que lleva al robot a su punto de referencia.

  MOVE: al presionar este botón se activa la pantalla que se observó en la figura 4.4. En esta pantalla se selecciona el tipo de variable en la que está almacenada la posición.
  También se debe seleccionar el tipo de movimiento y la velocidad del mismo. El código que genera es el siguiente:

  MOVL C0001 V=93.0

  En este ejemplo se seleccionó una variable de posición local con tipo de movimiento lineal y una velocidad de 93 MM/sec.

- HERRAMIENTAS: Tomar gripper, tomar ventosa, tomar desarmador, activar herramienta, abrir mordazas.

  Todos estos botones colocan en pantalla el código para mandar a las subrutinas para tomar las herramientas correspondientes, activarlas y/o dejarlas en su lugar.

  Tomar gripper: CALL JOB: TGRIPPER
  Dejar Gripper: CALL JOB: DGRIPPER
  Tomar ventosa: CALL JOB: TOMVENT
  Dejar Ventosa: CALL JOB: DVENTOSA
  Tomar desarmador: CALL JOB: TDESTORN
  Dejar desarmador: CALL JOB: DDESTORN
  Activar herramienta: CALL JOB: RBHERRON
  Desactivar Herramienta: CALL JOB: RBHERROF
  Abrir mordazas: CALL JOB: ENSCIPRE
  Cerrar mordazas: CALL JOB: ENSABPRE

  El nombre de estas funciones está establecido de acuerdo al nombre que le haya querido dar el programador, por lo tanto, si a la decisión del programador cambian estos nombres se deberán hacer los cambios correspondientes en el código del software.

4.3 INSTRUCCIONES DEL USO DE LA INTERFAZ MOTOMAN

Para facilitar la aplicación el usuario debe de estar familiarizado con algunos de los términos básicos del uso del Motoman, pero por el diseño de la interfaz se minimiza la necesidad de estos conocimientos.

Al correr el programa interfaz al usuario se le presenta la pantalla de control principal. En este punto los botones para trabajar en línea con el robot permanecen inhabilitados a excepción del botón de conexión y el del programador o editor de programas.

4.3.1 Establecer conexión con el robot

Para establecer contacto con el robot Motoman solo es necesario dar clic sobre el botón conexión. Al activarse este botón manda el carácter ENQ para esperar la respuesta ACK0, al ocurrir esto se puede decir que ya se ha establecido conexión con el manipulador.
Los motivos por los que no se podría establecer conexión son 4:

1. Que el cable serial que está conectado al robot Motoman no esté conectado al puerto serial de la computadora.
2. Que el controlador del robot Motoman no esté encendido.
3. Que el puerto serial de la computadora no esté configurado como COM1
4. Que en el play back box no esté activado el botón REMOTE.

Cuando la conexión falla el usuario tiene un numero de oportunidades indefinido para restablecer la conexión, si así se desea o de cancelar los intentos.

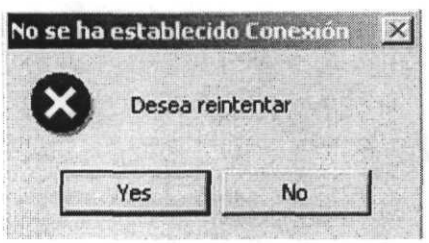


Fig. 4.10 Ventana de diálogo de reintento

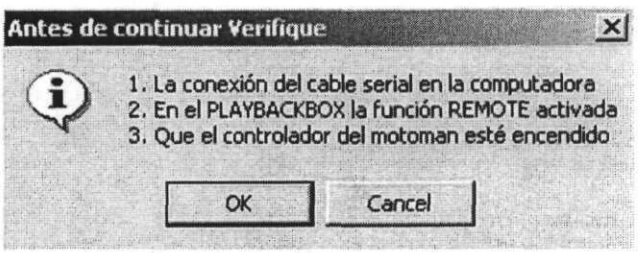


Fig. 4.11 Ventana de diálogo de reintento

4.3.2 Modo de trabajo, alarmas y errores.

Después de que se ha establecido conexión se puede trabajar con el robot de 2 modos. De modo PLAY para ejecutar programas en el controlado. El otro modo es TEACH el cuál se usa para enseñanza de puntos y edición de programas desde el control manual.

Al estar trabajando con el robot se pueden dar situaciones de alarma o de error. Dichas situaciones el software las detecta y las corrige cuando el usuario recurre al botón de emergencias. Los errores se dan por causa de errores lógicos en la operación del Hardware del robot. Las alarmas se dan por situaciones de errores lógicos en el software del robot y cuando hay impactos o falta de aire en el sistema neumático del robot.

4.3.3 Creación de un programa

Para crear un programa se comienza presionando desde el panel de control el botón programador, después aparece la pantalla del editor de Motoman. Siguiendo el flujo lógico de programación del robot Motoman se empieza por el encabezado. En esta sección, como ya se mencionó anteriormente, se da de alta el nombre del programa. Se debe tener en cuenta que el nombre del programa y el nombre del archivo deben de ser el mismo. También se coloca la fecha y la hora de elaboración. Después con el botón de inicio se coloca las primeras 2 instrucciones del programa. La siguiente instrucción corresponde a la de tomar la herramienta a utilizar. Suponiendo que se empieza con la herramienta GRIPPER se programa el comando para que el robot la recoja. Se debe tener en cuenta que solo estamos programando fuera de línea y que ninguno de los comandos que aparecen

en pantalla se está llevando a cabo en vivo. Después de estas instrucciones iniciales se colocan los comandos para llamar puntos. Intercaladas entre los puntos aparecerán otros comandos para hacer cambio de herramienta o activaciones de dispositivos externos como las mordazas ETC. A continuación se muestra un ejemplo de un programa hecho desde la interfaz.

```
/JOB
//NAME CAJA <------ Nombre del programa
//POS
///NPOS 0,0,0,10,0,0
///TOOL 0
///POSTYPE PULSE
///PULSE
//INST
///DATE 2003/04/11 11:08 <------ Fecha del programa
///ATTR SC, RW <------ Configuración de seguridad
///GROUP1 RB1 <------ Configuración del robot
NOP <------ Inicio
CALL JOB: HOME
CALL JOB: TGRIPPER
MOVL P001 V=375.0 <------ movimiento lineal al punto 1 a una velocidad de 375.0
MOVL P002 V=46.0
CALL JOB: RBHERRON
MOVL P001 V=187.0
MOVL P003 V=375.0
MOVL P004 V=93.0
CALL JOB: RBHERROF
MOVL P003 V=187.0
CALL JOB: ENSCIPRE
CALL JOB: HOME
CALL JOB: TOMVENT
MOVL P005 V=375.0
MOVJ P006 VJ=50.00 <------ Movimiento en forma joint al punto 6
MOVL P007 V=375.0
MOVL P008 V=46.0
CALL JOB: RBHERRON
MOVL P007 V=187.0
MOVJ P005 VJ=50.00
MOVL P009 V=375.0
MOVL P010 V=46.0
CALL JOB: RBHERROF
MOVL P009 V=187.0
CALL JOB: ENSABPRE
CALL JOB:HOME
CALL JOB: DVENTOSA
END
```

Este programa corresponde al ensamble de una caja con una tapa. El programa se muestra tal como es hecho desde el controlador.

4.3.4 Uniendo programa con la enseñanza de coordenadas.

Cuando se crea un programa desde la computadora generalmente no se tienen las coordenadas de las posiciones. Estas coordenadas se podrían obtener de un simulador o trabajando en línea con el robot. A estas alturas los puntos todavía serían imaginarios. Posteriormente al cargarlos al controlador se generarían las variables de posición todas con coordenadas cero:

```
P0000=0,0,0,0,0,0
P0001=0,0,0,0,0,0
```

Así sucesivamente. Se van creando variables huecas. Cuando el programa está en controlador manual se procede a editarlo con el fin de grabar las posiciones en los huecos.
Primero es necesario pasar el programa al controlador. Para esto se presiona la opción de enviar programa al robot.

Para este ejemplo el cursor del controlador se colocará en la línea del punto C000 y en el control manual se presionarán las teclas EDIT+MODIFY+PLAY SPEED+ENTER. El punto se sobre escribe.

Para esto el robot debe ser llevado de manera manual a las posiciones deseadas.
Para efectos prácticos puede ser que requiere ejecutar algún comando pero no junto con el programa y además de manera rápida. Para esto se utilizan los botones de comandos del panel de control y se ejecutarían acciones como las de tomar herramienta, activarla entre otras. Esto para grabar puntos en los que se requiere que el material esté en la herramienta del robot.

Uno de los atajos que se puede utilizar y que se ha hecho posible por la interfaz es que dos personas trabajen por separado para obtener el programa. El usuario #1 podría estar trabajando en la creación del programa, mientras que el usuario #2 podría estar grabando posiciones. Siendo esta la situación el usuario #2 se toparía con la dificultad de que las posiciones de variables locales requieren del programa para ubicarlas.

La solución está en grabar las posiciones en las variables globales P000 las cuales no necesitan estar dentro de la estructura de un programa. Teniendo esto el usuario #1 puede hacer la operación EDIT+MODIFY+PLAY SPEED+ENTER en el lugar adecuado teniendo solo por hacer el mandar llamar los puntos de variables globales que se grabaron con anterioridad por el usuario #2.

4.3.5 Recibiendo del controlador

Para recibir un programa del robot es necesario seleccionar la opción recibir programa, desde el panel de control o desde el editor. Este programa es el mismo que se envíen el inciso anterior pero ahora cuneta con posiciones definidas en sus puntos:

Aquí se observan las posiciones adquiridas y el uso de algunos comandos que no se han mencionado:

```
/JOB
//NAME  CAJA
//POS
///NPOS 0,0,0,10,0,0
///TOOL 0
///POSTYPE PULSE
///PULSE
P001=-33869,17043, -24832,69,289, -8770
P002=-33867,17165, -29439,74,1824, -8784
P003=-51696,20823, -26778,67,947, -8325
P004=-51695,21624, -30741,73,2268, -8337
P005=-23340,23293, -24030,51,62, -6736
P006=11341,23293, -24030,51,62, -6736
P007=12830,26425, -29758,58,1966, -6985
P008=12829,28173, -33240,65,3126, -6994
P009=-51309,20112, -28015,65,1336, -9240
P010=-51308,22523, -33674,78,3222, -9259
//INST
///DATE 2003/04/11 11:08
///ATTR SC, RW
///GROUP1 RB1
NOP
```

Posiciones almacenadas

Este programa es igual al mostrado en la Pág. 53. La diferencia es que esta es la forma del programa cuando se descarga del controlador manual a la computadora. Este es el programa de la cajita creado desde el robot. Al descargarse envía los puntos.

```
CALL JOB: HOME
CALL JOB: TGRIPPER
MOVL P001 V=375.0
MOVL P002 V=46.0
CALL JOB: RBHERRON
MOVL P001 V=187.0
MOVL P003 V=375.0
MOVL P004 V=93.0
CALL JOB: RBHERROF
MOVL P003 V=187.0
CALL JOB: ENSCIPRE
CALL JOB: HOME
CALL JOB: TOMVENT
MOVL P005 V=375.0
MOVJ P006 VJ=50.00
MOVL P007 V=375.0
MOVL P008 V=46.0
CALL JOB: RBHERRON
MOVL P007 V=187.0
MOVJ P005 VJ=50.00
MOVL P009 V=375.0
MOVL P010 V=46.0
CALL JOB:RBHERROF
MOVL P009 V=187.0
CALL JOB:ENSABPRE
CALL JOB:HOME
CALL JOB:DVENTOSA
END
```

NPOS: Que se refiere al número de posiciones y al tipo que en este caso son del tipo locales ya que el nombre que tienen es C000.

TOOL 0= La herramienta en turno en el portaherramientas del robot.

POSTYPE= Aquí se describe los ejes en los que están dadas las variables de posición. Los ejes son S.L.U.B.R.T y son llamados también Pulse.

No se ocupa de un archivo de puntos con extensión propia ya que las coordenadas se envían junto con el programa.

4.4 AQUITECTURA Y ESTRUCTURA DEL PROGRAMA

De acuerdo a lo que se trató en el capítulo anterior se construyeron las bases para el algoritmo de programación de un software intermediario entre el protocolo BCC y el usuario.
El lenguaje que se utilizó es el Visual Basic versión 6.0 profesional. Se escogió este programa por la facilidad en la creación de pantallas interactivas y por las ventajas de programación que el paquete ofrece.

La función central del programa se divide en 3 partes que son el envió de comandos, la descarga de archivos y la carga de archivos al robot. Para cada una de las partes se creó un conjunto de subrutinas diferentes. Las diferencias se justifican en que los algoritmos de programación para cada una de estas operaciones es diferente por motivos de sensibilidad a los cambios por parte del programa y variabilidad de la respuesta del robot.

4.4.1 Envío de comandos

El protocolo de comunicación está formado por una subrutina llamada MOTOTALK. Esta subrutina se usa es la base del envío de comandos. Esta formada por 3 subrutinas que son: BCC, GEN, COMAND, DIALOGO. Mototalk es una función anidada en Proto_master2, esta función contiene rutinas de dialogo inicial con el controlador y algo de dialogo con el usuario. A continuación se muestra la interacción:

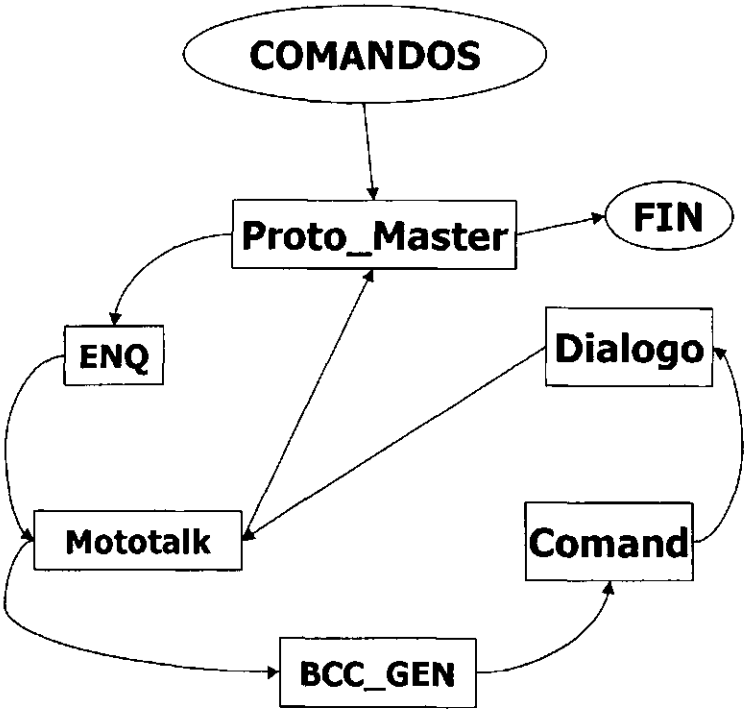


Fig. 4.12 Esquema de interacción para envío de comando

El algoritmo de esta rutina es el siguiente:

1. Por default la función vale 1
2. Abrir Puerto serial
3. Enviar ENQ
4. Esperar respuesta
5. Dependiendo de la respuesta cambia el valor de la función fijada en 1
6. Si función es 1 ejecuta mototalk, de lo contrario envía mensaje

```
Public Function proto_master2() As Integer
Proto_master2 = 1
Answer = 1
Abre _ puerto                    'Subrutina que abre el puerto de com
ENQ
Timer4.Enabled = True
If enqerror = 0 Then      'Función que verifica que haya comunicación con el robot regresa un valor si es
que no la hay
```

```
COMANDO = COMANDO1              'Puede llegar a ser útil
Mototalk                        'Es el procedimiento de comunicación
Else
'mensajes
proto_master2 = 0
Cierra_puerto
Exit Function
End If
End Function
```

BCC_GEN

Es la subrutina que se encarga de generar los dígitos BCC propios del protocolo de comunicación. El fragmento de código que corresponde a esta función es el siguiente:

El algoritmo de esta rutina es el siguiente:

1. Fijar Variables
2. Leer caracteres del comando a enviar
3. Transformar los caracteres a su equivalente numérico
4. Aritmética
5. Asignación de valores.

```
Private Sub BCC_GEN()
Dim j As Integer
j = 1
BCC = 303
Do Until j > Len(COMANDO)
        caracter = Mid(COMANDO, j, 1)
        BCC = BCC + Asc(caracter)
        j = j + 1
Loop
bcc1 = BCC Mod 256
cociente = BCC / 256
BCC2 = Fix(cociente)
End Sub
```

El BCC comienza en 303 porque es la suma de los caracteres que se colocan invariablemente en el protocolo de los comandos. Comando es una variable que contiene el nombre del comando seleccionado, esta información se va convirtiendo al valor numérico del código ASCII y se añade al 303. Por último se obtienen el cociente y el residuo con base 256 y se obtiene el primer número llamado bcc1 y BCC2 respectivamente.

COMAND

Es la subrutina que envía el comando completo al controlador del robot.
Este comando incluye el bcc1 y BCC2 generados en la rutina anterior.

```
Private Sub COMAND( )

MSComm1.Output = Chr$(1) + "01,000" + Chr$(2) + COMANDO + Chr$(13) + Chr$(3) + Chr$(bcc1) +
Chr$(BCC2)
End Sub
```

DIALOGO
Esta subrutina se encarga de mantener el dialogo con el robot después de que éste recibió el comando. Esta rutina está formada por una serie de lecturas del puerto serial y tiempos de espera. Posteriormente envía las respuestas que corresponderían al protocolo de envío de comandos.

```
Prívate Sub Dialogo()
TACK1
TLEENQ
tiempo
motodata = MSComm1.Input
MSComm1.Output = Chr$(16) & Chr$(49)
TEOT
End Sub
```

4.4.2 Descarga de archivos

La descarga de archivos maneja una estructura similar a la del envío de comandos. Lo que varía son los algoritmos de las subrutinas. La función que anida las subrutinas es proto_master1.
La cuál tiene la misma función que proto_, master2.
La rutina de comunicación es mototalk_job que contiene BCC_JOB, comand_job, dialogo_job

A continuación el esquema de descarga de archivos

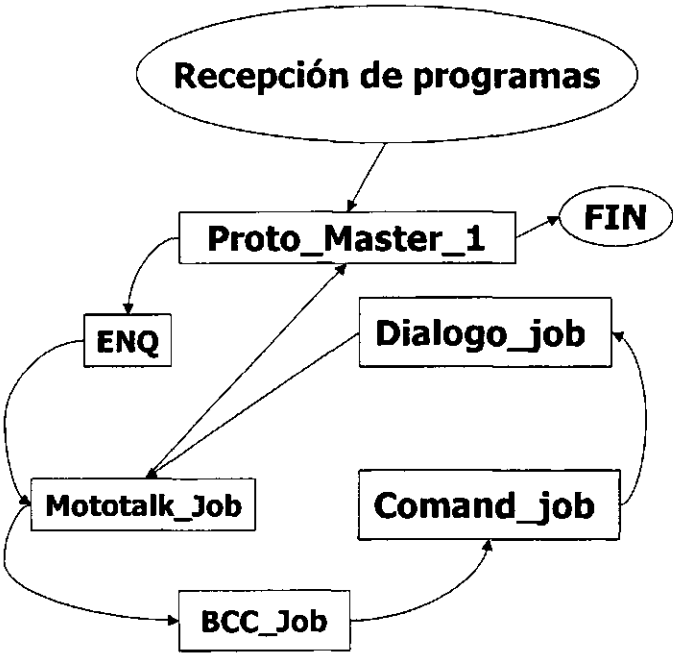


Fig. 4.13 Esquema de interacción para recepción de programas

BCC_JOB
Es la misma función que BCC_GEN solo que aquí la suma se inicializa a 310.

Comand_job
La función es la misma que comand solo que cambia el encabezado con el que se envía el comando.

```
Private Sub comand_job()
MSComm1.Output = Chr$(1) + "02,051" + Chr$(2) + COMANDO + Chr$(13) + Chr$(3) + Chr$(bcc1) +
Chr$(BCC2)
```

Dialogo_job

```
Private Sub Dialogo_job()
tiempo _ dialogo
respuesta = MSComm1.Input
MSComm1.Output = Chr$(4)
barra1
tiempo _ dialogo
respuesta = MSComm1.Input
MSComm1.Output = Chr$(16) & Chr$(48)
tiempo _ Job
respuesta = MSComm1.Input
MSComm1.Output = Chr$(16) & Chr$(49)
MSComm1.RThreshold = 1
```

En este caso todo se mantiene igual con excepción de la última línea. Esta línea activa el evento del serial cada vez que haya un carácter en el buffer de entrada. Esto convierte el programa en una "oreja" digital que está sensible a la información que mande el robot. Esto se hizo por la variabilidad del tamaño de bloque que el controlador maneja cuando los archivos son de tamaño variable. La rutina MSCOMM1 se puede observar en los anexos.

4.4.2 Carga de archivos

En esta parte de la comunicación la rutina general es proto_master3 que contiene mototalk_job_envio(), esta a su vez, contiene bcc_job_envio, comand_job_envio y dialogo_job_envio. Las funciones de estas rutinas son las mismas que en los casos anteriores. Solo se resaltará la última en la que hay cambios considerables en el algoritmo con respecto a las anteriores.

dialogo_job_envio

Con esta rutina se ejecuta el envío de programas de tamaño variable al robot.
Ya que el controlador solo puede recibir 256 caracteres por bloque, fue necesario ajustar el algoritmo de tal manera que seccionara el programa en partes de tamaño 256. Después de esto enviarlas a su destino con el protocolo de envío de cada bloque.

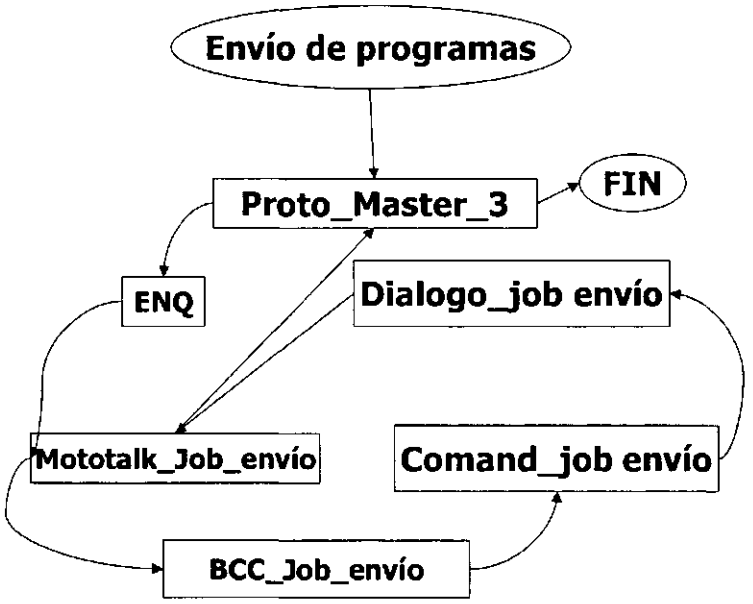


Fig. 4.14 Esquema de interacción para envío de programas

El algoritmo de la rutina es el siguiente:

1. Determina el numero de caracteres del archivo a enviar.
2. Obtiene el valor de paquetes tamaño 256 a enviar
3. Obtiene el residuo para paquetes de tamaño < 256
4. Si el programa es menor a 256 caracteres lo envía directamente, de lo contrario lo manda por paquetes

```
Private Sub Dialogo_job_envio()
Dim contador, contador2, ncar, entero, resto As Integer
Dim car_envio As String
Dim tamaño, pak, paquetes As Integer
tiempo_dialogo
respuesta = MSComm1.Input
tamaño = Len(strtext)
pak = tamaño / 256
paquetes = Fix(pak)
entero = paquetes * 256
resto = tamaño - entero
Select Case pak
Case Is <= 1
   car_envio = Mid(strtext, 1, tamaño)
    arch_envio = car_envio
     chksumfin = 1
```

```
        BCC_JOB_texto
         MSComm1.Output = Chr$(2) + arch_envio + Chr$(3) + Chr$(bcc1) + Chr$(BCC2)
          tiempo_dialogo
           respuesta = MSComm1.Input
            MSComm1.Output = Chr$(4)              'Envia EOT
             barra1
              tiempo_dialogo
               respuesta = MSComm1.Input              'Lee ENQ
                MSComm1.Output = Chr$(16) & Chr$(48)     'Envia ACK0
                 tiempo
                  motodata = MSComm1.Input              'Se lee lo que hay en el buffer
                 MSComm1.Output = Chr$(16) & Chr$(49)     'Responde ACK1
                tiempo_dialogo
               respuesta = MSComm1.Input              'Se lee EOT
             arch_envio = ""
            strtext = ""
        barra2

Case ls > 1
 ncar = 1
 Do
 contador2 = contador2 + 1
  car_envio = Mid(strtext, ncar, 256)
   ncar = ncar + 256
    arch_envio = arch_envio + car_envio
     chksumfin = 0
      BCC_JOB_texto
      MSComm1.Output = Chr$(2) + arch_envio + Chr$(23) + Chr$(bcc1) + Chr$(BCC2)
     tiempo_dialogo
    respuesta = MSComm1.Input
   arch_envio = ""
  Ret = DoEvents
Loop Until contador2 = paquetes

 arch_envio = ""
 contador2 = 0

car_envio = Mid(strtext, ncar, resto)
arch_envio = arch_envio + car_envio
chksumfin = 1
BCC_JOB_texto
MSComm1.Output = Chr$(2) + arch_envio + Chr$(3) + Chr$(bcc1) + Chr$(BCC2)
tiempo_dialogo
respuesta = MSComm1.Input
MSComm1.Output = Chr$(4)              'Envia EOT
tiempo_dialogo
respuesta = MSComm1.Input           'Lee ENQ
MSComm1.Output = Chr$(16) & Chr$(48)     'Envia ACK0
tiempo
motodata = MSComm1.Input              'Se lee lo que hay en el buffer
MSComm1.Output = Chr$(16) & Chr$(49)     'Responde ACK1
tiempo_dialogo
respuesta = MSComm1.Input
strtext = ""
arch_envio = ""
barra2
```

```
End Select
End Sub
```

En este capítulo se presentó la parte técnica de la interfaz. En los anexos de este documento se podrán ver los programas con más detalle con sus respectivas ventanillas de diálogo.
El programa cuenta con una ayuda que se puede usar cuando se está usando el paquete. Esta ayuda contiene los aspectos básicos de la interfaz a la que se puede accesar cuando el usuario lo requiera.
También se puede observar que el ambiente de programación del robot se hace más claro y que la necesidad de conocer con exactitud el lenguaje detrás de la programación del robot no es necesario para usuarios sin experiencia en el manejo de la máquina.

En el siguiente capítulo se verán los resultados del uso experimental de la interfaz con usuarios sin experiencia en el uso del robot, aplicando la interfaz a elaborar la rutina del ensamble de una caja con tapa.

CAPÍTULO 5 APLICACIÓN DE LA INTERFAZ MOTOMAN

5.1 INTRODUCCIÓN

En una celda de manufactura integrada los brazos mecánicos, cumplen versátilmente con las funciones de transporte, manipulación y ensamble de componentes. [9]
De manera particular en el Laboratorio De Sistemas Integrados De manufactura, el sistema del robot Motoman se encarga de ensamblar un producto que consta de una caja con dos engranes y una tapa. La celda cuenta con una fresadora y una caja para los engranes que se maquina en la fresadora. El ensamble de la caja de engranes es solo una de las múltiples operaciones que se pueden llevar a cabo con el robot Motoman.

Actualmente la programación para está práctica se lleva a cabo de manera directa con el controlador manual y con subrutinas sobre las que se puede construir un nuevo programa, el procedimiento es algo tardado por que la operación desde el controlador manual es complicada.
Siguiendo con el desarrollo del propósito de está tesis que es elaborar una interfaz que facilite a nuevos usuarios el aprendizaje en el Laboratorio de Sistemas integrados de Manufactura se aplicó una prueba a usuarios típicos de este robot para obtener resultados y experiencias acerca del uso de esta interfaz.

5.2 DESARROLLO DE LA PRÁCTICA

En el capítulo 2 se habló un poco de los componentes de la mesa de operaciones del robot Motoman.
Haciendo referencia a los esquemas de la Fig. 2.16 y 2.17 del capítulo 2 de la vista superior de la mesa de ensamble.

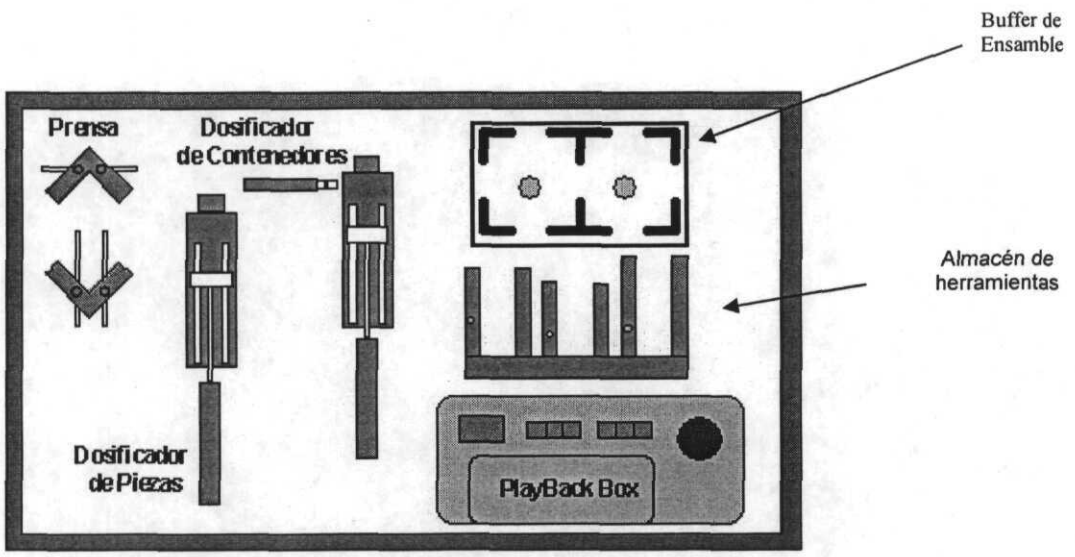


Fig. 5.1 Vista superior de la mesa de trabajo [9]

Como se observa en la figura en el buffer de ensamble es donde se coloca la caja y los engranes. Los pasos para elaborar este ensamble son los siguientes:

1. El robot toma su herramienta del almacén de herramientas.
2. Del buffer de ensamble el robot toma la caja y lo pasa a la prensa donde esta lo sujeta.

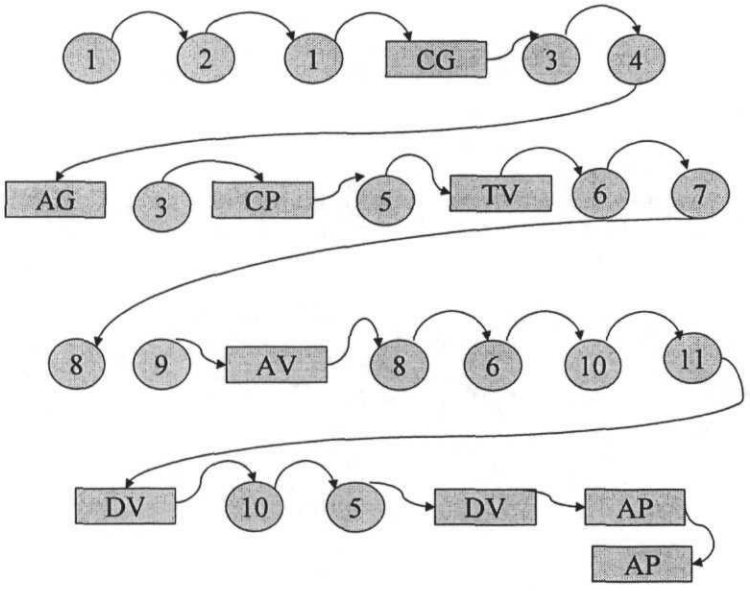
3. Posteriormente se toman del buffer de ensamble él engrane #1 y se coloca en la caja
4. Se toma un engrane #2 y se coloca en la caja.
5. El robot deja el gripper y toma la ventosa.
6. El robot succiona la ventosa de una parte que está a espaldas de el. Esta parte es la tapa de la caja de engranes. Coloca la tapa sobre la caja.
7. Deja la ventosa y toma el desarmador magnético.
8. Se abren las mordazas y el producto queda ensamblado.

Una vez que se conoce la rutina que se quiere realizar con el robot entonces se procede con los pasos necesarios para su programación. Estos pasos se muestran a continuación.

A. Hacer diagrama de puntos
B. Elaborar programa en el editor Motoman de la interfaz.
C. Subir programa al robot
D. Grabar puntos
E. Ejecutar el programa y ejecutar.

5.2.1 Diagrama de puntos

En la siguiente figura se muestra el diagrama de puntos utilizado para el desarrollo de la práctica de acuerdo a los pasos mostrados anteriormente.



La nomenclatura que describe el diagrama de puntos es la siguiente

1.1 CG:CIERRA GRIPPER
AG:Abre gripper
CP:Cierra prensa
TV:Toma ventosa
AV:Activa ventosa
DEV:Desactiva ventosa
DV:Deja ventosa
AP:Abre prensa

Fig. 5.2 Diagrama de puntos para la rutina

Como parte del diagrama de puntos se elabora un algoritmo en el que se muestra la secuencia de actividades del diagrama en forma escrita. En la figura 5.3 se muestra el diagrama del algoritmo. En este diagrama se presenta la ubicación de los puntos en el espacio donde actúa el robot. Este diagrama es también para cuando no se usa el software y se programa de manera convencional.

a) Inicio
b) Punto 1 Arriba de caja
c) Punto 2 Para tomar caja
d) Cierra el gripper
e) Punto 1 sube con caja
f) Punto 3 Arriba de mordazas con caja.
g) Punto 4 para dejar caja
h) Abre Gripper
i) Punto 3
j) Cierra prensa
k) Punto 5 (De seguridad o home)
l) Toma Ventosa
m) Punto 6 de partida para mov. Joint
n) Punto 7 de llegada para mov. Joint.
o) Punto 8 arriba de tapa
p) Punto 9 Baja para tomar tapa
q) Punto 8 Sube con tapa
r) Punto 6 Final de trayectoria Joint
s) Punto 10 arriba de caja para colocar tapa
t) Punto 11 Baja tapa
u) Desactiva Ventosa
v) Punto 10 Sube sin tapa
w) Punto 5 Punto de seguridad
x) Deja ventosa
y) Abre prensa
z) Final

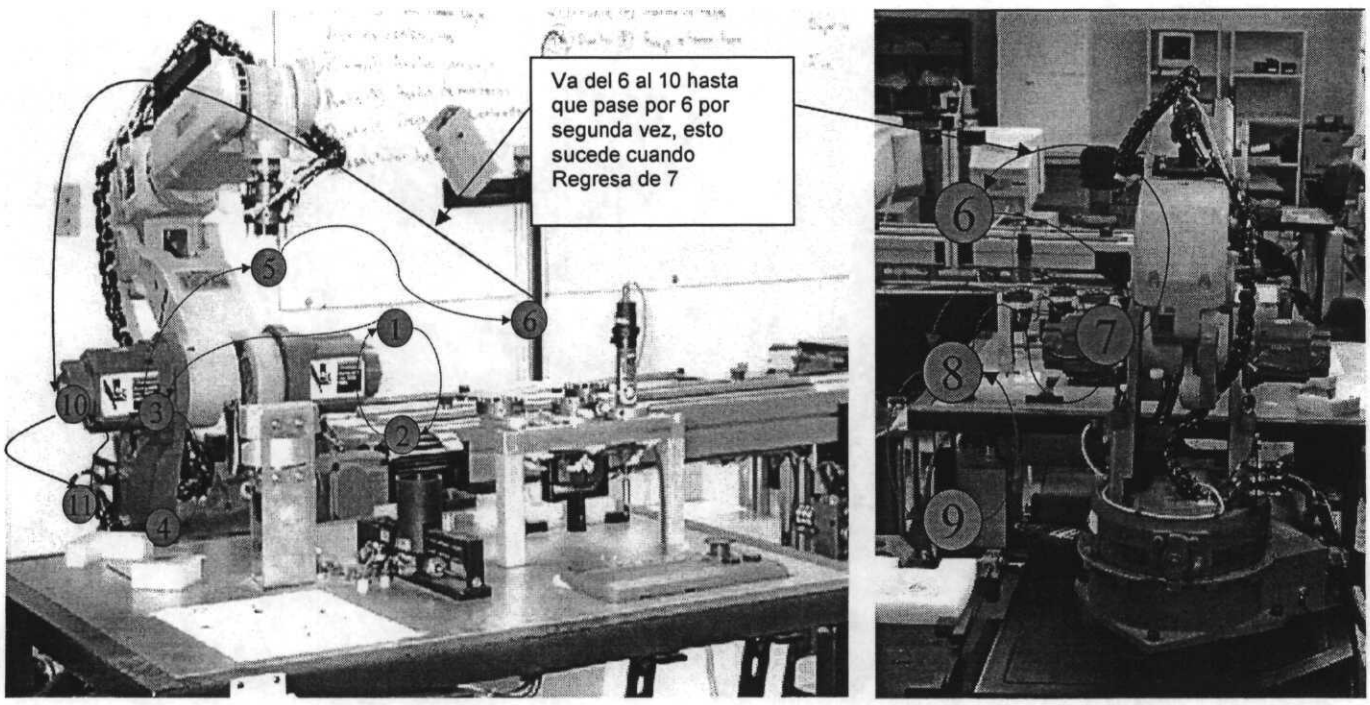


Fig. 5.3 Esquema de puntos de la rutina

5.3 CREACIÓN DEL PROGRAMA.

Como se indica en el inciso B del punto 5. 2 se procede a crear el programa en el editor Motoman de la interfaz. Los pasos para llevarlo a cabo se muestran a continuación.

1. Correr el programa mototalk
2. Oprimir el botón "Editor"
3. Oprimir el botón "encabezado" y colocar los siguientes datos:

   Nombre del programa = engranes
4. Oprima el botón "INICIO"
5. Oprima el botón "TOMAR GRIPPER"
6. Siguiendo el diagrama de puntos y teniendo en cuenta el tipo de operación oprima el botón MOVE y seleccione el número de acuerdo al diagrama, también seleccione la velocidad correspondiente. Todos los movimientos se harán en modo lineal menos el movimiento del punto 15 al 16 que será en modo joint.
7. Al terminar el programa se oprime el botón "FIN".
8. El programa se debe de guardar con el mismo nombre con que se le llamó en el encabezado. Se generará un archivo con extensión tipo JBI.
9. Después desde el panel de control se procede a enviar el programa al robot. Para esto presione el botón enviar y seleccione el programa que va a enviar. Se considera que el programa se envió exitosamente si no hay mensajes de error y si el programa aparece en el menú de programas del controlador.
10. Lleve el robot por los puntos que ejecutará en el programa y guarde desde la interfaz oprimiendo el botón "GUARDAR PUNTO" en la plantilla de puntos. Se abrirá una ventanilla para asignar un numero de punto a la posición seleccionada. También se deberán ejecutar algunas rutinas necesarias para grabar los puntos algunas como tomar herramienta y cerrar mordazas.
11. Al terminar de guardar los puntos se pueden ejecutar de uno en uno para comprobar las posiciones.
12. Después se procede a que el robot ejecute el programa completo presionando el botón de "Correr programa".

5.3.1 Jerarquía de comunicación

Para enviar comandos al robot se mencionó en la sección que hay 3 alternativas: El controlador manual, la caja de ejecución PLAYBACKBOX y la computadora. Solo depende a cuál de estas se le dé el mando para hacerlo.

5.3.2 Computadora y caja de ejecución.

Para ejecutar movimientos automáticos del robot desde la computadora es necesario que los botones en el controlador estén en orden. El botón "ENEABLE" del controlador manual deshabilitado y el botón "PLAY" en el PLAYBACKBOX.
Al terminar de guardar posiciones el programa se puede mandar llamar de este controlador las coordenadas que se almacenaron para tenerse en registro o para futuros programas.

Controlador remoto.
Para que el controlador remoto tenga el mando es necesario activar en la caja de ejecución el botón "TEACH", activar los servomotores y en el controlador remoto activar el botón ENEABLE.

5.3.3 Variables de posición.

Las posiciones de los ejes del robot se almacenan en variables, las posiciones que se almacenan corresponden a los 6 grados de libertad del robot. Las variables son del tipo "C" o del tipo "P".

Variables tipo C
Guardan posiciones relacionadas a un programa. No pueden ser usadas por un programa diferente. Se les llama variables de posiciónlocales. No tienen mucha flexibilidad para ser editadas y solo se pueden crear desde el programador manual del robot y solo se pueden llamar una vez dentro del cuerpo de un programa.

Variables tipo P
Estas variables son llamadas globales ya que pueden ser usadas por cualquier programa. Almacenan la misma información que una del tipo C pero estas se pueden cargar fuera del contexto del programa que se van a usar. Estas variables pueden ser llamadas desde cualquier programa. Por su flexibilidad se decidió usar estas para el trabajo normal del software con el robot y para economizar tiempo de operación.

5.4 RESULTADOS DE LA APLICACIÓN DE LA PRÁCTICA

Las pruebas se hicieron bajo las condiciones en las que normalmente los usuarios del laboratorio de Sistemas Integrados de Manufactura hacen uso de la interfaz.
Se cronometró el tiempo que le toma a una persona construir un programa con el editor de la interfaz. Esto corresponde al inciso B del 5.2. Después se cronometró la enseñanza de puntos que es lo correspondiente al inciso D. Se tomaron 20 muestras de ambos casos.

Prueba de tiempo total usando método convencional
Para el caso de la prueba del modo convencional era necesario contaran con participantes que estuvieran capacitados en el manejo del método convencional del robot con el fin de poder cronometrar la programación de la tarea completa en el tiempo disponible de los participantes. Se logró hacer esta prueba completa con 4 participantes con experiencia previa. Por limitantes del tiempo de los participantes muestreados para el modo interfaz no fue posible darles capacitación para hacer la prueba en el formato convencional.

Prueba de tiempo total usando Interfaz Motoman
Para la muestra en la modalidad interfaz no era necesario contar con experiencia previa en el uso del robot. Solo fue necesario el apoyo de un instructor para dudas de carácter técnico en el manejo del brazo mecánico. Para esta prueba se logró hacer un muestreo de 20 participantes. La prueba consistía de 2 pasos; 1) Elaborar el programa escrito y 2) La enseñanza de puntos. El tiempo total que le tomó alos participantes elaborar esta operación se cronometró y se muestra en la tabla 5.

A continuación se muestra la tabla de resultados.

Prueba	Tiempo total / interfaz	Total/Convencional
1	24.41	
2	24	
3	19.22	
4	23.11	
5	19.29	
6	18.35	
7	18.05	27.35
8	17.36	
9	23.28	
10	15.41	
11	18.2	
12	15.28	
13	22.26	33.43
14	16.33	
15	25.08	
16	19.29	
17	30.07	45.2
18	11.24	17.43
19	15.05	
20	12.13	

Tabla 5.1 Tiempos de programación en minutos usando la interfaz y de forma convencional

5.5 CONCLUSIONES DE LA PRÁCTICA

5.5.1 Tiempos de programación

Con los datos obtenidos para los participantes que completaron la prueba en las 2 modalidades se observó que el tiempo invertido en la programación del robot usando la interfaz es menor que el que se obtuvo de forma convencional.
Se analizarán las diferencias: En el caso 7 la diferencia es **9.3 minutos**, esto equivale a una disminución del **34.18%.** En el caso 13 la diferencia es **11.17 minutos,** esto equivale al **33.41%.** En la muestra 17 la diferencia es de **15.13 minutos** que equivale al **33.47%.** En la muestra 18 la diferencia es de **6.19** minutos que es un **35.51%.** Con estos resultados podemos concluir que la interfaz de Motoman reduce el tiempo necesario para la programación y enseñanza de puntos.

5.5.2 Nivel de uso de la interfaz

Criterios de ergonomía de las pantallas de diálogo

Generalmente la evaluación de un software se basa en la evaluación de áreas como la facilidad de uso del paquete, la rapidez para entenderlo, eficiencia, número de errores y ambiente de trabajo.
Estos criterios solo se pueden evaluar de manera efectiva escuchando la opinión de varios usuarios finales del paquete. Esta evaluación califica la usabilidad inherente de la interfaz. [14]

Las estrategias para incrementar la usabilidad inherente se dividen en 3:

1. La eficiencia cognitiva
2. La eficiencia operacional
3. La eficiencia de seguridad

Eficiencia cognitiva
Se enfoca en reducir la carga cognitiva del usuario e incluye la secuencia del vistazo, la estrategia de la familiaridad y la estrategia del agrupamiento. La secuencia del vistazo es similar a la curva del vistazo la cuál es conocida en el campo de las artes visuales. En la situación que se presenta cuando el usuario lee un documento o está observando la pantalla de una computadora se supone, por información de la curva del vistazo, que el usuario mueve sus ojos generalmente de desde la esquina superior izquierda de la pantalla a la esquina inferior derecha. [14]

La estrategia de la familiaridad significa que los objetos de la pantalla se agrupen de una manera en que el usuario esté acostumbrado a verlos. Un ejemplo se podría aplicar cuando se está decidiendo colocar una pantalla que contiene un panel numérico con la configuración telefónica o con la configuración de una calculadora. En este caso la mayoría de los usuarios tendrían familiaridad con el formato telefónico. [14]

La estrategia del agrupamiento viene de la ley del agrupamiento perceptual de la psicología de Gestalt. Esta estrategia recomienda que los botones con funciones similares deben agruparse usando la ley de la proximidad, la ley de la similaridad, ETC. [14]

Eficiencia operacional
Se enfoca en mejorar la forma de operar la interfaz acomodando los botones en el orden de operación y usando la información de los patrones de uso dominantes. [14]

Eficiencia de seguridad
Esta estrategia sugiere que los botones para cancelar operaciones o salir de pantallas sean colocados fuera del flujo principal de trabajo, esto para evitar presionarlos por error. [14]
Los métodos de evaluación inherente como ya se mencionó solo se aplican una vez que la interfaz está terminada. Lo ideal es contar con un método de calificación de la interfaz válido durante su construcción o en la etapa de mejora.

Se encontró un método para evaluar la interfaz de acuerdo los principios básicos con los que debe de contar una interfaz. Este método es propuesto por Richrard Holcomb y Alan L. Tharp bajo el título de "Un modelo amalgamado de usabilidad". La propuesta del método es que la interfaz se califique de acuerdo a la puntuación que reciba en cada una seria de criterios de análisis. La jerarquía de las categorías se ponderó de acuerdo a l impacto que esta tiene en el desempeño de la interfaz y en su interacción con el usuario. [15]

Los criterios de evaluación son:

Funcional
Consistente
Intuitivo
Memorización mínima
Retroalimentación
Ayuda al usuario
Nivel de control del usuario

Funcional: Se refiere a que la interfaz cumpla el propósito para el que fue creado en un tiempo aceptable y de una manera confiable. Ponderación 25.[15]

Consistente: Se refiere a que la presentación la interfaz debe mostrar uniformidad en sus comandos, en su lenguaje con el usuario. Esta es quizás una de las áreas más débiles de los paquetes computacionales. Ponderación 20. [15]

Intuitivo: Se refiere a la capacidad del programa para comportarse de la manera que el usuario espera, usando un lenguaje habitual y una estructura fácil de entender por la mayoría de los usuarios. Ponderación 14. [15]

Retroalimentación: Evalúa la calidad de la información que la interfaz envía al usuario con el fin de mantener informado de lo que sucede en la interacción. Esto incluye mensajes de verificación y advertencias. Ponderación 12

Memorización: Mide la capacidad de la interfaz para manejar toda la información que requiere para su función, evitando así al usuario memorizar información que tendrá que ingresar posteriormente. Ponderación 12

Ayuda de usuario: Mide la calidad de la información que la ayuda en línea de la interfaz brinda al usuario. Esta ayuda debe de ser la necesaria para la correcta operación de la interfaz..
Ponderación 10. [15]

Nivel de control de usuario: Evalúa la flexibilidad de control que la interfaz brinda al usuario como por ejemplo la omisión de menús conocidos, atajos a funciones, omitir pasos, entre otros. Ponderación 7

Evaluación del nivel de uso de la interfaz Motoman

La usabilidad se define como la facilidad de aprender y comprender un sistema [15].
La evaluación de la interfaz Motoman de acuerdo a como propone Holcomb y Thrap arrojó los siguientes resultados:

Funcionalidad=20/25 Ya que la interfaz no cubre el 100% de los comandos posibles propuestos por el manual de comunicaciones del robot Motoman y las rutinas de recepción de comandos son rígidas lo cual hace necesario que en un futuro se tengan que replantear algunas rutinas para poder establecer una comunicación digital que haga más flexible las rutinas de programación.

Consistente=18/20 Se le asigna este puntaje porque se mantiene consistencia en l a programación. El ambiente que se proporciona al usuario es sólido. No se usan los mismos botones para instrucciones o comandos distintos. Los botones se han agrupado conceptualmente para familiarizar al usuario rápidamente.

Intuitivo=16/16 El acomodo de los botones se ha hecho con el criterio de guiar al usuario de manera natural por el flujo de programación del robot. Es decir que en la trayectoria del primer vistazo se encuentra la secuencia de botones en el orden en que se deben de usar. Las instrucciones son fáciles de entender

Retroalimentación= 1 0/12 E l p rograma brinda al usuario i nformación mientras se está operando. E sta información se da en ventanas de diálogo para guiar al usuario en la manera en que debe de darse de alta la información que el programa requiere para llevar a cabo determinada operación. Si embargo, el programa no tiene esta misma s ensibilidad cuando se envían o reciben programas. Este aspecto no causa mayores problemas al usuario, pero le deja la responsabilidad de verificar que las comunicaciones de programas se hayan completando.

Memorización mínima=10/12 Este puntaje se asigna por el hecho de que en 2 casos se requiere que el usuario recuerde información que ya ingresó. Esto es en el caso de guardar un programa después de haber sido creado. El usuario debe de asignar un nombre al programa que está creando, este nombre lo debe de escribir de la misma manera cuando el programa pide el nombre bajo el que se guardará en la memoria de la computadora.

Ayuda a l usuario=8/10 L a a yuda de usuario Motoman cubre de manera sencilla y concisa l os puntos necesarios para dar ayuda al usuario en dudas básicas de programación. Por otro lado la ayuda no es sensible al contexto de trabajo por lo que deja al usuario la responsabilidad de usar la información que necesita.

Nivel de control=7/7 El software brinda al usuario completo control de la interfaz salvo en situaciones de precaución que son en las que se involucra el movimiento del robot, por lo que hay ventanas de dialogo que detienen el flujo de la operación y no pueden ser omitidas.

Sumando los puntajes individuales se obtiene un resultado de 89.
Identificando como áreas de oportunidad la retroalimentación al usuario y la memorización del usuario.
El usuario en estas áreas pudiese cometer errores del tipo diálogo con la interfaz y con la memorización.
Si embargo las áreas de mayor ponderación se muestran sólidas como lo son la funcionalidad y la consistencia del programa. Esto significa que el programa cumple con el final para el que fue hecho de manera consistente.

5.5.3 Comparación interfaz Vs. Controlador manual.

En la siguiente figura se observa el controlador manual del robot Motoman y algunas de los motivos principales que generan confusión en el usuario y provocan pérdidas de tiempo.

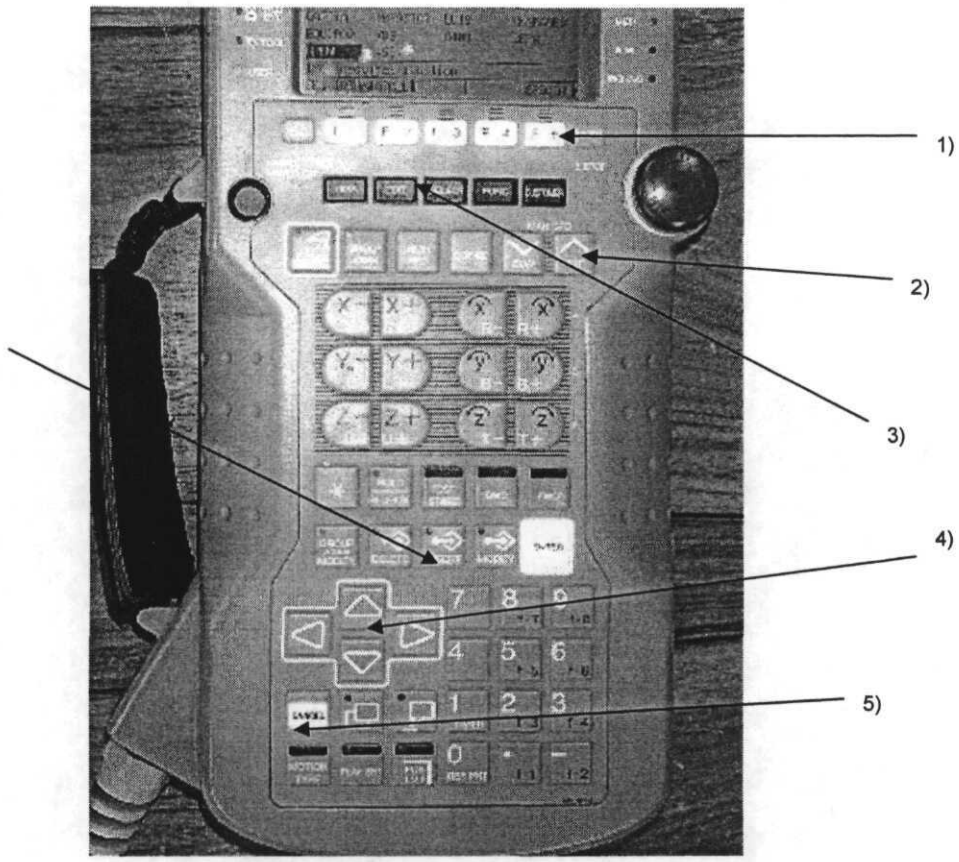


Fig. 5.4 Controlador manual del robot Motoman

1) Botones para accesar funciones en pantalla. Las funciones en pantalla no son fijas y cambian por otras funciones o por alarmas del sistema Motoman.

2) Botones de velocidad se confunden con botones para movimiento de cursor en pantalla.

3) Botones para desplegar menús de funciones.

4) Botones para edición se confunden con botones para movimiento del controlador.

5) Botón para cancelar errores se confunde con cancelación de alarmas. Para cancelar alarmas se usa botones para accesar funciones.

6) Los botones para editar llevan a confusiones en cuanto a que sucederá en pantalla al usarlos ya que los títulos que tienen no son muy claros.

En la siguiente figura se observa el editor Mototalk que pretende solucionar los problemas observados en el controlador manual.

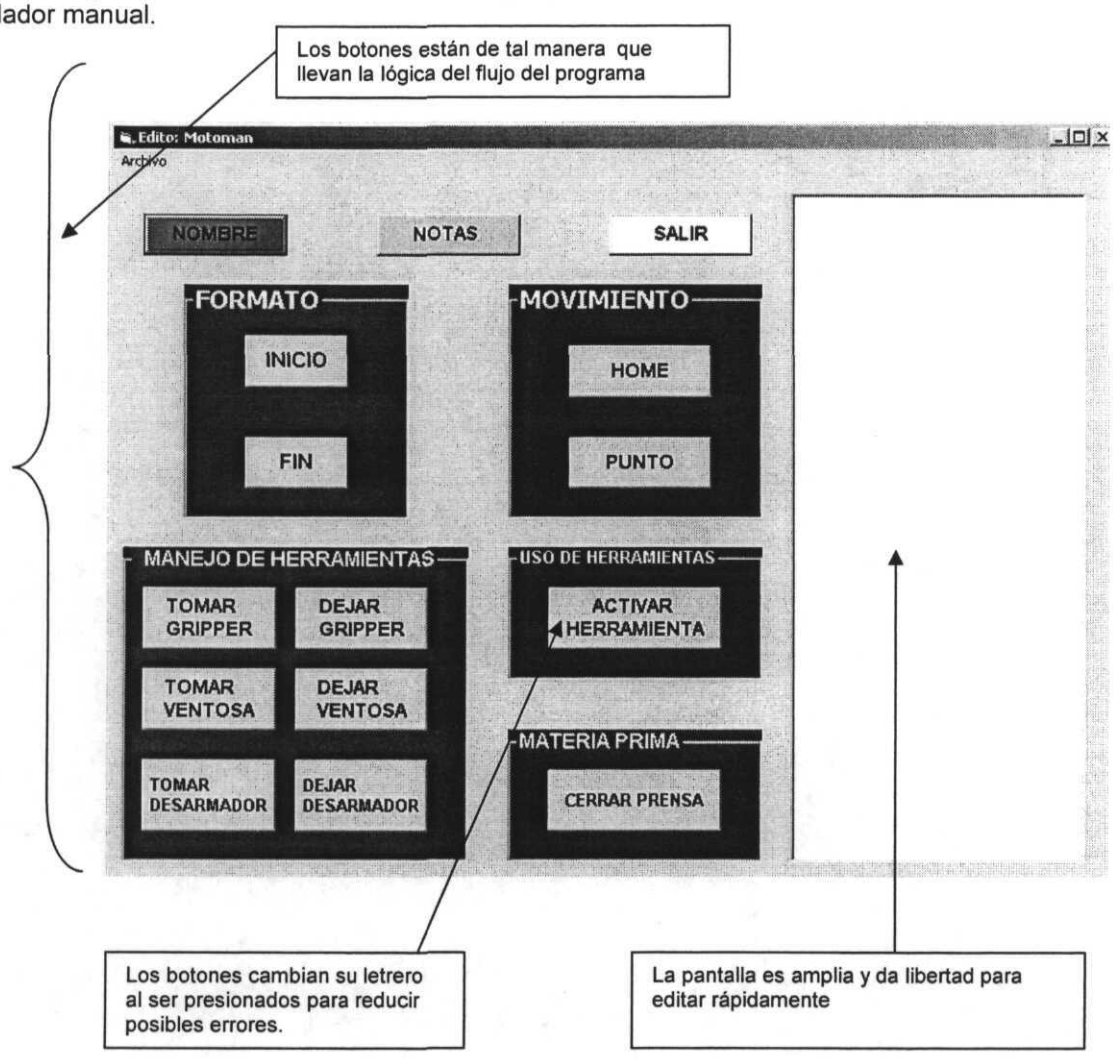


Fig. 5.5 Controlador manual del robot Motoman

La programación de funciones que requieren de varios datos para activarse, ahora se facilita, ya que al presionar un botón se abren ventanas de dialogo que piden todos los datos necesarios para dicha función. Como ejemplo la función de mover a un punto que se observa en la figura 5.5. En esta misma ventana de dialogo dan de alta datos de tipo de movimiento, número de punto y velocidad. En el controlador manual son necesarias 4 operaciones; 1) Estar en modo de edición, 2) Presionar el botón play speed y presionar el botón del tipo de movimiento, por último 4) Presionar ENTER. Todas estas funciones en menús diferentes.

En la figura 5.6 se muestra la ventana de dialogo para dar entrada a los datos que el robot necesita para ejecutar un punto. En la sección de Motion type se revisa bajo que tipo de movimiento se llevará a cabo el movimiento al punto.

En la parte del número de punto se coloca cualquier número del 1-99 que haya sido previamente guardado. En la parte de velocidad se selecciona del 1-100% para movimientos tipo Joint y del 1 a 1500 mm/seg para el tipo de movimiento lineal. Al presionar mover, la codificación se imprime en la pantalla del editor, si es que se está trabajando fuera de línea o bien se ejecuta por el robot, si es que se está trabajando en línea.

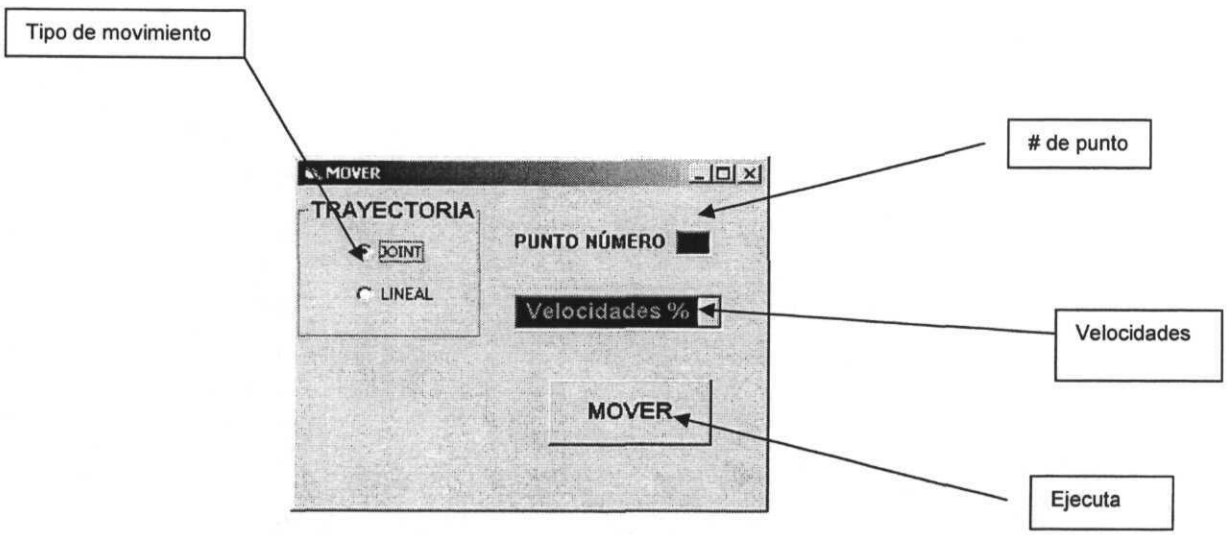


Fig. 5.6 Ventana de dialogo para guardar punto

En el siguiente capítulo se hará mención de las conclusiones más relevantes que ha arrojado esta investigación así como los panoramas para investigaciones futuras en esta rama.

CAPÍTULO 6 CONCLUSIONES

6.1 Conclusiones

No hay duda de que el desarrollo evolutivo de la especie humana está muy ligado con el progreso de sus herramientas y de sus máquinas. [16]
Cuando se habla de términos como aumentar la productividad y el rendimiento significan una mejor administración de los bienes y los recursos. También se refieren a saber utilizar estos recursos de la mejor manera posible.
Generalmente en la industria o en el aula se cuenta con equipo sofisticado para diversos fines y en la mayoría de los casos se conocen las ventajas y cualidades que el equipo brinda para hacer rendir mejor recursos como el tiempo, la energía etc. Pero hacer funcionar estas ventajas requiere de un esfuerzo inicial extra. En el uso de los robots este pensamiento es muy propicio ya que él usarlo incorrectamente constituye una enorme pérdida de tiempo y de dinero.

Un robot se vuelve necesario en situaciones muy específicas cómo condiciones de trabajo poco favorables para un humano, cuando la velocidad y la precisión juegan un papel importante, la automatización entre otras. Por este motivo empresas dedicadas al desarrollo de software invierten en hacer programas que ayuden a que se exploten al máximo las cualidades del equipo que sé este usando y poder aprovecharlas en las áreas ya mencionadas minimizando el esfuerzo inicial de un conocimiento profundo de la máquina.

Los programas mencionados se llaman interfaz y son un intermediario entre el usuario y la máquina. Proveen de un ambiente en que el usuario pueda enviar comandos a la máquina en términos que el usuario entiende y permiten que la máquina reciba estos comandos en términos que la máquina entiende. Estos programas se desarrollan con fines industriales y didácticos.

Para adaptar la interfaz a un nivel de capacitación es necesario diseñarla de una manera que guíe al usuario hacía el objetivo que quiere lograr con su uso. Para esto el lenguaje de comunicación con el usuario debe de ser lo menos técnico posible para agilizar el aprendizaje.

La programación de la interfaz se realizó en 6 meses cumpliendo con la cronología presentada en la propuesta de esta tesis. Entre los retos más importantes se destaca el conocimiento del lenguaje base de comunicación serial del robot en 3 niveles de dificultad. De entrada implicaba conocer un lenguaje nuevo en código ASCII el BSC que había que traducir a números que el software interfaz pudiera emular para intercambiar información con el robot. En este sentido el primer nivel de dificultad fue la parte de envío de comandos, fue también la primera rutina que se desarrolló porque los comandos se manejan como situaciones de programación constantes y esto solo implicaba conocer las restricciones del protocolo. El siguiente nivel de dificultad con respecto al código fue el envío de programas; esto se debió a que esta situación de comunicación es variable y se tenía que crear rutinas en la programación de Visual Basic 6.0 que respondieran a estas situaciones de dialogo variable entre el robot y la computadora.

Por último en este mismo contexto de protocolo de comunicación se encontró la mayor dificultad en el envío n de programas del robot a una computadora personal, por 2 motivos. Para hablar del primero se recuerda que en él envió de programas al robot la computadora lleva mano en la comunicación y las rutinas tienen cierto nivel de variabilidad pero en viceversa, envió de programas del robot a la computadora, la situación adquiere un nivel de dificultad mayor porque el software debe contener una rutina que lo convierta en una especie de oído que capture la información al ritmo del robot. Debido a esta situación la rutina de recepción se creó con un nivel de sensibilidad y variabilidad mayor que las dos anteriores. Un segundo motivo fue que los esquemas del protocolo de comunicación para envió de programas del robot a la computadora presentados en el manual de comunicaciones del robot Motoman había errores en algunos de los caracteres del protocolo lo cuál implicó hacer muchos experimentos de prueba y error para encontrar el motivo por el que al inicio no se llevaba a cabo exitosamente este envío de información.

Otro de los retos importantes fue el conocimiento del software que se utilizó para crear la interfaz el Visual Basic 6.0. El manejo del lenguaje es sencillo en su nivel básico pero no de igual manera para programación avanzada para comunicaciones seriales. En este sentido el lenguaje no es muy robusto y ofrece poca flexibilidad a diferencia del C++. En este caso el tiempo era un factor muy importante que influyó en optar por el Visual Basic 6.0 como el lenguaje base para la comunicación.

Por otra parte también cabe mencionar que los cálculos de los verificadores de error BCC que se mencionan en el capítulo 3 como parte de la explicación del protocolo BCC (BLOCK CHECK SUM CARÁCTER) fueron un punto crítico en el desarrollo del programa. Diferentes autores muestran alternativas para calcularlos, aún los autores del manual de comunicaciones del robot, Sin embargo las propuestas eran con operaciones de números binarios, era necesario encontrar una manera simple en formato decimal que se pudiera emular por parte el software interfaz. Por medio de un artificio matemático se logró construir una rutina sencilla que calculara el BCC.

El diseño de las pantallas de diálogo fue la última etapa de la construcción de la interfaz y no por ser la última etapa carece de importancia y de nivel de dificultad, de hecho tiene la misma importancia que las etapas anteriores, ya es una de las cosas que implicaba llevar a cabo este proyecto era lograrlo todo o nada, puesto que el haber diseñado una interfaz cuya única fortaleza hubiese sido la funcionalidad, no hubiera cumplido con el objetivo de esta tesis ya que como se mencionó en el capítulo anterior la interfaz se evalúa desde varios puntos de vista.

Las etapas más importantes durante la construcción del software se pueden resumir con el siguiente esquema

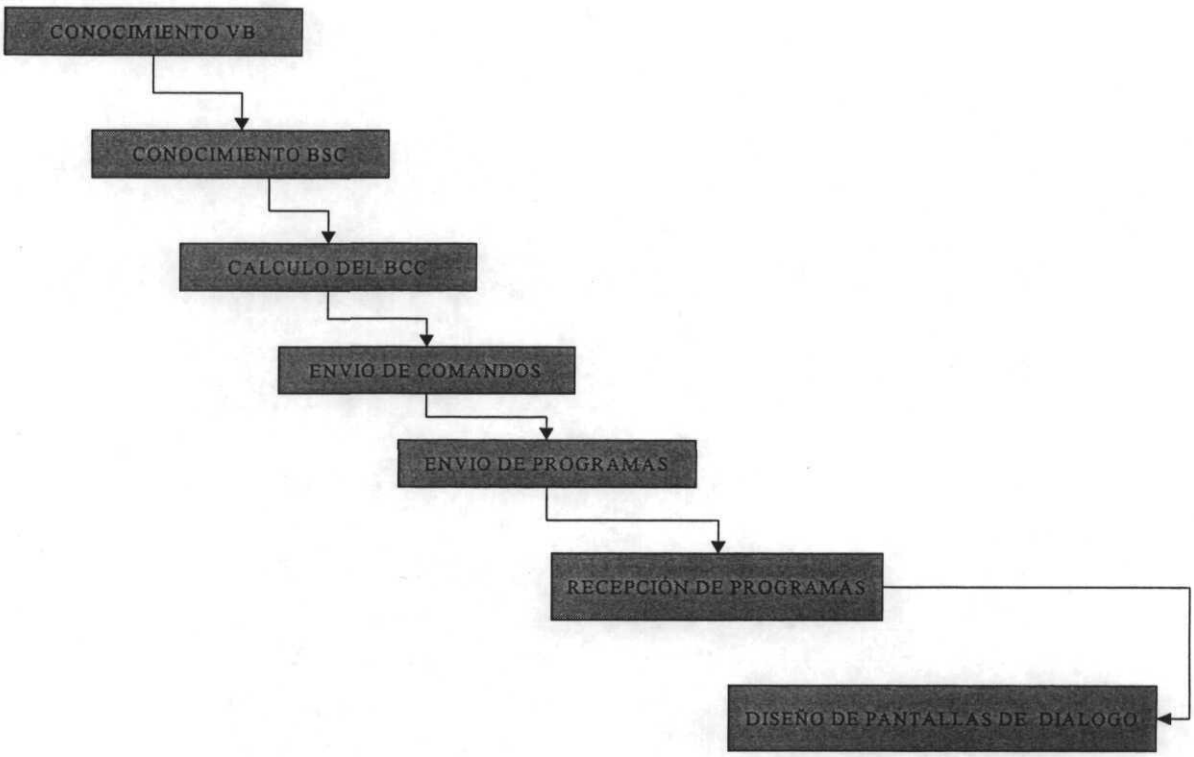


FIG. 6.1 Etapas de construcción de la interfaz

Con esta tesis se demostró que la interfaz funcionó para aprovechar mejor las cualidades del robot Motoman Yasnac modelo K3 y que se optimizó el método de enseñanza para este robot en el laboratorio de sistemas integrados de manufactura. Se comprueba desde el punto de vista de tiempos y de interfaz hombre-máquina que la implementación de la interfaz facilita la operación y la programación del robot por nuevos usuarios tal como se plantea en la hipótesis. La interfaz facilita la labor docente y el aprendizaje.

Con el análisis de usabilidad y el planteamiento de los criterios de diseño se logró solidificar las bases De construcción del panel de control y el editor del programa, cumpliendo el objetivo propuesto al inicio de aprovechar las ventajas de una interfaz en ambiente Windows para beneficio de los usuarios y los instructores del laboratorio de sistemas integrados de manufactura.

6.2 Investigaciones Futuras

En el capítulo 5 se trató una propuesta de evaluación de la interfaz, esta propuesta arrojó resultados que reflejan las fortalezas y las debilidades de la interfaz. Investigaciones posteriores apuntan a trabajar en estas áreas con el fin de ir mejorando esta interfaz, eso como proceso natural del desarrollo de la misma, ya que es poco probable que en el desarrollo de software los diseñadores logren agotar las áreas de oportunidad en su primera versión del software. Mototalk 1.0 deja espacio a mejoras para ser construidas sobre esta misma base.

Una de las áreas principales de investigación futura se debería de enfocar en la flexibilidad de comunicación del software con el robot.. Hay rutinas del software que no tiene porque ser diseñadas con capacidad de respuesta a condiciones variables. Estas rutinas son las de envío de comandos. Sin embargo, las rutinas de envío de programas, recepción de programas y estatus del robot pudieran y debieran tener mayor flexibilidad. Estas rutinas manejan información que cambia según factores de operación del robot, de comportamiento del robot, entre otras. Esto involucraría inicialmente lograr una comunicación de lazo cerrado con el robot al nivel de señales digitales y de comunicación con el PLC de la celda de manufactura donde se encuentra el robot. Esto repercutiría en un aumento en el nivel de retroalimentación entre el usuario y la interfaz.
En la escritura del programa se proporcionan notas técnicas que ayudarán a construir sobre esta misma base y lograr las mejoras propuestas.

Lo anterior permitirá la creación de un panel de monitoreo de señales que representara la mesa de trabajo donde se encuentra el robot y del estado de operación de las máquinas que lo rodean en ese mismo instante permitiendo al usuario tener desde un lugar remoto la idea exacta de lo que ocurre en el lugar de trabajo.

Otra propuesta de investigación es por la rama de la simulación y la programación fuera de línea. Completar esta interfaz con la capacidad de programar completamente al robot fuera de línea con el objetivo de crear programas completos sin contar con la presencia del robot ayudaría de manera significativa la labor de capacitación. También impactaría de manera significativa en el cuidado del equipo. Las sesiones donde los usuarios no tienen familiaridad ni experiencia con el robot se pudiesen cubrir con un simulador que recreara las situaciones reales bajo las cuales el robot se comportaría en respuesta a la operación del usuario.

RESUMEN

La operación de los robots es muy variada y depende de las necesidades de la empresa. Generalmente se usan en tareas que no se pueden realizar por un humano por motivos de condiciones ambientales, precisión, velocidad entre otras. Los robots han hecho su aparición por generaciones satisfaciendo las demandas de la manufactura y la automatización. Las capacidades de los robots han ido aumentando al paso del tiempo. Estos aumentos no solo se deben a las propiedades estructurales del robot sino también a la versatilidad de la interfaz con el usuario. Empresas dedicadas al desarrollo de software interfaz han sacado al mercado interfaces para los robots más usados en la industria que facilitan la interacción del usuario con el robot. También existen en el mercado programas de simulación que se usan para programar robots sin necesidad de estar en el lugar donde se encuentra la máquina.

La aplicación didáctica de los robots también ha aumentado considerablemente en los sistemas de educación moderna. Se busca que alumnos que cursan carreras relacionadas con la manufactura tengan contacto cercano con las máquinas utilizadas en esta rama, entre ellas, los robots. Este acercamiento usuario-máquina debe de ser lo más pedagógico posible, para hacer posible que en poco tiempo el alumno tenga una noción clara y concisa del funcionamiento de los diferentes tipos de robots e identifique su aplicación en la industria. Por este motivo se identificó la necesidad de crear una interfaz didáctica para ser utilizada en el Laboratorio De Sistemas Integrados De manufactura con el robot Motoman modelo Yasnac K3. Este robot se encuentra en este laboratorio con fines didácticos y es utilizado por los alumnos de la carrera de ingeniería industrial. El Robot es una de las estaciones de trabajo que forma parte de una celda flexible de manufactura localizada en este mismo laboratorio.

El proyecto de generar una interfaz para el robot ya constituía un enorme ahorro de tiempo en lo que se refiere en programar el robot. La interfaz con la que contaba es la proporcionada por el fabricante en su controlador manual, el cuál es una herramienta muy técnica como para usarse directamente para la capacitación del robot y cubrir el material en el tiempo requerido. Pero para aprovechar este ahorro de tiempo se decidió darle a la interfaz un enfoque didáctico para facilitar la labor de capacitación y aprendizaje.

La interfaz cuenta con una pantalla principal en la que se muestran botones a los comandos principales del robot como por ejemplo tomar las herramientas, enseñanza de posiciones, movimiento a posiciones predeterminadas, envío y recepción de programas, manejo de la memoria virtual del robot, entre otros. Además cuenta con ventanas de dialogo que agilizan la captura de datos usados para algunos de los comandos de programación que así lo requieren.

El uso de la interfaz se hizo con alumnos y con instructores del Laboratorio de Sistemas Integrados De Manufactura y se obtuvieron datos de los tiempos que tardaban en programar al robot haciendo uso de la interfaz. El número de datos de tiempos de programación en formato convencional era limitado, por lo que se hizo una comparación sencilla de estos datos con los obtenidos en la nueva manera. También se hizo una comparación de la interfaz para Windows Vs. La interfaz convencional analizando los problemas que se encontraban con el uso del controlador manual y las soluciones que plantea la interfaz.

BIBLIOGRAFÍA

[1] http://www.dei.uc.edu.py/tai99/robotica/introduc.html

[2] Automatización Y Robótica. Obtenido en Febrero del 2003,en http://www.geocities.com/soldadura17/rob/autorob.htm

[3] Clasificación De Los Robots. Obtenido en Febrero del 2003, en http://www.geocities.com/soldadura17/rob/clasrob.htm

[4] Daimler Chrysler JR Program-Robot Swap al SHAP, Obtenido en Marzo del 2003,en: http://www.robotics.org/public/articles/amt_shap_project_02251.pdf

[5] Axpe, Cabanes, Axpe. Control y programación de Robots. Obtenido en Marzo del 2003, en la Escuela Superior De Ingenieros De Bilbao en: http://www.disa.bi.ehu.es/spanish/ftp/material_asignaturas/Control_Programacion_Robots/Teor%EDa/Tema1_Introduccion.pdf

[6] Groover, P., Mikell. , Weiss, Mitchell. ,Naguel, N., Roger, Odrey, G., Nicholas. Industrial Robotics Thecnology, Programming and Applications, McGraw-Hill, Inc. , 1986.

[7] Types Of Robots. Obtenido en Marzo del 2003, en la NASA en: http://prime.jsc.nasa.gov/ROV/types.html

[8] Reg, A. James. Introduction To Robotics In CIM Systems, Prentice Hall 2000.

[9] Manual del laboratorio de sistemas integrados de manufactura.

[10] Manual de comunicaciones del robot Motoman

[11] Brown, Brian (1995-2000. Data Comunicatios. Obtenido en Febrero del 2003, del instituto central de tecnología de auto estudio de cursos de TI en: http://goforit.unk.edu/datacomm/dc_012.htm

[12] Alhoniemi, Erno. Error Detection And Control In Data Transfer. Obtenido en Febrero del 2003, en la página de La Universidad de Tecnología de Helsinki en: http://www.tml.hut.fi/Studies/Tik-110.300/1998/Essays/error_detection.html

[13] Operator Screen (HMI) Design Guidelines. Obtenido el 23 de mayo del 2003, en la página de Hexatec en: http://www.hexatec.com/Documents/Operator% 20screen%20Design.PDF.

[14] *Kurosu, M.; Kashimura, K.;*
Systems, Man and Cybernetics, 1995. 'Intelligent Systems for the 21st Century'. , IEEE International Conference on [Versión electrónica], Volume: 2, 22-25 Oct 1995
Page(s): 1509 -1514 vol.2

[15] *Holcomb, R.; Tharp, A.L.;*
Computer Software and Applications Conference, 1989. COMPSAC 89. , Proceedings of the 13th Annual International[Versión electrónica], 20-22 Sep.b 1989
Page(s): 559 -566

[16] Moriello, Sergio, Evolución Sinérgica hombre-máquina. Obtenido en Abril del 2003, de la página del Instituto De investigación sobre La Evolución Humana, A.C. en: http://www.iieh.com/doc/doc200302030300.html

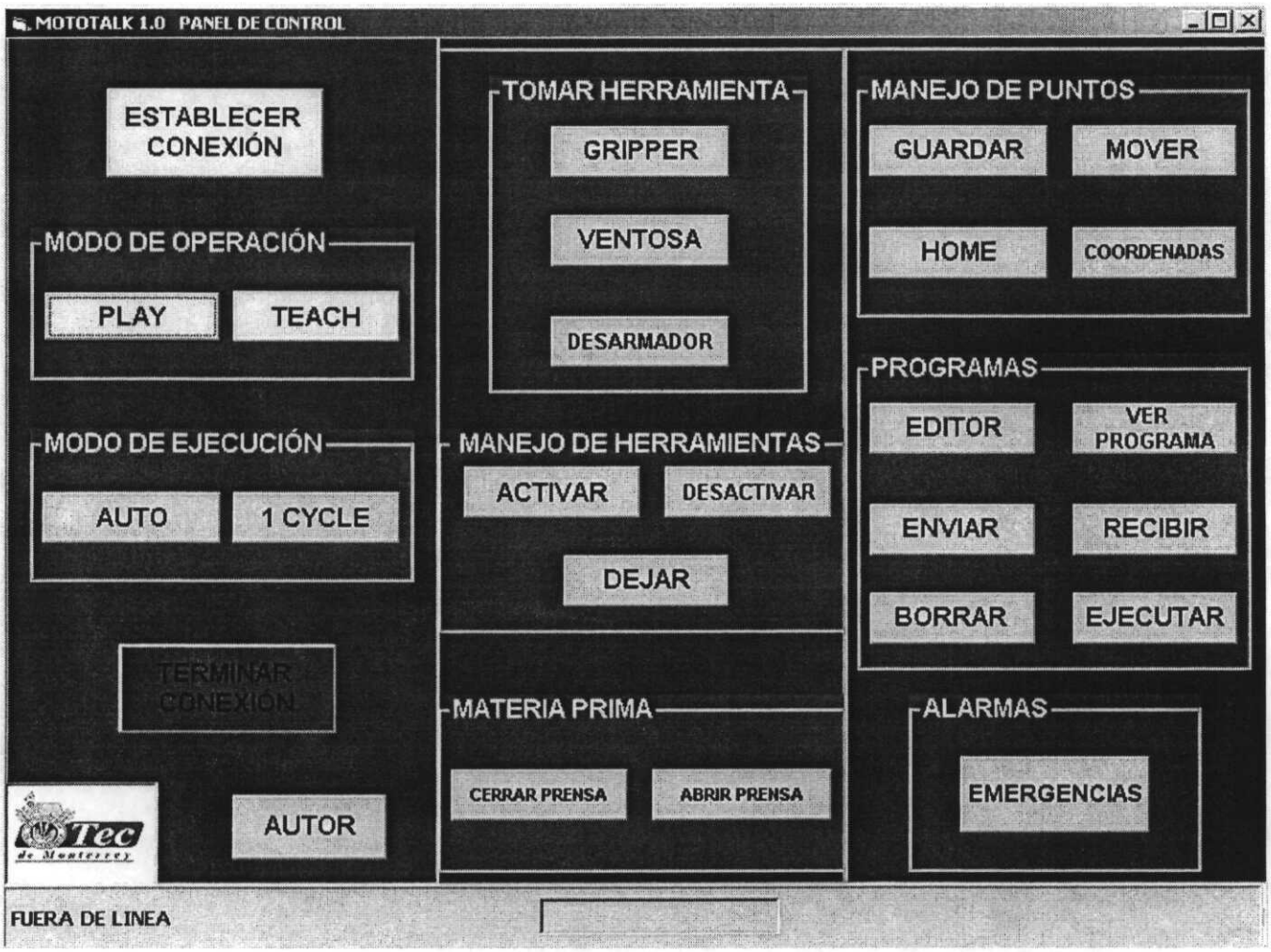
[17] Basic Data Communication Thecnology. Obtenido en Febrero del 2003 en la página de la Universidad De Sacramento en: http://216.239.51.100/search?q=cache:dsteBYPvRngC:www.csus.edu/indiv/t/tsain/m115s99/goldman3.ppt+BCC+checksum+example+&hl=en&ie=UTF-8

[18] BiSync, BSC. Obtenido en Febrero del 2003, de Connectivity knowlwdge platform en la página http: //ckp.made-it.com/bisync.html.

[19] Obtenido en Marzo del 2003 la página de Texas A&M University en: http://etidweb.tamu.edu/ftp/entc435/Class%20Notes/Exam%201/BSC-KERMIT.PDF

ANEXOS

ANEXO A
CÓDIGO Y DISEÑO DEL PANEL DE CONTROL DE LA INTERFAZ.



```
'Se declaran variables de uso global
Option Explicit
Dim Counter As Long
Dim Ret
Dim cociente, respuesta, motodata, caracter, data, progra, motodata2 As Variant
Dim punto(1000), TR As String
Dim COMANDO, COMANDO1, arch_envio, alarma, cord, p, jo, mezcla, strtext, nombrearch As
String
Dim BCC, bcc1, BCC2, usuario, tool, noread, INICIO, chksumfin, BORRAR, enqerror, answer,
usarveri, aviso As Long
'Código base de la pantalla inicial
Private Sub Form_Load()
'boton_out
interfaz.MousePointer = 1
StatusBar1.Panels(1).Width = 10000
StatusBar1.Panels(1).TEXT = "FUERA DE LINEA"
ESTADO.Enabled = True
interfaz.Top = True
interfaz.Width = Screen.Width
interfaz.Height = Screen.Height
usarveri = 1
'Aqui se configura el puerto de comunicación de la computadora en el puerto asignado con
COMM1
MSComm1.CommPort = 1
MSComm1.Settings = "9600,E,8,1"
MSComm1.RThreshold = 0          'Se deshabilita la sensibilidad de entrada de caracteres
MSComm1.SThreshold = 0          'Se deshabilita la sensibilidad de envío de caracteres
MSComm1.ParityReplace = "?"
MSComm1.InputLen = 0
MSComm1.DTREnable = True
MSComm1.NullDiscard = False
MSComm1.RTSEnable = True
MSComm1.InBufferSize = 1024
End Sub
```

```
'Solo para envío de comandos
Public Function proto_master2() As Integer
proto_master2 = 1
answer = 1
Abre_puerto                 'Subrutina que abre el puerto de comunicaciones
ENQ
If enqerror = 0 Then        'Función que verifica que haya comunicación con el robot, regresa
un valor si es que se estableció
 COMANDO = COMANDO1
 Mototalk                   'Es el procedimiento de comunicación principal
Else
proto_master2 = 0
Cierra_puerto
Exit Function
End If
End Function
```

```
'Es la subrutina que envia el ENQ inicial
Private Sub ENQ()
```

```
Dim intento, tamaño As Long
Dim car1, car2, buffer  As Variant
Dim data As String
On Error GoTo panic
MSComm1.RThreshold = 1
denuevo:                        'Apartir de aquí reintenta si la comunicación falla
MSComm1.Output = Chr$(5)
Timer4.Enabled = True           'Comienza el tiempo de espera de respuesta
Do
  Ret = DoEvents
    buffer = buffer + MSComm1.Input
     tamaño = Len(buffer)
       If Timer4.Enabled = False Then   'Se debe recibir respuesta antes de que acabe el tiempo
         If mensajes = 1 Then
           GoTo denuevo
             Else
               Cierra_puerto
                 interfaz.MousePointer = 1
                   boton_in
                   Form_Load
                enqerror = 1
           Exit Sub
         End If
       Else
    End If
Loop While tamaño < 2

Do
  Ret = DoEvents
   car1 = CStr(Asc(Mid(buffer, 1)))   'Con esta función se convierte a cadena el número recibido
  data = data + car1                  'Acumula el número recibido
 Loop Until InStr(data, 16)           'Equivale a recibir el ACK
 Do
   Ret = DoEvents
    car2 = CStr(Asc(Mid(buffer, 2)))
  data = data + car2
Loop Until data = 1648                'Equivale a recibir el ACK0

panic:
HANDLER
End Sub
```

```
Private Sub Mototalk()
BCC_GEN                 'Subrutina que genera el BCC para mandar al robot
COMAND                  'Sub rutina que formula el diálogo inicial con motoman
Dialogo                 'Función que es escucha las respuestas del robot y las contesta
End Sub
```

```
'Función que genera el BCC para enviarlo al robot
Private Sub BCC_GEN()
Dim j As Long
j = 1                           'Inicializa el contador
```

```
BCC = 303                              'Este es una constante decimal resultado de summar 01,000
Stx Cr Etx
Do Until j > Len(COMANDO)                    'Cuenta el número de caracteres en el comando
        caracter = Mid(COMANDO, j, 1)        'De acuerdo al contador va sacando uno por uno los
caracteres del comando
        BCC = BCC + Asc(caracter)            'A la constante decimal se le suma el equivalente
decimal de cada caracter
        j = j + 1                      'Se incrementa el contador
                                      'El entero de este cociente corresponde al caracter #2 del BCC
de 8 bits
Loop
bcc1 = BCC Mod 256                           'De la suma completa se obtiene el caracter #1 de 8
bits del BCC por medio del residuo de los posibles 256 caracteres que se pueden enviar
cociente = BCC / 256                         'De la suma completa se obtiene el cociente
BCC2 = Fix(cociente)                         'El entero de este cociente corresponde al caracter #2
del BCC de 8 bits                                'Primera linea de comandos con el cálculo de BCC
incluido
End Sub
```

```
Private Sub COMAND() 'Formato de comando solo falta asignar palabra clave
MSComm1.Output = Chr$(1) + "01,000" + Chr$(2) + COMANDO + Chr$(13) + Chr$(3) +
Chr$(bcc1) + Chr$(BCC2)
End Sub
```

```
'Subrutina de cierre de comunicación
Private Sub Dialogo()
TACK1                          'Tiempo de espera para recibir el ACK0
TLEENQ                          'Tiempo de espera para recibir ENQ
tiempo                        'Tiempo de sincronía
motodata = MSComm1.Input              'Se lee lo que hay en el buffer
If usarveri = 1 Then
Else
End If
MSComm1.Output = Chr$(16) & Chr$(49)     'Responde ACK1
TEOT
If usarveri = 1 Then
If verifica = 4 Then
proto_master2
Else
End If
Else
End If
End Sub
```

```
'Ciclo que se detiene cuando se recibe la información esperada
Private Sub TACK1()
Dim intento, tamaño As Long
Dim car1, car2, buffer  As Variant
Dim data As String
On Error GoTo panic
Do
 Ret = DoEvents
  buffer = buffer + MSComm1.Input
 tamaño = Len(buffer)
```

```
Loop While tamaño < 2
Do
  Ret = DoEvents
   car1 = CStr(Asc(Mid(buffer, 1)))
 data = data + car1
Loop Until data = 16

Do
 Ret = DoEvents
  car2 = CStr(Asc(Mid(buffer, 2)))
  data = data + car2
Loop Until data = 1649
 MSComm1.Output = Chr$(4)


panic:
 HANDLER
End Sub
```

```
'Tiempo de espera del ENQ del diálogo
Private Sub TLEENQ()
Dim intento As Long
Dim car, buffer As Variant
Dim data As String
On Error GoTo panic
Do
 Ret = DoEvents
 buffer = MSComm1.Input
Loop While buffer = ""
Do
 Ret = DoEvents
  car = CStr(Asc(Mid(buffer, 1)))
  data = data + car
Loop Until InStr(data, 5)
MSComm1.Output = Chr$(16) + Chr$(48)
panic:
 HANDLER
End Sub
```

```
'Es el tiempo que necesita el programa para leer lo que hay en el puerto
Private Sub tiempo()
Timer1.Enabled = True
Counter = 0
Counter = Timer + 0.8        'Espera 8 segundos
Do While Timer < Counter
Ret = DoEvents
Loop
Timer1.Enabled = False
End Sub
```

```
'Tiempo de espera para que finalice la comunicación
Private Sub TEOT()
Dim intento As Long
Dim car, buffer  As Variant
Dim data As String
```

```
On Error GoTo panic
Do
 Ret = DoEvents
 buffer = MSComm1.Input
Loop While buffer = ""   'No avanza hasta que el robot envíe un caracter

 Do
intento = intento + 1
 Ret = DoEvents
  car = CStr(Asc(Mid(buffer, 1)))
 data = data + car
Loop Until InStr(data, 4)
tiempo_ejecución
interfaz.MousePointer = 1
panic:
 HANDLER
```

```
'Solo para recepción de programas
Public Function proto_master1()
proto_master1 = 1
Abre_puerto
interfaz.MousePointer = 11
ENQ
If enqerror = 0 Then
COMANDO = COMANDO1
mototalk_job
Else
proto_master1 = 0
Cierra_puerto
interfaz.MousePointer = 1
ProgressBar1.Visible = False
StatusBar1.Panels(1).TEXT = "FUERA DE LINEA"
Exit Function
End If
End Function
```

```
Private Sub mototalk_job()
BCC_JOB
comand_job
Dialogo_job
End Sub
```

```
'Es el generador de BCC´s cuando se descarga un trabajo del robot
Private Sub BCC_JOB()
Dim j As Long
j = 1
BCC = 310
Do Until j > Len(mezcla)
        caracter = Mid(mezcla, j, 1)
        BCC = BCC + Asc(caracter)
        j = j + 1
Loop
bcc1 = BCC Mod 256      'De la suma completa se obtiene el caracter #1 de 8 bits del BCC por
medio del residuo de los posibles 256 caracteres que se pueden enviar
cociente = BCC / 256    'De la suma completa se obtiene el cociente
BCC2 = Fix(cociente)
End Sub
```

```
'Formato para descarga de programas del robot solo falta nombre de archivo
Private Sub comand_job()
MSComm1.Output = Chr$(1) + "02,051" + Chr$(2) + COMANDO + Chr$(13) + Chr$(3) +
Chr$(bcc1) + Chr$(BCC2)
End Sub
```

```
Private Sub Dialogo_job()
tiempo_dialogo
respuesta = MSComm1.Input              'Limpia buffer y lee ACK1
MSComm1.Output = Chr$(4)               'Envia EOT
'barra1
tiempo_dialogo
respuesta = MSComm1.Input              'Lee ENQ
MSComm1.Output = Chr$(16) & Chr$(48)   'Envia ACK0
tiempo_job
respuesta = MSComm1.Input              'Se corrobora el programa que se espera
MSComm1.Output = Chr$(16) & Chr$(49)   'Responde ACK1
ProgressBar1.Value = 75
answer = 2
MSComm1.RThreshold = 1
End Sub
```

```
Private Sub tiempo_dialogo()
Timer1.Enabled = True
Counter = 0
Counter = Timer + 0.9
Do While Timer < Counter
Ret = DoEvents
Loop
Timer1.Enabled = False
End Sub
```

```
Private Sub tiempo_job()
Timer1.Enabled = True
Counter = 0
Counter = Timer + 5
Do While Timer < Counter
Ret = DoEvents
Loop
Timer1.Enabled = False
End Sub
```

```
'Solo para cargar programas al robot
Public Function proto_master3()
'barra1
proto_master3 = 1
Abre_puerto
ENQ
If enqerror = 0 Then
COMANDO = COMANDO1
mototalk_job_envio
Else
proto_master3 = 0
```

```
Cierra_puerto
boton_in
ProgressBar1.Visible = False
Exit Function
End If
End Function
```

```
Private Sub mototalk_job_envio()
BCC_JOB_envio
comand_job_envio
Dialogo_job_envio
End Sub
```

```
'Corrobora información cuando envía el comando de recepción de texto
Private Sub BCC_JOB_envio()
Dim j As Long
j = 1
BCC = 325
Do Until j > Len(mezcla)                    'Cuenta el número de caracteres en el comando
        caracter = Mid(mezcla, j, 1)        'De acuerdo al contador va sacando uno por uno los
caracteres del comando
        BCC = BCC + Asc(caracter)           'A la constante decimal se le suma el equivalente
decimal de cada caracter

        j = j + 1                           'Se incrementa el contador
                                            'El entero de este cociente corresponde al caracter #2 del BCC de
8 bits
Loop
bcc1 = BCC Mod 256                          'De la suma completa se obtiene el caracter #1 de 8
bits del BCC por medio del residuo de los posibles 256 caracteres que se pueden enviar
cociente = BCC / 256                        'De la suma completa se obtiene el cociente
BCC2 = Fix(cociente)

End Sub
```

```
'Corrobora info cuando envía fragmentos
Private Sub BCC_JOB_texto()
Dim j As Long
j = 1
If chksumfin = 0 Then
BCC = 23
Else
BCC = 3
End If

Do Until j > Len(arch_envio)                'Cuenta el número de caracteres en el comando
        caracter = Mid(arch_envio, j, 1)    'De acuerdo al contador va sacando uno por uno los
caracteres del comando
        BCC = BCC + Asc(caracter)           'A la constante decimal se le suma el equivalente
decimal de cada caracter
        j = j + 1                           'Se incrementa el contador                          'El
entero de este cociente corresponde al caracter #2 del BCC de 8 bits
Loop
bcc1 = BCC Mod 256                          'De la suma completa se obtiene el caracter #1 de 8
bits del BCC por medio del residuo de los posibles 256 caracteres que se pueden enviar
```

```
cociente = BCC / 256                    'De la suma completa se obtiene el cociente
BCC2 = Fix(cociente)
End Sub
```

```
'Dialogo para envío de comandos
Private Sub Dialogo_job_envio()
Dim contador, contador2, ncar, entero, resto As Long
Dim car_envio As String
Dim tamaño, pak, paquetes As Long
tiempo_dialogo
respuesta = MSComm1.Input
tamaño = Len(strtext)
pak = tamaño / 256
paquetes = Fix(pak)
entero = paquetes * 256
resto = tamaño - entero
ProgressBar1.Value = 50                 'Activa barra de avance'
Select Case pak
Case Is <= 1
    car_envio = Mid(strtext, 1, tamaño)
     arch_envio = car_envio
      chksumfin = 1
       BCC_JOB_texto
        MSComm1.Output = Chr$(2) + arch_envio + Chr$(3) + Chr$(bcc1) + Chr$(BCC2)

         tiempo_dialogo
          respuesta = MSComm1.Input
            MSComm1.Output = Chr$(4)        'Envia EOT
              TLEENQ
                 tiempo
                   motodata = MSComm1.Input          'Se lee lo que hay en el buffer
                   verifica
                 MSComm1.Output = Chr$(16) & Chr$(49)     'Responde ACK1
               tiempo_dialogo
              respuesta = MSComm1.Input                 'Se lee EOT
            arch_envio = ""
          strtext = ""
    Case Is > 1
 ncar = 1
 Do
 contador2 = contador2 + 1
  car_envio = Mid(strtext, ncar, 256)
    ncar = ncar + 256
     arch_envio = arch_envio + car_envio
      chksumfin = 0
       BCC_JOB_texto
       MSComm1.Output = Chr$(2) + arch_envio + Chr$(23) + Chr$(bcc1) + Chr$(BCC2)
      tiempo_dialogo
     respuesta = MSComm1.Input
   arch_envio = ""
  Ret = DoEvents
Loop Until contador2 = paquetes

arch_envio = ""
contador2 = 0
```

```
car_envio = Mid(strtext, ncar, resto)
arch_envio = arch_envio + car_envio
chksumfin = 1
ProgressBar1.Value = 75
BCC_JOB_texto
MSComm1.Output = Chr$(2) + arch_envio + Chr$(3) + Chr$(bcc1) + Chr$(BCC2)
tiempo_dialogo
respuesta = MSComm1.Input
MSComm1.Output = Chr$(4)              'Envia EOT
tiempo_dialogo
respuesta = MSComm1.Input             'Lee ENQ
MSComm1.Output = Chr$(16) & Chr$(48)    'Envia ACK0
tiempo
motodata = MSComm1.Input              'Se lee lo que hay en el buffer
verifica
MSComm1.Output = Chr$(16) & Chr$(49)    'Responde ACK1
tiempo_dialogo
respuesta = MSComm1.Input
strtext = ""
arch_envio = ""
End Select
interfaz.MousePointer = 1
ProgressBar1.Value = 100
boton_in
ProgressBar1.Visible = False
StatusBar1.Panels(1).TEXT = "EN LINEA"
End Sub
```

```
'Formato de envío de programas solo falta nombre del programa
Private Sub comand_job_envio()
MSComm1.Output = Chr$(1) + "02,001" + Chr$(2) + COMANDO + Chr$(13) + Chr$(23) +
Chr$(bcc1) + Chr$(BCC2)
End Sub
```

```
'Se activa con un solo caracter en el buffer se usa para recepción de programas
Private Sub MSComm1_OnComm()
Dim car, buffer As Variant
Dim numvez As Long
Dim fin As String
On Error GoTo panic
If answer = 1 Then
Select Case MSComm1.CommEvent          'Activa el caso de recepción
Case comEvReceive
interfaz.MousePointer = 11
StatusBar1.Panels(1).TEXT = " COMUNICANDO"
End Select
Else
Select Case MSComm1.CommEvent
Case comEvSend
MSComm1.Output = Chr$(16) + Chr$(49)
Case comEvReceive                   'Se activa cuando ha recibido 1 caracter
  buffer = MSComm1.Input
    car = CStr(Asc(Mid(buffer, 1)))     'Convierte a numéros en forma string los caracteres leidos
en el buffer 1 por 1
If car = 1 Or car = 2 Or car = 3 Or car = 4 Or car = 5 Or car = 6 Or car = 16 Or car = 23 Then
'Excluye ACK,EOT,ENQ
  GoTo 2
```

```
    Else
     If noread = 0 Then                'Verifica que no este activada la bandera de no lectura
      motodata = motodata + buffer      'Lectura de datos
    Else
   End If
End If

2:
Select Case car
 Case 2
  noread = 0                          'Desactiva la bandera de no lectura
'Activa la bandera de no lectura
       Case 4
   If motodata <> "/" Then
         INICIO = InStr(motodata, "/")
            fin = InStr(motodata, "END")
              fin = fin + 2
                progra = Mid(motodata, INICIO, fin)
                 Open nombrearch For Output As #1     'Abre el archivo
                  Print #1, progra                    'escribe en el archivo
                   Close #1                           'Cierra el archivo
                    MSComm1.RThreshold = 0            'Desactiva la sensibilidad de eventos del
puerto
                     progra = ""
                     INICIO = 0
                      motodata = ""
                      ProgressBar1.Value = 100
                      interfaz.MousePointer = 1
                      ProgressBar1.Visible = False
                      MsgBox ("PROGRAMA RECIBIDO")
                     boton_in
                     Else
                    Exit Sub
                   End If
                  Case 5
                 MSComm1.Output = Chr$(16) + Chr$(48)
                Case 23
               noread = 1
              tiempo_enq
             MSComm1.Output = Chr$(16) + Chr$(49)
           Case 3
          noread = 1
        Do
       Ret = DoEvents
     Loop Until MSComm1.InBufferCount = 0   'Espera que pasen los dos caracteres del BCC
después del ETX
    tiempo_arch                       'tiempo de seguridad
   MSComm1.Output = Chr$(16) + Chr$(49)     'Responde ACK1
  End Select
End Select
End If
interfaz.MousePointer = 1
StatusBar1.Panels(1).TEXT = "EN LINEA"
panic:
HANDLER
End Sub
```

```
'Tiempo de recepción de archivos
Private Sub tiempo_arch()
Timer1.Enabled = True        'Habilita el cronómetro
Counter = 0                  'Reinicia el cronómetro
Counter = Timer + 0.5        'Da .8 segundos para leer el buffer
Do While Timer < Counter     'El programa en espera
Ret = DoEvents               'Permite que se ejecuten otras tareas
Loop
Timer1.Enabled = False
End Sub
```

```
'Esta subrutina al ser ejecutada trae la información de la posición del robot en forma SLRUBT, y
de acuerdo al número de punto asignado en puntos(form1.frm) la guarda como variable de
posición en motoman
Public Function posición(p) As String
Dim k As Long
Dim conteo As Long
boton_out
StatusBar1.Panels(1).TEXT = "GRABANDO PUNTO"
COMANDO1 = "RPOSJ"                    'Palabra clave para conocer posiciones
proto_master2
conteo = Len(motodata)
If conteo > 0 Then
k = 9
cord = ""
  Do Until k > conteo - 14            'Para dejar fuera 2 ceros y 2 comas
     caracter = Mid(motodata, k, 1)
      cord = cord + caracter
        k = k + 1
        Loop

        punto(p) = cord

COMANDO1 = "LOADV 4," + p + ",0," + cord      'Es lo que saldrá del buffer para que guarde el
punto en la memoria
tiempo
proto_master2                         'Ejecuta de acuerdo a las circunstancias significa variable del
tipo posición
Else
End If
boton_in
boton_in
End Function
```

```
Public Function a_punto(p As String, vel As String, sel As Integer) As String
Dim k As Long
Dim conteo As Long
On Error GoTo panic

boton_out
COMANDO1 = "SAVEV 4," + p             'Obtiene la información de coordenadas del punto
pregrabado
If proto_master2 = 1 Then             'Comprueba comunicación
conteo = Len(motodata)
```

```
If conteo > 0 Then
k = 11
cord = ""
Do Until k > conteo - 6              'Para dejar fuera 2 ceros y 2 comas
  caracter = Mid(motodata, k, 1)
 cord = cord + caracter              'Pasa las coordenadas a cadena
k = k + 1
Loop
If sel = 1 Then
COMANDO1 = "PMOVJ " + vel + "," + cord + ",0,0,0,0,0,0,0"
Else
COMANDO1 = "PMOVL" + " 0," + vel + "," + cord + ",0,0,0,0,0,0,0"
End If
tiempo
usarveri = 2
proto_master2
End If
Else
boton_in
Exit Function
End If
boton_in
panic:
HANDLER
End Function
```

```
'Maneja errores ENQ sin respuesta
Private Function ENQ_error_check() As Integer
ENQ_error_check = 1
tiempo_enq                           'Es el valor default si es que no hubo errores
If MSComm1.InBufferCount = 0 Then          'Si no hay nada en el buffer es que no hubo
respuesta al ENQ
ENQ_error_check = 0
'Es el valor para decirque si hubo errores en la respuesta
'Son los mensajes de errores de comunicación iniciales
usuario = MsgBox("¿Esta Motoman encendido?", vbYesNo, "Error de Comunicación")
  Select Case usuario
   Case 6
    usuario = MsgBox("¿Esta Motoman en modo REMOTE + PLAY", vbYesNo, "Error de
Comunicación")
       Select Case usuario
        Case 6
         usuario = MsgBox("Reinicie su computadora y el robot", vbExclamation, "Error de
Comunicación")
          usuario = MsgBox("Puede que el puerto serial no esté funcionando o no esté conectado
al robot", vbExclamation, "Soluciones")
          Case 7
           usuario = MsgBox("Active REMOTE + PLAY e intente de nuevo", vbExclamation,
"Solución")
          End Select
         Case 7
        usuario = MsgBox("Encienda Motoman e intente de nuevo", vbExclamation, "Soluciónes")
       End Select
      Else
    End If
  End Function
```

```
'Mensajes cuando falla la conexión con el robot
  Private Function mensajes() As Integer
  usuario = MsgBox("Desea reintentar", vbYesNo + vbCritical, "No se ha establecido Conexión")
 Select Case usuario
  Case 6
     usuario = MsgBox("1. La conexión del cable serial en la computadora" + vbCrLf + "2. En el
PLAYBACKBOX la función REMOTE activada" + vbCr + "3. Que el controlador del motoman esté
encendido" + vbCr, vbOKCancel + vbInformation, "Antes de continuar Verifique")
   Select Case usuario
   Case 1
   mensajes = 1
   Case 2
   interfaz.Show
   End Select

   Case 7
     End Select
  End Function
```

```
'Cierra puerto serial
Private Sub Cierra_puerto()
If MSComm1.PortOpen = True Then
MSComm1.PortOpen = False
End If
End Sub
```

```
'Abre puerto serial
Public Sub Abre_puerto()
If MSComm1.PortOpen = False Then
MSComm1.PortOpen = True
End If
End Sub
```

```
'Rutinas para ver, borrar y ejecutar.
Public Function TEXTEXT(jo) As String
Select Case BORRAR
Case 0
COMANDO1 = "JSEQ " + jo + " ,0"
          boton_out
          proto_master2
          boton_in
Case 1
COMANDO1 = "DELETE " + jo
boton_out
proto_master2
boton_in

Case 3
COMANDO1 = "START " + jo
boton_out
proto_master2
boton_in
End Select

End Function
```

```
'Es la subrutina principal de retroalimentación al usuario
Private Function verifica()
Dim error As Variant
```

```
Dim conteo As Long
conteo = Len(motodata)
If conteo > 0 Then
caracter = Mid(motodata, 9, 4)    'Obtiene valor del error o la alrma o estatus
error = caracter
Select Case error
Case 1011
Ret = DoEvents
MsgBox ("El comando seleccionado no se puede ejecutar por restricciones de motoman")
Case 1012
MsgBox ("El punto no existe en el controlador")
Case 2070
Ret = DoEvents
verifica = MsgBox("Encender en PLAYBACK BOX el boton SERVO y presione Retry",
vbRetryCancel)
Case 2080
Ret = DoEvents
verifica = MsgBox("Debe cambiar de modo TEACH a PLAY o Viceversa en la pantalla
principal,corrija y oprima retry", vbRetryCancel)

Case 2010
Ret = DoEvents
verifica = MsgBox("No válida con el robot en operación, espere a que el robot termine, corrija y
oprima retry", vbRetryCancel)

Case 2020
Ret = DoEvents
verifica = MsgBox("Robot detenido por paro de emergencia en controlador manual, desactive el
botón y oprima retry", vbRetryCancel)

Case 2030
Ret = DoEvents
verifica = MsgBox("Robot detenido por paro de emergencia del PLAYBACK box en la mesa de
trabajo,desactive el botón y oprima retry", vbRetryCancel)
Case 2040
Ret = DoEvents
verifica = MsgBox("Robot detenido por paro de emergencia remoto, desactive el botón y oprima
retry", vbRetryCancel)

Case 2060
Ret = DoEvents
MsgBox ("El robot está inhabilitado por un error o alarma, Presione el botón emergencia y
reintente")
Case 2130
Ret = DoEvents
verifica = MsgBox("Operación no válida en modo de edicion presione DISP en el controlador
manual y presione retry", vbRetryCancel)
Case 4020
Ret = DoEvents
MsgBox ("Candado de Edición")
Case 4030
Ret = DoEvents
verifica = MsgBox("El trabajo ya existe en la memoria, cambie de nombre desde el controlador
manual y presione retry", vbRetryCancel)
Case 5110
Ret = DoEvents
```

```
verifica = MsgBox("Error de sintáxis en el programa, corrija el error y presione retry",
vbRetryCancel)
Case 5130
Ret = DoEvents
verifica = MsgBox("Al programa le falta la instrucción NOP o END, corrija y presione retry",
vbRetryCancel)
End Select
Else
End If
End Function
```

```
'Se encarga de manejar la situación de error que se haya presentado
Private Sub HANDLER()
Dim A As Long
If Err.Number <> 0 Then
If Err.Number <> cdlCancel Then
A = MsgBox("Error" & Err.Number & vbCr & vbCr & Err.Description, vbCritical + vbOKOnly, "Error
interno")
Err.Clear
End If
End If
boton_in
End Sub
```

```
'Comienzan subrutinas relacionadas con botones
Private Sub HOME_Click()
Dim elec As Long
Dim aviso As Long
aviso = MsgBox("Antes de continuar verifique" + vbCr + "1. El botón ENEABLE del control
manual esté apagado" + vbCr + "2. El botón PLAY esté activado", vbOKCancel + vbExclamation,
"IMORTANTE")
  If aviso = 1 Then
    elec = MsgBox("El robot comenzará a moverse, Verifique que no haya peligro de colisión y
oprima OK", vbOKCancel + vbExclamation, "Precaución")
      If elec = 1 Then
        COMANDO1 = "START HOME"    'Si se cambia el nombre de la rutina se cambia el
nombre después de START
          StatusBar1.Panels(1).TEXT = "PRECAUCION MOTOMAN EN MOVIMIENTO A HOME"
           boton_out
         proto_master2
       boton_in
      Else
     End If
   Else
 End If
End Sub
```

```
'Para ejecutar un programa en el robot
Private Sub EJECUTAR_Click()
aviso = MsgBox("Antes de continuar verifique" + vbCr + "1. El botón ENEABLE del control
manual esté apagado" + vbCr + "2. El botón PLAY esté activado", vbOKCancel + vbExclamation,
"IMORTANTE")
If aviso = 1 Then
Dim elec
elec = MsgBox("El robot comenzará a ejecutar un trabajo, Verifique que no haya peligro de
colisión y que todos los elementos de trabajo estén en el lugar correcto. Después presione OK",
vbOKCancel + vbExclamation, "PRECAUCIÓN")
If elec = 1 Then
```

```
BORRAR = 3
JOB.Show
Else
End If
Else
End If
End Sub
```

```
'Subrutina para desactivar herramienta
Private Sub herdesac_Click()
aviso = MsgBox("Antes de continuar verifique" + vbCr + "1. El botón ENEABLE del control
manual esté apagado" + vbCr + "2. El botón PLAY esté activado", vbOKCancel + vbExclamation,
"IMORTANTE")
If aviso = 1 Then
 StatusBar1.Panels(1).TEXT = "PRECAUCION MOTOMAN ACTIVANDO HERRAMIENTA"
   If tool = 1 Or tool = 2 Or tool = 3 Then
    COMANDO1 = "START RBHERROF"  'Si se cambia el nombre de la rutina se cambia el
nombre después de START
      boton_out
       proto_master2
       boton_in
      Else
    MsgBox ("No hay herramienta en Motoman")
  heract.Value = 0
  End If
 Else
End If
End Sub
```

```
'Rutina para activar herramienta
Private Sub HERACT_Click()
aviso = MsgBox("Antes de continuar verifique" + vbCr + "1. El botón ENEABLE del control
manual esté apagado" + vbCr + "2. El botón PLAY esté activado", vbOKCancel + vbExclamation,
"IMORTANTE")
If aviso = 1 Then
StatusBar1.Panels(1).TEXT = "PRECAUCION MOTOMAN ACTIVANDO HERRAMIENTA"
          If tool = 1 Or tool = 2 Or tool = 3 Then

        COMANDO1 = "START RBHERRON"    'Si se cambia el nombre de la rutina se cambia
el nombre después de START
        boton_out
        proto_master2
        boton_in
Else
MsgBox ("No hay herramienta en Motoman")
heract.Value = 0
End If
Else
End If
End Sub
```

```
'Rutina para cerrar mordazas
Private Sub MORD_Click()
aviso = MsgBox("Antes de continuar verifique" + vbCr + "1. El botón ENEABLE del control
manual esté apagado" + vbCr + "2. El botón PLAY esté activado", vbOKCancel + vbExclamation,
"IMORTANTE")
If aviso = 1 Then
```

```
            COMANDO1 = "START ENSCIPRE"    'Si se cambia el nombre de la rutina se cambia
el nombre después de START
              boton_out
               proto_master2
                boton_in
Else
End If
End Sub
```

```
'Rutina para abrir mordazas
Private Sub MORD2_Click()
aviso = MsgBox("Antes de continuar verifique" + vbCr + "1. El botón ENEABLE del control
manual esté apagado" + vbCr + "2. El botón PLAY esté activado", vbOKCancel + vbExclamation,
"IMORTANTE")
If aviso = 1 Then
              COMANDO1 = "START ENSABPRE"    'Si se cambia el nombre de la rutina se
cambia el nombre después de START
              boton_out
              proto_master2
             boton_in
             Else

End If
End Sub
```

```
'Al activar este botón se abre la ventana de dialogo del editor
Private Sub programador_Click()
program.Show
End Sub
```

```
'Es la selección de colocar trabajo en la pantalla del controlador
Private Sub Fijar_Click()
 BORRAR = 0
JOB.Show
End Sub
```

```
'Es la selección para borrar programa del controlador
Private Sub erasejob_Click()
Dim elec As Long
elec = MsgBox("¿Está usted seguro?" + vbCr + "La información se perderá del controlador",
vbOKCancel + vbExclamation, "Precaución")
If elec = 1 Then
BORRAR = 1
JOB.Show
Else
End If
End Sub
```

```
'Rutina para asignar un nombre al programa que se va a descargar del robot y manda llamar la
subrutina protomaster
Private Sub RECIBIR_Click()
Dim punto, large As Long
boton_out
On Error GoTo panic
StatusBar1.Panels(1).TEXT = "RECIBIENDO PROGRAMA"
CommonDialog1.DialogTitle = "Recepción de programade MOTOMAN a disco"
CommonDialog1.DefaultExt = "JBI"
CommonDialog1.Filter = "Programas MOTOMAN|*.JBI"
CommonDialog1.InitDir = ""
CommonDialog1.Flags = cdlOFNOverwritePrompt
```

```
CommonDialog1.FileName = ""
CommonDialog1.ShowSave
nombrearch = CommonDialog1.FileTitle
If nombrearch <> "" Then
COMANDO1 = " "
punto = InStr(nombrearch, ".")
large = punto - 1                    'Para que el archivo termine antes que el punto del texto
COMANDO1 = Mid(nombrearch, 1, large)
mezcla = COMANDO1
proto_master1
Else
MsgBox ("DEBE PROPORCIONAR UN NOMBRE")
 StatusBar1.Panels(1).TEXT = "EN LINEA"
      boton_in
Exit Sub
End If
ProgressBar1.Value = 25
panic:
HANDLER
End Sub
```

```
Private Sub Salir_Click()
End
End Sub
```

```
Private Sub SAVEJOB_Click()
JOB.Show
End Sub
```

```
'Para tomar gripper
Private Sub Gripper_Click()
Dim elec As Long

aviso = MsgBox("Antes de continuar verifique" + vbCr + "1. El botón ENEABLE del control
manual esté apagado" + vbCr + "2. El botón PLAY esté activado", vbOKCancel + vbExclamation,
"IMORTANTE")
If aviso = 1 Then

elec = MsgBox("El robot comenzará a moverse, Verifique que no haya peligro de colisión y
oprima OK", vbOKCancel + vbExclamation, "Precaución")
If elec = 1 Then
COMANDO1 = "START TOMGRIP"  'Si se cambia el nombre de la rutina se cambia el nombre
después de START
StatusBar1.Panels(1).TEXT = "PRECAUCION MOTOMAN TOMANDO GRIPPER"
boton_out
If proto_master2 = 1 Then
tool = 1
Else
tool = 0
End If
boton_in
If enqerror = 0 Then
StatusBar1.Panels(1).TEXT = "En línea"
Else
StatusBar1.Panels(1).TEXT = "Fuera de línea"
End If
Else
End If
```

```
Else
End If
End Sub
Private Sub tomar_grip_Click()
End Sub
'Para terminar interfaz
Private Sub TERMINAR_Click()
End
End Sub
Private Sub Timer4_Timer()
Timer4.Enabled = False
End Sub
'Se usa cuando se va a subir un programa al robot
Public Sub UPPRG_Click()
Dim punto, large, aviso As Long
Dim arch, strbuffer, archivo As String
Dim numarch%
aviso = MsgBox("Antes de continuar verifique" + vbCr + "Que después de la instrucción END no
haya espacios", vbOKCancel + vbExclamation, "IMORTANTE")
Select Case aviso
Case 2
GoTo final
End Select
On Error GoTo panic
boton_out
ProgressBar1.Visible = True
interfaz.MousePointer = 11
StatusBar1.Panels(1).TEXT = "ENVIANDO PROGRAMA"
CommonDialog1.DialogTitle = "Programa Motoman"
CommonDialog1.Filter = "Programas Motoman|*.JBI"
CommonDialog1.FileName = ""
CommonDialog1.ShowOpen
CommonDialog1.Flags = cdlOFNHideReadOnly
CommonDialog1.DefaultExt = ".JBI"
 archivo = CommonDialog1.FileTitle
 If archivo <> "" Then
   COMANDO1 = " "
    punto = InStr(archivo, ".")
     large = punto - 1                    'Para que el archivo termine antes que el punto
      COMANDO1 = Mid(archivo, 1, large)
       mezcla = COMANDO1
        numarch% = FreeFile
          Open archivo For Input As numarch%   'Abre el archivo
           Do While Not EOF(numarch%)
            Line Input #numarch%, strbuffer
             strtext = strtext + strbuffer + vbCrLf
              Loop
             Close #numarch%
           proto_master3
           Else
           StatusBar1.Panels(1).TEXT = "EN LINEA"
           ProgressBar1.Visible = False
           boton_in
            interfaz.MousePointer = 1
      Exit Sub
```

```
    End If
panic:
    HANDLER
final:
End Sub
```

```
'Para tomar ventosa
Private Sub VENTOSA_Click()
Dim elec
aviso = MsgBox("Antes de continuar verifique" + vbCr + "1. El botón ENEABLE del control
manual esté apagado" + vbCr + "2. El botón PLAY esté activado", vbOKCancel + vbExclamation,
"IMORTANTE")
If aviso = 1 Then
elec = MsgBox("El robot comenzará a moverse, Verifique que no haya peligro de colisión y
oprima OK", vbOKCancel + vbExclamation, "Precaución")
If elec = 1 Then
COMANDO1 = "START TOMVENT"  'Si se cambia el nombre de la rutina se cambia el nombre
después de START
StatusBar1.Panels(1).TEXT = "PRECAUCION MOTOMAN TOMANDO VENTOSA"
boton_out
If proto_master2 = 1 Then
tool = 3
Else
tool = 0
End If
boton_in
If enqerror = 0 Then
StatusBar1.Panels(1).TEXT = "EN LINEA"
Else
StatusBar1.Panels(1).TEXT = "FUERA DE LINEA"
End If
Else
End If
Else
End If
End Sub
```

```
'Para tomar desarmador
Private Sub Destorn_Click()
Dim elec As Long
aviso = MsgBox("Antes de continuar verifique" + vbCr + "1. El botón ENEABLE del control
manual esté apagado" + vbCr + "2. El botón PLAY esté activado", vbOKCancel + vbExclamation,
"IMORTANTE")
If aviso = 1 Then
elec = MsgBox("El robot comenzará a moverse, Verifique que no haya peligro de colisión y
oprima OK", vbOKCancel + vbExclamation, "Precaución")
If elec = 1 Then
COMANDO1 = "START TOMDESM"  'Si se cambia el nombre de la rutina se cambia el nombre
después de START
StatusBar1.Panels(1).TEXT = "PRECAUCION MOTOMAN TOMANDO DESARMADOR"
boton_out
If proto_master2 = 1 Then
tool = 2
Else
tool = 0
End If
```

```
boton_in
If enqerror = 0 Then
StatusBar1.Panels(1).TEXT = "EN LINEA"
Else
StatusBar1.Panels(1).TEXT = "FUERA DE LINEA"
End If
Else
End If
Else
End If
End Sub
```

```
'Para dejar herramienta
Private Sub dejaherr_Click()
Dim elec As Long
aviso = MsgBox("Antes de continuar verifique" + vbCr + "1. El botón ENEABLE del control
manual esté apagado" + vbCr + "2. El botón PLAY esté activado", vbOKCancel + vbExclamation,
"IMORTANTE")
If aviso = 1 Then
elec = MsgBox("El robot comenzará a moverse, Verifique que no haya peligro de colisión y
oprima OK", vbOKCancel + vbExclamation, "Precaución")
If elec = 1 Then
Select Case tool                    'Verifica que herramienta fué tomada por medio de la revisión
de la variable global tool
Case 0
MsgBox ("Motoman no tiene Herramienta")
enqerror = 1
Case 1
 StatusBar1.Panels(1).TEXT = "PRECAUCION MOTOMAN DEJANDO GRIPPER"
   COMANDO1 = "START DGRIPPER"          'Si se cambia el nombre de la rutina se cambia el
nombre después de START
   boton_out
     If proto_master2 = 1 Then
      tool = 0
        Else
         End If
         boton_in
           Case 3
           COMANDO1 = "START DVENTOSA"   'Si se cambia el nombre de la rutina se cambia
el nombre después de START
           boton_out
           If proto_master2 = 1 Then
           tool = 0
          Else
         End If
         boton_in
        Case 2
        boton_out
      COMANDO1 = "START DDESTORN"        'Si se cambia el nombre de la rutina se cambia
el nombre después de START
    If proto_master2 = 1 Then
   tool = 0
  Else
 End If
 boton_in
End Select
If enqerror = 0 Then
```

```
StatusBar1.Panels(1).TEXT = "EN LINEA"
Else
StatusBar1.Panels(1).TEXT = "FUERA DE LINEA"
End If
Else
End If
Else
End If
End Sub
```

```
'Para colocar el robot en modo Play
Private Sub play_Click()
Dim aviso As Long
aviso = MsgBox("Antes de continuar verifique que el botón ENEABLE del control manual esté
desactivado" + vbCr, vbOKCancel + vbExclamation, "IMORTANTE")
If aviso = 1 Then
COMANDO1 = "MODE 2"     'No estamos mandando llamar una subrutina por lo tanto este no
cambia
boton_out
proto_master2
boton_in
If enqerror = 0 Then
play.BackColor = &H8000&
teacher.BackColor = &HE0E0E0
StatusBar1.Panels(1).TEXT = "EN LINEA"
Else
StatusBar1.Panels(1).TEXT = "FUERA DE LINEA"
End If
Else
End If
End Sub
```

```
'botón para cambiar a modo teach
Private Sub teacher_Click()
boton_out
COMANDO1 = "MODE 1"
proto_master2
boton_in
If enqerror = 0 Then
play.BackColor = &HE0E0E0
teacher.BackColor = &H8000&
StatusBar1.Panels(1).TEXT = "EN LINEA"
Else
StatusBar1.Panels(1).TEXT = "FUERA DE LINEA"
End If
End Sub
```

```
'Botón para cambiar a modo automático
Private Sub AUTO_Click()
COMANDO1 = "CYCLE 3"  'Esta palabra clave no cambia
boton_out
proto_master2
boton_in
If enqerror = 0 Then
StatusBar1.Panels(1).TEXT = "EN LINEA"
AUTO.BackColor = &H8000&
CYCLE1.BackColor = &HE0E0E0
Else
```

```
StatusBar1.Panels(1).TEXT = "FUERA DE LINEA"
End If
End Sub
'Para cambiar de modo
Private Sub CYCLE1_Click()
boton_out
COMANDO1 = "CYCLE 2"
proto_master2
boton_in
If enqerror = 0 Then
StatusBar1.Panels(1).TEXT = "EN LINEA"
AUTO.BackColor = &HE0E0E0
CYCLE1.BackColor = &H8000&
Else
StatusBar1.Panels(1).TEXT = "FUERA DE LINEA"
End If
End Sub
'Esta subrutina envía información sobre el estado del robot
Private Sub ESTADO_Click()
Dim k, e, conteo, est2, tamaño As Long
Dim est1 As Long
Dim stat, s(2), herr As String
Dim valor As Variant
On Error GoTo panic
usarveri = 1
boton_out
STATUS.Text1 = " "
COMANDO1 = "RSTATS"  'Esta rutina no cambia ya que es palabra clave
proto_master2
conteo = Len(motodata)
If conteo > 0 Then
k = 9
stat = ""
  Do Until k > conteo - 4
     caracter = Mid(motodata, k, 1)
       stat = stat + caracter
          k = k + 1
Loop
k = 1
e = 1
s(e) = " "
caracter = " "

Do Until e > 2
  Ret = DoEvents
   conteo = Len(stat)
     Do Until k > conteo
      Ret = DoEvents
       caracter = Mid(stat, k, 1)
         valor = Asc(Mid(stat, k, 1))
          If valor = 44 Then                    'Cada Slot esá separado por una coma y se lee
hasta la coma.
           k = k + 1
            Exit Do
             Else
```

```
            s(e) = s(e) + caracter                    'Se asigna una variable diferente a cada Slot
             k = k + 1
              End If
              Loop
e = e + 1
Loop

est1 = CInt(s(1))
est2 = CInt(s(2))

Select Case est1

Case 161
STATUS.Text2 = "TEACH /STEP"
teacher.BackColor = &H8000&
play.BackColor = &HE0E0E0
Case 162

STATUS.Text2 = "TEACH /1CYCLE"
teacher.BackColor = &H8000&
CYCLE1.BackColor = &H8000&
play.BackColor = &HE0E0E0
AUTO.BackColor = &HE0E0E0
Case 164
STATUS.Text2 = "TEACH /AUTO"
teacher.BackColor = &H8000&
AUTO.BackColor = &H8000&
play.BackColor = &HE0E0E0
CYCLE1.BackColor = &HE0E0E0
Case 193
STATUS.Text2 = "PLAY /STEP"
play.BackColor = &H8000&
teacher.BackColor = &HE0E0E0
AUTO.BackColor = &H8000000B
CYCLE1.BackColor = &H8000000F
Case 194
STATUS.Text2 = "PLAY /1CYCLE"
play.BackColor = &H8000&
CYCLE1.BackColor = &H8000&
teacher.BackColor = &HE0E0E0
AUTO.BackColor = &HE0E0E0
Case 196
STATUS.Text2 = "PLAY/ AUTO"
play.BackColor = &H8000&
AUTO.BackColor = &H8000&
teacher.BackColor = &HE0E0E0
CYCLE1.BackColor = &HE0E0E0
End Select
Select Case est2
Case 64
STATUS.Shape1.FillColor = &HC000&
Case 65 Or 66
STATUS.Shape2.FillColor = &HC000&
Case 96
STATUS.Shape4.FillColor = &HC000&
STATUS.Text1 = "??????"
```

```
GoTo Line1
Case 80
STATUS.Shape3.FillColor = &HC000&
STATUS.Text1 = "??????"
GoTo Line1
End Select
 COMANDO1 = "SAVEV 0,1"
 tiempo
 proto_master2
 k = 1
 caracter = " "
 conteo = Len(motodata)
k = 9
herr = ""
  Do Until k > conteo - 4
     caracter = Mid(motodata, k, 1)
      herr = herr + caracter
          k = k + 1
          Loop
tool = CInt(herr)
Select Case tool
Case 0
STATUS.Text1 = "NINGUNA"
Case 1
STATUS.Text1 = "GRIPPER"
Case 2
STATUS.Text1 = "DESARMADOR"
Case 3
STATUS.Text1 = "VENTOSA"
End Select
Else
boton_in
Exit Sub
End If
Line1: STATUS.Show
StatusBar1.Panels(1).TEXT = "EN LINEA"
boton_in
ESTADO.BackColor = &H8000&
panic:
HANDLER
End Sub
```

```
Private Sub punto_save_Click()
Puntos.Show                'manda llamar la pantalla de captura de puntos
End Sub
```

```
Private Sub punto_move_Click()
Dim aviso As Long
aviso = MsgBox("Antes de continuar verifique" + vbCr + "1. El botón ENEABLE del control
manual esté apagado" + vbCr + "2. El botón PLAY esté activado" + vbCr + "3. El botón 1CYCLE
esté activado", vbOKCancel + vbExclamation, "IMORTANTE")
If aviso = 1 Then
Pmove.Show
Else
End If
End Sub
```

```
'Esta es la subrutina que solo da lectura a la posición del robot
Private Sub pos_Click()
Dim k, conteo As Long
Dim e, aviso As Long
Dim valor As Variant
Dim c(7) As String
On Error GoTo panic
boton_out
usarveri = 1
COMANDO1 = "RPOSJ"
proto_master2
conteo = Len(motodata)
If conteo > 0 Then
k = 9
cord = ""
  Do Until k > conteo - 14
     caracter = Mid(motodata, k, 1)
       cord = cord + caracter
          k = k + 1
Loop
k = 1
e = 1
Do Until e > 6
  Ret = DoEvents
   conteo = Len(cord)
     Do Until k > conteo
      Ret = DoEvents
       caracter = Mid(cord, k, 1)
         valor = Asc(Mid(cord, k, 1))
          If valor = 44 Then                    'Cada Slot esá separado por una coma y se lee
hasta la coma.
           k = k + 1
            Exit Do
             Else
            c(e) = c(e) + caracter               'Se asigna una variable diferente a cada Slot
             k = k + 1
               End If
Loop
e = e + 1
Loop
'En estas variables se almacenan los valores de posición
SLURBT.Text1 = c(1)
SLURBT.Text2 = c(2)
SLURBT.Text3 = c(3)
SLURBT.Text4 = c(4)
SLURBT.Text5 = c(5)
SLURBT.Text6 = c(6)
SLURBT.Show
Else
Exit Sub

End If
boton_in
panic:
HANDLER
```

```
End Sub
Private Sub tiempo_enq()
Timer1.Enabled = True
Counter = 0
Counter = Timer + 0.8
Do While Timer < Counter
Ret = DoEvents
Loop
Timer1.Enabled = False
End Sub
Private Sub tiempo_ejecución()
Timer3.Enabled = True
Counter = 0
Counter = Timer + 0.4
Do While Timer < Counter
Ret = DoEvents
Loop
Timer3.Enabled = False
End Sub
'Esta subrutina manda llamar la alarma y fragmenta los mensajes
Private Sub EME_Click()
Dim k As Long
Dim e As Long
Dim valor As Variant
Dim conteo, error, alarma1, alarma2, alarma3, alarma4, alarma5, alarma6, alarma7, alarma8,
alarma9 As Long
Dim A(9) As String     'Array tamaño 9
On Error GoTo panic
boton_out
COMANDO1 = "RALARM"
proto_master2
conteo = Len(motodata)
If conteo > 0 Then
k = 9
alarma = ""                                    'Valor inicial de alarma
  Do Until k > conteo - 4                      'Son los caracteres que sobran como EOT
    caracter = Mid(motodata, k, 1)
      alarma = alarma + caracter               'Aquí se tienen los 9 slots de la alarma sin el
protocolo de comunicación
       k = k + 1
        Loop
k = 1
e = 1
A(e) = " "
caracter = " "

Do Until e > 9
 Ret = DoEvents
  conteo = Len(alarma)
    Do Until k > conteo
     Ret = DoEvents
      caracter = Mid(alarma, k, 1)
       valor = Asc(Mid(alarma, k, 1))
        If valor = 44 Then                     'Cada Slot esá separado por una coma y se lee
hasta la coma.
```

```
        k = k + 1
         Exit Do
          Else
          A(e) = A(e) + caracter                    'Se asigna una variable diferente a cada Slot
           k = k + 1
            End If
Loop
e = e + 1
Loop
'Códigos de error más comunes obtenidos del manual de comunicaciones
error = CInt(A(1))
StatusBar1.Panels(1).TEXT = "ESTADO: ERROR: " + A(1)
Select Case error
Case 20
respuesta = MsgBox("TEACH modo de operación incorrecto", vbExclamation, "ERROR")
RESET_Click
Case 40
respuesta = MsgBox("Posición no definida", vbExclamation, "ERROR")

Case 50
respuesta = MsgBox("Operación no valida con Modify Presionado", vbExclamation, "ERROR")

Case 100
respuesta = MsgBox("Operación no válida con Overrun recovery", vbExclamation, "ERROR")

Case 110
respuesta = MsgBox("Operación no valida con SERVO desactivado", vbExclamation, "ERROR")

Case 120
respuesta = MsgBox("Modo de operación incorrecto Presione Cancel y active Play",
vbExclamation, "ERROR")

Case 150
respuesta = MsgBox("Modo de operación incorrecto Presione Cancel y active Play",
vbExclamation, "ERROR")

Case 160
respuesta = MsgBox("Operación no válida con Enable activado", vbExclamation, "ERROR")

Case 380
respuesta = MsgBox("Posición no revisada", vbExclamation, "ERROR")

Case 1010
respuesta = MsgBox("No está permitido editar este programa", vbExclamation, "ERROR")

Case 2070
respuesta = MsgBox("Operación no válida con modify desactivado", vbExclamation, "ERROR")

Case 2080
respuesta = MsgBox("Operación no válida con Insert o Modify activado", vbExclamation,
"ERROR")

Case 2100
respuesta = MsgBox("Este trabajo no puede ser editado", vbExclamation, "ERROR")

Case 2140
```

```
respuesta = MsgBox("Debe presionar Enable para modificar", vbExclamation, "ERROR")

Case 2150
respuesta = MsgBox("No es posible insertar desde esa posición.Debe estar una lína antes de
END", vbExclamation, "ERROR")
Case 3110
respuesta = MsgBox("Error de sintáxis", vbExclamation, "ERROR")
Case 3200
respuesta = MsgBox("No se encontró instrucción NOP o END en el programa presione Cancel",
vbExclamation, "ERROR")
End Select
If error = 0 Then
respuesta = MsgBox("Motoman no encontró errores", vbExclamation, "ERROR")
Else
cancelar_click
MsgBox ("Error cancelado")
End If
alarma2 = CInt(A(2))
alarma4 = CInt(A(4))
alarma6 = CInt(A(6))
alarma8 = CInt(A(8))
'Casos de alarma más comunes
StatusBar1.Panels(1).TEXT = "ESTADO: ALARMA: " + A(2) + " " + A(4) + " " + A(6) + " " + A(8)
Select Case alarma2 Or alarma4 Or alarma6 Or alarma8
Case 0
respuesta = MsgBox("Motoman no encontró alarmas", vbExclamation, "ALARMA")
Case 2130
respuesta = MsgBox("Motoman editando", vbExclamation, "Mensaje")
Case 4020
respuesta = MsgBox("Archivo protegido contra escritura", vbExclamation, "Mensaje")
Case 4030
respuesta = MsgBox("El trabajo ya existe", vbExclamation, "Mensaje")
Case 4550
respuesta = MsgBox("Alarma por causa ajena al usuario", vbExclamation, "ALARMA")
RESET_Click
Case 4200 To 4202
respuesta = MsgBox("Distancia entre puntos incorrecta", vbExclamation, "ALARMA")
RESET_Click
Case 4210 To 4212
respuesta = MsgBox("Movimineto lineal imposible para los ejes L y U", vbExclamation,
"ALARMA")
RESET_Click
Case 4220 To 4222
respuesta = MsgBox("Movimineto lineal imposible para los ejes S Y L", vbExclamation,
"ALARMA")
RESET_Click
Case 5110
respuesta = MsgBox("Error de sintaxis", vbExclamation, "Mensaje")
Case 5100
respuesta = MsgBox("Impacto o falta de aire en la Muñeca de Motoman. Revise la alimentación
de aire", vbExclamation, "ALARMA")
RESET_Click
Case 5130
respuesta = MsgBox("Falta NOP o END, error de sintaxis en el archivo", vbExclamation,
"Mensaje")
Case 6690 To 6729
```

```
respuesta = MsgBox("Alarma de comunicación, ajena al usuario", vbExclamation, "ALARMA")
RESET_Click
Case 6730
respuesta = MsgBox("Alarma de comunicación. Error en sintáxis del archivo", vbExclamation,
"Mensaje")
RESET_Click
End Select
Else
boton_in
Exit Sub
End If
boton_in
StatusBar1.Panels(1).TEXT = "EN LINEA"
panic:
HANDLER
End Sub
```

```
'Desactiva algunos botones mintras se ejecuta una acción
Private Sub boton_out()
interfaz.MousePointer = 11
StatusBar1.Panels(1).Width = 10000
play.Enabled = False
teacher.Enabled = False
AUTO.Enabled = False
CYCLE1.Enabled = False
punto_save.Enabled = False
punto_move.Enabled = False
pos.Enabled = False
EME.Enabled = False
ESTADO.Enabled = False
Gripper.Enabled = False
Ventosa.Enabled = False
Destorn.Enabled = False
dejaherr.Enabled = False
Fijar.Enabled = False
UPPRG.Enabled = False
RECIBIR.Enabled = False
erasejob.Enabled = False
EJECUTAR.Enabled = False
home.Enabled = False
heract.Enabled = False
heract.Enabled = False
MORD.Enabled = False
herdesac.Enabled = False
MORD2.Enabled = False
End Sub
```

```
'Reactiva los botones que se desactivaron
Private Sub boton_in()
interfaz.MousePointer = 1
play.Enabled = True
teacher.Enabled = True
AUTO.Enabled = True
CYCLE1.Enabled = True
punto_save.Enabled = True
punto_move.Enabled = True
```

```
pos.Enabled = True
EME.Enabled = True
ESTADO.Enabled = True
Gripper.Enabled = True
Ventosa.Enabled = True
Destorn.Enabled = True
dejaherr.Enabled = True
Fijar.Enabled = True
UPPRG.Enabled = True
RECIBIR.Enabled = True
erasejob.Enabled = True
EJECUTAR.Enabled = True
home.Enabled = True
heract.Enabled = True
MORD.Enabled = True
herdesac.Enabled = True
MORD2.Enabled = True
End Sub
```

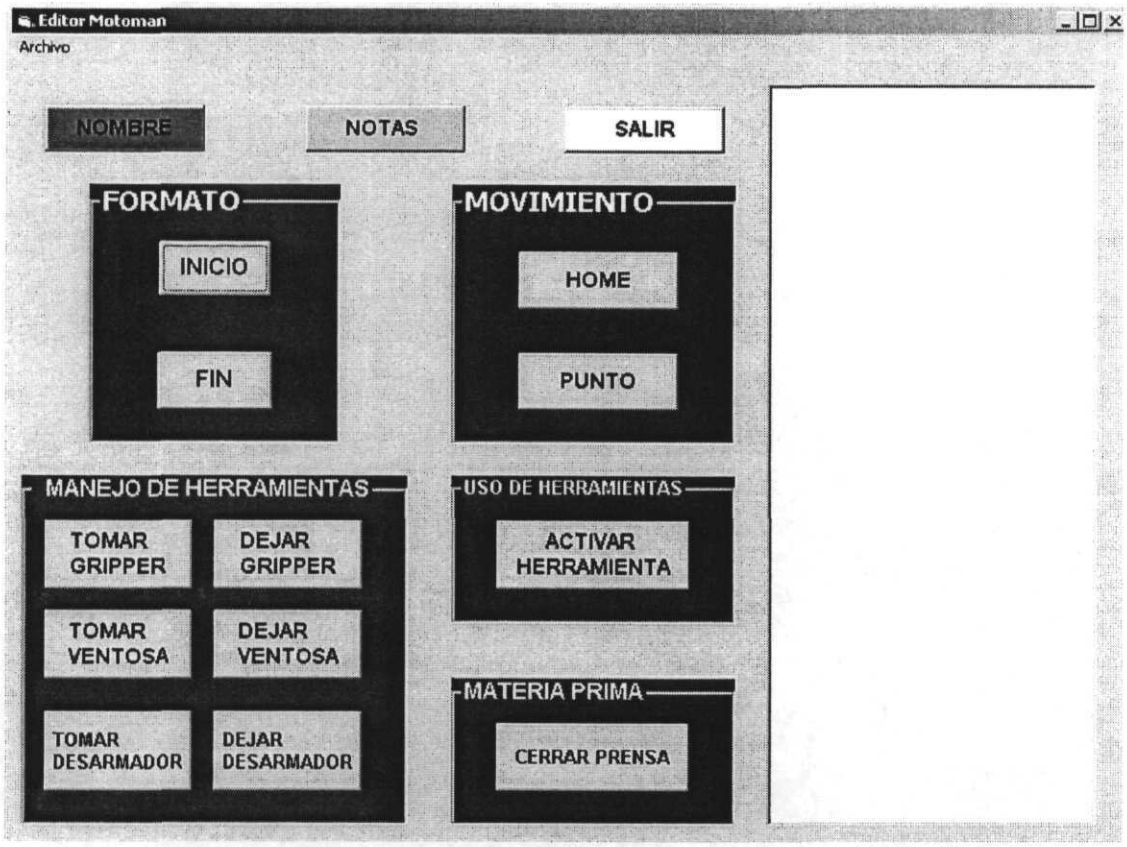
```
'Rutina para anular alarma
Private Sub RESET_Click()
boton_out
COMANDO1 = "RESET"
proto_master2
boton_in
End Sub
```

```
'Rutina para anular el error
Private Sub cancelar_click()
boton_out
COMANDO1 = "CANCEL"
proto_master2
boton_in
End Sub
```

```
'Se coloca en lugar de tiempo y se recomienda agragar un tiempito más ya que no dará
oportunidad al BCC
Private Sub TEXT()
Dim car, buffer  As Variant
Dim data, Text2 As String
On Error GoTo panic
Do
Ret = DoEvents
denuevo:
buffer = MSComm1.Input
motodata = motodata + buffer
If buffer = "" Then
GoTo denuevo
Else
End If
car = CStr(Asc(Mid(buffer, 1)))
data = data + car
 Loop Until InStr(data, 3)
 MSComm1.Output = Chr$(16) & Chr$(49)

panic:
 HANDLER
End Sub
```

ANEXO B
CÓDIGO Y DISEÑO DEL EDITOR DE LA INTERFAZ




```
Option Explicit

Dim COMANDO, nombrearch As String
Private Sub Form_Load()
program.Top = True
program.Height = Screen.Height
program.Width = Screen.Width
End sub

Private Sub Command4_Click()
End Sub
'Rutina para abrir
'Rutina para abrir archivo
Private Sub Abrearch_Click()
CommonDialog1.DialogTitle = "Editor de programas MOTOMAN"
CommonDialog1.DefaultExt = "JBI"
CommonDialog1.Filter = "Programas MOTOMAN|*.JBI"
CommonDialog1.InitDir = ""
CommonDialog1.FileName = ""
CommonDialog1.ShowOpen
```

```
nombrearch = CommonDialog1.FileName
VENTANA.LoadFile nombrearch
End Sub
```

```
'Rutina que conecta a manejo de puntos
Private Sub apunto_Click()
Pmove.Show
End Sub
```

```
'Para escribir código de dejar Desarmador
Private Sub DDESTORN_Click()
COMANDO = "CALL JOB:DDESTORN"
insertar (COMANDO)
End Sub
```

```
'Para dejar Desarmador
Private Sub DESARMADOR_Click()
COMANDO = "CALL JOB:TDESTORN"
insertar (COMANDO)
End Sub
```

```
'Para dejar gripper
Private Sub DGRIPPER_Click()
COMANDO = "CALL JOB:DGRIPPER"
insertar (COMANDO)
End Sub
```

```
'Para dejar venotsa
Private Sub DVENTOSA_Click()
COMANDO = "CALL JOB:DVENTOSA"
insertar (COMANDO)
End Sub
```

```
'Para escribitr encabezado
Private Sub Encabeza_Click()
Dim fecha As String
EDITOR.Show
End Sub
```

```
Private Sub END_Click()
COMANDO = "END"
insertar (COMANDO)
End Sub
```

```
'Para tomar gripper
Private Sub Gripper_Click()
COMANDO = "CALL JOB:TGRIPPER"
insertar (COMANDO)
End Sub
```

```
'Para almacena run programa
Private Sub guardacomo_Click()
Dim programa As String
denuevo:
CommonDialog1.DialogTitle = "Editor de programas Motoman"
CommonDialog1.DefaultExt = "JBI"
CommonDialog1.Filter = "Programas MOTOMAN|*.JBI"
CommonDialog1.InitDir = "C:"
CommonDialog1.Flags = cdlOFNOverwritePrompt
```

```
CommonDialog1.FileName = ""
CommonDialog1.ShowSave
nombrearch = CommonDialog1.FileName
VENTANA.SaveFile nombrearch, 1
End Sub
```

```
Private Sub Guardar_Click()
interfaz.UPPRG_Click
interfaz.Show
End Sub
```

```
Private Sub HERRONOFF_Click()
If HERRONOFF.Value = 1 Then
COMANDO = "CALL JOB:RBHERRON"
insertar (COMANDO)
HERRONOFF.Caption = "DESACTIVAR HERRAMINETA"
Else
COMANDO = "CALL JOB:RBHERROF"
insertar (COMANDO)
HERRONOFF.Caption = "ACTIVAR HERRAMINETA"
End If
End Sub
```

```
Private Sub HOME_Click()
COMANDO = "CALL JOB:HOME"
insertar (COMANDO)
End Sub
```

```
Private Sub Imprime_Click()
 CommonDialog1.Flags = cdlPDReturnDC + cdlPDNoPageNums
  If VENTANA.SelLength = 0 Then
    CommonDialog1.Flags = CommonDialog1.Flags + cdlPDAllPages
  Else
    CommonDialog1.Flags = CommonDialog1.Flags + cdlPDSelection
  End If
  CommonDialog1.ShowPrinter
  Printer.Print ""
  VENTANA.SelPrint CommonDialog1.hDC

End Sub
```

```
Private Sub INICIO_Click()
COMANDO = "NOP" + vbCrLf + "CALL JOB:HOME"
insertar (COMANDO)
End Sub
```

```
Private Sub MOVL_Click()
Pmove.Show
VENTANA.TEXT = "MOVL"
End Sub
```

```
Private Sub Nuevo_archivo_Click()
EDITOR.Show
End Sub
```

```
Private Sub numblock_Click()
```

```
VENTANA.TEXT = VENTANA.TEXT
End Sub
```

```
Private Sub OPENM_Click()
If OPENM.Value = 1 Then
COMANDO = "CALL JOB:ENSCIPRE"
insertar (COMANDO)
OPENM.Caption = "ABRIR PRENSA"
Else
COMANDO = "CALL JOB:ENSABPRE"
insertar (COMANDO)
OPENM.Caption = "CERRAR PRENSA"
End If
End Sub
```

```
Private Sub Salir_Click()
Dim eleccion As Integer
eleccion = MsgBox("Desea guardar los cambios hechos al programa " +
CommonDialog1.FileTitle, vbYesNoCancel)
Select Case eleccion
Case 6
VENTANA.SaveFile nombrearch, 1
Unload Me
Unload Pmove
Case 7
Unload Me
Unload Pmove
Case 2
Exit Sub
End Select
End Sub
```

```
Private Sub CODE()
VENTANA.TEXT = VENTANA.TEXT + "CALL JOB:" + COMANDO + vbCrLf
End Sub
```

```
Private Sub sologuarda_Click()
VENTANA.SaveFile nombrearch, 1
End Sub
```

```
Public Function escrito(Notas) As String
COMANDO = "'" + Notas
insertar (COMANDO)
End Function
```

```
Private Sub texto_Click()
Notas.Show
End Sub
```

```
Private Sub VENTOSA_Click()
COMANDO = "CALL JOB:TOMVENT"
insertar (COMANDO)
End Sub
```

```
Public Function punto(p As String, vel As String, sel As Integer) As String
Dim pun As String
pun = "P"
If p > 9 Then
```

```
p = pun + "00" + p
Else
p = pun + "000" + p
End If
If Pmove.Option1 = True Then
COMANDO = "MOVJ " + p + " VJ=" + vel + "0"
insertar (COMANDO)
Else
COMANDO = "MOVL " + p + " V=" + vel
insertar (COMANDO)
End If
End Function
```

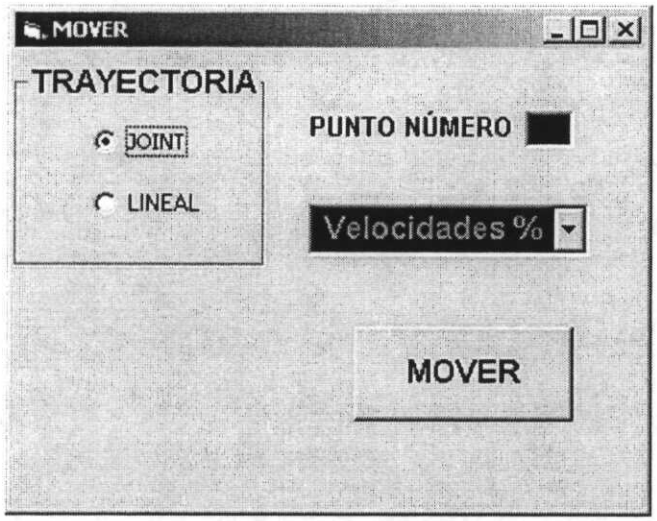
```
Public Function datos(nombre As String) As String
Dim numpos As String
Dim DIA, MES, AÑO, HORA, MINUTO
Dim fecha As String
nombrearch = nombre
DIA = Day(Now)
DIA = CStr(DIA)
If DIA < 10 The
DIA = "0" + DIA
Else
End If
MES = Month(Now)
MES = CStr(MES)
If MES < 10 Then
MES = "0" + MES
Else
End If
AÑO = Year(Now)
AÑO = CStr(AÑO)
HORA = Hour(Now)
HORA = CStr(HORA)
MINUTO = Minute(Now)
MINUTO = CStr(MINUTO)
fecha = AÑO + "/" + MES + "/" + DIA + " " + HORA + ":" + MINUTO

'If EDITOR.Option1 = True Then
'numpos = "///NPOS " + ",0,0,0,0,0"
'Else
'numpos = "///NPOS " + "0,0,0," + ",0,0"
'End If

COMANDO = "/JOB" + vbCrLf + "//NAME " + nombre + vbCrLf + "//INST" + vbCrLf + "///DATE " +
fecha + vbCrLf + "///ATTR SC,RW" + vbCrLf + "///GROUP1 RB1"
insertar (COMANDO)

End Function
```

ANEXO C
CÓDIGO Y DISEÑO PANTALLA PARA LLAMAR POSICIONES




```
Dim vel As String
Dim sel As Integer
```

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

```
Private Sub move_it_Click()
Dim X As String    'Solo para que acepte la función con entrada de 2 variables
p = captura.TEXT                    'Captura el número de punto que se quiere mandar llamar

If captura.TEXT <> "" Then
If Velocidades1.Visible = True Then          'aparecen velocidades en forma JOINT por la elección
del usuario
Select Case Velocidades1.ListIndex
Case 0
 vel = "0.78"
  Case 1
    vel = "1.56"
     Case 2
      vel = "3.12"
       Case 3
        vel = "6.25"
         Case 4
          vel = "12.50"
          Case 5
         vel = "25.0"
     Case 6
    vel = "50.0"
 Case 7
vel = "100.0"
Case Is <> 0, 1, 2, 3, 4, 5, 6, 7
```

```
MsgBox ("DEBE SELECCIONAR UNA VELOCIDAD")
Exit Sub
End Select
    Else
     Select Case Velocidades2.ListIndex                'Aparecen velocidades de la fomra LINEAL
por la elección del usuario
      Case 0
       vel = "11.0"
        Case 1
         vel = "23.0"
          Case 2
           vel = "46.0"
            Case 3
             vel = "93.0"
            Case 4
           vel = "187.0"
          Case 5
         vel = "375.0"
        Case 6
       vel = "750.0"
      Case 7
     vel = "1500.0"
     Case Is <> 0, 1, 2, 3, 4, 5, 6, 7
MsgBox ("DEBE SELECCIONAR UNA VELOCIDAD")
Exit Sub
     End Select
    End If
    If program.Visible = False Then
X = interfaz.a_punto(p:=captura.TEXT, vel:=vel, sel:=sel)        'Manda llamar una función de
interfaz (motoman) y le alimenta las selecciones del usuario de velocidad y tipo de moviemiento
Unload Me
Else
X = program.punto(p:=captura.TEXT, vel:=vel, sel:=sel)
End If
Else
MsgBox ("DEBE ASIGNAR UN NUMERO DE PUNTO")
End If
End Sub
```

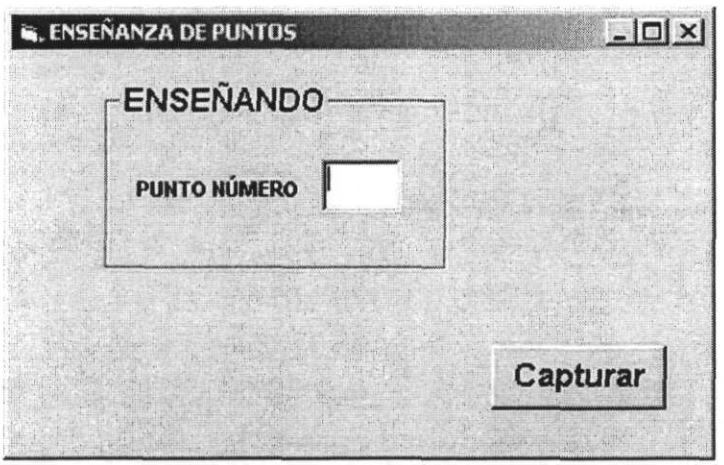
```
Private Sub Option1_Click()
sel = 1
Velocidades1.Visible = True
Velocidades2.Visible = False
End Sub
```

```
Private Sub Option2_Click()
sel = 2
Velocidades1.Visible = False
Velocidades2.Visible = True
End Sub
```

```
Private Sub Text1_Change()
End Sub
```

```
Private Sub StatusBar1_PanelClick(ByVal Panel As MSComctlLib.Panel)
End Sub
```

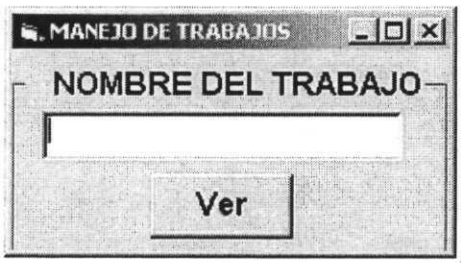
ANEXO D
CÓDIGO Y DISEÑO DE LA PANTALLA PARA GRABAR POSICIONES




```
Private Sub Form_Load()
EndSub
```

```
Public Sub Capturar_Click()
If punto.TEXT <> "" Then
interfaz.MousePointer = 11
Capturar.Enabled = False
p = punto.TEXT          'Envia la información capturada a a motoman.frm
interfaz.posición (p)
Unload Me
Else
MsgBox ("DEBE ASIGNAR UN NÚMERO AL PUNTO")
Exit Sub
End If
End Sub
```

ANEXO E
CÓDIGO Y DISEÑO DEL PANEL MANEJO DE TRABAJOS.



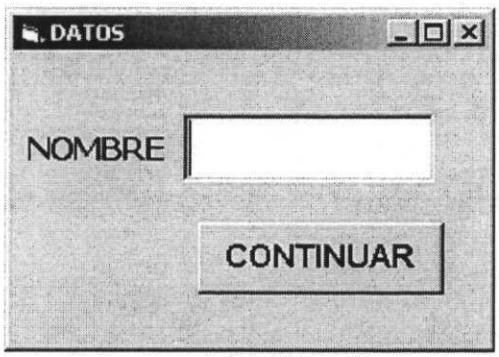

```
Private Sub Label1_Click()
End Sub
```

```
Public Function trabajo() As String
End Function.
```

```
Private Sub Command1_Click()
If Text1.TEXT <> "" Then
jo = Text1.TEXT
interfaz.TEXTEXT (jo)
Unload Me
Else
MsgBox ("DEBE PROPORCIONAR UN NOMBRE")
Exit Sub
End If
End Sub
```

```
Private Sub Form_Load()
JOB.Top = 1800
JOB.Left = 8000
End Sub
```

ANEXO F
CÓDIGO Y DISEÑO DEL PANEL DE ENCABEZADO DE PROGRAMA.




```
Private Sub Command1_Click()
Dim X As String
If nombre.TEXT <> "" Then
X = program.datos(nombre:=nombre.TEXT)
Unload Me
Else
MsgBox ("DEBE PROPORCIONAR UN NOMBRE")
Exit Sub
End If
End Sub
Public Sub Option1_Click()
End Sub
```

ANEXO G
AYUDA MOTOTALK

MOTOTALK 1.0

Software de comunicación para el robot Motoman Yasnac modelo K3

Bienvenida a Mototalk

Bienvenido al software de comunicación para Motoman versión 1.0.
Está diseñado para ser usado en el Laboratorio de Sistemas Integrados de Manufactura del ITESM.

Este programa te permitirá:

- Crear programas para Motoman sin necesidad de conocer a fondo el lenguaje de
- programación.
- Enviar estos programas al controlador del robot o viceversa.
- Enseñanza rápida de posiciones.
- Ordenes de ejecución agilizadas por medio de botones de acceso.
- Edición de programas existentes.
- Manipulación de la memoria del controlador.
- Forma rápida y divertida de aprender el funcionamiento del robot.

CONEXIÓN

Para establecer una conexión debe de verificar lo siguiente:

1. Que el cable serial del robot Motoman que se encuentra en el controlador esté conectado al puerto serial de su computadora COM 1.
2. Que haya presión de aire en el sistema.
3. Que el robot se encuentre encendido.
4. Que el botón REMOTE colocado en el playbackbox se encuentre activado.
5. Que los paros de emergencia estén desactivados.
6. Que el control manual no presente señales de alarma.
7. Que en su computadora se esté ejecutando MOTOTALK.
8. Presione el botón ESTABLECER CONEXIÓN en el panel de control.

PROBLEMAS CON LA CONEXIÓN

¿Qué sucede?

- Envío comandos desde MOTOTALK y no responde
- Causa posible: Probablemente se perdió la conexión:

1. Oprima terminar sesión en la pantalla de control.
2. Desactive y active el botón remote.
3. Pulse el botón conectar en el panel de control.

Se activó una alarma en el control manual pero el botón remote en el controlador principal permanece encendido
Causa posible: error o alarma generada por error del usuario o de la rutina

1. Oprima el botón emergencias en el panel de control.

Se activó una alarma en el controlador manual y el botón remote del controlador principal está desactivado.
Causa posible: Error de comunicación con el robot.

1. Presione terminar conexión en el panel de control de MOTOTALK.
2. Presione el botón F6 en el controlador manual del robot.
3. Presione Remote en el controlador principal.
4. Presione Conexión en el panel de control de Mototalk.

Enciendo el robot y aparece en el controlador: SHOCK SENSOR ACTION
Causa posible: El robot no tiene alimentación de aire

1. Verificar que el compresor que alimenta al robot esté encendido por medio de la revisión de los manómetros de las tuberías de alimentación.
2. Si el compresor está encendido verificar que las válvulas de las tuberías estén paralelas a la tubería (Abiertas.
3. Presione F6 en el control manual del robot.

CREACIÓN DE UN PROGRAMA

1. Para que el robot ejecute una rutina es necesario que cuente con un programa.
2. Los programas se pueden construir en línea o fuera de línea.
3. La creación de un programa se hace por medio del editor de textos.

PASOS PARA CREAR UN PROGRAMA

1. Desde el panel de control oprima el botón "EDITOR".
2. Desde el editor oprima el botón Nombre y escriba el nombre del programa.
3. Presione el botón inicio.
4. Siguiendo un diagrama de puntos establezca las instrucciones de movimiento o los comandos necesarios.

5. Al terminar oprima el botón FIN
6. Guarde el programa.
7. Envíelo al robot por medio el botón "ENVIAR PROGRAMA AL ROBOT".

PASOS PARA INSTRUCCIONES DE MOVIMIENTO

1. Presione el botón MOVE.
2. Seleccione el tipo de movimiento.
3. Asigne número al punto.
4. Establezca la velocidad del punto.

PASOS PARA COMANDOS

1. Seleccione el nombre de la operación que desea y presione el robot correspondiente.

COMANDOS
ENVIO DE PROGRAMAS AL CONTROLADOR.

1. Verifique que hay conexión con el robot Motoman.
2. Presione en el panel de control enviar programa.
3. Seleccione el programa desde el menú de la computadora.
4. Espere a que el programa se cargue en el controlador.
5. En el controlador manual presione el botón "SELECT"
6. Con las flechas puede verificar que el programa se encuentre en la memoria.

RECEPCIÓN DE PROGRAMAS DEL CONTROLADOR.

1. Verifique que hay conexión con el robot Motoman.
2. Presione en el panel de control recibir programa.
3. Coloque en la ventana el nombre del programa que desea recibir.
4. Espere a que el programa sea recibido.

OPERACIÓN

Para Cambiar de modo de operación oprima: PLAY para ejecutar automáticamente o TEACH para enseñanza de puntos.

MODO

Para seleccionar el modo de operación oprima AUTO para ejecutar de manera continua, CYCLE 1 para ejecutar única vez y STEP para ejecutar por pasos.

GUARDAR PUNTOS

1. Lleve al robot al punto deseado.
2. Para guardar un punto presione el botón "Guardar Punto".
3. Espere a que la pantalla desaparezca.

MOVER A PUNTO

1. Presione el botón mover a punto
2. Seleccione el tipo de movimiento (Joint/Lineal)
3. Coloque el número de punto al que quiere moverse
4. Establezca la velocidad dl movimiento.
5. Verifique que el botón "ENEABLE" del control manual esté desactivado
6. Verifique que el botón "PLAY" del controlador esté activado
7. Presione "MOVE" El robot comenzará a moverse.

HERRAMINETAS

1. Seleccionar la herramienta que desea que el robot tome (Gripper, ventosa, desarmador.
2. Verifique que el botón "ENEABLE" del control manual esté desactivado.
3. Verifique que el botón "PLAY" del controlador esté activado.
4. Verifique que no haya obstáculos entre el lugar donde se encuentra el robot y su destino.

ALARMAS Y ERRORES

• Alarmas: Son ocasionadas por problemas en el software y algunas terminan con la existencia de la conexión remota entre el robot y la computadora y otras no. Estás se pueden eliminar desde el panel de control de MOTOTALK. También se pueden eliminar desde el controlador remoto por medio del botón F6

• Errores: Son generados por errores en la aplicación de la lógica del robot. Son marcados por el controlador remoto como errores. Se pueden eliminar desde el panel de control de Mototalk.
También se pueden eliminar por medio del botón CANCEL en el controlador remoto.