5

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

DIVISION DE INGENIERIA ELECTRICA

DESARROLLO DE SOFTWARE CONTROLADOR DEL ROBOT RHINO XR-3

T E S I S

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACION

P R E S E N T A:

HUGO AYALA AMBROCIO

DIRECTOR DE TESIS: DR. JESUS SAVAGE CARMONA

MEXICO, D. F. 2001

ÍNDICE

1. INTRODUCCIÓN

En el momento actual, la mayoría de las tareas de fabricación automatizadas se realizan mediante máquinas de uso especial, diseñadas para realizar funciones predeterminadas en un proceso de manufacturación. La inflexibilidad y generalmente el alto costo de estas máquinas, a menudo llamadas *sistemas de automatización duros*, han llevado a un interés creciente en el uso de robots, capaces de efectuar una variedad de funciones de fabricación en un entorno de trabajo más flexibles y a un menor costo de producción.

Mecánicamente, un robot, se compone de un brazo y una muñeca más una herramienta. Se diseña para alcanzar una pieza de trabajo localizada dentro de su volumen de trabajo. El volumen de trabajo es la esfera de influencia de un robot cuyo brazo puede colocar el submontaje de la muñeca en cualquier punto dentro de la esfera.

La finalidad esencial de un robot es trasladar objetos o herramientas de una posición a otra en el espacio.

Para controlar los movimientos de las diferentes partes de la estructura es necesario estudiar las ecuaciones cinemáticas. Un movimiento mandado, definido por un conjunto de coordenadas tal como "mover el objeto verticalmente hacia arriba tres milímetros manteniendo constante su orientación", en una estructura de robot concreta se transformará en un movimiento de las coordenadas de la articulación, dando una orden tal como "girar la articulación en 30 grados a la derecha, alargar el eslabón 2 en un milímetro y girar la articulación 3 en 30 grados a la izquierda". También es importante ser capaz de hacer una transformación en sentido opuesto; esto es determinar cómo se mueve el objeto en el espacio cuando las articulaciones se mueven con valores determinados. También debe estudiarse el comportamiento dinámico del robot. Varias medidas de ejecución están relacionadas a cómo se mueve el efector terminal en función del tiempo, así debemos encontrar cómo es afectado esto por las fuerzas y pares aplicados por los accionadores.

1.1. HISTORIA

La palabra **robot** proviene de la palabra checa *robota*, que significa trabajo, esta palabra se introdujo en la lengua inglesa en 1921, con el drama satírico R.U.R. de Karel Capek (Rossum Universal Robots). La obra de Capek, es en gran medida responsable de algunas de las creencias mantenidas popularmente acerca de los mismos en nuestro tiempo, incluyendo la perfección de los robots como máquinas humanoides dotadas con inteligencia y personalidades individuales. Esta imagen se reforzó en la película alemana de robots "Metrópolis" de 1926, con el robot andador eléctrico y su perro "Sparko", representada en 1939, en la Feria Mundial de Nueva York , y más recientemente por el robot C3PO, protagonista en la película de 1977, "La guerra de las Galaxias". Ciertamente los robots industriales modernos parecen primitivos cuando se comparan con las expectativas creadas por los medios de comunicación durante las pasadas décadas.

1.2 DESARROLLO DE LOS BRAZOS ROBOT

Los primeros trabajos que condujeron a los robots industriales, de hoy día, se remontan al periodo que siguió inmediatamente a la Segunda Guerra Mundial. Durante los años finales de la década de los cuarenta, comenzaron programas de investigación en Oak Ridge y Argonne National Laboratories, para desarrollar manipuladores mecánicos controlados de forma remota, para manejar materiales radiactivos. Estos sistemas eran del tipo "maestro – esclavo", diseñados para reproducir fielmente los movimientos de mano y brazos realizados por un operario humano. El manipulador maestro era guiado por el usuario a través de una secuencia de movimientos, mientras que el manipulador esclavo duplicaba a la unidad maestra tan fidedignamente tal como era posible.

El trabajo sobre manipuladores maestro-esclavo fue seguido rápidamente por sistemas más sofisticados, capaces de operaciones repetitivas autónomas. A mediados de los años cincuenta, George C. Devol, desarrollo un dispositivo que él llamó "dispositivo de transferencia programada articulada", un manipulador cuya operación podía ser programada (y, por tanto, cambiada) y que podía seguir una secuencia de pasos de movimientos determinados por las instrucciones en el programa. Posteriores desarrollos de este concepto llevan a introducir el primer robot industrial en 1959, por Unimation Inc. el cual usaba una computadora, lo que significaba que podía ser "enseñada" para realizar una variedad de tareas de forma automática.

En 1968, Pieper, estudió el problema cinemático de un manipulador controlado por computadora, mientras que Khan y Roth, en 1971, analizaban la dinámica y el control de un brazo restringido utilizando control bang-bang (casi de tiempo mínimo).

Mientras tanto, otros países (en particular Japón) comenzaron a ver el potencial de los robots industriales. Ya en 1968, la compañía japonesa Kawasaki Heavy Industries, negocio una licencia con Unimation para sus robots.

Durante los años setenta se centro un gran esfuerzo de investigación sobre el uso de sensores externos, para facilitar las operaciones manipulativas. En Stanford, Bolles y Paul (1973), utilizando realimentación tanto visual como de fuerza, demostraron que el brazo Stanford controlado por computadora, conectado a una PDP-10, efectuaba el montaje de bombas de agua de automóvil.

Hoy día tenemos la robótica como un campo de trabajo mucho más amplio que el que teníamos simplemente hace uno pocos años, tratando con investigación y desarrollo en una serie de áreas interdisciplinarias, que incluyen cinemática, dinámica, planificación de sistemas de control, sensores, lenguajes de programación e inteligencia de máquina.

1.3 DEFINICIÓN DEL PROBLEMA.

El problema es desarrollar un software, que sea capaz de controlar un brazo mecánico (robot), permitiendo una interface gráfica con el usuario y que en un momento dado pueda ser modificado por otro diseñador para un fin en especifico. Por ejemplo, acoplarle al robot una cámara y usando un algoritmo de reconocimiento de patrones, podrá reconocer objetos y tomar decisiones, basándose en lo anterior podrá realizar los movimientos que se requieran.

El robot que estamos usando es un RHINO XR-3, con una unidad de control MARK III, ello complica la situación, ya que el desarrollo del software debe de ser compatible con las señales del controlador, de lo contrario el robot no realizará el movimiento deseado.

El Software, estará diseñado para que pueda ser ejecutado desde cualquier computadora que cuente como mínimo el sistema operativo Windows 95, esto debido a que el desarrollo del software de baso en las normas establecidas por el propio sistema operativo, tales como; uso de ventanas dinámicas (que permitan moverlas de un lugar a otro, maximizarlas, minimizarlas, redimencionar, etc.), Manejo de archivos de ayuda para el manejo del software, Compatibilidad de la aplicación con el modo de direccionamiento de 32 bits perteneciente a Windows 95 y posteriores (un motivo más para usar Visual Basic 5 edición empresarial)

Para desarrollar el software, que permitirá controlar el robot Rhino – XR3, es necesario contar con un lenguaje de programación que cumpliera con los requerimientos anteriores y además fuera de aprendizaje rápido y entendible, por ello es que se opto por usar Visual Basic 5 edición empresarial y para el desarrollo de la ayuda se empleo el editor de ayudas llamado RoboHelp, el cual es compatible con el lenguaje de programación ya mencionado.

Por lo anterior, podemos garantizar que se obtendrán resultados satisfactorios como es el contar con el software que controle el robot, el código fuente de la aplicación para futuras mejoras o aplicaciones diversas, ya que se dejara código tanto en Visual Basic 5 en su edición empresarial así como los avances realizados en lenguaje 'C' y de manera análoga los programas desarrollados para la lectura de códigos de interface RS-232.

1.4 JUSTIFICACIÓN DE LA SOLUCIÓN.

La solución se basa en las necesidades del control del robot, es decir conforme se presentan los problemas se van planteando alternativas de solución, que conforme demuestren que son las adecuadas se toman en cuenta como "las soluciones optimas" y por lo tanto son las que describiremos en este trabajo.

Hay que recalcar que hemos trabajado en primer lugar en el desarrollo del software y posteriormente en el presente escrito.

Para encontrar la solución optima consideramos el tipo de comunicación necesaria entre la computadora y el MARK – III, la interface física, etc. Por lo que consideramos importante definir en este apartado las características del robot a las cuales nos tendremos que ajustar:

PIEZA	
Brazo mecánico	➢ Fabricado por Rhino Robots Inc. ➢ La señal para cada motor se realiza por un cable a cada motor que sale de MARK - III. ➢ Robot con 5 grados de libertad.
Controlador MARK - III.	➢ Fabricado por Rhino Robots Inc. Cuenta con un Dip-Switch, el cual tiene los siguientes valores por defecto: ➢ Velocidad 9,600 baudios. ➢ Paridad par. ➢ Bits Stop 2. ➢ Bits de datos 7 ➢ La interface de comunicación es RS232

Para mayores detalles obsérvese el apéndice "A".

2. Interfaces

Quizá la pregunta inicial será, ¿Porqué un capitulo de interfaces? La respuesta a esta pregunta es por necesidad, me explico:

Para tener una adecuada comunicación entre la computadora y el controlador es necesario desarrollar los medios de comunicación adecuados a dicha comunicación es por ello que se vuelve necesario hablar de interfaces físicas, en donde tendremos que entrar en detalle para mencionar los protocolos y velocidades necesarios para dicha comunicación.

2.1 Definición

Durante la carrera nos han dado diversas definiciones de interfaces, pero todas coinciden en el hecho de que sirve para entablar una comunicación, la cual puede ser entre usuario y máquina (interface gráfica), entre máquina con máquina (interface física), esta última es la que describiré en éste capitulo, y en especifico la interface del tipo RS - 232

Una interface la entenderemos como el medio de comunicación entre un objeto (en este caso la computadora) y otro objeto (para este caso el controlador del robot, el MARK III).

2.2 *Tipos y Características*

Existen muchos tipos de interfaces físicas, pero las más comunes por su uso son:

- RS - 232, con sus variantes (C, D)
- X.25
- V.24
- RS – 422
- V.11

Todas las interfaces (aún las no mencionadas), coinciden en que tienen protocolos de uso, es decir cuentan con reglas que deben ser respetadas para obtener un funcionamiento optimo, estas reglas (protocolos) hacen referencia al tipo y nivel de voltaje que debe usar como mínimo y máximo la interface, la distancia máxima que soporta la interface (cantidades mayores, no es seguro que se tenga una buena recepción de los datos transmitidos) uso de Token Ring, del pin (es) de comunicación, el bit se sincronia (en caso de existir), por ello veremos un caso especial el de la interface RS-232C (y su equivalente V.24), que es la interface física que utilice para comunicar la computadora (en este caso la consideramos el DTE) con el controlador Mark III (DCE)

2.3 Caso de estudio: RS-232C.

2.3.1 Justificación

El único motivo, real, que nos llevo a elegir la interface RS-232C fue que esta es el medio de comunicación definido por el controlador del robot, el Mark III, y dado que la idea es utilizar los instrumentos y herramientas que ya se encuentran implementadas. En caso de no haberlo hecho de esta forma hubiéramos tenido que diseñar, construir y probar un nuevo controlador.

2.3.2 Características

La interface RS-232C, definida por EIA (Electronic Industries Association)[1] y la interface V.24 (definida por la International Telegraph and Telephone Consultive Committee), fueron definidos originalmente como la interface standard de conexión de un DTE a un módem (DCE). La separación física entre el DTE y el DCE debe ser relativamente corta y la velocidad máxima de transmisión es relativamente baja (9,600 bps).

Los niveles de las señales usadas por la interface RS-232C (v.24) se muestran en la figura 2.1. Se observa que los voltajes de las líneas son simétricas en 3 volts, respecto a la señal de tierra de referencia, para el cero (0) esta en +3V y para el 1 es –3V. En la practica el encontrar voltajes de ± 12 V o ± 15 V no es raro. Los relativos niveles altos de voltaje usados en estas interfaces hacen que los efectos de las señales (atenuación y ruido) sean menos significativas que en los niveles de los circuitos TTL. Para realizar la conexión entre los equipos comúnmente se usa cable plano o multihilos. Los estándar RS-232C y V.24 especifican una separación física de máximo 15 metros y un rango de transmisión de hasta 9.6 kbps.

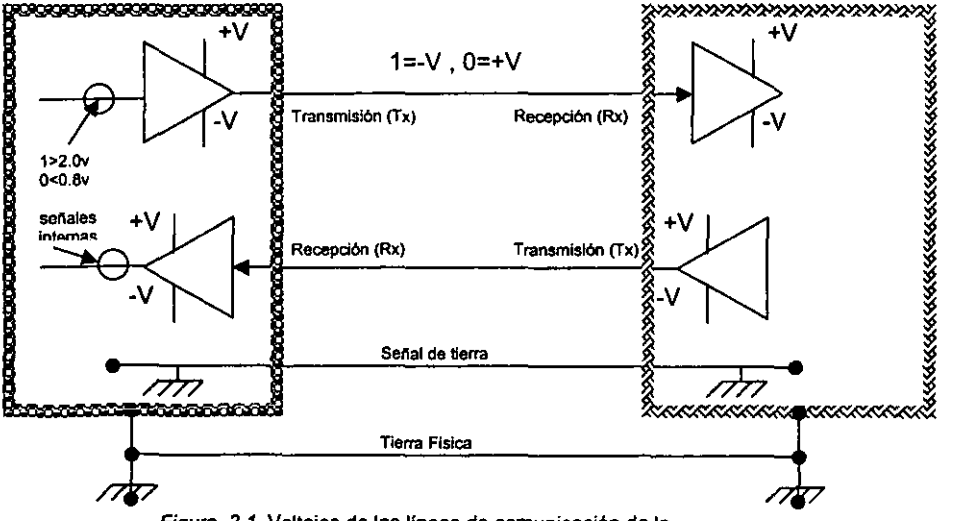


Figura 2.1. Voltajes de las líneas de comunicación de la interface RS-232.[2]

Las líneas de transmisión de datos (Tx) y la de recepción de datos (Rx) son, quizá, las más importantes, sin embargo existen otras líneas de conectividad que realizan el cronometrado y funciones asociadas con el envío y limpiado de la conexión establecida entre el DTE y el DCE. Todas las líneas usan alguna señal eléctrica que se describe en su nombre (fig. 2.2)

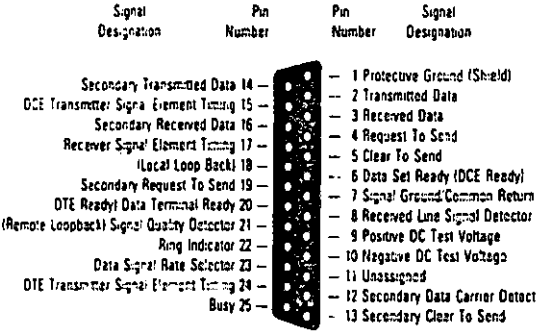


Figura 2.2. El nombre de la señal eléctrica describe su función.[3]

Normalmente las aplicaciones con RS-232C son usadas en comunicaciones binarias de tipo serie entre DCEs y DTEs cuyo rango esta entre 0 y los 20,000 bps. De esta forma los datos más comunes para los que se aplica RS-232C son a la velocidad de 19.2 kbps. De igual forma la limitación en la longitud del cable es de 50 pies.

La EIA, especifica las situaciones en las que RS-232C no es aplicable, donde el aislamiento eléctrico entre los equipos opuestos a la interface es requerido. El hecho de que pensaran que la interface RS-232C, se pudiera dirigir a comunicaciones de datos a largas distancias que involucran la red telefónica, no ha sido impedimento para aplicarla a una gran variedad de comunicaciones a corta distancia, como por ejemplo: de una computadora a una terminal, de computadora a impresora, etc. Realmente nunca pensaron que RS-232C se volvería el estándar de interface a corta distancia en el que se ha convertido, esto es una evidencia más para apoyar el refrán que dice: "cualquier norma es mejor que ninguna norma".

➢ **Características de las señales eléctricas.**

El siguiente resumen de las características de RS-232C se muestran en la fig. 2.3.

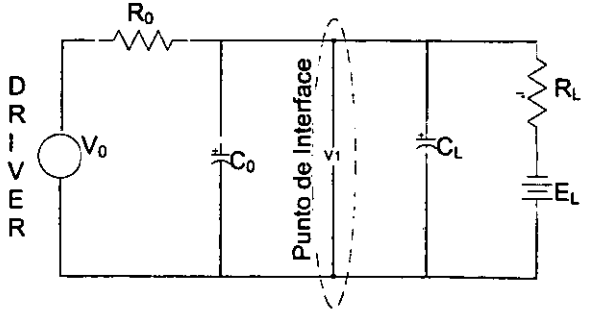


Figura 2.3
Resumen gráfico de las características de la interface RS-232C[4]

1. Una señal en cualquier pin del conector RS-232C tiene un estado asociado con él. Se dice que el estado es uno de los siguientes pares posibles:
   MARCA/ ESPACIO
   ENCENDIDO/ APAGADO
   0 LÓGICO/ 1 LOGICO
   La tabla 2.2 muestra la relación entre los pares antes mencionados y los voltajes señalados. Note que RS-232C emplea lógica negativa, en donde la condición de encendido es asociada con el 0 lógico, mientras que la condición de apagado es asociada con el 1 lógico. El voltaje V_1 señalado es medido con respecto a la señal de tierra del circuito que se discutirá posteriormente. El rango de –3 a +3 V. es la región de transición en la cuál ningún estado señalado se cumple.

Tabla 2.2. Relación entre pares.[5]

Estado (Status)	Voltaje de la Señal	
	$-25V < V_1 < -3$ V	$3V < V_1 < 25$ V
Binario lógico	1	0
Condición de Señal	MARCA	ESPACIO
Función	Apagado	Encendido

2. Para representar un 1 lógico o condición de marca, un driver (controlador) debe aplicar un voltaje entre –5 y –15 volts. Para representar un 0 lógico o condición de espacio el controlador debe aplicar un voltaje entre +5 y +15 v.
   Note que está regla junto con la regla anterior, implica que existe un margen de ruido de 2V. En la norma. La figura 2.4 es una representación gráfica de la relación entre los niveles de voltaje y el estado de la señal. Así, el controlador de la línea, o la fuente de la señal, envía un 0 lógico aplicando un voltaje en el rango de +5 a +15 V. La línea receptora, o destino de la señal, recibe un 0 lógico lo que provoca que busque en el rango de +3 a +15 V. El resultado de esto provoca un defasamiento de 2V. En las señales de origen destino. El defasamiento es similar para la transmisión del 1 lógico.

Habiendo leído lo anterior referente a los voltajes y su importancia en la interface RS-232C, es razonable hacer dos preguntas específicas:

A) ¿Por qué la lógica TTL usa niveles en el rango de 0 a +5 V?
B) ¿Por qué se escoge de –15 a –3 y de +3 a +15 V?

En primer lugar, la salida TTL fue motivada por la necesidad de mejorar la inmunidad al ruido y capacidades de distancia. Segundo, los rangos de voltaje de –15 a –3 y de +3 a +15 estaban normalmente disponibles en la mayoría de los circuitos de la computadora en el momento que la norma RS-232C fue desarrollada. Adicionalmente muchos transistores son capaces de trabajar a estos voltajes, manteniendo las corrientes requeridas. Los rangos de voltaje proporcionan una buena inmunidad al ruido y permiten el funcionamiento a velocidades aceptables (a 20,000 bps.). Es más, las marcas y espacios son representados a través de flujos

opuestos y son diferenciados por un mínimo de 6 V. Esto contribuye a la fiabilidad de la transferencia de los datos.

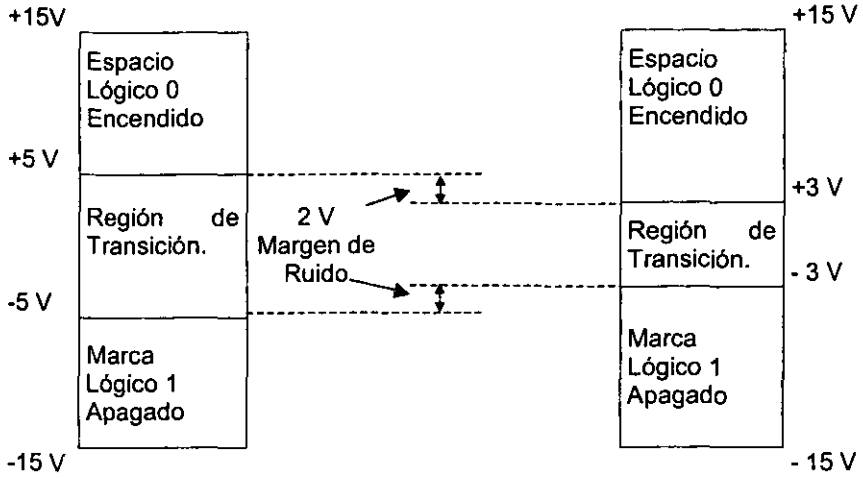


Figura 2.4. representación gráfica de la relación entre los niveles de voltaje y el estado de la señal[6]

3. La capacitancia de desviación C_L del lado terminador de un circuito RS-232C no debe exceder 2,500 pf incluida la capacitancia del cable. Note que ésta regla contribuye significativamente a la limitación de 50 pies de longitud del cable.
4. El voltaje en circuito abierto (o sin carga) V_o no debe exceder los 25 V.
5. Un circuito controlador RS-232C debe poder resistir un corto circuito con cualquier alambre en el cable sin sufrir daño en él o cualquier equipo asociado (incluye terminal, módem, puerto de la computadora y cualquier otro dispositivo que pudiera conectarse con el cable)

➢ **Características Mecánicas de la Interface.**

Las asignaciones de los pines en la interface RS-232C se muestra en la tabla 2.4. Note que esta asignación no hace mención al DB-25 (también llamado conector tipo D de 25 pines), a pesar de que son típicamente asociados a él. El conector DB-25 que se usa siempre con las interfaces RS-232C es compatible con ISO 2113, una norma promulgada por la Organización Internacional para la Regulación (ISO, por sus siglas en ingles). El documento para obtener información más detallada es:

ISO Draft International Standard 2110, "Data Communication: 25-pin DTE/DCE Interface Connector an pin Assignment" (Revision de ISO 2110-1972). Febrero de 1979.[7]

Note que un conector macho es asociado con un DTE y un conector hembra es asociada con DCE.

Tabla 2.4

Pin	Circuito	Descripción
1	AA	Tierra de Protección (Física)
2	BA	Transmisión de Datos
3	BB	Recepción de Datos
4	CA	Demanda de Envío
5	CB	Limpiar Envío
6	CC	Los Datos están listos
7	AB	Señal de Tierra (lógica)
8	CF	Línea recibida por el Detector de señales
9 / 10	-	Reservado para Datos de prueba
11	-	Sin señalización
12	SCF	Línea Secundaria del Detector de Señales
13	SCB	Línea Secundaria de Limpiar Envío
14	SBA	Línea Secundaria de Transmisión de Datos
15	DB	Transmisión de señal de elemento cronometrado
16	SBB	Línea Secundaria de Datos Recibidos
17	DD	Recepción de señal de elemento cronometrado
18	-	Sin Señal
19	SCA	Línea Secundaria de Demanda de envío
20	CD	Terminación de Datos
21	CG	Detector de Señal de Calidad
22	CE	Indicador de llamada (Ring Indicator)
23	CH / CI	Los datos señalan proporción
24	DA	Transmisión de Señal de elemento cronometrado
25	-	Sin Señal

Basándose en lo anterior y consultando el manual de usuario de una computadora Gateway 2000, observé que existe un manejo análogo para un DB-9, lo que me permitió deducir la tabla 2.5 de correspondencia entre un DB-25 y un DB-9. En ella se especifica como es la conexión en las configuraciones DB-9 con DB-9 y DB-9 con DB-25 y DB-25 con DB-25. Esto debido a que es común que el puerto de salida de la computadora (DCE) serie número 1 sea de salida DB-9 macho, y el puerto de salida de la computadora (DCE) serie número 2 es DB-25 macho, y la entrada del controlador Mark-III (DTE) es un DB-25.

Tabla 2.5. Correspondencia de pines de DB-9 a DB-25

DB-9 (PIN)	DB-25 (Corresponde al PIN)	Descripción
1	8	Línea recibida por el Detector de señales
2	3	Recepción de Datos
3	2	Transmisión de Datos
4	20	Terminación de Datos
5	7	Señal de Tierra (lógica)
6	6	Los Datos están listos
7	4	Demanda de Envío
8	5	Limpiar Envío
9	22	Indicador de llamada (Ring Indicator)

➢ **Características Funcionales de los Circuitos**

Los circuitos mostrados en la tabla 2.4, pueden ser divididos en cinco categorías:

- ✓ De Tierra o Retorno Común (A)
- ✓ Circuitos de Datos (B)
- ✓ Circuitos de Control (C)
- ✓ Circuitos Temporizadores (D)
- ✓ Circuitos de Canal Secundario (S)

La letra en el paréntesis después de cada categoría es el primer circuito de dos o tres designaciones que es usada en la mayoría de la literatura que discute los signos de la interface RS-232C.

3. Robots

La maquinaria para la automatización rígida dio paso al robot con el desarrollo de controladores rápidos, basados en el microprocesador, así como con el empleo de servos en bucle cerrado, que permiten establecer con exactitud la posición real de los elementos del robot y establecer el error con la posición deseada. Esta evolución ha dado origen a una serie de tipos de robots:

3.1 Manipuladores

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

a) Manual. Cuando el operario controla directamente la tarea del manipulador.
b) De secuencia fija. Cuando se repite, de forma variable, el proceso de trabajos preparado previamente.
c) De secuencia variable. Se pueden alterar algunas características de los ciclos de trabajo.

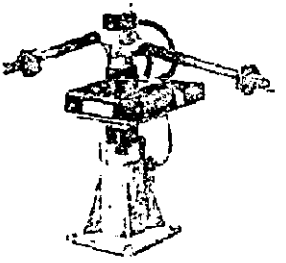


Figura 3.1
Robot tipo Neumático.[8]

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

3.2. Clasificación de los Robots

Robots de repetición o aprendizaje[9]

Son manipuladores que se limitan a repetir una secuencia de movimientos, previamente ejecutada por un operador humano, haciendo uso de un controlador manual o un dispositivo auxiliar. En este tipo de robots, el operario en la fase de enseñanza, se vale de una *pistola de programación* con diversos pulsadores o teclas o bien, de joystics, o bien utiliza un maniquí, o a veces desplaza directamente la mano del robot.

Los robots de aprendizaje son los más conocidos, hoy día, en los ambientes industriales y el tipo de programación que incorporan, recibe el nombre de "gestual".

Robots con control por computadora[10]

Son manipuladores o sistemas mecánicos multifuncionales, controlados por una computadora. En este tipo de robots, el programador no necesita mover realmente el elemento de la máquina, cuando la prepara para realizar un trabajo. El control por computadora dispone de un lenguaje especifico, compuesto por varias instrucciones adaptadas al robot, con las que se puede confeccionar un programa de aplicación utilizando sólo la computadora, no el brazo. A esta programación se le denomina textual y se crea "off-line", es decir, sin la intervención del manipulador

Robots inteligentes[11]

Son similares a los del grupo anterior, pero, además, son capaces de relacionarse con el mundo que les rodea a través de sensores y tomar decisiones en tiempo real (autoprogramables). La visión, el sonido de máquina y la inteligencia artificial, son las ciencias que más se están estudiando para su aplicación en los robots inteligentes.

Micro – Robots[12]

Con fines educacionales, de entrenamiento o investigación, existen numerosos robots de formación o micro - robots a un precio accesible y, cuya estructura y funcionamiento son similares a los de aplicación industrial.

Existen diferentes formas de clasificar a los robots, pero inicialmente la que se ha expuesto es la que facilita una primera aproximación al estudio de la Robótica.

La definición del "Robot Industrial" y consecuentemente su clasificación, variará con el progreso y lo que hoy se considera un robot avanzado, parará a ser un recuerdo histórico, como ha sucedido con las computadoras.

3.3 Grados de libertad de un robot.

Son los parámetros que se precisan para determinar la posición y la orientación del elemento terminal del robot (manipulador). También se pueden definir los grados de libertad, como los posibles movimientos básicos (giros y desplazamiento) independientes. En la figura 3.2 se muestra el esquema de un robot de estructura moderna con 6 grados de libertad; tres de ellos determinan la posición en el espacio del aprehensor (q_1, q_2 y q_3) y los otros 3, la orientación del mismo (q_4, q_5 y q_6).

Un mayor número de grados de libertad conlleva un aumento de la flexibilidad en el posicionamiento del elemento terminal. Aunque la mayoría de las aplicaciones industriales requieren 6 grados de libertad, como la soldadura, mecanizado y paletización, otras más complejas exigen un número mayor, tal es el caso en las labores de montaje. Tareas más sencillas y con movimientos más limitados, como las de pintura y paletización, suelen exigir 4 ó 5 grados de libertad.

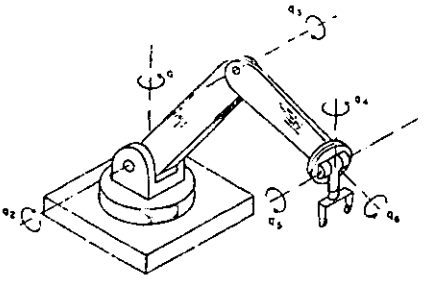


Figura 3.2
Esquema de un manipulador con 6 grados de libertad.[13]

3.4 Anatomía de los Robots

Articulaciones[14]

Son comunes dos tipos de articulaciones: la prismática y la giratoria. Una junta prismática, también conocida como junta deslizante, posibilita a un eslabón deslizarse en línea recta sobre otro. Una junta giratoria, si consideramos el caso el caso de un grado de libertad, toma la forma de una bisagra entre un eslabón y el próximo.

Una junta de rotula (bola-cazoleta) tiene el mismo efecto que combinar tres juntas de revolución sencillas. Dos o más articulaciones de estas pueden combinarse estrechamente.

Eslabones[15]

Con objeto de lograr la respuesta más rápida posible para un movimiento dado y un sistema de accionamiento, los eslabones que forman la estructura deben mantenerse lo más ligero posible. Con objeto de que las deformaciones de los eslabones bajo cargas estáticas y dinámicas se mantengan al mínimo, los eslabones deben ser también tan rígidos como sea posible. Estas dos exigencias son antagónicas y debe alcanzarse un compromiso.

Zona de trabajo y dimensiones del manipulador[16]

Las dimensiones de los elementos del manipulador, junto a los grados de libertad, definen la zona de trabajo del robot, característica fundamental en las fases de selección e implantación del modelo adecuado.

La zona de trabajo se subdivide en áreas diferenciadas entre sí, por la accesibilidad especifica del elemento terminal en cada una de ellas. Por ejemplo, la zona en la que se puede orientar horizontalmente el elemento terminal (aprehensor o herramienta), es diferente a la que permite orientarlo verticalmente o con el determinado ángulo de inclinación. De igual forma queda la zona de trabajo por los límites de giro y desplazamiento que existen en las articulaciones.

3.5 Características:

Capacidad de Carga.[17]

El peso, en kilogramos, que puede transportar la garra del manipulador recibe el nombre de capacidad de carga. A veces, este dato lo proporcionan los fabricantes, incluyendo el peso de la propia garra.

La capacidad de carga es una de las características que más se tienen en cuenta en la selección del robot, según la tarea a la que se destine. En soldadura y mecanizado es común precisar capacidades de carga superiores a los 50 Kgs.

Precisión en la repetibilidad.[18]

Esta magnitud establece el grado de exactitud en la repetición de los movimientos de un manipulador al realizar una tarea programada.

Dependiendo del trabajo que se deba realizar, la precisión en la repetibilidad de los movimientos es mayor o menor. Así por ejemplo, en labores de ensamblaje de piezas, dicha característica ha de ser menor que $\pm$ 0.1 mm. En soldadura, pintura y manipulación de piezas, la precisión en la repetibilidad esta comprendida entre 1 y 3 mm. Y en las operaciones de mecanizado, la precisión ha de ser menor de 1 mm.

Velocidad.[19]

En muchas ocasiones, una velocidad de trabajo elevada, aumenta extraordinariamente el rendimiento del robot, por lo que esta magnitud se valora considerablemente en la elección del mismo.

En tareas de soldadura y manipulación de piezas es muy aconsejable que la velocidad de trabajo sea alta, En pintura, mecanizado y ensamblaje, la velocidad debe ser media e incluso baja.

Coordenadas de los movimientos[20]

La estructura del manipulador y la relación entre sus elementos proporciona una configuración mecánica, que da origen al establecimiento de los parámetros que hay que conocer para definir la posición y orientación del elemento terminal. Fundamentalmente, existen cuatro estructuras clásicas en los manipuladores, que se relacionan con los correspondientes modelos de coordenadas en el espacio y que son los que muestran en la figura 3.3:

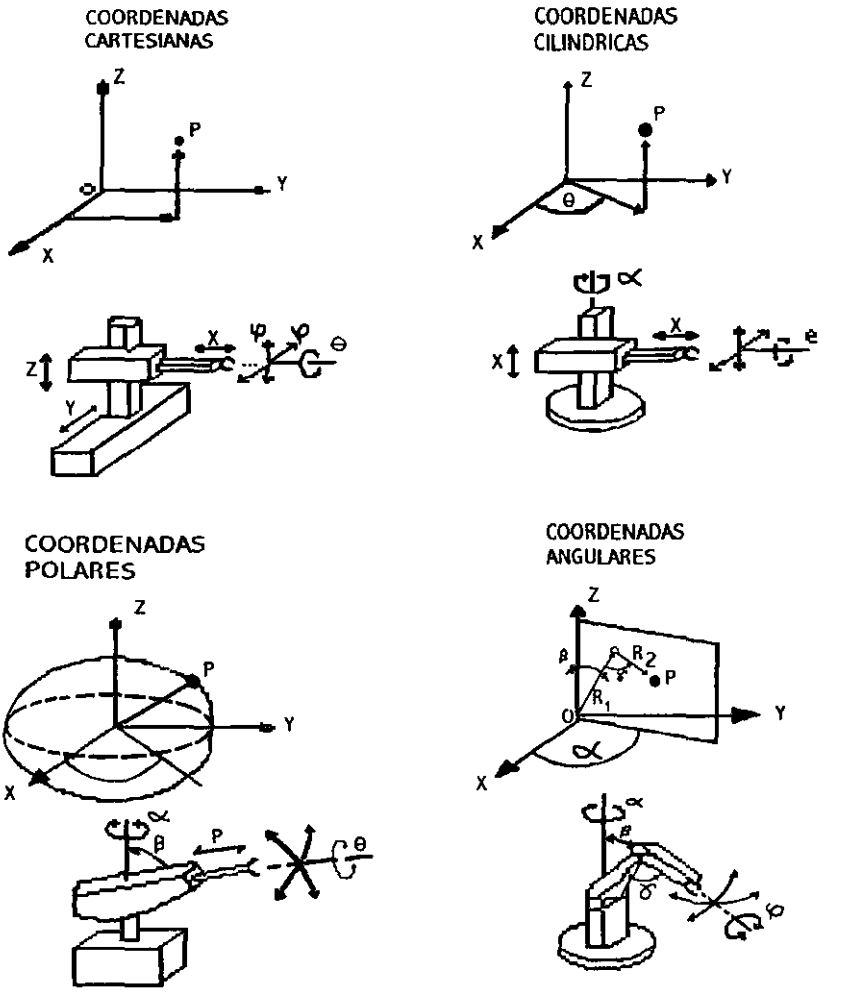


Figura 3.3. Modelos de coordenadas empleadas para una ubicación espacial[21].

4. CINEMÁTICA DEL ROBOT[22]:

4.1 Localización Espacial

La manipulación de piezas llevada a cabo por un robot, implica el movimiento espacial de su extremo. Asimismo, para que el robot pueda recoger una pieza, es necesario conocer la posición y orientación de ésta con respecto a la base del robot. Se aprecia entonces la necesidad de contar con una serie de herramientas matemáticas que permitan especificar la posición y orientación en el espacio de cualquier objeto.

Estas herramientas deben ser lo suficientemente potentes como para permitir obtener de forma sencilla relaciones espaciales entre distintos objetos y especial entre éstos y el robot. Es necesario resaltar que éstas son de aplicación general para el tratamiento de problemas de localización espacial y que, por tanto, no son de aplicación exclusiva en el campo de la robótica.

Matrices de Transformación Homogénea

Las coordenadas homogéneas se han introducido para resolver el problema de representar de forma conjunta la posición como la orientación de un objeto. La representación mediante coordenadas homogéneas de la localización de sólidos en un espacio n-dimensional se realiza a través de un espacio (n+1)-dimensional. Es decir, un espacio n-dimensional esta representado en coordenadas homogéneas por (n+1) dimensiones, de tal forma que un vector p(x,y,z) vendrá representado por (wx,wy,wz,w), donde w tiene un valor arbitrario y representa un factor de escala. De forma general, un vector p=ai+bj+ck, se representa en coordenadas homogéneas mediante el vector columna:

$$p = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} aw \\ bw \\ cw \\ w \end{bmatrix} = \begin{bmatrix} a \\ b \\ c \\ 1 \end{bmatrix} \qquad 4.0$$

Por ejemplo, el vector 2i+3j+4k se puede representar en coordenadas homogéneas como$[2,3,4,1]^T$ o como $[4,6,8,2]^T$, etc. Los vectores nulo se representan como $[0,0,0,n]^T$ donde n es no-nulo. Los vectores de la forma $[a,b,c,0]^T$ sirven para representar direcciones, pues representan vectores de longitud infinita.

A partir de la definición de las coordenadas homogéneas surge el concepto de transformación homogénea. Se define como matriz de transformación homogénea T a una matriz de dimensión 4X4 que representa la transformación de un vector de coordenadas homogéneas de un sistema de coordenadas a otro.

$$T = \begin{bmatrix} R_{3X3} & P_{3X1} \\ f_{1X3} & w_{1X1} \end{bmatrix} = \begin{bmatrix} Rotación & Traslación \\ Perspectiva & Escalado \end{bmatrix} \qquad 4.1$$

De lo anterior se puede considerar que una matriz homogénea se haya compuesta de cuatro submatrices de distinto tamaño: una submatriz R_{3X3} que corresponde a una matriz de rotación; una submatriz P_{3X1} que corresponde al vector de traslación; una submatriz f_{1X3} que representa una transformación de perspectiva, y una submatriz w_{1X1} que representa el escalado global. En robótica generalmente sólo interesa conocer el valor de R_{3X3} y de P_{3X1}, considerándose las componentes de f_{1X3} nulas y la de w_{1X1} la unidad, por lo que la matriz homogénea T resulta ser de la forma:

$$T = \begin{bmatrix} R_{3X3} & P_{3X1} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} Rotación & Traslación \\ 0 & 1 \end{bmatrix} \qquad 4.2$$

Que representa la orientación y posición de un sistema O'UVW rotado y trasladado con respecto al sistema de referencia OXYZ. Esta matriz sirve para conocer las coordenadas (r_x,r_y,r_z) del vector r en el sistema OXYZ a partir de sus coordenadas (r_u,r_v,r_w) en el sistema O'XYZ:

$$\begin{bmatrix} r_x \\ r_y \\ r_z \\ 1 \end{bmatrix} = T \begin{bmatrix} r_u \\ r_v \\ r_w \\ 1 \end{bmatrix} \qquad 4.3$$

También se puede utilizar para expresar la rotación y traslación de un vector respecto de un sistema de referencia fijo OXYZ, de tal manera que un vector r_{xyz} rotado según R_{3X3} y trasladado según P_{3X1} se convierte en el vector r'_{xyz} dado por:

$$\begin{bmatrix} r'_x \\ r'_y \\ r'_z \\ 1 \end{bmatrix} = T \begin{bmatrix} r_u \\ r_v \\ r_w \\ 1 \end{bmatrix} \qquad 4.4$$

En resumen, una matriz de transformación homogénea se puede aplicar para:

1. Representar la posición y orientación de un sistema girado y trasladado O'UVW con respecto a un sistema fijo de referencia OXYZ, que es lo mismo que representar una rotación y traslación realizada sobre un sistema de referencia.
2. Transformar un vector expresado en coordenadas con respecto a un sistema O'UVW, a su expresión en coordenadas del sistema de referencia OXYZ.
3. Rotar y trasladar un vector con respecto a un sistema de referencia fijo OXYZ.

Se hace notar que se utilizan coordenadas homogéneas con factor de escalado igual a la unidad, y que por tanto los vectores que intervienen en las transformaciones han de poseer cuatro componentes. Por comodidad, se elige el factor de escalado w=1.

Sí analizamos con más detalle el empleo de las matrices homogéneas como herramientas para representar la localización de objetos en el especio tridimensional, así como para realizar proyecciones y escalados, tendremos que para:

Traslación:

Supóngase que el sistema O'UVW únicamente se encuentra trasladado en vector $p=p_x i+p_y j+p_z k$ con respecto al sistema OXYZ, esto implica que la matriz T correspondiente (según 4.1) es de tal forma que sólo estará presente de manera predominante la parte de la traslación:

$$T(p)=\begin{bmatrix} 1 & 0 & 0 & P_x \\ 0 & 1 & 0 & P_y \\ 0 & 0 & 1 & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad 4.5$$

La cual llamamos matriz básica de traslación.

Un vector cualquiera r, representado en el sistema O'UVW por r_{uvw}, tendrá como componentes del vector con respecto al sistema OXYZ:

$$\begin{bmatrix} r_x \\ r_y \\ r_z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & P_x \\ 0 & 1 & 0 & P_y \\ 0 & 0 & 1 & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_u \\ r_v \\ r_w \\ 1 \end{bmatrix} = \begin{bmatrix} r_u + P_x \\ r_v + P_y \\ r_w + P_z \\ 1 \end{bmatrix} \qquad 4.6$$

Rotación:

La matriz de Rotación define la orientación del sistema OUVW con respecto al sistema OXYZ. Recibe el nombre de matriz de cosenos directores y se trata de una matriz ortonormal, tal que la inversa de la matriz R es igual a su transpuesta: $\mathbf{R^{-1}} = \mathbf{R^T}$.

La principal utilidad de esta matriz de rotación corresponde a la representación de la orientación de sistemas girados únicamente sobre uno de los ejes principales del sistema de referencia.

En la figura 4.1-b, la orientación del sistema OUVW, con el eje OU coincide con el eje OX, vendrá representada mediante la matriz:

$$R(x,\alpha)=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos\alpha & -sen\alpha\\ 0 & sen\alpha & \cos\alpha\end{bmatrix} \qquad 4.7$$

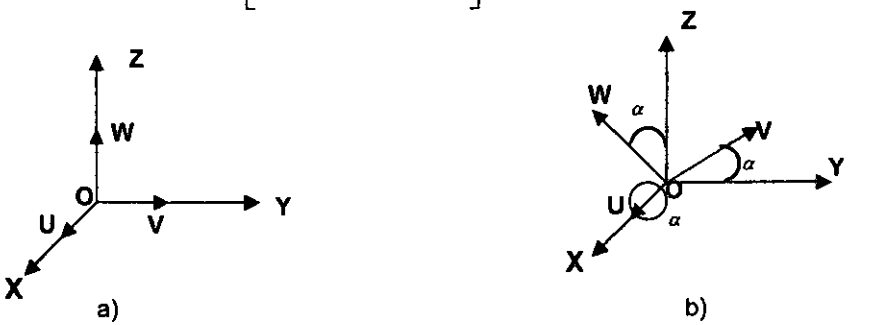


Figura 4.1. Sistema de referencia OXYZ y solidario al objeto OUVW

En la figura 4.2-a se observa la orientación del sistema OUVW, con el eje OV coincidente con el eje OY, esto vendrá representada mediante la matriz:

$$R(y,\phi)=\begin{bmatrix}\cos\phi & 0 & sen\phi\\ 0 & 1 & 0\\ -sen\phi & 0 & \cos\phi\end{bmatrix} \qquad 4.8$$

En la parte b de la figura 4.2, la orientación del sistema OUVW, con el eje OW coincide con el eje OZ, será representada mediante la matriz:

$$R(z,\theta)=\begin{bmatrix}\cos\theta & -sen\theta & 0\\ sen\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix} \qquad 4.9$$

Figura 4.2, Rotación del sistema OUVW con respecto a los ejes OY y OZ, respectivamente.

Supóngase ahora que el sistema O'UVW sólo se encuentra rotado con respecto al sistema OXYZ. La submatriz de rotación R_{3x3} será la que defina la rotación, y se corresponde al tipo matriz de rotación presentada anteriormente (4.7,4.8 y 4.9). De igual forma como se hizo allí, se pueden definir tres matrices homogéneas básicas de rotación según se realice ésta según cada uno de los tres ejes coordenados OX, OY y OZ del sistema de referencia OXYZ:

$$T(x,\alpha)=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & \cos\alpha & -sen\alpha & 0\\ 0 & sen\alpha & \cos\alpha & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad 4.10$$

$$T(y,\phi)=\begin{bmatrix}\cos\phi & 0 & sen\phi & 0\\ 0 & 1 & 0 & 0\\ -sen\phi & 0 & \cos\phi & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad 4.11$$

$$T(z,\theta)=\begin{bmatrix}\cos\theta & -sen\theta & 0 & 0\\ sen\theta & \cos\theta & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad 4.12$$

Un vector cualquiera r, representado en el sistema girado O'UVW por r_{uvw} tendrá como componentes (r_x, r_y, r_z) en el sistema OXYZ las siguientes:

$$\begin{bmatrix} r_x \\ r_y \\ r_z \\ 1 \end{bmatrix} = T \begin{bmatrix} r_u \\ r_v \\ r_w \\ 1 \end{bmatrix} \qquad 4.13$$

Y a su vez un vector $r_{x,y,z}$ rotado según T vendrá expresado por $r'_{x,y,z}$ según:

$$\begin{bmatrix} r'_x \\ r'_y \\ r'_z \\ 1 \end{bmatrix} = T \begin{bmatrix} r_x \\ r_y \\ r_z \\ 1 \end{bmatrix} \qquad 4.14$$

La principal ventaja de las matrices homogéneas reside en su capacidad de representación conjunta de posición y orientación. Ésta representación se realiza utilizando al mismo tiempo la matriz de rotación $\mathbf{R}_{3x3}$ y el vector de traslación $\mathbf{p}_{3x1}$ en una misma matriz de transformación homogénea (según 4.1).

La traslación y la rotación son transformaciones que se realizan en relación a un sistema de referencia. Por lo tanto, si se quiere expresar la posición y orientación de un sistema O'UVW, originalmente coincidente con el de referencia y que ha sido rotado y trasladado según éste, habrá que tener en cuenta si primero se ha realizado la rotación y después la traslación o viceversa, pues se trata de transformaciones espaciales no conmutativas.

Como ya se ha mencionado, una matriz homogénea sirve para transformar un vector expresado en coordenadas homogéneas con respecto a un sistema O'UVW, a su expresión en las coordenadas del sistema de referencia OXYZ. También se puede utilizar para rotar y girar un vector referido a un sistema de referencia fijo, y en definitiva sirve para expresar la orientación y posición de un sistema de referencia O'UVW con respecto a otro fijo OXYZ. La matriz T de transformación se suele escribir de la siguiente forma:

$$T = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n & o & a & p \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad 4.15$$

Donde **n,o,a**, es una terna ortonormal que representa la orientación y **p** es un vector que representa la posición.

Si se considera un vector $r_{uvw}=[0,0,0,1]^T$, es decir, el origen del sistema O'UVW, la aplicación de la matriz T que representa la transformación (traslación + rotación) de O'UVW con respecto a OXYZ, se obtiene r_{xyz}:

$$r_{xyz}=\begin{bmatrix} n & o & a & p \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}=\begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} \qquad 4.16$$

que coincide con el vector columna p de T. Por tanto, este vector columna representa la posición del origen de O'UVW con respecto del sistema OXYZ.

Si de igual manera, se considera el vector de coordenadas homogéneas $[1,0,0,1]^T$ con respecto del sistema OUVW, es decir, el vector director del eje coordenado O'U del sistema O'UVW, y suponiendo el vector p de traslación nulo, se tendrá:

$$\begin{bmatrix} r_x \\ r_y \\ r_z \\ 1 \end{bmatrix}=\begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}=\begin{bmatrix} n_x \\ n_y \\ n_z \\ 1 \end{bmatrix} \qquad 4.17$$

Es decir, el vector columna **n** representa las coordenadas del eje O'U del sistema O'UVW con respecto del sistema OXYZ. De igual forma, si se realiza la transformación de los vectores $[0,1,0,1]^T$ y $[0,0,1,1]^T$ referidos al sistema O'UVW, se obtiene que el vector columna **o** representa las coordenadas del eje OY del sistema O'UVW con respecto del sistema OXYZ, y que el vector columna **a** representa las coordenadas del eje O'W del sistema O'UVW con respecto del sistema OXYZ

Consecuentemente, los vectores **n,o** y **a** definen una terna ortonormal a derechas, lo que significa que:

$$|n|=|o|=|a|=1$$
$$\mathbf{n \times o = a} \qquad 4.18$$

la submatriz de rotación **[n,o,a]** corresponde a una matriz ortonormal, que cumple:

$$[n\ o\ a]^{-1} = [n\ o\ a]^{t} \qquad 4.19$$

La matriz inversa de la matriz homogénea de transformación T es fácilmente obtenible, y corresponde a la siguiente expresión:

$$T^{-1} = \begin{bmatrix} n_x & n_y & n_z & -n^T p_x \\ o_x & o_y & o_z & -o^T p_y \\ a_x & a_y & a_z & -a^T p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad 4.20$$

Si se tiene la relación $r_{xyz} = T\ r_{uvw}$ y se multiplica en ambos miembros por T^{-1}, se tiene:

$$T^{-1}\ r_{xyz} = r_{uvw} \qquad 4.21$$

por lo que, realizando el mismo proceso que se hizo anteriormente, se deduce que los vectores fila de la submatriz de rotación de la matriz T (vectores columna de la submatriz de rotación T^{-1}), representan los ejes principales del sistema de coordenadas de referencia OXYZ con respecto a OUVW. Es decir, los vectores fila de la matriz [n o a] representan otra terna ortonormal a derechas.

Ya se ha mencionado que una matriz de transformación homogénea sirve, entre otras cosas para representar el giro y la traslación realizados sobre un sistema de referencia. Esta utilidad de las matrices homogéneas cobra aún más importancia cuando se componen las matrices homogéneas para describir diversos giros y traslaciones consecutivos sobre un sistema de referencia determinado.

De esta forma, una transformación compleja podrá descomponerse en la aplicación consecutiva de transformaciones simples (giros básicos y traslaciones).

Por ejemplo, una matriz que representa un giro de un ángulo α sobre el eje OX, seguido de un giro de ángulo ϕ sobre el eje OY y de un giro de un ángulo θ sobre el eje OZ, puede obtenerse por la composición de las matrices básicas de rotación (composición de 4.10, 14.11 y 4.12):

$$T = T(z,\theta)T(y,\phi)T(x,\alpha) = \begin{bmatrix} \cos\theta & -\operatorname{sen}\theta & 0 & 0 \\ \operatorname{sen}\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\phi & 0 & \operatorname{sen}\phi & 0 \\ 0 & 1 & 0 & 0 \\ -\operatorname{sen}\phi & 0 & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\operatorname{sen}\alpha & 0 \\ 0 & \operatorname{sen}\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \qquad 4.22$$

$$= \begin{bmatrix} \cos\phi\cos\theta & -\operatorname{sen}\theta\cos\alpha + \cos\theta\operatorname{sen}\phi\operatorname{sen}\alpha & \operatorname{sen}\theta\operatorname{sen}\alpha + \cos\theta\operatorname{sen}\phi\cos\alpha & 0 \\ \operatorname{sen}\theta\cos\phi & \cos\theta\cos\alpha + \operatorname{sen}\theta\operatorname{sen}\phi\operatorname{sen}\alpha & -\cos\theta\operatorname{sen}\alpha + \operatorname{sen}\theta\operatorname{sen}\phi\cos\alpha & 0 \\ -\operatorname{sen}\phi & \cos\phi\operatorname{sen}\alpha & \cos\alpha\cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Debido a que el producto de matrices no es conmutativo, tampoco lo es la composición de transformaciones.

De forma general, a la hora de componer diversas transformaciones mediante matrices homogéneas, se ha de tener en cuenta los siguientes criterios:

1. Si el sistema fijo OXYZ y el sistema transformado O'UVW son coincidentes, la matriz homogénea de transformación será la matriz 4x4 identidad I_4.
2. Si el sistema O'UVW, se obtiene mediante rotaciones y traslaciones definidas con respecto al sistema fijo OXYZ, la matriz homogénea que representa cada transformación se deberá pre-multiplicar sobre las matrices de las transformaciones previas.
3. Si el sistema O'UVW, se obtiene mediante rotaciones y traslaciones definidas con respecto al sistema móvil, la matriz homogénea que representa cada transformación se deberá post-multiplicar sobre las matrices de transformaciones previas.

Siguiendo estas indicaciones, cualquier composición de matrices homogéneas puede estudiarse como si se realiza cada transformación con respecto al sistema fijo o se realiza cada transformación con respecto al sistema móvil.

4.2 Cinemática Inversa (métodos)

La cinemática del robot, estudia el movimiento del mismo con respecto a un sistema de referencia. Así, la cinemática se interesa por la descripción analítica del movimiento espacial del robot como una función del tiempo, y en particular por las relaciones entre la posición y la orientación del extremo final del robot con los valores que toman sus coordenadas articulares.

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

Denavit y Hartenberg propusieron un método sistemático para describir y representar la geometría espacial de los elementos de una cadena cinemática, y en particular de un robot, con respecto a un sistema de referencia fijo. Éste método utiliza una matriz de transformación homogénea para describir la relación espacial entre dos elementos rígidos adyacentes, reduciéndose el problema cinemático directo a encontrar una matriz de transformación homogénea de 4X4 que relacione la localización espacial del extremo del robot con respecto al sistema de coordenadas de su base.

El objetivo del problema cinemático inverso consiste en encontrar los valores que deben adoptar las coordenadas articulares del robot $q=[q_1,q_2, ..., q_n]^T$ para que su extremo se posicione y oriente según una determinada localización espacial.

Así como es posible abordar el problema cinemático directo de una manera sistemática a partir de la utilización de matrices de transformación homogéneas, e independientemente de la configuración del robot, no ocurre lo mismo con el problema cinemático inverso, siendo el procedimiento de obtención de las ecuaciones fuertemente dependiente de la configuración del robot.

Se han desarrollado algunos procedimientos genéricos susceptibles de ser programados, de modo que la computadora pueda, a partir del conocimiento de la cinemática del robot (con sus parámetros D-H, por ejemplo) obtener la n-upla de valores articulares que posicionan y orientan su extremo. El inconveniente de estor procedimientos es que se trata de métodos numéricos iterativos, cuya velocidad de convergencia e incluso su convergencia en sí no está siempre garantizada.

A la hora de resolver el problema de cinemática inversa es mucho más adecuado encontrar una solución cerrada. Esto es, encontrar una relación matemática explícita de la forma:

$$q_k = f_k(x,y,z,\alpha,\beta,\gamma) \qquad 4.23$$
$$k=1...n \text{ (Grados de Libertad)}$$

A éste tipo de solución se presentan los siguientes puntos que podemos considerara como ventajas:

1. En muchas aplicaciones, el problema cinemático inverso ha de resolverse en tiempo real (por ejemplo, en el seguimiento de una determinada trayectoria). Una solución de tipo iterativo no garantiza tener la solución en el momento adecuado.
2. Al contrario de lo que ocurría en el problema cinemático directo, con cierta frecuencia la solución del problema cinemático inverso no es única; existiendo diferentes n-uplas $[q_1, q_2, ..., q_n]^T$ que posicionan y orientan el extremo del robot del mismo modo. En estos casos una solución cerrada permite incluir determinadas reglas o restricciones que aseguren que la solución obtenida sea la más adecuada de entre las posibles (por ejemplo, límites en los recorridos articulares).

No obstante, a pesar de las dificultades comentadas, la mayor parte de los robots poseen cinemáticas relativamente simples que facilitan en cierta medida la resolución

de su problema cinemático inverso. Por ejemplo, si se consideran sólo los tres primeros grados de libertad de muchos robots, éstos tienen una estructura planar, esto es, los tres primeros elementos quedan contenidos en un plano. Esta circunstancia facilita la resolución del problema. Asimismo, en muchos robots se da la circunstancia de que los tres grados de libertad últimos, dedicados fundamentalmente a orientar el extremo del robot, corresponden a giros sobre ejes que se cortan en un punto. De nuevo esta situación facilita el cálculo de la n-upla $[q_1, q_2, ..., q_n]^T$ correspondiente a la posición y orientación deseadas. Por lo tanto es posible establecer ciertas pautas generales que permitan plantear y resolver el problema de cinemática inversa de una manera sistemática.

Los métodos geométricos permiten obtener normalmente los valores de las primeras variables articulares, que son las que consiguen posicionar el robot (prescindiendo de la orientación de su extremo). Para ello utilizan relaciones trigonométricas y geométricas sobre los elementos del robot. Se suele recurrir a la resolución de triángulos formados por los elementos y articulaciones del robot.

Solución por Métodos Geométricos:

Éste procedimiento es adecuado para robots de pocos grados de libertad o como complemento para el método de desacoplamiento cinemático, ya que puede ser empleado cuando se consideren sólo los primeros grados de libertad, dedicados a posicionar el extremo.

El procedimiento en sí, se basa en encontrar suficiente número de relaciones geométricas en las que intervendrán las coordenadas del extremo del robot, sus coordenadas articulares y las dimensiones físicas de sus elementos.

Para mostrar el procedimiento a seguir, se va aplicar el método a la resolución del problema de Rhino XR-3, en sus tres primeros Grados de Libertad (estructura típica articular). La figura 4.3 muestra la configuración del robot. El dato de partida son las coordenadas (p_x, p_y, p_z) referidas a $\{S_0\}$ (origen) en las que se quiere posicionar su extremo.

Como se ve, este robot posee una estructura planar, quedando este plano definido por el ángulo de la primera variable articular q_1.

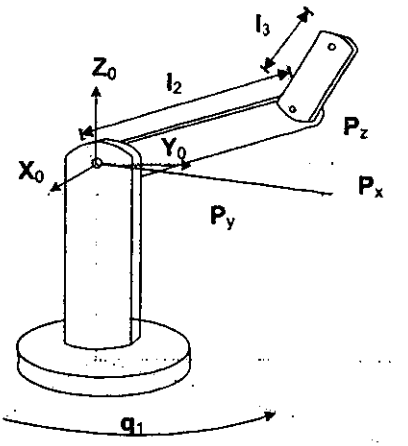


Figura 4.3 Parte articular (3 primeros grados de libertad) del robot Rhino XR-3

El valor de q_1 se obtiene inmediatamente como:

$$q_1 = arctang\left(\frac{p_y}{p_x}\right) \qquad 4.24$$

Considerando ahora únicamente los elemento 2 y 3 que están situados en un plano (figura 4.4-a), utilizando el teorema del coseno, se tendrá:

$$r^2 = p_x^2 + p_y^2$$

$$r^2 + p_z^2 = l_2^2 + l_3^2 + 2l_2 l_3 \cos q_3$$

$$\Rightarrow$$

$$\cos q_3 = \frac{p_x^2 + p_y^2 + p_z^2 - l_2^2 - l_3^2}{2l_2 l_3} \qquad 4.25$$

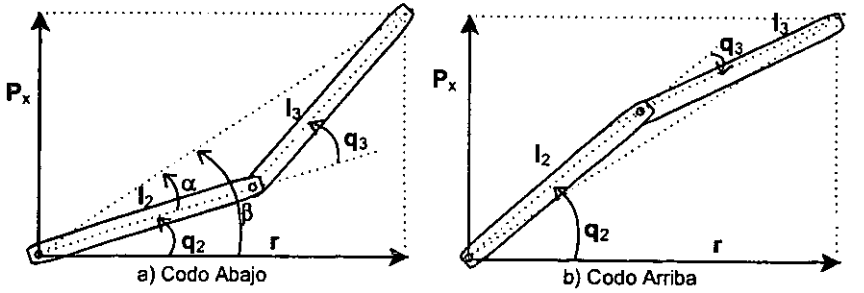


Figura 4.4. Elementos 2 y 3 del robot de la figura 4.3 contenidos en un plano y en a) configuración codo abajo y b) configuración codo arriba

Esta expresión permite obtener q_3 en función del vector de posición del extremo p. No obstante, y por motivos de ventajas computacionales, es más conveniente utilizar la expresión de la arcotangente en lugar del arcoseno.

Puesto que

$$\operatorname{sen} q_3 = \pm\sqrt{1-\cos^2 q_3} \qquad 4.26$$

Se tendrá que

$$q_3 = arctang\left(\frac{\pm\sqrt{1-\cos^2 q_3}}{\cos q_3}\right) \qquad 4.27$$

$$con \qquad \cos q_3 = \frac{p_x^2 + p_y^2 + p_z^2 - l_2^2 - l_3^2}{2l_2 l_3}$$

Como se ve, existen 2 posibles soluciones para q_3 según se tome el signo del radical. Éstas corresponden a las configuraciones de codo arriba (figura 4.4-a) y codo abajo (figura 4.4-b) del robot.

El calculo de q_2 se hace a partir de la diferencia entre β y α:

$$q_2 = \beta - \alpha \qquad 4.28$$

Siendo:

$$\beta = arctang\left(\frac{p_z}{r}\right) = arctang\left(\frac{p_z}{\pm\sqrt{p_x^2 + p_y^2}}\right)$$

$$\alpha = arctang\left(\frac{l_3 \operatorname{sen} q_1}{l_2 + l_3 \cos q_1}\right) \qquad 4.29$$

Luego, finalmente:

$$q_2 = arctang\left(\frac{p_z}{\pm\sqrt{p_x^2 + p_y^2}}\right) - arctang\left(\frac{l_3 \operatorname{sen} q_1}{l_2 + l_3 \cos q_1}\right) \qquad 4.30$$

De nuevo los dos posibles valores según la elección del signo dan lugar a dos valores diferentes de q_2 correspondientes a las configuraciones codo arriba y abajo. Por lo tanto las expresiones [4.24], [4.27] y [4.30] resuelven el problema cinemático inverso para el robot de 3 grados de libertad considerado.

Solución por Matriz de Transformación Homogénea:

En principio es posible tratar de obtener el modelo cinemático inverso de un robot a partir del conocimiento de su modelo directo.

Si embargo, en la práctica esta tarea no es trivial siendo en muchas ocasiones tan complejas que obliga a desecharla. Además, puesto que el problema cinemático directo, contiene en el caso de un robot de 6 grados de libertad 12 ecuaciones, y se buscan sólo 6 relaciones (una por cada grado de libertad), existirán necesariamente ciertas dependencias entre las 12 expresiones de partida (resultado de la condición de ortonormalidad de los vectores **n,o** y **a**) con lo cual la elección de qué ecuaciones escoger debe hacerse con sumo cuidado.

Se va aplicar este procedimiento al robot de 3 grados de libertad de configuración esférica (2 giros y un desplazamiento) mostrado en la figura 4.4. El robot queda siempre contenido en un plano determinado por el ángulo q_1.

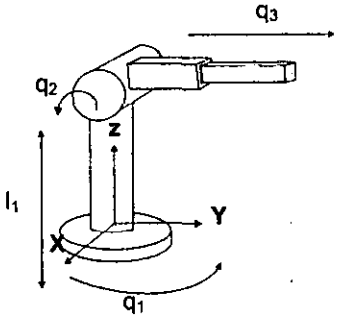


Figura 4.5 Robot polar de 3 Grados de Libertad

El primer paso a dar para resolver el problema cinemático inverso es obtener la matriz T que relaciona el sistema de referencia $\{S_0\}$ asociado a la base con el sistema de referencia $\{S_3\}$ asociado a su extremo. La figura 4.6 representa la asignación de sistemas de referencia según los criterios Denavit- Hartenberg, con el robot situado en su posición de partida (q_1=q_2=0), la tabla 4.1 muestra los valores de los parámetros de Denavit- Hartenberg.

Tabla 4.1. Parámetros D-H del robot de la figura 4.6

Articulación	θ	d	a	α
1	q_1	l_1	0	90
2	q_2	0	0	-90
3	0	q_3	0	0

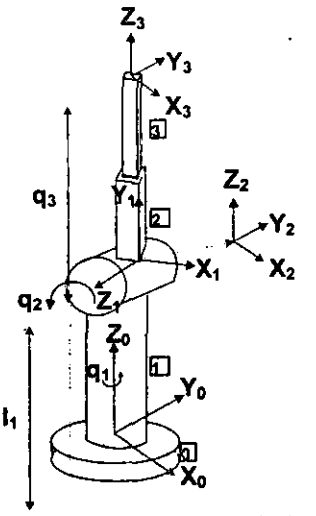


Figura 4.6. Asignación de sistemas de referencia del robot polar de la figura 4.5

A partir de estos es inmediato obtener las matrices A y T

$$ {}^{0}A_1 = \begin{bmatrix} \cos q_1 & 0 & sen q_1 & 0 \\ sen q_1 & 0 & -\cos q_1 & 0 \\ 0 & 1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{1}A_2 = \begin{bmatrix} \cos q_2 & 0 & -sen q_2 & 0 \\ sen q_2 & 0 & \cos q_2 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{1}A_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{0}A_1 = \begin{bmatrix} \cos q_1 \cos q_2 & -sen q_1 & \cos q_1 sen q_2 & 0 \\ sen q_1 \cos q_2 & \cos q_1 & -sen q_1 sen q_2 & 0 \\ sen q_2 & 0 & \cos q_2 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T = {}^{0}A_3 = \begin{bmatrix} \cos q_1 \cos q_2 & -sen q_1 & \cos q_1 sen q_2 & -q_3 \cos q_1 sen q_2 \\ sen q_1 \cos q_2 & \cos q_1 & -sen q_1 sen q_2 & -q_3 sen q_1 sen q_2 \\ sen q_2 & 0 & \cos q_2 & -q_3 \cos q_2 + l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad 4.31 $$

Obtenida la expresión de **T** en función de las coordenadas articulares (q_1,q_2,q_3), y supuesta una localización de destino para el extremo del robot definida por los vectores **n.o.a** y **p** se podría intentar manipular directamente las 12 ecuaciones resultantes de **T** a fin de despejar q_1,q_2 y q_3 en función de **n,o,a** y **p**.

Sin embargo, este procedimiento directo es complicado, apareciendo ecuaciones trascendentes. En lugar de ello, suele ser adecuado aplicar el siguiente procedimiento:

Puesto que $T={}^0A_1\ {}^1A_2\ {}^2A_3$ se tendrá que:

$$\left({}^0A_1\right)^{-1}T={}^1A_2\ {}^2A_3 \qquad 4.32$$

$$\left({}^1A_2\right)^{-1}\left({}^0A_1\right)^{-1}T={}^2A_3$$

Puesto que $T=\begin{bmatrix} n & o & a & p \\ 0 & 0 & 0 & 1 \end{bmatrix}$ es conocida, los miembros a la izquierda en las expresiones [4.32] son función de las variables articulares (q_1, ... , q_k) mientras que los miembros de la derecha lo son de las variables articulares (q_{k+1}, ... , q_n).

De este modo, de la primera de las expresiones de [4.32] se tendrá q_1 aislado del resto de las variables articulares y tal vez sera posible obtener su valor sin la complejidad que se tendría abordando directamente la manipulación de la expresión [4.31]. Asu vez, una vez obtenida q_1, la segunda expresión de [4.31] permitirá tener el valor de q_2 aislado respecto de q_3. Por ultimo, conocidor los valores de q_1 y q_2 se podrá tener q_3 de la expresión [4.31] sin excesiva dificultad.

Para poder aplicar este procedimiento, es necesario en primer lugar obtener las inversas de las matrices ${}^{i-1}A_i$. Esto es sencillo si se considera que la inversa de una matriz de transformación homogenea viene dada por [4.20]:

$$\begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}=\begin{bmatrix} n_x & n_y & n_z & -n^Tp \\ o_x & o_y & o_z & -n^Tp \\ a_x & a_y & a_z & -n^Tp \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad 4.33$$

Luego se tiene que:

$${}^0A_1=\begin{bmatrix} \cos q_1 & 0 & \operatorname{sen} q_1 & 0 \\ \operatorname{sen} q_1 & 0 & -\cos q_1 & 0 \\ 0 & 1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}=\begin{bmatrix} \cos q_1 & \operatorname{sen} q_1 & 0 & 0 \\ 0 & 0 & 1 & -l_1 \\ \operatorname{sen} q_1 & -\cos q_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad 4.34$$

$$ {}^{1}A_2 = \begin{bmatrix} \cos q_2 & 0 & -\text{sen}q_2 & 0 \\ \text{sen}q_2 & 0 & \cos q_2 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} \cos q_2 & \text{sen}q_2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ -\text{sen}q & \cos q_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{1}A_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Por lo tanto, utilizando la primera de las ecuaciones de [4.32] se tiene que:

$$ ({}^{0}A_1)^{-1}\,{}^{0}T_3 = {}^{1}A_2\,{}^{2}A_3 = \begin{bmatrix} \cos q_1 & \text{sen}q_1 & 0 & 0 \\ 0 & 0 & 1 & -l_1 \\ \text{sen}q_1 & -\cos q_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \qquad 4.35 $$

$$ = \begin{bmatrix} \cos q_2 & 0 & -\text{sen}q_2 & 0 \\ \text{sen}q_2 & 0 & \cos q_2 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos q_2 & 0 & -\text{sen}q_2 & -\text{sen}q_2 q_3 \\ \text{sen}q_2 & 0 & \cos q_2 & \cos q_2 q_3 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

De las 12 relaciones establecidas en la ecuación [4.35] interesan aquellas que expresan q_1 en función de constantes (y no de q_2 y q_3). Así por ejemplo, tomando el elemento (3,4) se tiene:

$$ (\text{Sen}q_1)(p_x)-(\text{Cos}q_1)(p_y)=0 \qquad \Rightarrow $$

$$ \tan(q_1) = \left(\frac{P_y}{P_x}\right) \quad \Rightarrow $$

$$ q_1 = \arctan\left(\frac{P_y}{P_x}\right) \qquad 4.36 $$

Utilizando ahora la segunda ecuación de [4.32] se tendrá:

$$({}^{1}A_2)^{-1}({}^{0}A_1)^{-1}\,{}^{0}T = {}^{2}A_3 = \begin{bmatrix} \cos q_2 & \operatorname{sen} q_2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ -\operatorname{sen} q_2 & \cos q_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos q_1 & \operatorname{sen} q_1 & 0 & 0 \\ 0 & 0 & 1 & -l_1 \\ \operatorname{sen} q_1 & -\cos q_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos q_2 \cos q_1 & \cos q_2 \operatorname{sen} q_1 & \operatorname{sen} q_2 & -l_1 \operatorname{sen} q_2 \\ -\operatorname{sen} q_1 & \cos q_1 & 0 & 0 \\ -\operatorname{sen} q_2 \cos q_1 & -\operatorname{sen} q_2 \operatorname{sen} q_1 & \cos q_2 & -\cos q_2 l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos q_1 & \operatorname{sen} q_1 & 0 & 0 \\ 0 & 0 & 1 & -l_1 \\ \operatorname{sen} q_1 & -\cos q_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad 4.37$$

Tomando el elemento (1,4) se tiene:

$$\cos q_2 \cos q_1 p_x + \cos q_2 \operatorname{sen} q_1 p_y + \operatorname{sen} q_2 p_z - l_1 \operatorname{sen} q_2 = 0 \quad \Rightarrow$$

$$\cos q_2 (\cos q_1 p_x + \operatorname{sen} q_1 p_y) + \operatorname{sen} q_2 (p_z - l_1) = 0 \quad \Rightarrow$$

$$\tan q_2 = -\frac{\cos q_1 p_x + \operatorname{sen} q_1 p_y}{(p_z - l_1)} \qquad 4.38$$

y considerando que por [4.36]:

$$(\operatorname{Sen} q_1)(p_x) - (\operatorname{Cos} q_1)(p_y) = 0 \quad \Rightarrow$$

$$(\operatorname{sen} q_1 p_x - \cos q_1 p_y)^2 = \operatorname{sen}^2 q_1 p_x^2 - \cos^2 q_1 p_y^2 - 2 \operatorname{sen} q_1 \cos q_1 p_x p_y = 0 \quad \Rightarrow$$

$$(1 - \cos^2 q_1) p_x^2 + (1 - \operatorname{sen}^2 q_1) p_y^2 = 2 \operatorname{sen} q_1 \cos q_1 p_x p_y \quad \Rightarrow$$

$$\cos^2 q_1 p_x^2 + \operatorname{sen}^2 q_1 p_y^2 + 2 \operatorname{sen} q_1 \cos q_1 p_x p_y = p_x^2 + p_y^2 \quad \Rightarrow$$

$$\cos q_1 p_x + \operatorname{sen} q_1 p_y = \sqrt{p_x^2 + p_y^2} \qquad 4.39$$

Se tiene finalmente:

$$q_2 = \arctan \frac{\sqrt{p_x^2 + p_y^2}}{l_1 - p_z} \qquad 4.40$$

Por ultimo, tomando de [4.37] el elemento (3,4) se tiene:

$$
\begin{aligned}
&-\operatorname{sen} q_2 \cos q_1 p_x - \operatorname{sen} q_2 \operatorname{sen} q_1 p_y + \cos q_2 p_z - \cos q_2 l_1 = q_3 \quad \Rightarrow \\
&\cos q_2 (p_z - l_1) - \operatorname{sen} q_2 (\cos q_1 p_x + \operatorname{sen} q_1 p_y) = q_3 \quad \Rightarrow \\
&q_3 = \cos q_2 (p_z - l_1) - \operatorname{sen} q_2 \sqrt{p_x^2 + p_y^2}
\end{aligned}
\qquad 4.41
$$

Las expresiones [4.36], [4.40] y [4.41] corresponden a la solución del problema cinemático inverso del robot considerado. A continuación se reproducen estas expresiones:

$$
\begin{aligned}
q_1 &= arctan\left(\frac{P_y}{P_x}\right) \\
q_2 &= arctan\frac{\sqrt{p_x^2 + p_y^2}}{l_1 - p_z} \\
q_3 &= \cos q_2 (p_z - l_1) - \operatorname{sen} q_2 \sqrt{p_x^2 + p_y^2}
\end{aligned}
\qquad 4.42
$$

A los mismos resultados se podría llegar mediante consideraciones geométricas.

Solución por Desacoplo Cinemático:

Los procedimientos vistos permiten obtener los valores de las 3 primeras variables articulares del robot, aquellas que posicionan su extremo en unas coordenadas (px, py, pz) determinadas, aunque pueden ser igualmente utilizadas para la obtención de un mayor número a costa de una mayor complejidad.

Ahora bien, como es sabido, en general no basta con posicionar el extremo del robot en un punto del espacio, sino que casi siempre es preciso también conseguir que la herramienta que aquel porta se oriente de una manera determinada. Para ello, los robots cuentan normalmente con otros tres grados de libertad adicionales, situados al final de la cadena cinemática y cuyos ejes, generalmente, se cortan en un punto, que de manera informal se denomina "muñeca del robot". Si bien la variación de estos tres últimos grados de libertad origina un cambio en la posición final del extremo real del robot, su verdadero objetivo es poder orientar la herramienta del robot libremente en el espacio.

El método de desacoplo cinemático saca partido de este hecho, separando ambos problemas: posición y orientación. Para ello, dada una posición y orientación final deseadas, establece las coordenadas del punto de corte de los 3 últimos ejes calculándose los valores de las tres primeras variables articulares (q_1,q_2,q_3) que consiguen posicionar este punto. A continuación, a partir de los datos de orientación y de los ya calculados (q_1,q_2,q_3) obtiene los valores del resto de las variables articulares.

En la figura 4.7 se representa un robot, que reúne las citadas características, con indicación de los sistemas de coordenadas asociados según el procedimiento de Denavit-Hartemberg, cuyos parámetros se pueden observar en la tabla 4.2

Tabla 4.2. Parámetros D-H del robot de la figura 4.7.

Articulación	θ	d	a	α
1	θ_1	l_1	0	-90
2	θ_2	0	l_2	0
3	θ_3	0	0	90
4	θ_4	l_3	0	-90
5	θ_5	0	0	90
6	θ_6	l_4	0	0

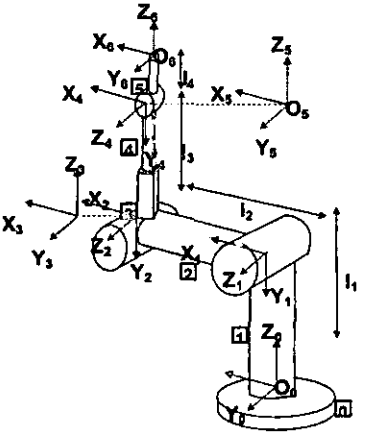


Figura 4.7. Cinemática del robot IRB2400, cuya inversa se puede desacoplar

El punto central de la muñeca del robot corresponde al origen del sistema $\{S_5\}$:O_5. Por su parte, el punto final del robot será el origen del sistema $\{S_6\}$:O_6. En lo siguiente se utilizarán los vectores:

$$p_m = \overline{O_0 O_5} \qquad 4.43$$
$$p_r = \overline{O_0 O_6}$$

que van desde el origen del sistema asociado a la base del robot $\{S_0\}$ hasta los puntos centros de la muñeca y fin del robot, respectivamente.

Puesto que la dirección del eje $\mathbf{z}_6$ debe coincidir con la de $\mathbf{z}_5$ y la distancia entre O_5 y O_6 medida a lo largo de $\mathbf{z}_5$ es precisamente $d_4=l_4$, se tendrá que:

$$p_m = p_r - l_4 z_6 \qquad 4.44$$

estando todos los vectores referidos a las coordenadas del sistema $\{S_0\}$.

En la expresión [4.43] p_r son las coordenadas del punto donde se pretende que se posicione el robot expresadas en $\{S_0\}$. Por lo tanto

$$p_r = [p_x, p_y, p_z]^T \qquad 4.45$$

El vector director z_6 es el vector a correspondiente a la orientación deseada $z_6=[a_x,a_y,a_z]^T$ y l_4 es un parámetro asociado con el robot. Por lo tanto, las coordenadas del punto central de la muñeca (p_{mx},p_{my},p_{mz}) son fácilmente obtenibles.

Tal y como se mostró es posible, mediante método geométrico, por ejemplo, calcular los valores de (q_1,q_2,q_3) que consiguen posicionar el robot en el p_m deseado.

Queda ahora obtener los valores de q_4, q_5 y q_6 que consiguen la orientación deseada. Para ello, denominamos ${}^0R_6$ a la submatriz de rotación de ${}^0T_6$ se tendrá:

$${}^0R_6=[n\ o\ a] = {}^0R_3\,{}^3R_6 \qquad 4.46$$

donde ${}^0R_6$ es conocida por ser la orientación deseada del extremo del robot, y ${}^0R_3$ definida por:

$${}^0R_3= {}^0A_1\,{}^1A_2\,{}^2A_3 \qquad 4.47$$

También lo será a partir de los valores ya obtenidos de q_1,q_2 y q_3. Por lo tanto:

$${}^3R_6 = [r_{ij}] = ({}^0R_3)^{-1}\,{}^0R_6 = ({}^0R)^T [n \quad o \quad a] \qquad 4.48$$

tendrá sus componentes conocidas.

Por otra parte, ${}^3R_6$ corresponde con la submatriz (3X3) de rotación de la matriz de transformación homogénea ${}^3T_6$ que relaciona el sistema $\{S_3\}$ con el $\{S_0\}$. Por lo tanto:

$$4.49$$

$${}^3R_6= {}^3R_4\,{}^4R_5\,{}^5R_6$$

donde ${}^{i-1}R_i$ es la submatriz de rotación de la matriz de Denavit- Hartenberg ${}^{i-1}A_i$, cuyos valores son:

$${}^3R_4 = \begin{bmatrix} \cos\theta_4 & 0 & -\text{sen}\,\theta_4 \\ \text{sen}\,\theta_4 & 0 & \cos\theta_4 \\ 0 & -1 & 0 \end{bmatrix} \quad {}^4R_5 = \begin{bmatrix} \cos\theta_5 & 0 & \text{sen}\,\theta_5 \\ \text{sen}\,\theta_5 & 0 & -\cos\theta_5 \\ 0 & 1 & 0 \end{bmatrix} \quad {}^5R_6 = \begin{bmatrix} \cos\theta_6 & -\text{sen}\,\theta_6 & 0 \\ \text{sen}\,\theta_6 & \cos\theta_6 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad 4.50$$

Luego se tiene que:

$$ {}^{3}R_{6} = \begin{bmatrix} \cos\theta_4\cos\theta_5\cos\theta_6 - \text{sen}\theta_4\,\text{sen}\theta_6 & -\cos\theta_4\cos\theta_5\,\text{sen}\theta_6 - \text{sen}\theta_4\cos\theta_6 & \cos\theta_4\,\text{sen}\theta_5 \\ \text{sen}\theta_4\cos\theta_5\cos\theta_6 + \cos\theta_4\,\text{sen}\theta_6 & -\text{sen}\theta_4\cos\theta_5\,\text{sen}\theta_6 + \cos\theta_4\cos\theta_6 & -\text{sen}\theta_4\cos\theta_5 \\ -\text{sen}\theta_5\cos\theta_6 & \text{sen}\theta_5\,\text{sen}\theta_6 & \cos\theta_5 \end{bmatrix} \quad 4.51 $$

donde r_{ij} serán por [4.48] valores numericos conocidos:

$$ [r_{ij}] = \begin{bmatrix} \cos\theta_4\cos\theta_5\cos\theta_6 - \text{sen}\theta_4\,\text{sen}\theta_6 & -\cos\theta_4\cos\theta_5\,\text{sen}\theta_6 - \text{sen}\theta_4\cos\theta_6 & \cos\theta_4\,\text{sen}\theta_5 \\ \text{sen}\theta_4\cos\theta_5\cos\theta_6 + \cos\theta_4\,\text{sen}\theta_6 & -\text{sen}\theta_4\cos\theta_5\,\text{sen}\theta_6 + \cos\theta_4\cos\theta_6 & -\text{sen}\theta_4\cos\theta_5 \\ -\text{sen}\theta_5\cos\theta_6 & \text{sen}\theta_5\,\text{sen}\theta_6 & \cos\theta_5 \end{bmatrix} \quad 4.52 $$

De las nueve relaciones expresadas en [4.52] se pueden tomar las correspondientes a r_{13}, r_{23}, r_{33}, r_{31}, r_{32}:

$$ r_{13} = \cos\theta_4\,\text{sen}\theta_5 \qquad r_{23} = -\text{sen}\theta_4\cos\theta_5 \qquad r_{33} = \cos\theta_5 $$
$$ r_{31} = -\text{sen}\theta_5\cos\theta_6 \qquad r_{32} = \text{sen}\theta_5\,\text{sen}\theta_6 \qquad 4.53 $$

Del conjunto de ecuaciones [4.53] es inmediato obtener los valores de los parámetros articulares (se recomienda convertir todas las funciones trigonométricas inversas en su arcotangente, por ser ésta computacionalmente más robusta):

$$ q_4 = \text{arcsen}\left(\frac{r_{23}}{r_{33}}\right) $$
$$ q_5 = ar\cos(r_{33}) \qquad 4.54 $$
$$ q_6 = arctan\left(-\frac{r_{32}}{r_{31}}\right) $$

Esta expresión, junto con las [4.24], [4.27] y [4.30], y teniendo en cuenta que las posiciones de cero son distintas, constituyen la solución completa del problema cinemático inverso del robot articulado de la figura 4.7.

4.3 Cinemática inversa (solución del Rhino XR-3)

Para encontrar una solución, que fuera acorde con lo ya expuesto, se opto por el desacoplamiento Cinemático, de esta forma tenemos que los primeros tres grados de libertad fueron resueltos a traves de una solución geometrica, por lo que sólo faltaria resolver para los últimos dos grados, esto se realizará utilizando matrices de Transformación Homogenea.

Por lo tanto, planteamos el modelo físico en la figura 4.8

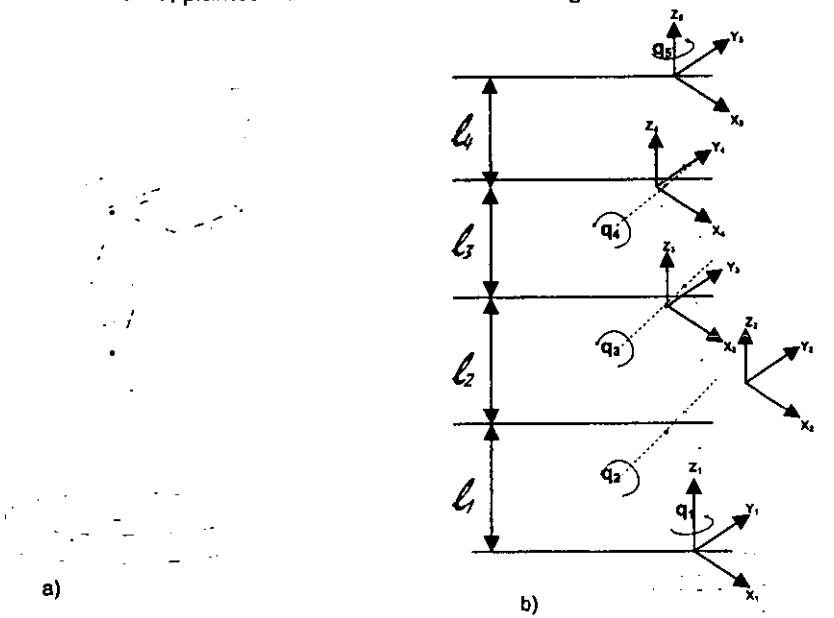


Figura 4.8. Cinemática del robot Rhino XR-3,
a) cuya inversa será desacoplada ; b) Con los parámetros D-H

De la figura 4.8 b). Se observa que existe un orden en los giros a realizarse, comenzando en la base (giro en Z_1), en los puntos 2,3 y 4 sobre el eje Y, en el punto 5 de nuevo sobre el eje X

Para resolver q_4 y q_5, definimos los parámetros de Denavit – Hartenberg, así tenemos:

Tabla 4.3. Parámetros D-H del robot de la figura 4.8. en sus ultimas dos articulaciones.

Articulación	θ	d	a	α
4	q_4	0	0	-90
5	q_5	l_5	0	90

Por lo que al formar las matrices homogéneas tenemos:

$$ {}^{3}A_{4} = \begin{bmatrix} \cos q_4 & 0 & senq_4 & 0 \\ 0 & 1 & 0 & 0 \\ -senq_4 & 0 & \cos q_4 & l_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{y} \quad {}^{4}A_{5} = \begin{bmatrix} \cos q_5 & -senq_5 & 0 & 0 \\ senq_5 & \cos q_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad 4.55 $$

Sabemos que para encontrar T es necesario realizar el producto entre estas matrices, es decir:

$$ {}^{3}T_{5} = {}^{3}A_{4}\,{}^{4}A_{5} = \begin{bmatrix} \cos q_4 \cos q_5 & -\cos q_4 senq_5 & senq_4 & 0 \\ senq_5 & \cos q_5 & 0 & 0 \\ -senq_4 \cos q_5 & senq_4 senq_5 & \cos q_4 & l_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad 4.56 $$

Este procedimiento directo trae consigo la aparición de ecuaciones trascendentales, por lo que se nos recomienda usar el procedimiento descrito en la sección referente a la solución por matriz de transformación homogénea:

$$ \left({}^{3}A_{4}\right)^{-1}{}^{3}T_{5} = {}^{4}A_{5} = \begin{bmatrix} \cos q_4 & 0 & -senq_4 & senq_4 l_4 \\ 0 & 1 & 0 & 0 \\ senq_4 & 0 & \cos q_4 & -\cos q_4 l_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos q_5 & -senq_5 & 0 & 0 \\ senq_5 & \cos q_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad 4.57 $$

De éstas relaciones nos interesan aquellas que expresan q_4 en función de constantes, por lo que tomamos el elemento (3,4) se tiene:

$$ p_x\left(senq_4\right) + p_z\left(\cos q_4\right) - l_4\left(\cos q\right) = 0 $$

$$ \Rightarrow $$

$$ senq_4\left(p_x\right) + \cos q_4\left(p_z - l_4\right) = 0 $$

$$ \therefore $$

$$ \tan q_4 = \frac{l_4 - p_z}{p_x} $$

$$ \Rightarrow $$

$$ q_4 = Arc\ \tan\left(\frac{l_4 - p_z}{p_x}\right) \qquad 4.58 $$

y para obtener a q_5 lo hacemos directamente de la primera matriz:

$$p_x(sen q_3) + p_y(\cos q_3) = 0$$

$$\Rightarrow$$

$$\tan q_3 = \frac{p_y}{p_x}$$

$$\therefore$$

$$q_3 = Arc\tan\left(\frac{p_y}{p_x}\right) \qquad 4.59$$

Por lo tanto los valores de los grados de libertad están determinados por:

$$q_1 = arctang\left(\frac{p_y}{p_x}\right) \qquad 4.24$$

$$q_3 = arctang\left(\frac{\pm\sqrt{1-\cos^2 q_3}}{\cos q_3}\right) \qquad 4.27$$

$$con \qquad \cos q_3 = \frac{p_x^2 + p_y^2 + p_z^2 - l_2^2 - l_3^2}{2l_2 l_3}$$

$$q_2 = arctang\left(\frac{p_z}{\pm\sqrt{p_x^2 + p_y^2}}\right) - arctang\left(\frac{l_3 \operatorname{sen} q_1}{l_2 + l_3 \cos q_1}\right) \qquad 4.30$$

$$q_4 = Arc\tan\left(\frac{l_4 - p_z}{p_x}\right) \qquad 4.58$$

$$q_5 = Arc\tan\left(\frac{p_y}{p_x}\right) \qquad 4.59$$

Con:

l_1= 26 Cms. l_2= 23 Cms. l_3= 23 Cms.

y

l_4= 9 Cms.

5. DISEÑO DE SOFTWARE

Para la realización del software es necesario considerar varios factores, pero el mas importante es la finalidad buscada. El software debe ser capaz de controlar el robot en su modalidad de brazo mecánico (Rhino XR-3) en la gran mayoría de sus movimientos posibles, digo en la gran mayoría debido que he comprobado que para tener un control absoluto del robot lo recomendable es diseñar desde el inicio **todo** el robot, es decir, es necesario que el robot sea diseñado desde su parte mecánica, electrónica y su unidad controladora. De esta forma podremos garantizar que el software realizado para ese tipo de robot cumplirá con todos los movimientos. De no ser así (como es mi caso) se necesita leer todas las señales y de ser necesario analizar todos los casos para controlar desde el manipulador terminal (gripper) hasta los motores de forma independiente. O de ser necesario, sí es que hay que agregarle otro componente, también determinar que señales son más convenientes, para que el controlador Mark III lo pueda manipular.

El diseño es muy importante, debido a que éste nos dirá como queremos que quede el software, para ello hacemos un pequeño esquema de lo que nos gustaría que tuviera el software. Lo anterior lo observamos en las dos primeras pantallas que están hechas a mano y manifiestan un primer acercamiento de lo que nos gustaría que tuviera el software.

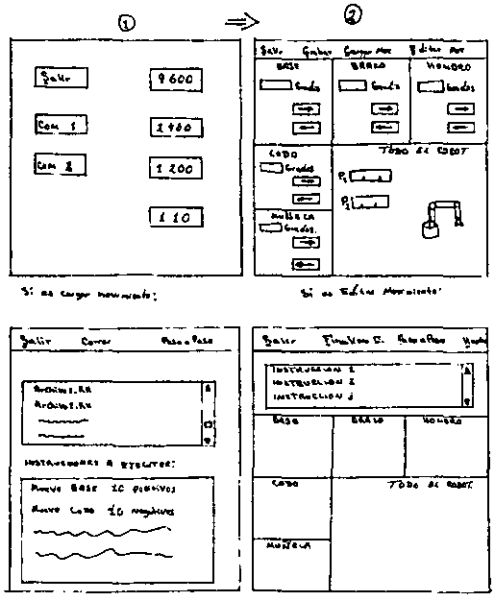


Figura 5.1. Primer acercamiento para determinar los requerimientos del software.

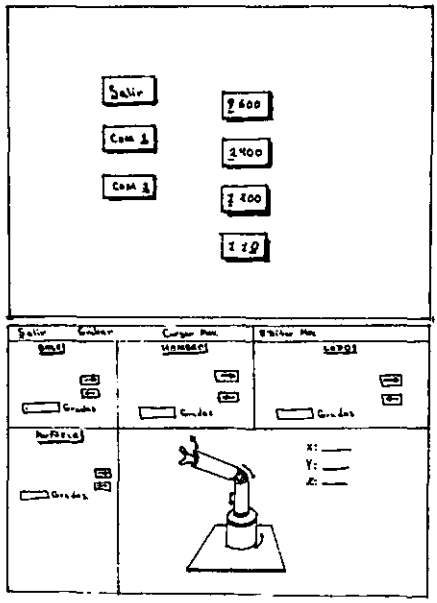


Figura 5.2 Segundo acercamiento en el cual se ve la funcionalidad de dejar todo en una sola plantilla.

Como podemos ver con las pantallas 5.1 y 5.2 observamos dos diferentes formas de manifestar lo deseado dentro del software esto es, en la pantalla 5.1 observamos que tenemos 4 pantallas diferentes en las cuales uno manifiesta mover el robot pedir la velocidad que debe llevar el alguno de los puertos seriales (COM 1 o COM 2) según sea el caso, los movimientos que se deben realizar cargar los movimientos y obviamente quizás ver un movimiento simulado de robot dentro de la pantalla, mientras que la pantalla 5.2 traducimos todas esas pantallas a una sola en esa pantalla podemos observar que dividimos esa pantalla en cinco sub-ventanas, cada una de ellas manifiesta un elemento del robot como es base hombro, muñeca y codo.

En cada uno de ellas se manifiesta el movimiento de alguna de sus partes. También es importante resaltar que en ambos casos las pantallas muestran una división de las actividades (cada grado de libertad se ve de forma independiente)

5.1 Requerimientos mínimos del software:

El software debe contar mínimamente con las siguientes características:

- Poder elegir el puerto de comunicaciones serial ya sea COM1 o COM2.
- La velocidad de comunicación debe de ser fija a 9,600 baudios, dado que debemos ajustarnos a los requerimientos del Mark III.
- Debe soportar el protocolo de comunicación de la interface RS232C, la cual ya se presento en el capitulo 2.

El software como requerimiento mínimo debe de ser capaz de controlar cada elemento del robot, es decir, la base, el hombro, el codo, la muñeca, el brazo, deben moverse de forma independiente. Ya sea dando los pasos de un solo golpe, por ejemplo si queremos que se mueva el robot 50 pasos los dará como si fuera un solo movimiento fluido. O bien llevar paso por paso con flechas como indicador ya sea izquierda o derecha, es decir debe ser capaz de manejar sentido como la cantidad de pasos que se requieran para un grado o 'X' cantidad de grados. Ahora bien hay algo interesante aquí aunque no sea especificado realmente si vamos a necesitar la participación de la dinámica inversa o dinámica directa para mover el robot de un punto a otro en este caso quiero partir de que quizá si vamos a tener la posibilidad de dar uno solo aunque no sea en esta tesis la posibilidad de que pueda implementar para eso dejo un marco en donde podemos especificar el algoritmo de alcanzar un punto a partir de un origen.

Como otro punto dentro de los requerimientos del software, es necesario también implementar la interface gráfica entre el usuario y la máquina de manera y amena. De ésta forma garantizamos que cualquier usuario es capaz de controlar el robot esto es, mientras una persona es neófita en el asunto pueda determinar como desea controlar el robot de la misma forma que una persona que ya es experta y que sabe como controlar ese robot. Por lo que es necesario un manual, el cual se pretende que sea de una forma interactiva, esto nos lleva a adoptar los estándares de Windows 95 específicamente, he desarrollado el software con los requerimientos de Windows 95, es

decir, un manual interactivo, un manual que permita damos una introducción rápida y sencilla del uso del software para el control del robot.

5.2 Selección del lenguaje a utilizar.

El lenguaje a utilizar va ha estar en función de la sencillez para realizar una aplicación, del ambiente en el que va a ejecutarse la aplicación y hasta cierto punto la compatibilidad de una máquina a otra, aunque esto último se asume como un hecho (hablando de PC's).

A lo anterior tenemos que agregar a especificaciones, como es la disponibilidad de los códigos fuentes, y el usar un lenguaje que pueda ser entendido por otra persona que deseara modificarlo es por ello que comencé a realizar un programa inicial en lenguaje "C", pero las circunstancias me han orillado a decidirme por un lenguaje netamente para Windows, el Visual Basic 5 en su versión empresarial en español. Esto último nos proporcionará una aplicación con presentación muy parecida a las aplicaciones comerciales para Windows (como por ejemplo: Word, Excel, Corel Draw, etc.) y con la ventaja de disponer los códigos fuentes en un lenguaje que poco a poco a tenido más adeptos (aunque de manera personal creo que existen otros lenguajes con mayor capacidad y que son netamente para aplicaciones elaboradas como son el lenguaje "C" de las diversas compañías o bien smalltalk), incluso se tiene un programa que controla el robot a desde el sistema operativo DOS (fragmentos de este programa se muestran en la figura 5.3) y que desde mi punto de vista esta funciona adecuadamente y fue realizado en lenguaje "C".

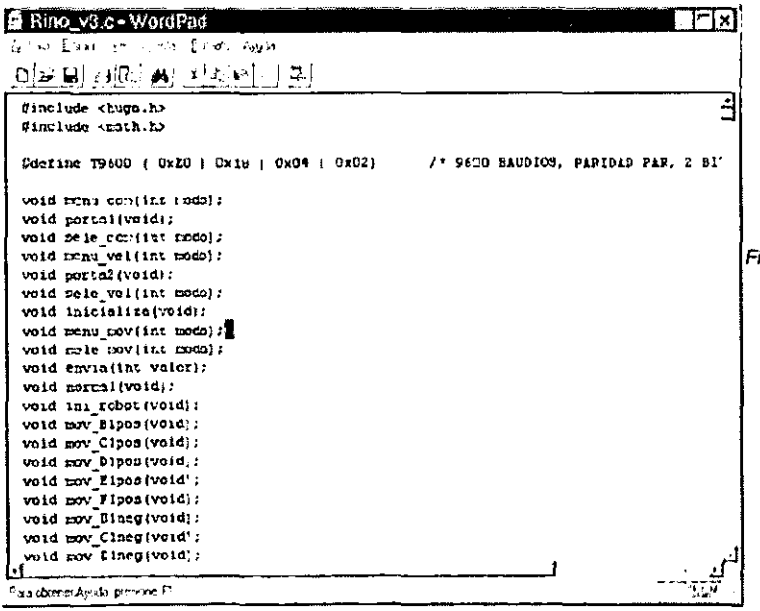


Figura 5.3 a. Definición de los procedimientos en lenguaje "C" del software controlador del Robot Rhino XR-3.

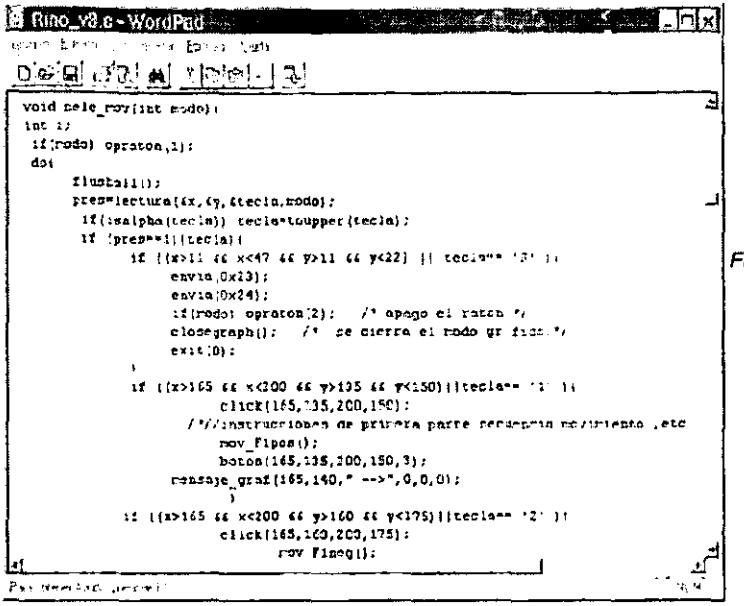


Figura 5.3 b. Proceso que muestra la selección de los elementos que componen el software controlador del Robot Rhino XR-3. Dicho Software se ejecuta en modo MS-DOS.

5.3 Selección de Plataforma y Hardware

En este caso la selección de plataforma y hardware no tiene gran relevancia dado que ha sido definida por mi director de tesis, la aplicación debe ser capaz de ejecutarse en una PC con sistema operativo Windows 95 o superior, pretendo que sea lo más sencillo y lo más rápido posible. De esta forma mostrare que desarrollando la aplicación en la plataforma antes mencionada será fácil traducirlo para otra plataforma. Debo destacar que estaré sujeto a las normas de Windows 95 por lo que al llevar esta aplicación a otra plataforma se deberá tomar en cuenta los estándares de la nueva plataforma.

En lo que respecta a hardware, cualquier máquina compatible con IBM PC, nos será buena, aunque en realidad no nos importa mucho cual sea el hardware que utilizamos dado que si podemos pasar de una plataforma a otra también se puede pasar de un hardware a otro es decir de una IBM a una máquina a compatible a Machintosh o a una Work Station o algo similar, obviamente es que no tenemos son los grandes recursos pues usaremos el equipo que tenemos al alcance, por lo que la máquina que tenemos es una máquina con procesador Pentium II, trabaja a 166 Mhz con 32 MB en RAM, Windows 95 y básicamente y el software como ya mencione se realiza como Visual Basic 5 edición empresarial en español, como podemos ver en este caso la selección de la plataforma y del software no estuvieron en función de mis gustos, si no que más bien estuvieron en función de cual es el equipo con el que contamos y en este caso ya se menciono. Entonces siempre tenemos que ajustamos a lo que ya tenemos o en su defecto lo que podríamos comprar para el caso particular no compramos nada, más que cable para hacer las interfaces.

6. DESARROLLO DEL SOFTWARE PARA EL CONTROL DE RHINO XR-3

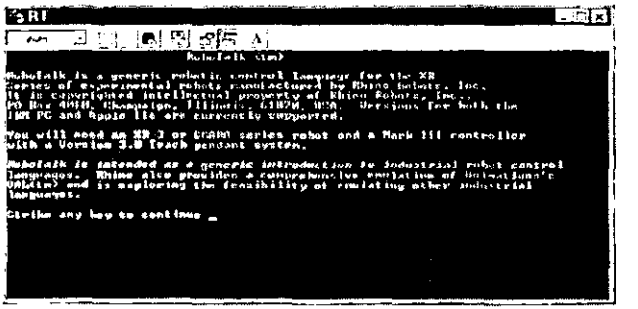
Quizás este capitulo por la naturaleza de la tesis sea el más largo y tedioso de explicar pues el objetivo de esta tesis es desarrollar un software que nos permita controlar al robot Rhino XR-3.

Por consiguiente en este capitulo trataremos de explicar muchos de los altibajos que he tenido al desarrollar el software mencionado, así que procurare llevar un orden:

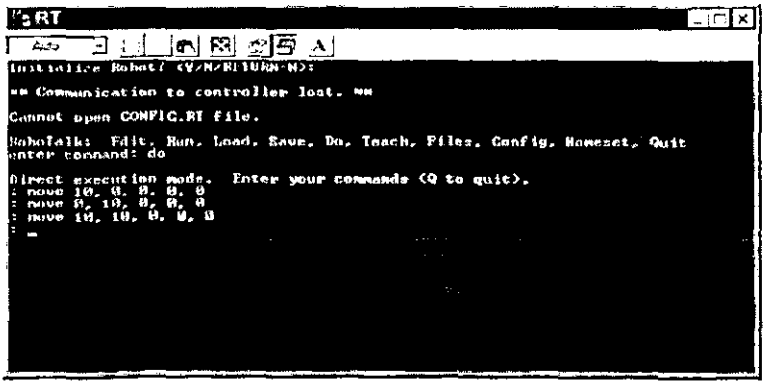
- **Desarrollo de las interfaces que permiten analizar la comunicación entre la computadora y el robot:**

Como ya se mencionó, se contaba con información limitada sobre el robot, de la cual nos fue útil el manual y el software proporcionado por el fabricante. Aunque es necesario recordar que en lo que respecta al manual de uso le hacen falta por lo menos el capitulo marcado como el "control del robot a través del software", además de que no existe una sección donde se trate las partes físicas que componen al robot (sólo hay esquemas), en lo que respecta al software lo considero muy completo, aunque su presentación nos es muy atractiva ya que sólo se puede ejecutar en el sistema operativo DOS, pero es capaz de controlar a la perfección el robot RHINO XR-3 (ver figuras 6.1 a y b).

Figura 6.1 a y b.
Pantallas de interface entre el usuario y el programa controlador de robot llamado ROBOTALK



a.



b.

Para poder desarrollar el software, se tuvo que construir una interface que me permitiera observar cual era el protocolo de las señales que ordenaban el movimiento del robot (figura 6.2), lo que me obligo a desarrollar un programa que me mostrará dichos códigos (figura 6.3) y fuera capaz de guardarlos en un archivo.

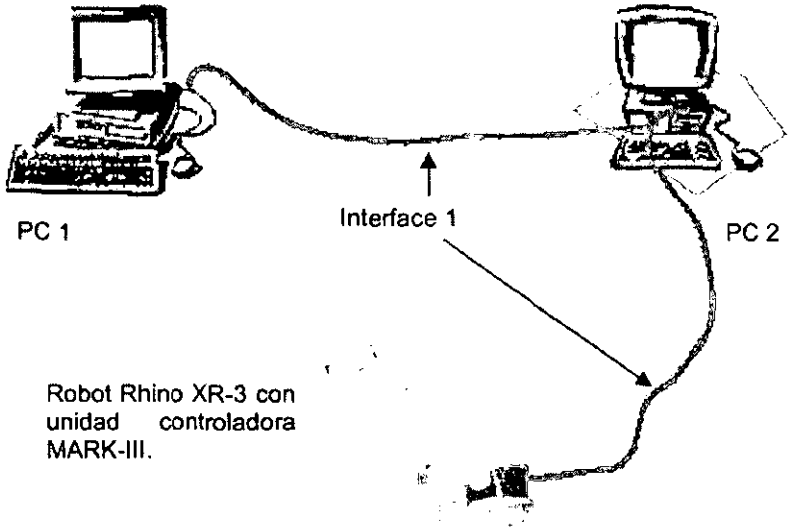
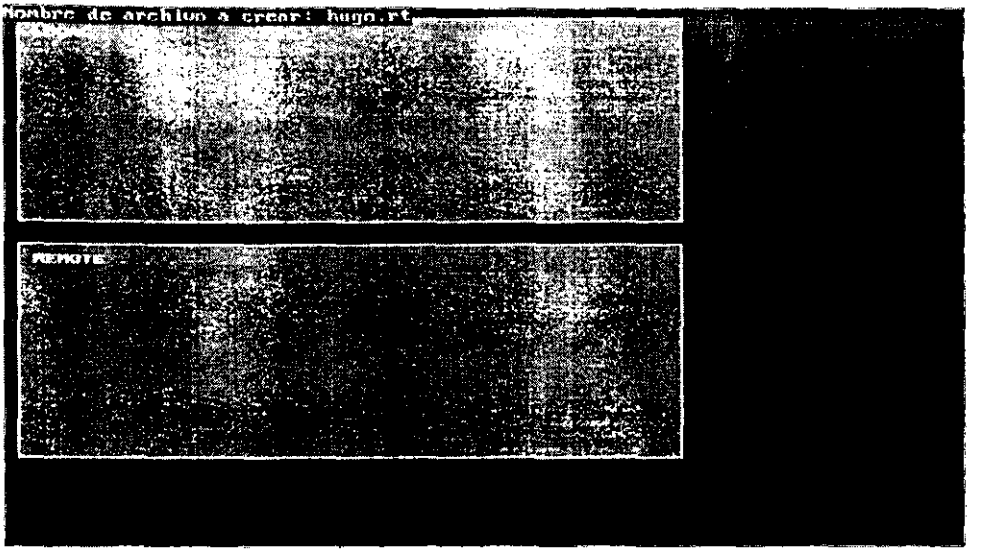
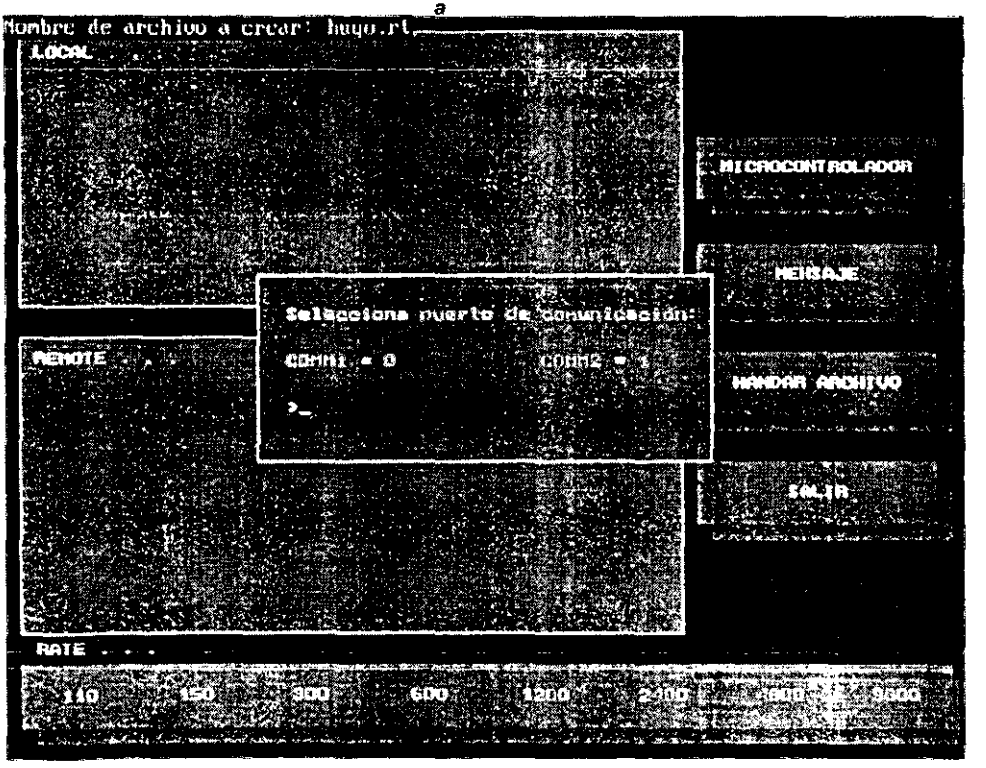


Figura 6.2
Muestra la configuración que se implemento para permitir la lectura de las señales controladoras del robot.



a

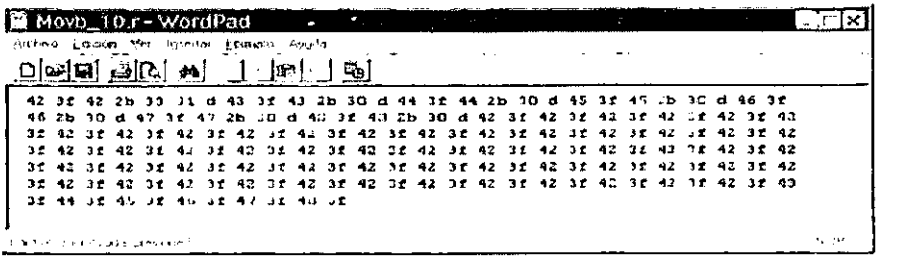


b

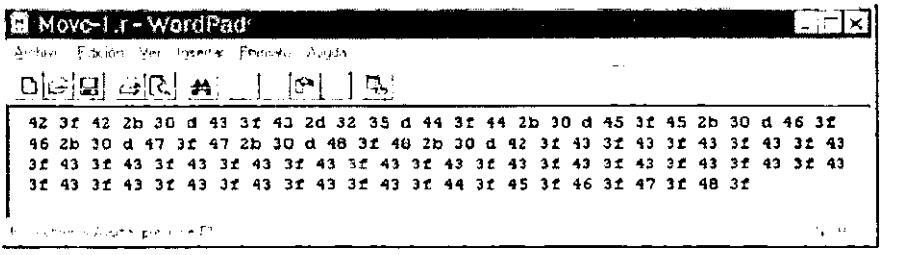
Figura 6.3 a y b. Muestran las pantallas del software que se tuvo que realizar para la lectura de las señales entre la computadora y el controlador usando Robotalk.

En la Figura 6.2 podemos observar que hay una interface 1 que comunica una PC llamada PC2, realizando la comunicación directa al controlador Mark-III, manda la señal para controlar el Mark III y a la vez para controlar el brazo mecánico, robot Rhino XR3, y de igual forma se puede decir que se realiza un puente en la interface para mandar las señales a la PC1. De ésta figura se puede decir que mientras en la PC2 ejecutamos el software que se incluye con el robot (Robotalk), en la PC1 ejecutamos el programa que me lee las señales enviadas de PC2 al controlador y las guarda en un archivo (Figuras 6.3 a y b). Dicho archivo lo almacene en código ASCII o en su defecto en Hexadecimal (Figuras 6.4 incisos a, b, c, d).

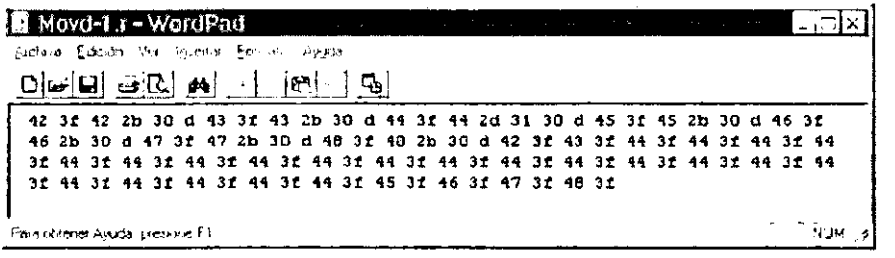
a)



b)



c)



d)

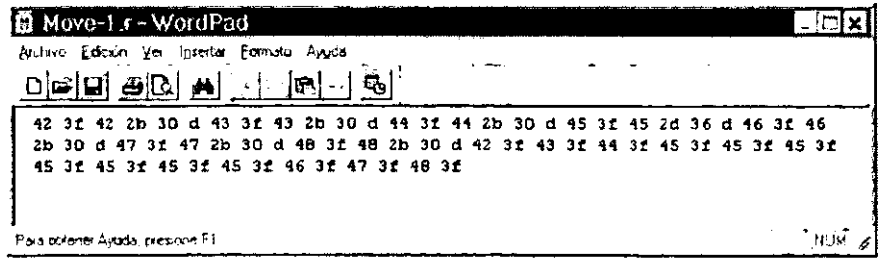


Figura 6.4 a) Código de movimientos del motor B, *b)* Código de movimientos del motor C, *c)* Código de movimientos del motor D, *d)* Código de movimientos del motor E. Todas las señales fueron leídas en Hexadecimal.

Una vez que se han leído las señales y almacenado sigue el análisis de cada una de las señales empleadas, de dicho análisis se obtiene la tabla siguiente:

Tabla 6.1. Correspondencia de la señal enviada al motor.

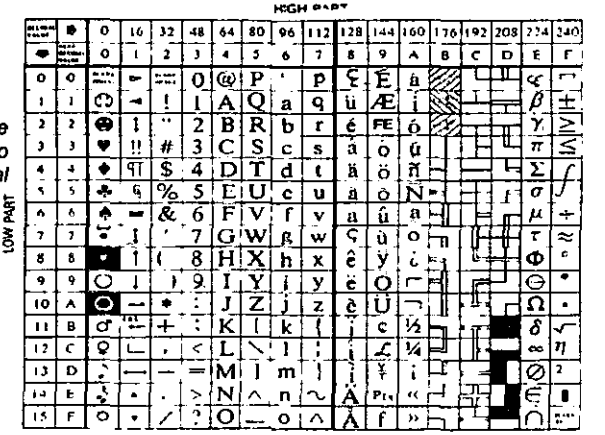
MOTOR	Código Asociado	
	Hexadecimal	ASCII
B	42	"B"
C	43	"C"
D	44	"D"
E	45	"E"
F	46	"F"

En la tabla anterior observemos que para controlar el motor B el código que se repite es el que corresponde a la letra "B" (en hexadecimal el número 42), de acuerdo a la tabla de caracteres de IBM PC (Ver tabla 6.2) y en los otros motores es prácticamente lo mismo.

IBM PC Display Characters

Tabla 6.2.

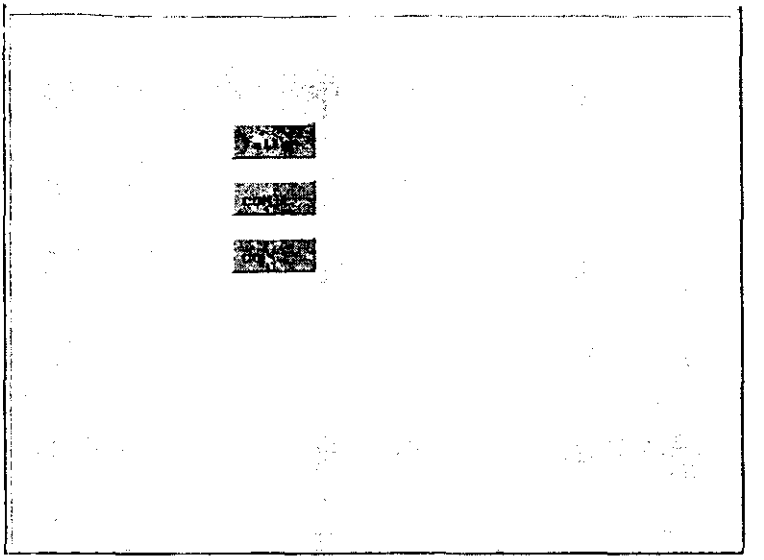
Desplegado de caracteres de acuerdo con el número correspondiente en hexadecimal y decimal.



➢ **Acercamientos :**

Al comenzar a implementar una solución trate de que fuera la optima, esto implicaba que debería contar con los elementos de control básicos (El control por separado de cada motor), así como presentar las opciones de guardar las instrucciones en un archivo, y mover el robot desde un punto a otro.

El primer acercamiento se realizó en lenguaje "C" obteniéndose el resultado que se muestra en la figura 6.5 incisos a) y b):



a)

b)

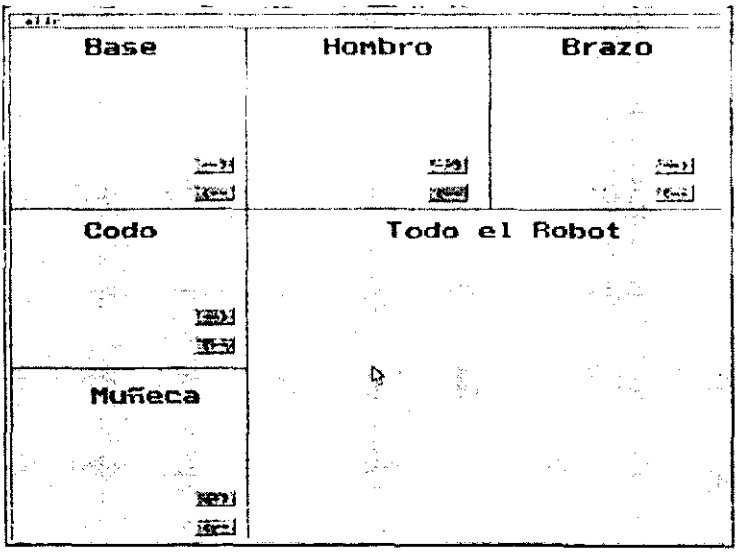
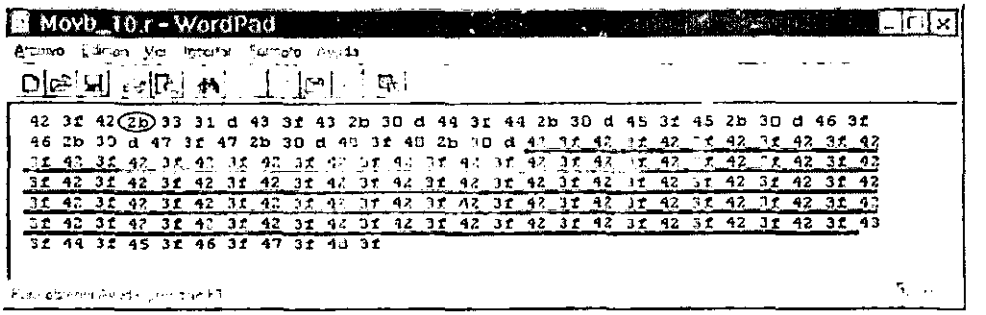


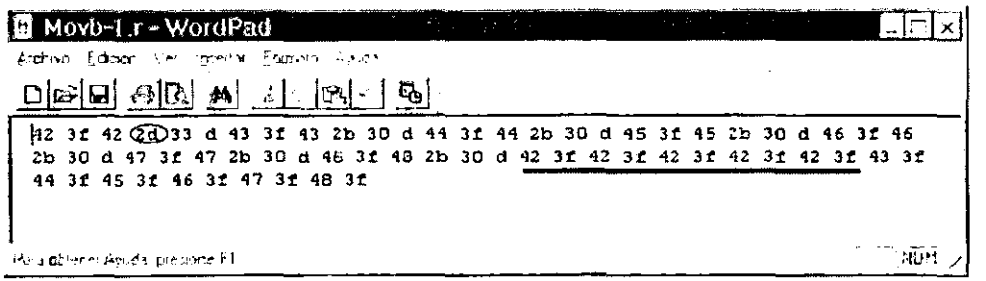
Figura 6.5. Pantallas del software realizado en lenguaje 'C', para controlar el robot Rhino XR-3.

Para obtener lo anterior fue necesario realizar un análisis exhaustivo, en el cual me pude dar cuenta de algunos parámetros (ver figura 6.6) como son:

El indicador de sentido de los motores (de la figura 6.6, sección encerrada en un ovalo), está determinado por el código 2d y 2b del encabezado de cada motor, la repetición del mismo elemento de acuerdo al motor que se ha especificado (de la figura 6.6, sección subrayada), de igual forma se observa que es proporcional el incremento de dichas señales conforme se le pide que aumente el número de pasos y/o grados que se moverá el motor requerido.



a) Mueve el motor B 10 pasos en sentido positivo.



b) Mueve el motor B 1 pasos en sentido negativo.

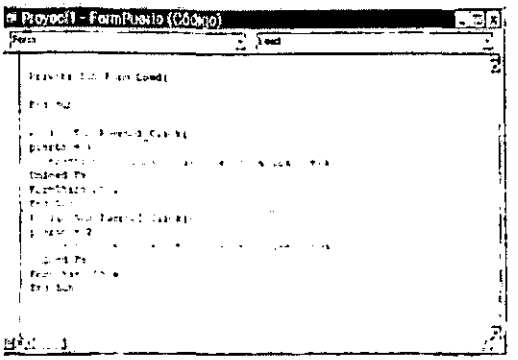
Figura 6.6 a) y b).Figuras representativas de las señales enviadas para mover el motor B del robot.

También debe notarse que existe un encabezado (que en ambas figuras corresponden a la primera linea y media) y un final de protocolo (que de igual manera, en ambos casos son las ultimas 12 señales).

El Software Deseado:

El software deseado se obtiene usando los antecedentes del punto anterior, una vez que se ha identificado los parámetros(señales) que indican sentido, motor y cantidad a repetir de los elementos.

El siguiente paso es desarrollar el sistema en un lenguaje popular como el Visual Basic en su versión 5 empresarial, por lo tanto el desarrollo hecho en C será codificado en Visual Basic (VB),



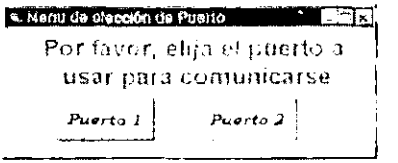
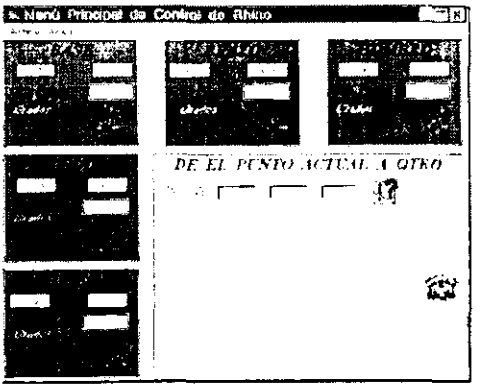
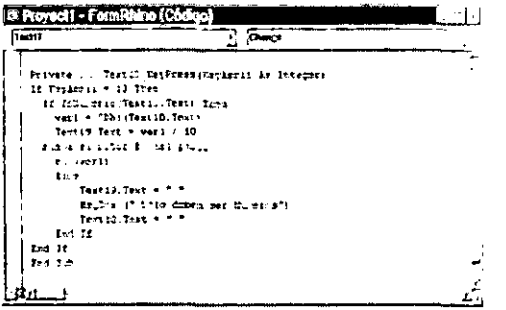


Figura 6.7.
Código fuente (Izquierda) de la pantalla de selección de puerto de comunicaciones (Arriba).

Como es evidente me tuve que ajustar a las reglas de programación y diseño de Visual Basic, para con ello lograr obtener el software que nos permita controlar el Robot Rhino XR-3





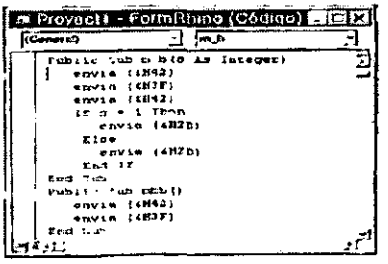


Figura 6.8.
El software controlador del Robot Rhino XR-3 (Superior Izquierda) cuenta con los acercamientos realizados en lenguaje 'C', pero con las reglas de Visual Basic 5 versión empresarial (Arriba e Izquierda Inferior, segmentos del código fuente del software).

7. RESULTADOS OBTENIDOS Y CONCLUSIONES

RESULTADOS:

- *Software Controlador del Robot Rhino XR-3.*

  Se ha realizado el software que permite controlar el robot, dicho software fue realizado en Visual Basic 5, esto fue cumpliendo las especificaciones marcadas por el sistema operativo Windows 95, aunque corre sin problemas en su versión 98.

- *Fuentes del Software Controlador del Robot Rhino XR-3.*

  Al desarrollar el software se pudo observar las diversas dificultades para lograr el objetivo, pero una vez alcanzado, logramos más de lo planeado originalmente, al tener los códigos fuentes de éste software abrimos las puertas para desarrollar aplicaciones especificas para el robot Rhino XR-3, como son combinar algoritmos de reconocimientos de patrones y una cámara al robot para que éste último tome objetos o realice alguna función, otro ejemplo sería combinar el software obtenido con redes neuronales que permitan la toma de decisiones del brazo mecánico, etc.

- *Manejo de la unidad controladora Mark-III.*

  Cuando se elabora el software uno piensa que de la computadora sale la señal y entra directamente al robot (o tan al menos eso pensé al inicio de éste proyecto), sin embargo, en éste caso no fue así, tenia que considerar la unidad controladora llamada Mark-III, ella es necesaria, ya que las señales de la computadora tienen que ser enviadas a la etapa de potencia del robot en un formato (RS-232), el cual es reconocido en la unidad controladora y que además cuenta con la etapa de potencia, de ella sale la señal a los motores del brazo mecánico.

- *Diversos programas desarrollados en 'C'.*

  Para obtener el objetivo planteado al inicio (un software que permita controlar el robot Rhino XR-3, así como los fuentes) fue necesario desarrollar varios programas que me permitieran acercarme al objetivo, de esta forma es como se realizó el programa que me permite leer las señales del cable (interface RS-232) y el primer acercamiento para controlar el robot.

CONCLUSIONES:

- *Diseño y Elaboración del Robot desde sus Bases.*

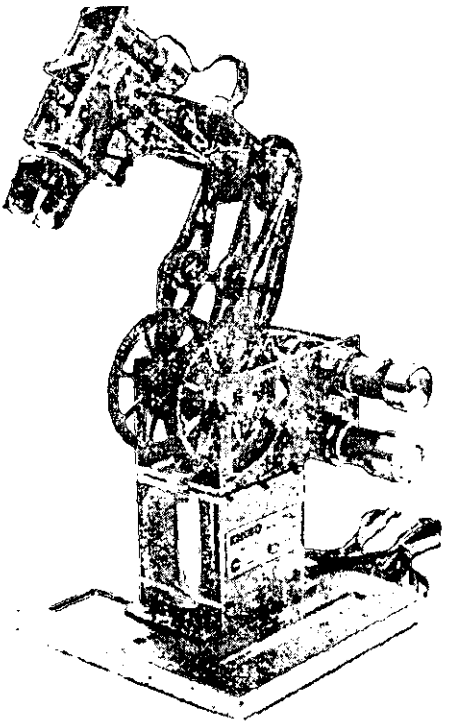
  Al realizar el presente proyecto, me he encontrado con diversos altibajos que me permiten asegurar que para tener un control absoluto de un robot lo conveniente es participar en su desarrollo desde el inicio, es decir, mientras que al tomar un robot ya armado tenemos que adecuarnos a las reglas del diseñador (en mi caso, por ejemplo, respetar el formato de la interface RS-232 o las señales a enviar para controlar el motor, etc.). En cambio, si la universidad se hubiera dado a la tarea de diseñar el robot desde sus bases, y la presente tesis hubiera participado desde el inicio, nosotros como diseñadores seriamos los que hubiéramos dado las normas a seguir y por consiguiente el estándar que se manejaría seria el que el equipo diseñador hubiera determinado (esto incluye las señales a enviar al robot, el tipo de controlador o etapa de potencia, etc.).

APÉNDICE "A"

Características del robot RHINO XR-3

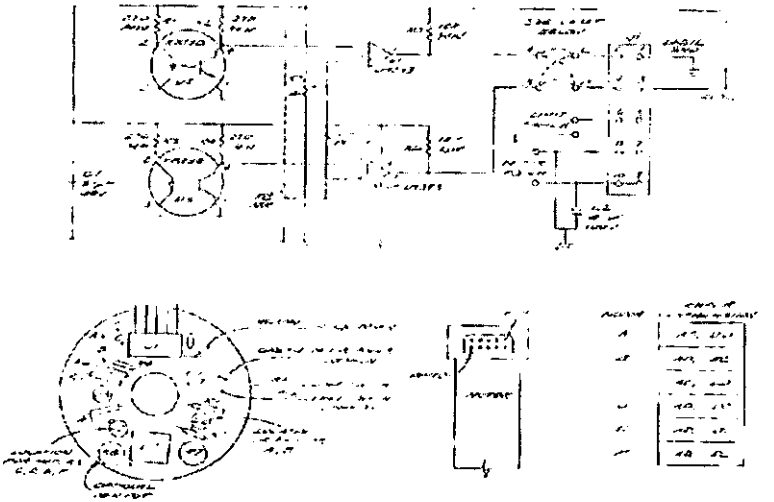
Brazo robot perteneciente a la serie XR-3,
Fabricante Rhino Robotics

El robot, cuenta con gran versatilidad y sofisticación, diseñado para la enseñanza e investigación. Cuenta con cinco grados de libertad y una mano estándar. La construcción del robot que muestra todo el mecanismo de movimientos permite realizar comparaciones y analogías con los robots de tipo industrial.



Construcción en aluminio, un servo motor por cada grado de libertad con codificadores reemplazables. Una línea de comunicación entre cada motor y el controlador.

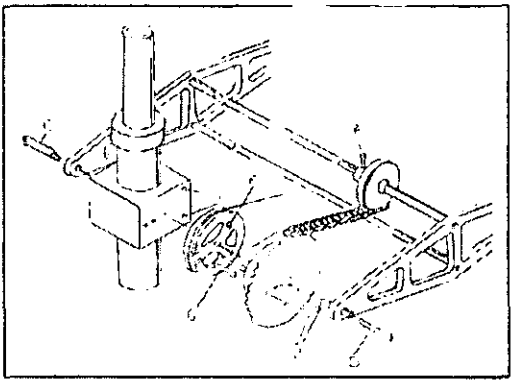
DIAGRAMA DE UN SERVO MOTOR DEL ROBOT RHINO XR - 3



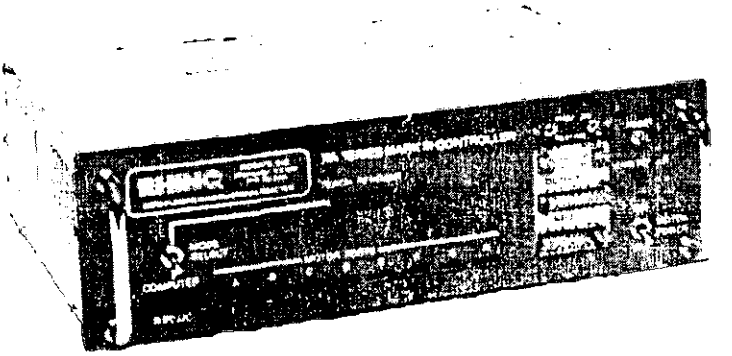
En éste esquema se muestra la estructura eléctrica del motor, de aquí he obtenido datos que considero importantes:

Elemento	Función
J1	Es el jumper que realiza la función de enlace entre el controlador Mark III con el motor respectivo del Rhino XR-3.
1	Es Tierra Lógica.
2 y 4	Parecen ser los pasos del motor.
3	Es la alimentación a 5 V DC.
5	No hay conexión.
6	Es el switch indicador del límite (cables rojo y negro), cuya función debería ser que cuando se llega al límite mecánico de la pieza (motor) el switch se cierra provocando el desvío de la corriente eléctrica, por lo que el motor se debe detener.
7 y 8	Están puenteados y son el común del motor (cable verde).
9 y 10	Están puenteados y al parecer llevan los +20 (-20) que mueven el motor.

Gráfica que muestra el ensamblado de las piezas del robot Rhino XR-3.



Controlador Mark III



Con la creciente fabricación de sofisticados sistemas automatizados, se ha estimulado la necesidad de que los trabajadores entiendan los conceptos que hacen que tales sistemas trabajen.

En Las escuelas y programas de entrenamiento se necesita preparar a cada vez más estudiantes y aprendices para el uso de éstas tecnologías de manufacturación. . Como el aprendizaje de la robótica dentro de la industria trae consigo costos elevados, Rhino Robots Inc. Ha fabricado sus robots educativos series XR, para los estudiantes y los aprendices industriales, y ha estandarizado su etapa de potencia a la serie MARK, la cual cuenta con 8 puertos identificados visualmente (6 para el brazo y 2 para accesorios), 8 líneas de entrada y 8 de salida, switch para activar el modo de teach pendat o computadora, puerto de comunicaciones para interface RS232C.

El controlador Mark-III, se puede describir mejor en el siguiente tabla:

Modelo	Controlador Mark III
Número de Parte	FGO792
Aplicaciones	Educativos, Entrenamiento Industrial, Investigación, Modelado de cadenas de producción.
Configuración	8 Puertos para motores del robot XR y accesorios, cada puerto provee 1.6 Amperes y un voltaje nominal de 20 Volts, cada puerto incluye líneas de lectura de los microswitch 2 Puertos no codificados para motores que operan dentro de una línea de producción, cada puerto provee 1.6 amperes como máximo y 20 volts nominales DC. 8 Entradas para la lectura de la línea de producción. 8 Salidas TTL para enviar señales a la línea de producción, cada línea provee 24 miliampers en bajo y 14 miliampers en alto.
Microprocesador	6502[a]
Interface de Comunicación	RS232C usando caracteres ASCII
Compatibilidad	Puede ejecutarce en cualquier máquina con un puerto serial e interface RS232C
Disponibilidad de Software	Robotalk, es el lenguaje controlador, disponible para maquinas compatibles con IBM y Apple con Rhino Com Lenguaje Card (sólo Apple).

APÉNDICE "B"

Manual de operación del Robot RHINO.

Bienvenido

Bienvenido a éste programa de control para el robot Rhino XR-III, La finalidad es la de proporcionar una herramienta para el control del robot ya mencionado, adicionando la ventaja de que al ser desarrollado dentro de la facultad de Ingeniería, se cuenta con el código fuente para futuras mejoras.

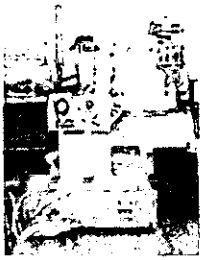
Los Requisitos mínimos para ejecutar la presente aplicación son:

- Una máquina con Windows 95.
- Por lo menos un puerto de comunicación serie no debe estar ocupado por algún dispositivo.
- El monitor debe ser a color, de preferencia UVGA.

Recuerde revisar el proceso de encendido para evitar contratiempos. De igual manera le recomiendo que vea la sección del uso del Software de esta forma estará enterado de cual la forma de controlar el robot.

Rhino XR-III

Conjunto de elementos desarrollado por Rhino Robotics inc. y que se compone del brazo mecánico Rhino XR-III y el controlador



PROCESO DE ENCENDIDO

Usted debe recordar siempre que inicie su sesión con el brazo mecánico Rhino XR III, ya que esto evitará contratiempos:

1. Asegúrese que todos los elementos (computadora y controlador Mark III), cuenten con energía eléctrica.

2. Conecte el controlador a la computadora, por medio de la interface que he diseñado usando el puerto serie de la computadora (el que se encuentre disponible), con el controlador Mark III (el conector DB25 que esta marcado como "Computer")

3. Verifique que este activada la modalidad computer del controlador (el switch, debe indicar "computer").

4. Encienda el Controlador: Primero active el switch que esta en la parte trasera a la posición ON y posteriormente el switch que esta en la parte frontal a la derecha parte inferior, igual pásela a la posición ON.

5. Encienda la computadora (Se ha probado con la computadora encendida y hasta el momento no me ha causado ningún problema), las ocasiones que he trabajado ha sido sin red, por lo que no se como responda conectada la máquina en red NT.

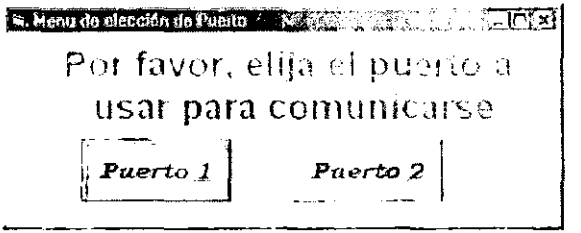
Ejecute el programa y ¡Comience a utilizar el Software!

Uso del Software

La idea es tener un software que permita controlar el robot de una forma intuitiva por ello es que se ha usado la siguiente configuración:

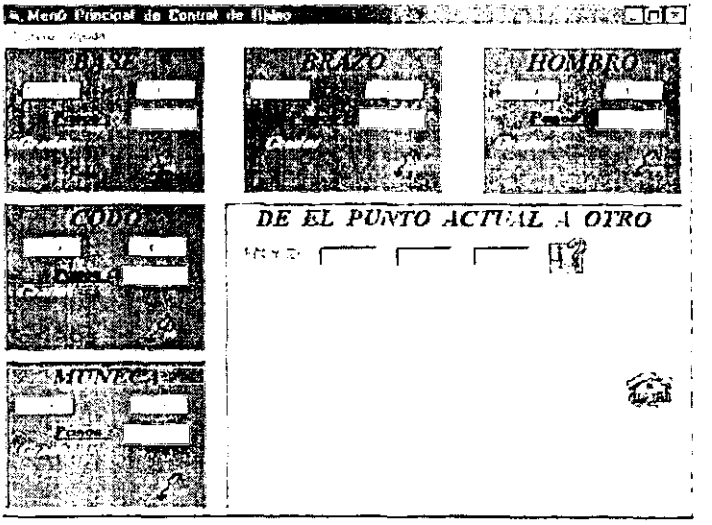
Parte 1: *Selección del puerto de comunicación.*

Es la primera pantalla que usted encontrará al ejecutar la aplicación, en ella usted tiene que seleccionar el puerto por el que realizará la comunicación con el Mark-III, debe recordar que dicha comunicación es por el puesto serie, se ha diseñado una interface que comunica por el puerto número 2.



Parte 2: Menú Principal.

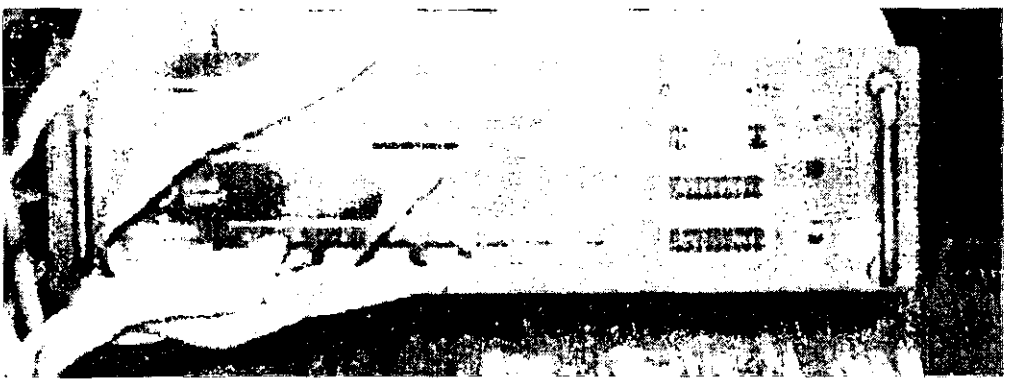
Una vez elegido el puerto de comunicaciones, aparece la pantalla siguiente que es la principal para el control del robot Rhino XR-III, en la cual se puede ver el control de cada uno de los elementos que forman el robot o en su caso poder cargar un archivo con los movimientos o en su defecto mover el robot de un punto a otro.



Cada elemento tiene una función especial, por ejemplo, la sección llamada base puede controlar los movimientos de toda la parte inferior, tanto a la izquierda o derecha (se dará cuenta que podemos decir positivo o negativo, pero en realidad esto dependerá del operador) y de igual manera podremos controlar cada uno de los elementos (Base, Brazo, Hombro, Codo y Muñeca), podrá darse cuenta que el actuador no esta configurado en éste menú, debido a que nunca hubo una especificación al respecto, sin embargo el Mark III cuenta con dos puertos extras que bien pueden servirnos para controlar el actuador.

Mark III

Es el controlador creado por Rhino Robotics[9], cuya finalidad es proporcionar la potencia y distribuir las señales a cada uno de sus 8 puertos (que están numerados desde la A hasta la H), para el brazo mecánico Rhino XR-III sólo he empleado 6 de ellos uno para cada motor, la forma en que entran las señales esta definida por el dip-swich que esta dentro del controlador el cual tiene la configuración de comunicación serie RS 232 a 9,600 baudios, con paridad par, bit de parada 2 y 7 bits de datos.



DB25

También conocido como conector tipo D de 25 pin's (como el mostrado en la siguiente figura 1), es compatible con el estándar ISO 2113 promulgado por la International Organization for Standardization (ISO). El documento donde se encuentran los detalles es:

Draft International Standard 2110, "Data Communication: 25 pin DTE/DCE interface connector and pin assignments" (La Norma Internacional 2110, " Comunicación de Datos: conector de 25 pin para DTE/DCE "), revisada por ISO 2110 en 1979.

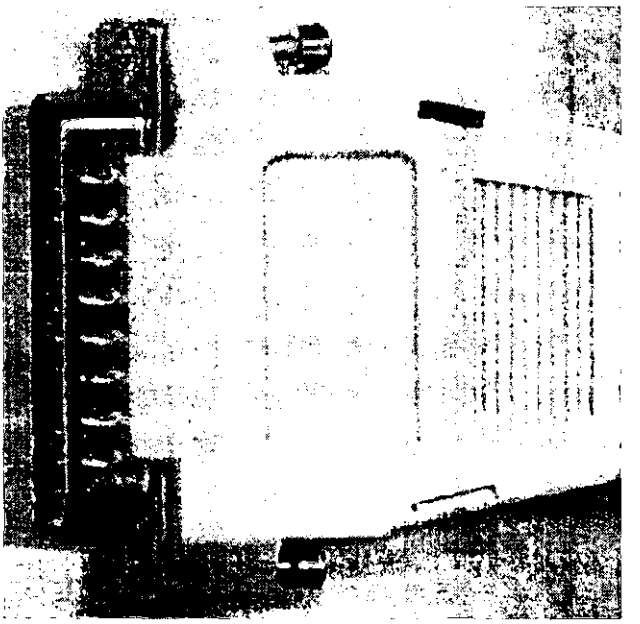
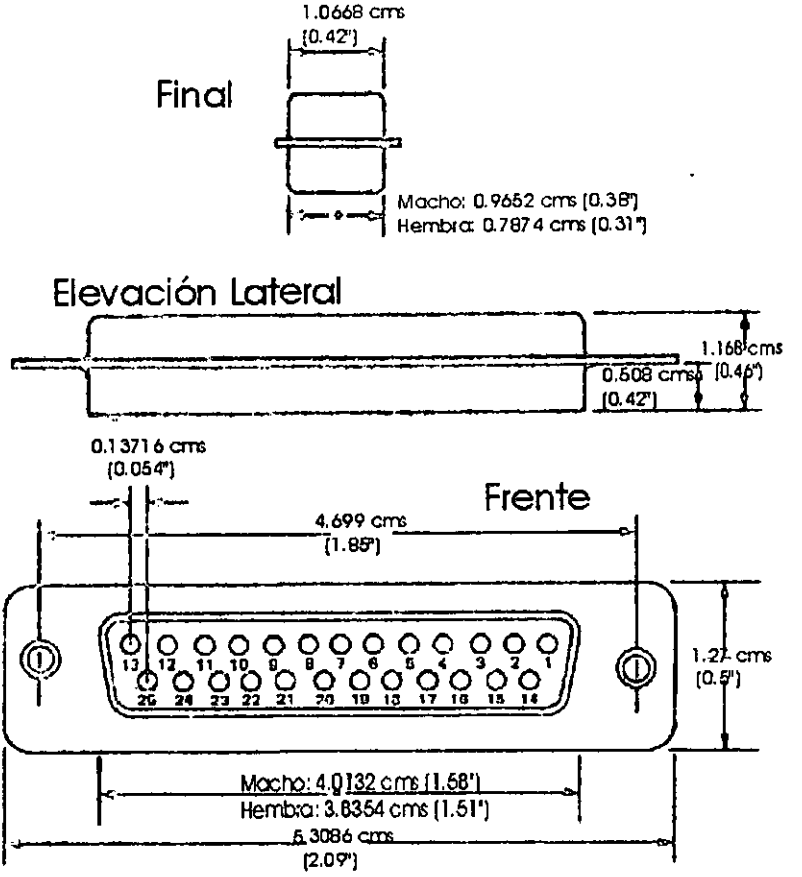


Figura 1

Las características físicas de ésta interface se muestra a continuación



Las características lógicas están señaladas en la siguiente tabla:

Tabla 2.4 (Ver documento de tesis, capitulo 2)

Pin	Circuito	Descripción
1	AA	Tierra de Protección (Física)
2	BA	Transmisión de Datos
3	BB	Recepción de Datos
4	CA	Demanda de Envío
5	CB	Limpiar Envío
6	CC	Los Datos están listos
7	AB	Señal de Tierra (lógica)
8	CF	Línea recibida por el Detector de señales

9 / 10	-	Reservado para Datos de prueba
11	-	Sin señalización
12	SCF	Línea Secundaria del Detector de Señales
13	SCB	Línea Secundaria de Limpiar Envío
14	SBA	Línea Secundaria de Transmisión de Datos
15	DB	Transmisión de señal de elemento cronometrado
16	SBB	Línea Secundaria de Datos Recibidos
17	DD	Recepción de señal de elemento cronometrado
18	-	Sin Señal
19	SCA	Línea Secundaria de Demanda de envío
20	CD	Terminación de Datos
21	CG	Detector de Señal de Calidad
22	CE	Indicador de llamada (Ring Indicator)
23	CH / CI	Los datos señalan proporción
24	DA	Transmisión de Señal de elemento cronometrado
25	-	Sin Señal

APÉNDICE "C"

CÓDIGO EN "C" PARA EL PROGRAMA QUE PERMITE LA LECTURA DE LOS CODIGOS DE ROBOTALK

```
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <bios.h>
#include <conio.h>
#include <ctype.h>

#define COM1     0
#define COM2     1
#define FiLE   0
#define MICRO     1
#define DATA_READY 0x100
#define TRUE     1
#define FALSE     0
#define T9600 ( 0xE0 | 0x08 | 0x00 | 0x03)     /* 9600 BAUDIOS, PARIDAD IMPAR, 1
BIT STOP, 8 BITS */
#define T4800 ( 0xC0 | 0x08 | 0x00 | 0x03)
#define T2400 ( 0xA0 | 0x08 | 0x00 | 0x03)
#define T1200 ( 0x80 | 0x08 | 0x00 | 0x03)
#define T600  ( 0x60 | 0x08 | 0x00 | 0x03)
#define T300  ( 0x40 | 0x08 | 0x00 | 0x03)
#define T150  ( 0x20 | 0x08 | 0x00 | 0x03)
#define T110  ( 0x00 | 0x08 | 0x00 | 0x03)
#define MTRANSMMITION      ( 0x80 | 0x00 | 0x00 | 0x03) /* 1200 BAUDIOS,SIN
PARIDAD,1 BIT STOP, 8 BITS */
#define ESC   '\x1b'
#define UP     '\x48'
#define DOWN      '\x50'
#define LEFT      '\x4B'
#define RIGHT     '\x4D'
#define ENTER     '\xD'
#define BACK        '\x8'


void graficos(void);
void shadow(void);
void remote(void);
void local(void);
void menu(void);
void contact(int settings);
```

```
void choice(void);
void coose(void);
void options(void);
void chooserate(int type);
void localdisplay(char data);
void remotedisplay(char data);
void sendfile(char *path, int settings);
void sendcode(char *path, int settings);
void capture(int settings, int type);
void message(void);
int setport(void);

int updown=0;
int leftright=0;
int COMM=COM1;

char a1[30];
FILE *info1;

int main(void)
{

        graficos();
        setbkcolor(BLUE);

        /* set color */
        setcolor(YELLOW);
        rectangle(0,0,getmaxx(),getmaxy());

        remote();
        local();
        printf("Nombre de archivo a crear: ");
        scanf("%s",a1);
        if((info1= fopen(a1,"wb+"))==NULL) {
         printf("no se abre");
         getch();
         exit(1);
         }
        menu();
        fclose(info1);

                /* clean up */
        closegraph();
        return 0;
}


void graficos(void){
```

```
   /* request auto detection */
   int gdriver = DETECT, gmode, errorcode;

   /* initialize graphics, local variables*/
   initgraph(&gdriver, &gmode, "");

         /* read result of initialization */
   errorcode = graphresult();
   if (errorcode != grOk)  /* an error
      occurred */
   {
                printf("error de graficos: %s\n", grapherrormsg(errorcode));
                printf("Presione una tecla:");
                getch();
                exit(1); /* terminate with error code */
   }
}

void local(void){
      /* set color */
      setcolor(YELLOW);

      /* set the fill style */
                setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE);

     /* draw the 3-d bar */
     bar3d(10,10, getmaxx()-190,getmaxy()/2-50,0, 1);

     /* draw title */
     outtextxy(20,20,"LOCAL . . .");
}

void remote(void){
      /* set color */
      setcolor(YELLOW);

      /* set the fill style */
     setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE);

     /* draw the 3-d bar */
                bar3d(10,getmaxy()/2-30, getmaxx()-190,getmaxy()-80,0, 1);

     /* draw title */
     outtextxy(20,getmaxy()/2-20,"REMOTE . . .");
}


void menu(void){
```

```
options();
COMM=setport();

while(1){
        switch(getch()){
case DOWN:          /* flecha abajo */
        updown++;
        if(updown>4)
        updown=1;
break;
case UP:            /* flecha arriba */
        updown--;
        if(updown<1)
        updown=4;
break;
}/* end switch get key */

switch(updown){             /* cual flecha fue presionada */

case 1:  /*     Ejecutar Programa seleccionado */
/* set the fill style */
setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE);

        /* draw the 3-d bar */
bar(getmaxx()-180,getmaxy()/2-20, getmaxx()-20,getmaxy()/2+20);
bar(getmaxx()-180,getmaxy()/2-50, getmaxx()-20,getmaxy()/2-90);
bar(getmaxx()-180,getmaxy()/2+50, getmaxx()-20,getmaxy()/2+90);

/* set the fill style */
setfillstyle(SOLID_FILL,WHITE);
bar(getmaxx()-180,getmaxy()/2-120, getmaxx()-20,getmaxy()/2-160);

/* justify text */
settextjustify(CENTER_TEXT,CENTER_TEXT);

/* set color */
setcolor(YELLOW);

/* title */
outtextxy(getmaxx()-100,getmaxy()/2,"MANDAR ARCHIVO");
        outtextxy(getmaxx()-100,getmaxy()/2-70,"MENSAJE");
outtextxy(getmaxx()-100,getmaxy()/2+70,"SALIR");

/* set color */
setcolor(RED);

outtextxy(getmaxx()-100,getmaxy()/2-140,"MICROCONTROLADOR");
```

```
while(!kbhit()){
if(getch()==ENTER)
capture(MTRANSMMITION,MICRO);
}
break;


case 2:  /*      mensaje seleccionado     */

        /* set the fill style */
setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE);

/* draw the 3-d bar */
bar(getmaxx()-180,getmaxy()/2-20, getmaxx()-20,getmaxy()/2+20);
bar(getmaxx()-180,getmaxy()/2+50, getmaxx()-20,getmaxy()/2+90);
bar(getmaxx()-180,getmaxy()/2-120, getmaxx()-20,getmaxy()/2-160);

/* set the fill style */
setfillstyle(SOLID_FILL,WHITE);

bar(getmaxx()-180,getmaxy()/2-50, getmaxx()-20,getmaxy()/2-90);


/* set color */
setcolor(YELLOW);

        /* justify text */
settextjustify(CENTER_TEXT,CENTER_TEXT);

/* title */
outtextxy(getmaxx()-100,getmaxy()/2-140,"MICROCONTROLADOR");
outtextxy(getmaxx()-100,getmaxy()/2,"MANDAR ARCHIVO");
outtextxy(getmaxx()-100,getmaxy()/2+70,"SALIR");

/* set color */
setcolor(RED);
outtextxy(getmaxx()-100,getmaxy()/2-70,"MENSAJE");

while(!kbhit()){
if(getch()==ENTER)
chooserate(0);
}

        break;

case 3:  /*      mandar archivo seleccionado     */

/* set the fill style */
```

```
setfillstyle(SOLID_FILL,WHITE);

/* draw the 3-d bar */
bar(getmaxx()-180,getmaxy()/2-20, getmaxx()-20,getmaxy()/2+20);

      /* set the fill style */
setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE);

bar(getmaxx()-180,getmaxy()/2-50, getmaxx()-20,getmaxy()/2-90);
bar(getmaxx()-180,getmaxy()/2+50, getmaxx()-20,getmaxy()/2+90);
bar(getmaxx()-180,getmaxy()/2-120, getmaxx()-20,getmaxy()/2-160);


/* set color */
setcolor(RED);

/* justify text */
settextjustify(CENTER_TEXT,CENTER_TEXT);

/* title */
outtextxy(getmaxx()-100,getmaxy()/2,"MANDAR ARCHIVO");

/* set color */
setcolor(YELLOW);

outtextxy(getmaxx()-100,getmaxy()/2-140,"MICROCONTROLADOR");
outtextxy(getmaxx()-100,getmaxy()/2-70,"MENSAJE");
outtextxy(getmaxx()-100,getmaxy()/2+70,"SALIR");

        while(!kbhit()){
if(getch()==ENTER)
chooserate(1);
}

break;

case 4:  /*      salir seleccionado   */
/* set the fill style */
setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE);

/* draw the 3-d bar */
bar(getmaxx()-180,getmaxy()/2-20, getmaxx()-20,getmaxy()/2+20);
bar(getmaxx()-180,getmaxy()/2-50, getmaxx()-20,getmaxy()/2-90);
bar(getmaxx()-180,getmaxy()/2-120, getmaxx()-20,getmaxy()/2-160);

/* set the fill style */
        setfillstyle(SOLID_FILL,WHITE);
```

```
    bar(getmaxx()-180,getmaxy()/2+50, getmaxx()-20,getmaxy()/2+90);



    /* justify text */
    settextjustify(CENTER_TEXT,CENTER_TEXT);

    /* set color */
    setcolor(YELLOW);

    /* title */

    outtextxy(getmaxx()-100,getmaxy()/2-140,"MICROCONTROLADOR");
    outtextxy(getmaxx()-100,getmaxy()/2,"MANDAR ARCHIVO");
    outtextxy(getmaxx()-100,getmaxy()/2-70,"MENSAJE");

    /* set color */
    setcolor(RED);
    outtextxy(getmaxx()-100,getmaxy()/2+70,"SALIR");

    while(!kbhit()){
    if(getch()==ENTER)
    return;
    }
    break;

                        }/*end switch updown*/

      }/*    end while    */
}


void contact(int settings){
      int in, out, status, DONE = FALSE;

      bioscom(0, settings, COMM);                    /* INICIALIZA EL PUERTO
A LA VELOCIDAD SELECCIONADA */


      while (!DONE)
      {
            status = bioscom(3, 0, COMM);            /* VERIFICA EL ESTADO
DEL PUERTO */
            if (status & DATA_READY)                     /* PETICION DE
ENTRADA DE DATOS */
       if ((out = bioscom(2, 0, COMM) & 0xFF) != 0){  /* SI NO ES UN CARACTER
NULO  */
             printf("%d ",out);
```

```
                fprintf(info1,"%d ",out);
                remotedisplay(out);                    /* DESPLEGA EL DATO
QUE LLEGO */
                }
        if (kbhit())
        {
        if ((in = getch()) == ESC || in==LEFT || in==RIGHT || in==UP || in==DOWN ){
                        DONE = TRUE;
                        continue;}
                bioscom(1, in, COMM);                  /* ENVIA DATOS */
                localdisplay(in);                      /* DESPLEGA DATOS */
        }
        }
}

void choice(void){

}

void coose(void){
    /*  set align */
    settextjustify(CENTER_TEXT,CENTER_TEXT);

     /* set the fill style */
    setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE);

    /* draw the 3-d bar */
    bar(10,getmaxy()-60, getmaxx()-10,getmaxy()-20);

    setcolor(YELLOW);

    /* draw title */
    outtextxy(60,getmaxy()-70,"RATE . . .");
    outtextxy(48.75,getmaxy()-40,"110");
    outtextxy(48.75+77.5,getmaxy()-40,"150");
    outtextxy(48.75+77.5*2,getmaxy()-40,"300");
    outtextxy(48.75+77.5*3,getmaxy()-40,"600");
    outtextxy(48.75+77.5*4,getmaxy()-40,"1200");
    outtextxy(48.75+77.5*5,getmaxy()-40,"2400");
    outtextxy(48.75+77.5*6,getmaxy()-40,"4800");
    outtextxy(48.75+77.5*7,getmaxy()-40,"9600");
}

void shadow(){
    /* set the fill style */
    setfillstyle(SOLID_FILL,DARKGRAY);

    /* draw shadow */
```

```
    bar(getmaxx()-170,getmaxy()/2-10, getmaxx()-10,getmaxy()/2+30);
    bar(getmaxx()-170,getmaxy()/2-40, getmaxx()-10,getmaxy()/2-80);
    bar(getmaxx()-170,getmaxy()/2+60, getmaxx()-10,getmaxy()/2+100);
    bar(getmaxx()-170,getmaxy()/2-110, getmaxx()-10,getmaxy()/2-150);
    bar(20,getmaxy()-50, getmaxx()-5,getmaxy()-10);

}

void options(void){
    shadow();

    /* set the fill style */
    setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE);

    /* draw the 3-d bar */
    bar(getmaxx()-180,getmaxy()/2-20, getmaxx()-20,getmaxy()/2+20);
    bar(getmaxx()-180,getmaxy()/2-50, getmaxx()-20,getmaxy()/2-90);
    bar(getmaxx()-180,getmaxy()/2+50, getmaxx()-20,getmaxy()/2+90);
    bar(getmaxx()-180,getmaxy()/2-120, getmaxx()-20,getmaxy()/2-160);


    /* set color */
    setcolor(YELLOW);

    /* justify text */
    settextjustify(CENTER_TEXT,CENTER_TEXT);

    /* title */
    outtextxy(getmaxx()-100,getmaxy()/2-140,"MICROCONTROLADOR");
    outtextxy(getmaxx()-100,getmaxy()/2,"MANDAR ARCHIVO");
    outtextxy(getmaxx()-100,getmaxy()/2-70,"MENSAJE");
    outtextxy(getmaxx()-100,getmaxy()/2+70,"SALIR");

    coose();
}

void chooserate(int type){
while(1){
    switch(getch()){
    case RIGHT:          /* flecha derecha */
            leftright++;
            if(leftright>8)
            leftright=1;
    break;
    case LEFT:           /* flecha izquierda */
            leftright--;
            if(leftright<1)
```

```
            leftright=8;
    break;
    case ESC:
    case UP:
    case DOWN:
    coose();
    return;
    break;
    }/* end switch key */


    switch(leftright){
    case 1:   coose();
              /*  set align */
              settextjustify(CENTER_TEXT,CENTER_TEXT);

              /* set the fill style */
              setfillstyle(SOLID_FILL,WHITE);

              /* draw the 3-d bar */
              bar(10,getmaxy()-60,10+(getmaxx()-20)/8,getmaxy()-20);

              /* set the color*/
              setcolor(RED);

              /* draw title */
              outtextxy(48.75,getmaxy()-40,"110");
    while(!kbhit())
    if(getch()==ENTER)
              switch(type){
              case 0:
              contact(T110);
              break;
              case 1:
              capture(T110,FiLE);
              break;
              }

    break;
    case 2:  coose();

              /*  set align */
              settextjustify(CENTER_TEXT,CENTER_TEXT);

              /* set the fill style */
              setfillstyle(SOLID_FILL,WHITE);
```

```
                /* draw the 3-d bar */
                bar(10 + (getmaxx()-20)/8,getmaxy()-60,10 + (getmaxx()-
20)/8*2,getmaxy()-20);

            /* set the/colo*/
                setcolor(RED);

                /* draw title */
                outtextxy(48.75+77.5,getmaxy()-40,"150");
    while(!kbhit())
    if(getch()==ENTER)
                switch(type){
                case 0:
                contact(T150);
                break;
                case 1:
                capture(T150,FiLE);
                break;
                }

    break;
    case 3:   coose();

                /*  set align */
                settextjustify(CENTER_TEXT,CENTER_TEXT);

                /* set the fill style */
                setfillstyle(SOLID_FILL,WHITE);

                /* draw the 3-d bar */
                bar(10 + (getmaxx()-20)/8*2,getmaxy()-60,10 + (getmaxx()-
20)/8*3,getmaxy()-20);

            /* set the/colo*/
                setcolor(RED);

                /* draw title */
                outtextxy(48.75+77.5*2,getmaxy()-40,"300");
    while(!kbhit())
    if(getch()==ENTER)
                switch(type){
                case 0:
                contact(T300);
                break;
                case 1:
                capture(T300,FiLE);
                break;
                }
```

```
    break;
    case 4:  coose();

             /*  set align */
             settextjustify(CENTER_TEXT,CENTER_TEXT);

             /* set the fill style */
             setfillstyle(SOLID_FILL,WHITE);

             /* draw the 3-d bar */
             bar(10 + (getmaxx()-20)/8*3,getmaxy()-60,10 + (getmaxx()-
20)/8*4,getmaxy()-20);

          /* set the/colo*/
             setcolor(RED);

             /* draw title */
             outtextxy(48.75+77.5*3,getmaxy()-40,"600");
    while(!kbhit())
    if(getch()==ENTER)
             switch(type){
             case 0:
             contact(T600);
             break;
             case 1:
             capture(T600,FiLE);
             break;
             }

    break;
    case 5:  coose();

             /*  set align */
             settextjustify(CENTER_TEXT,CENTER_TEXT);

             /* set the fill style */
             setfillstyle(SOLID_FILL,WHITE);

             /* draw the 3-d bar */
             bar(10 + (getmaxx()-20)/8*4,getmaxy()-60,10 + (getmaxx()-
20)/8*5,getmaxy()-20);

             /* set the colo*/
             setcolor(RED);

             /* draw title */
             outtextxy(48.75+77.5*4,getmaxy()-40,"1200");
    while(!kbhit())
```

```
    if(getch()==ENTER)
                switch(type){
                case 0:
                contact(T1200);
                break;
                case 1:
                capture(T1200,FiLE);
                break;
                }

    break;
    case 6:   coose();

                /*  set align */
                settextjustify(CENTER_TEXT,CENTER_TEXT);

                /* set the fill style */
                setfillstyle(SOLID_FILL,WHITE);

                /* draw the 3-d bar */
                bar(10 + (getmaxx()-20)/8*5,getmaxy()-60,10 + (getmaxx()-
20)/8*6,getmaxy()-20);

            /* set the/colo*/
                setcolor(RED);

                /* draw title */
                outtextxy(48.75+77.5*5,getmaxy()-40,"2400");
    while(!kbhit())
    if(getch()==ENTER)
                switch(type){
                case 0:
                contact(T2400);
                break;
                case 1:
                capture(T2400,FiLE);
                break;
                }
    break;
    case 7:   coose();

                /*  set align */
                settextjustify(CENTER_TEXT,CENTER_TEXT);

                /* set the fill style */
                setfillstyle(SOLID_FILL,WHITE);

                /* draw the 3-d bar */
```

```
            bar(10 + (getmaxx()-20)/8*6,getmaxy()-60,10 + (getmaxx()-
20)/8*7,getmaxy()-20);

        /* set the/colo*/
            setcolor(RED);

            /* draw title */
            outtextxy(48.75+77.5*6,getmaxy()-40,"4800");

    while(!kbhit())
    if(getch()==ENTER)
            switch(type){
            case 0:
            contact(T4800);
            break;
            case 1:
            capture(T4800,FiLE);
            break;
            }
    break;
    case 8:  coose();
            /*  set align */
            settextjustify(CENTER_TEXT,CENTER_TEXT);

            /* set the fill style */
            setfillstyle(SOLID_FILL,WHITE);

            /* draw the 3-d bar */
            bar(10 + (getmaxx()-20)/8*7,getmaxy()-60,getmaxx()-10,getmaxy()-20);

        /* set the/colo*/
            setcolor(RED);

            /* draw title */
            outtextxy(48.75+77.5*7,getmaxy()-40,"9600");

    while(!kbhit())
    if(getch()==ENTER)
            switch(type){
            case 0:
            contact(T9600);
            break;
            case 1:
            capture(T9600,FiLE);
            break;
            }
```

```
                break;
                        }/* end switch leftright*/


}/* end while*/
}


void localdisplay(char data){
char msg[1];
/*
        inicio en Local 30,30
        inicio en Remote

*/
static int x=30,y=50;
msg[1]='\0';


    /*  set align */
    settextjustify(LEFT_TEXT,BOTTOM_TEXT);
    /*  set color */
    setcolor(WHITE);

    msg[0]=data;

       switch(msg[0]){
       case ESC:
       return;
       break;
       case ENTER:
       if(y>getmaxy()/2-70)
       y=50;
       x=30;
       y+=10;

       /* set the fill style */
       setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE );

     /* draw the 3-d bar */
       bar(30,y-10,getmaxx()-210,y+10);

       break;
       case BACK:
                        /* set the fill style */
                        setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE);

                        /* draw the 3-d bar */
```

```
                    bar(x-textwidth(msg)-1,y-10,x,y);

                    x-=textwidth(msg)+1;
        break;
        default:
                    if(isprint(msg[0])){
                    if(x>getmaxx()-220){

                        /* set the fill style */
                        setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE );

                        /* draw the 3-d bar */
                        bar(30,y,getmaxx()-210,y+20);
                        x=30;
                        y+=10;

                        }
                    if(y>getmaxy()/2-70){
                        y=50;

                        /* set the fill style */
                        setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE );

                        /* draw the 3-d bar */
                        bar(30,y-10,getmaxx()-210,y+10);
                    }
                    outtextxy(x,y,msg);
                    x+=textwidth(msg)+1;
                    }/* en if */
        break;
        }/* end switch */

}

void remotedisplay(char data){
char msg[1];
/*
        inicio en Local 30,getmaxxy()/2+10
        inicio en Remote

*/
static int x=30,y=249;
msg[1]='\0';

    /*  set align */
    settextjustify(LEFT_TEXT,BOTTOM_TEXT);
    /*  set color */
    setcolor(WHITE);
```

```
msg[0]=data;

   switch(msg[0]){
   case ESC:
   return;
   break;
   case ENTER:
   if(y>getmaxy()-100)
   y=getmaxy()/2+10;
   x=30;
   y+=10;

   /* set the fill style */
   setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE );

 /* draw the 3-d bar */
   bar(30,y-10,getmaxx()-210,y+10);

   break;
   case BACK:
                /* set the fill style */
                setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE);

                /* draw the 3-d bar */
                bar(x-textwidth(msg)-1,y-10,x,y);

                x-=textwidth(msg)+1;
   break;
   default:
                if(isprint(msg[0])){
                if(x>getmaxx()-220){

                        /* set the fill style */
                        setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE );

                        /* draw the 3-d bar */
                        bar(30,y,getmaxx()-210,y+20);
                        x=30;
                        y+=10;

                        }
                if(y>getmaxy()-100){
                        y=getmaxy()/2+10;

                        /* set the fill style */
                        setfillstyle(SOLID_FILL,LIGHTGREEN + WHITE );
```

```
                        /* draw the 3-d bar */
                        bar(30,y-10,getmaxx()-210,y+10);
                    }
                    outtextxy(x,y,msg);
                    x+=textwidth(msg)+1;
                    }/* end if */
        break;
        }/* end switch */

}

void sendfile(char *path,int settings){
FILE *point;
char c;


        if ((point = fopen(path, "rt")) == NULL){
        message();
        return;
        }
  bioscom(0, settings, COMM);                    /* INICIALIZA EL PUERTO A LA
VELOCIDAD SELECCIONADA */

        while (!feof(point)){
                c=getc(point);
                bioscom(1, c, COMM);              /* ENVIA DATOS */
                localdisplay(c);                /* DESPLEGA DATOS */
                }
fclose(point);
}

void capture( int settings, int type){
  void *window;
  unsigned int size;
  char c, *path;
  int ptr=0;

  /* calculate the size of the image */
  size = imagesize(getmaxx()/2-150,getmaxy()/2-70,getmaxx()/2+150,getmaxy()/2+50);

  /* allocate memory to hold the image */
  window = malloc(size);

  /* grab the image */
  getimage(getmaxx()/2-150,getmaxy()/2-70,getmaxx()/2+150,getmaxy()/2+50,window);

  /* set the fill style */
```

```
  setfillstyle(SOLID_FILL,RED);
  setcolor(WHITE);

  /* set align */
  settextjustify(LEFT_TEXT,BOTTOM_TEXT);

  /* draw window */
  bar3d(getmaxx()/2-150,getmaxy()/2-70,getmaxx()/2+150,getmaxy()/2+50,0,1);

  /* put label */
  outtextxy(getmaxx()/2-130,getmaxy()/2-40,"Dame el nombre del archivo.");
  outtextxy(getmaxx()/2-130,getmaxy()/2-10,">_");

  while((c=getch())!=ENTER ){
if(c!=ESC){
  if(c!=BACK){
       if(ptr<25){
            path[ptr++]=c;
            path[ptr]='\0';
            outtextxy(getmaxx()/2-110,getmaxy()/2-10,path);
             }
       else{
            c=ESC;
            outtextxy(getmaxx()/2-130,getmaxy()/2+30,"Verifique nombre.");
            getch();
            continue;
       }/* end if ptr */
   }/* end if BACK */
   else{
            path[--ptr]='\0';
            bar(getmaxx()/2-110,getmaxy()/2-20,getmaxx()/2+120,getmaxy()/2+20);
            outtextxy(getmaxx()/2-110,getmaxy()/2-10,path);
   }
}/* end if ESC */
else {
  putimage(getmaxx()/2-150,getmaxy()/2-70,window,COPY_PUT);

  /* clean up */
  free(window);
}
}/* end while */

  putimage(getmaxx()/2-150,getmaxy()/2-70,window,COPY_PUT);

  /* clean up */
  free(window);

  switch(type){
```

```
  case FILE:
            sendfile(path,settings);
            break;
  case MICRO:
            sendcode(path,settings);
            break;
  }
}

void sendcode(char *path, int settings){
FILE *input, *output;
int row=0,i,length=0;
long hex;
char
      lastline[40]="\0",
      line[40]="\0",
      c[2]="\0",
      *endptr;

      if ((input = fopen(path, "rt")) == NULL){
      message();
      return;
      }
      if((output = fopen("code.hex", "wt")) == NULL)
      message();

      while (!feof(input)){
      if(fgetc(input)=='S')
      row++;
      }

      /* in the begining of file */
      fseek(input,0,SEEK_SET);

      /* put permission character at the beginning */
      fputc('F',output);
      fputc('F',output);

      while(row-1){
      if(fgetc(input)=='S'){
            fseek(input,7,SEEK_CUR);
      fgets(line,33,input);
      if((length=strlen(line))==32)
            fputs(line,output);
      else
      {
            strncpy(lastline,line,length-3);
            fputs(lastline,output);
```

```
        }
        row--;
        }/* en if */

        }/* end while */
        fclose(input);
        fclose(output);

        output = fopen("code.hex","rt");
        bioscom(0, settings, COMM);                    /* INICIALIZA EL PUERTO
A LA VELOCIDAD SELECCIONADA */


        while((c[0] = fgetc(output)) != EOF){
        c[1]=fgetc(output);
        c[2]='\0';

                /* strtol converts string to long integer  */
                hex = strtol(c, &endptr, 16);


          //    bioscom(1, hex, COMM);                   /* ENVIA DATOS */
        }

        /* in the begining of file */
        fseek(output,0,SEEK_SET);
        while(!feof(output))
                localdisplay(getc(output));

        fclose(output);

}
void message(void){
void *window;
unsigned int size;

  /* calculate the size of the image */
  size = imagesize(getmaxx()/2-150,getmaxy()/2-30,getmaxx()/2+150,getmaxy()/2+30);

  /* allocate memory to hold the image */
  window = malloc(size);

  /* grab the image */
  getimage(getmaxx()/2-150,getmaxy()/2-30,getmaxx()/2+150,getmaxy()/2+30,window);

  /* set the fill style */
  setfillstyle(SOLID_FILL,RED);
  setcolor(WHITE);
```

```
  /*  set align */
  settextjustify(LEFT_TEXT,BOTTOM_TEXT);

  /* draw window */
  bar3d(getmaxx()/2-150,getmaxy()/2-30,getmaxx()/2+150,getmaxy()/2+30,0,1);

  /* put label */
  outtextxy(getmaxx()/2-130,getmaxy()/2,"No se encontro el archivo!.");

  getch();

  /* restore the image */
  putimage(getmaxx()/2-150,getmaxy()/2-30,window,COPY_PUT);

  /* clean up */
  free(window);
}
int setport(void){
  void *window;
  unsigned int size;
  char port;
  int portid=0;

  /* calculate the size of the image */
  size = imagesize(getmaxx()/2-150,getmaxy()/2-70,getmaxx()/2+150,getmaxy()/2+50);

  /* allocate memory to hold the image */
  window = malloc(size);

  /* grab the image */
  getimage(getmaxx()/2-150,getmaxy()/2-70,getmaxx()/2+150,getmaxy()/2+50,window);

  /* set the fill style */
  setfillstyle(SOLID_FILL,GREEN);
  setcolor(WHITE);

  /*  set align */
  settextjustify(LEFT_TEXT,BOTTOM_TEXT);

  /* draw window */
  bar3d(getmaxx()/2-150,getmaxy()/2-70,getmaxx()/2+150,getmaxy()/2+50,0,1);

  /* put label */
  outtextxy(getmaxx()/2-130,getmaxy()/2-40,"Selecciona puerto de comunicaci¢n:");
  outtextxy(getmaxx()/2-130,getmaxy()/2-10,"COMM1 = 0          COMM2 = 1");

  outtextxy(getmaxx()/2-130,getmaxy()/2+20,">_");
```

```
  while((port=getch())!=ENTER ){
if(port!=ESC){
  if(port!=BACK){
        switch(port){
        case '0':
                outtextxy(getmaxx()/2-110,getmaxy()/2+20,"COMM1");
                portid=COM1;
                break;
        case '1':
                outtextxy(getmaxx()/2-110,getmaxy()/2+20,"COMM2");
                portid=COM2;
                break;
        default:
                outtextxy(getmaxx()/2-130,getmaxy()/2+40,"Verifique puerto.");
                getch();
                continue;
                break;
        }
   }/* end if BACK */
   else  /* else BACK */
                bar(getmaxx()/2-110,getmaxy()/2+10,getmaxx()/2+120,getmaxy()/2+30);
}
else {
        bar(getmaxx()/2-130,getmaxy()/2+30,getmaxx()/2+120,getmaxy()/2+40);
}
}/* end while */

  putimage(getmaxx()/2-150,getmaxy()/2-70,window,COPY_PUT);

  /* clean up */
  free(window);

  return portid;
}
```

CÓDIGO EN "VISUAL BASIC" PARA EL PROGRAMA QUE PERMITE EL CONTROL DEL ROBOT RHINO XR-III.

De la forma de elección de puerto de comunicaciones:

```
Public Sub MSComm1_OnComm()

End Sub

Private Sub Form_Load()

End Sub

Public Sub Puerto1_Click()
puerto = 1
' Desactivo la forma actual y entro a una nueva
Unload Me
FormRhino.Show
End Sub
Public Sub Puerto2_Click()
puerto = 2
' Desactivo la forma actual y entro a una nueva
Unload Me
FormRhino.Show
End Sub
```

De la Forma Principal para el control del Robot

```
Public indicArchivo As Integer
Public nombreArchivo As String
Public i As Integer
Public signo As Integer
Public Px, Py, Pz, I1, I2, I3, I4, I5 As Integer
Private Sub ini_robot()
envia (&H23)
envia (&H23)
envia (&H49)
envia (&H4D)
envia (&H4F)
envia (&H50)
envia (&H31)
envia (&H32)
envia (&H50)
envia (&H33)
envia (&H50)
envia (&H34)
envia (&H50)
```

```
envia (&H35)
envia (&H50)
envia (&H36)
envia (&H50)
envia (&H37)
envia (&H50)
envia (&H38)
envia (&H23)
End Sub

Public Sub inicializa()
   MSComm1.CommPort = puerto
   MSComm1.Settings = "9600,e,7,2"
   MSComm1.PortOpen = True
End Sub
Public Sub envia(caracter As Integer)
  Static c As Variant
    c = Chr(caracter)
    MSComm1.Output = c
End Sub
Public Sub m_b(s As Integer)
  envia (&H42)
  envia (&H3F)
  envia (&H42)
  If s = 1 Then
     envia (&H2B)
   Else
     envia (&H2D)
   End If
End Sub
Public Sub mbb()
  envia (&H42)
  envia (&H3F)
End Sub
Public Sub mcc()
  envia (&H43)
  envia (&H3F)
End Sub

Public Sub mdd()
  envia (&H44)
  envia (&H3F)
End Sub

Public Sub mee()
  envia (&H45)
  envia (&H3F)
End Sub
```

```
Public Sub mff()
  envia (&H46)
  envia (&H3F)
End Sub

Public Sub mgg()
  envia (&H47)
  envia (&H3F)
End Sub

Public Sub mhh()
  envia (&H48)
  envia (&H3F)
End Sub

Public Sub m_c(s As Integer)
  envia (&H43)
  envia (&H3F)
  envia (&H43)
  If s = 1 Then
    envia (&H2B)
  Else
    envia (&H2D)
  End If
End Sub
Public Sub m_h(s As Integer)
  envia (&H48)
  envia (&H3F)
  envia (&H48)
  If s = 1 Then
    envia (&H2B)
  Else
    envia (&H2D)
  End If
End Sub
Public Sub m_g(s As Integer)
  envia (&H47)
  envia (&H3F)
  envia (&H47)
  If s = 1 Then
    envia (&H2B)
  Else
    envia (&H2D)
  End If
End Sub
Public Sub m_f(s As Integer)
  envia (&H46)
```

```
    envia (&H3F)
    envia (&H46)
    If s = 1 Then
      envia (&H2B)
    Else
      envia (&H2D)
    End If
  End Sub
  Public Sub m_e(s As Integer)
    envia (&H45)
    envia (&H3F)
    envia (&H45)
    If s = 1 Then
      envia (&H2B)
    Else
      envia (&H2D)
    End If
  End Sub
  Public Sub m_d(s As Integer)
    envia (&H44)
    envia (&H3F)
    envia (&H44)
    If s = 1 Then
      envia (&H2B)
    Else
      envia (&H2D)
    End If
  End Sub


  Public Sub br(var1 As Double)
  If Abs(var1) <= 500 Then
        If var1 < 0 Then
           signo = 2
        ElseIf var1 >= 0 Then
           signo = 1
        End If
        ' Mueve el motor E
        m_b (1)
        envia (&H30)
        envia (&HD)
        m_c (1)
        envia (&H30)
        envia (&HD)
        m_d (1)
        envia (&H30)
        envia (&HD)
        m_e (signo)
```

```
        valor = Abs(var1)
        If valor < 10 Then
          numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
          numero1 = Val(numero1) + 48
          envia (numero1)
        ElseIf valor > 9 And valor < 100 Then
          numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
          numero1 = Val(numero1) + 48
          numero2 = Mid(Str(valor), 3, 1)  ' Devuelve segundo numero.
          numero2 = Val(numero2) + 48
          envia (numero1)
          envia (numero2)
        ElseIf valor > 99 And valor <= 500 Then
          numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
          numero1 = Val(numero1) + 48
          numero2 = Mid(Str(valor), 3, 1)  ' Devuelve segundo numero.
          numero2 = Val(numero2) + 48
          numero3 = Mid(Str(valor), 4, 1)  ' Devuelve tercer numero.
          numero3 = Val(numero3) + 48
          envia (numero1)
          envia (numero2)
          envia (numero3)
        End If
        envia (&HD)
        m_f (1)
        envia (&H30)
        envia (&HD)
        m_g (1)
        envia (&H30)
        envia (&HD)
        m_h (1)
        envia (&H30)
        envia (&HD)
        mbb
        mcc
        mdd
        For i = 0 To 23
          mee
        Next i
        mff
        mgg
        mhh
Else
        MsgBox (" Debe colocar un número Válido (-500 a 500)")
        Text4.Text = " "
End If
End Sub
```

```
Public Sub ba(var1 As Double)
If Abs(var1) <= 500 Then
    If var1 < 0 Then
      signo = 2
    ElseIf var1 >= 0 Then
      signo = 1
    End If
    ' Mueve el motor F var1 pasos
    m_b (1)
    envia (&H30)
    envia (&HD)
    m_c (1)
    envia (&H30)
    envia (&HD)
    m_d (1)
    envia (&H30)
    envia (&HD)
    m_e (1)
    envia (&H30)
    envia (&HD)
    m_f (signo)
valor = Abs(var1)
    If valor < 10 Then
      numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
      numero1 = Val(numero1) + 48
      envia (numero1)
    ElseIf valor > 9 And valor < 100 Then
      numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
      numero1 = Val(numero1) + 48
      numero2 = Mid(Str(valor), 3, 1)  ' Devuelve segundo numero.
      numero2 = Val(numero2) + 48
      envia (numero1)
      envia (numero2)
    ElseIf valor > 99 And valor <= 500 Then
      numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
      numero1 = Val(numero1) + 48
      numero2 = Mid(Str(valor), 3, 1)  ' Devuelve segundo numero.
      numero2 = Val(numero2) + 48
      numero3 = Mid(Str(valor), 4, 1)  ' Devuelve tercer numero.
      numero3 = Val(numero3) + 48
      envia (numero1)
      envia (numero2)
      envia (numero3)
    End If
    envia (&HD)
    m_g (1)
    envia (&H30)
    envia (&HD)
```

```
        m_h (1)
        envia (&H30)
        envia (&HD)
        mbb
        mcc
        mdd
        mee
        For i = 0 To 23
        mff
        Next i
        mgg
        mhh
      Else
        MsgBox (" Debe colocar un número Válido (-500 a 500)")
        Text1.Text = " "
      End If
End Sub
Public Sub compara(cadena As Variant)
    Dim valor As Double
    'si la cadena comienza con * se trata de un comentario
    If Mid(cadena, 1, 1) = "*" Then
        'No hay intrucciones aquí
    Else
      If Mid(StrConv(cadena, 2), 1, 3) = "mu(" Then
        valor = Val(Mid(cadena, 4, 5))
        mu (valor)
      ElseIf Mid(StrConv(cadena, 2), 1, 3) = "br(" Then
        valor = Val(Mid(cadena, 4, 5))
        br (valor)
      ElseIf Mid(StrConv(cadena, 2), 1, 3) = "ba(" Then
        valor = Val(Mid(cadena, 4, 5))
        ba (valor)
      ElseIf Mid(StrConv(cadena, 2), 1, 3) = "co(" Then
        valor = Val(Mid(cadena, 4, 5))
        co (valor)
      ElseIf Mid(StrConv(cadena, 2), 1, 3) = "ho(" Then
        valor = Val(Mid(cadena, 4, 5))
        ho (valor)
      End If
    End If
End Sub
Public Sub mu(var1 As Double)
If Abs(var1) <= 500 Then
      If var1 < 0 Then
        signo = 2
      ElseIf var1 >= 0 Then
        signo = 1
      End If
```

```
        ' Mueve el motor B
        m_b (signo)
valor = Abs(var1)
        If valor < 10 Then
          numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
          numero1 = Val(numero1) + 48
          envia (numero1)
        ElseIf valor > 9 And valor < 100 Then
          numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
          numero1 = Val(numero1) + 48
          numero2 = Mid(Str(valor), 3, 1)  ' Devuelve segundo numero.
          numero2 = Val(numero2) + 48
          envia (numero1)
          envia (numero2)
        ElseIf valor > 99 And valor <= 500 Then
          numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
          numero1 = Val(numero1) + 48
          numero2 = Mid(Str(valor), 3, 1)  ' Devuelve segundo numero.
          numero2 = Val(numero2) + 48
          numero3 = Mid(Str(valor), 4, 1)  ' Devuelve tercer numero.
          numero3 = Val(numero3) + 48
          envia (numero1)
          envia (numero2)
          envia (numero3)
        End If
        envia (&HD)
        m_c (1)
        envia (&H30)
        envia (&HD)
        m_d (1)
        envia (&H30)
        envia (&HD)
        m_e (1)
        envia (&H30)
        envia (&HD)
        m_f (1)
        envia (&H30)
        envia (&HD)
        m_g (1)
        envia (&H30)
        envia (&HD)
        m_h (1)
        envia (&H30)
        envia (&HD)
        For i = 0 To 23
          mbb
        Next i
        mcc
```

```
        mdd
        mee
        mff
        mgg
        mhh
      Else
        MsgBox (" Debe colocar un número Válido (-500 a 500)")
        Text10.Text = " "
      End If
End Sub

Public Sub co(var1 As Double)
If Abs(var1) <= 500 Then
      If var1 < 0 Then
         signo = 2
      ElseIf var1 >= 0 Then
         signo = 1
      End If
      ' Mueve el motor C
      m_b (1)
      envia (&H30)
      envia (&HD)
      m_c (signo)
      valor = Abs(var1)
      If valor < 10 Then
         numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
         numero1 = Val(numero1) + 48
         envia (numero1)
      ElseIf valor > 9 And valor < 100 Then
         numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
         numero1 = Val(numero1) + 48
         numero2 = Mid(Str(valor), 3, 1)  ' Devuelve segundo numero.
         numero2 = Val(numero2) + 48
         envia (numero1)
         envia (numero2)
      ElseIf valor > 99 And valor <= 500 Then
         numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
         numero1 = Val(numero1) + 48
         numero2 = Mid(Str(valor), 3, 1)  ' Devuelve segundo numero.
         numero2 = Val(numero2) + 48
         numero3 = Mid(Str(valor), 4, 1)  ' Devuelve tercer numero.
         numero3 = Val(numero3) + 48
         envia (numero1)
         envia (numero2)
         envia (numero3)
      End If
      envia (&HD)
      m_d (1)
```

```
        envia (&H30)
        envia (&HD)
        m_e (1)
        envia (&H30)
        envia (&HD)
        m_f (1)
        envia (&H30)
        envia (&HD)
        m_g (1)
        envia (&H30)
        envia (&HD)
        m_h (1)
        envia (&H30)
        envia (&HD)
        mbb
        For i = 0 To 23
        mcc
        Next i
        mdd
        mee
        mff
        mgg
        mhh
    Else
        MsgBox (" Debe colocar un número Válido (-500 a 500)")
        Text8.Text = " "
    End If
End Sub

Public Sub ho(var1 As Double)
If Abs(var1) <= 500 Then
        If var1 < 0 Then
            signo = 2
        ElseIf var1 >= 0 Then
            signo = 1
        End If
        ' Mueve el motor D
        m_b (1)
        envia (&H30)
        envia (&HD)
        m_c (1)
        envia (&H30)
        envia (&HD)
        m_d (signo)
valor = Abs(var1)
        If valor < 10 Then
            numero1 = Mid(Str(valor), 2, 1)   ' Devuelve primer numero.
            numero1 = Val(numero1) + 48
```

```
        envia (numero1)
      ElseIf valor > 9 And valor < 100 Then
        numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
        numero1 = Val(numero1) + 48
        numero2 = Mid(Str(valor), 3, 1)  ' Devuelve segundo numero.
        numero2 = Val(numero2) + 48
        envia (numero1)
        envia (numero2)
      ElseIf valor > 99 And valor <= 500 Then
        numero1 = Mid(Str(valor), 2, 1)  ' Devuelve primer numero.
        numero1 = Val(numero1) + 48
        numero2 = Mid(Str(valor), 3, 1)  ' Devuelve segundo numero.
        numero2 = Val(numero2) + 48
        numero3 = Mid(Str(valor), 4, 1)  ' Devuelve tercer numero.
        numero3 = Val(numero3) + 48
        envia (numero1)
        envia (numero2)
        envia (numero3)
      End If
      envia (&HD)
      m_e (1)
      envia (&H30)
      envia (&HD)
      m_f (1)
      envia (&H30)
      envia (&HD)
      m_g (1)
      envia (&H30)
      envia (&HD)
      m_h (1)
      envia (&H30)
      envia (&HD)
      mbb
      mcc
      For i = 0 To 23
        mdd
      Next i
      mee
      mff
      mgg
      mhh
    Else
      MsgBox (" Debe colocar un número Válido (-500 a 500)")
      Text6.Text = " "
    End If
    End Sub

Private Sub acerca_Click()
```

```
  Formacerca.Show
End Sub

Private Sub archivo_Click()
If PresentArchivo.Visible = True Then
  cargar.Enabled = False
Else
   cargar.Enabled = True
   cerrar.Enabled = False
   guardar.Enabled = False
End If
End Sub

Private Sub BorraOrigen_Click()
   Origen.Visible = True
   BorraOrigen.Visible = False
   RegresaOrigen.Visible = False
   Respuesta = MsgBox("Esta usted seguro de borrar el origen", vbYesNo)
   If Respuesta = vbYes Then   ' El usuario eligió el botón Sí.
      ' Ejecuta la acción de borrar.
      MsgBox ("Se eliminará el origen")
   Else    ' El usuario eligió el botón No.
      MsgBox ("No se eliminará el origen, uffff")
   End If
End Sub

Private Sub cargar_Click()
 ' Si ocurre un error ejecutar ManipularErrorAbrir
 On Error GoTo ManipularErrorAbrir
 ' Filtros
 CommonDialog1.Filter = "Archivos Rhino (*.rh)|" & _
                "*.rh|Todos los Archivos (*.*)|*.*"
 ' Filtro por defecto
 CommonDialog1.FilterIndex = 1
 ' Visualizar la caja de diálogo
 CommonDialog1.ShowOpen

 ' CommonDialog1.filetitle contiene el nombre del
 ' archivo elegido. Leer el contenido de este archivo.
 Dim NumArchivo As Integer
 ' Obtener un número libre de Archivo
 NumArchivo = FreeFile
 ' Abrir el Archivo para escribir
 Open CommonDialog1.FileTitle For Input As NumArchivo
 nombreArchivo = CommonDialog1.filename
 indicArchivo = 1
 ' Guardar el texto en el Archivo
 PresentArchivo.Visible = True
```

```
    Trabajar.Visible = True
    cerrar.Enabled = True
    guardar.Enabled = True
    FormRhino!PresentArchivo.Text = Input(LOF(1), #NumArchivo)
    ' Cerrar el Archivo
    Close NumArchivo

SalirAbrir:
    Exit Sub

ManipularErrorAbrir:
    Dim Msg As String
    ' Manipular el error
    If Err.Number = 7 Then
      Msg = "El Archivo es demasiado grande"
    ElseIf Err.Number = 53 Then
      Msg = "El Archivo no existe"
    Else
      Msg = Err.Description
    End If
    MsgBox Msg, vbExclamation, "Editor"
    Close
    Resume SalirAbrir

End Sub

Private Sub cerrar_Click()
    PresentArchivo.Visible = False
    Trabajar.Visible = False
End Sub


Private Sub Command1_Click()
' Mueve el motor F 1 grado positivo
    m_b (1)
    envia (&H30)
    envia (&HD)
    m_c (1)
    envia (&H30)
    envia (&HD)
    m_d (1)
    envia (&H30)
    envia (&HD)
    m_e (1)
    envia (&H30)
    envia (&HD)
    m_f (1)
    envia (&H31)
```

```
    envia (&H30)
    envia (&HD)
    m_g (1)
    envia (&H30)
    envia (&HD)
    m_h (1)
    envia (&H30)
    envia (&HD)
    mbb
    mcc
    mdd
    mee
    For i = 0 To 23
    mff
    Next i
    mgg
    mhh
End Sub


Private Sub Command10_Click()
' Mueve el motor B 1 grado negativo
    m_b (2)
    envia (&H33)
    envia (&HD)
    m_c (1)
    envia (&H30)
    envia (&HD)
    m_d (1)
    envia (&H30)
    envia (&HD)
    m_e (1)
    envia (&H30)
    envia (&HD)
    m_f (1)
    envia (&H30)
    envia (&HD)
    m_g (1)
    envia (&H30)
    envia (&HD)
    m_h (1)
    envia (&H30)
    envia (&HD)
    For i = 0 To 3
       mbb
    Next i
    mcc
    mdd
```

```
    mee
    mff
    mgg
    mhh
End Sub

Private Sub Command2_Click()
' Mueve el motor F 1 grado negativo
    m_b (1)
    envia (&H30)
    envia (&HD)
    m_c (1)
    envia (&H30)
    envia (&HD)
    m_d (1)
    envia (&H30)
    envia (&HD)
    m_e (1)
    envia (&H30)
    envia (&HD)
    m_f (2)
    envia (&H31)
    envia (&H30)
    envia (&HD)
    m_g (1)
    envia (&H30)
    envia (&HD)
    m_h (1)
    envia (&H30)
    envia (&HD)
    mbb
    mcc
    mdd
    mee
    For i = 0 To 23
    mff
    Next i
    mgg
    mhh
End Sub


Private Sub Command3_Click()
' Mueve el motor E 1 grado positivo
    m_b (1)
    envia (&H30)
    envia (&HD)
    m_c (1)
```

```
    envia (&H30)
    envia (&HD)
    m_d (1)
    envia (&H30)
    envia (&HD)
    m_e (1)
    envia (&H31)
    envia (&H30)
    envia (&HD)
    m_f (1)
    envia (&H30)
    envia (&HD)
    m_g (1)
    envia (&H30)
    envia (&HD)
    m_h (1)
    envia (&H30)
    envia (&HD)
    mbb
    mcc
    mdd
    For i = 0 To 12
    mee
    Next i
    mff
    mgg
    mhh
End Sub


Private Sub Command4_Click()
' Mueve el motor E 1 grado negativo
    m_b (1)
    envia (&H30)
    envia (&HD)
    m_c (1)
    envia (&H30)
    envia (&HD)
    m_d (1)
    envia (&H30)
    envia (&HD)
    m_e (2)
    envia (&H31)
    envia (&H30)
    envia (&HD)
    m_f (1)
    envia (&H30)
    envia (&HD)
```

```
    m_g (1)
    envia (&H30)
    envia (&HD)
    m_h (1)
    envia (&H30)
    envia (&HD)
    mbb
    mcc
    mdd
    For i = 0 To 12
    mee
    Next i
    mff
    mgg
    mhh
End Sub

Private Sub Command5_Click()
' Mueve el motor D 1 grado positivo
    m_b (1)
    envia (&H30)
    envia (&HD)
    m_c (1)
    envia (&H30)
    envia (&HD)
    m_d (1)
    envia (&H31)
    envia (&H30)
    envia (&HD)
    m_e (1)
    envia (&H30)
    envia (&HD)
    m_f (1)
    envia (&H30)
    envia (&HD)
    m_g (1)
    envia (&H30)
    envia (&HD)
    m_h (1)
    envia (&H30)
    envia (&HD)
    mbb
    mcc
    For i = 0 To 10
    mdd
    Next i
    mee
    mff
```

```
    mgg
    mhh
End Sub


Private Sub Command6_Click()
' Mueve el motor D 1 grado negativo
    m_b (1)
    envia (&H30)
    envia (&HD)
    m_c (1)
    envia (&H30)
    envia (&HD)
    m_d (2)
    envia (&H31)
    envia (&H30)
    envia (&HD)
    m_e (1)
    envia (&H30)
    envia (&HD)
    m_f (1)
    envia (&H30)
    envia (&HD)
    m_g (1)
    envia (&H30)
    envia (&HD)
    m_h (1)
    envia (&H30)
    envia (&HD)
    mbb
    mcc
    For i = 0 To 21
    mdd
    Next i
    mee
    mff
    mgg
    mhh
End Sub

Private Sub Command7_Click()
' Mueve el motor C 1 grado positivo
    m_b (1)
    envia (&H30)
    envia (&HD)
    m_c (1)
    envia (&H31)
    envia (&H30)
```

```
    envia (&HD)
    m_d (1)
    envia (&H30)
    envia (&HD)
    m_e (1)
    envia (&H30)
    envia (&HD)
    m_f (1)
    envia (&H30)
    envia (&HD)
    m_g (1)
    envia (&H30)
    envia (&HD)
    m_h (1)
    envia (&H30)
    envia (&HD)
      mbb
    For i = 0 To 4
    mcc
    Next i
    mdd
    mee
    mff
    mgg
    mhh
End Sub


Private Sub Command8_Click()
' Mueve el motor C 1 grado negativo
    m_b (1)
    envia (&H30)
    envia (&HD)
    m_c (2)
    envia (&H31)
    envia (&H30)
    envia (&HD)
    m_d (1)
    envia (&H30)
    envia (&HD)
    m_e (1)
    envia (&H30)
    envia (&HD)
    m_f (1)
    envia (&H30)
    envia (&HD)
    m_g (1)
    envia (&H30)
```

```
    envia (&HD)
    m_h (1)
    envia (&H30)
    envia (&HD)
    mbb
    For i = 0 To 13
    mcc
    Next i
    mdd
    mee
    mff
    mgg
    mhh
End Sub

Private Sub Command9_Click()
' Mueve el motor B 1 grado positivo
    m_b (1)
    envia (&H33)
    envia (&HD)
    m_c (1)
    envia (&H30)
    envia (&HD)
    m_d (1)
    envia (&H30)
    envia (&HD)
    m_e (1)
    envia (&H30)
    envia (&HD)
    m_f (1)
    envia (&H30)
    envia (&HD)
    m_g (1)
    envia (&H30)
    envia (&HD)
    m_h (1)
    envia (&H30)
    envia (&HD)
    For i = 0 To 4
       mbb
    Next i
    mcc
    mdd
    mee
    mff
    mgg
    mhh
End Sub
```

```
Private Sub ejemplo_Click()

End Sub

Private Sub Form_Load()
    inicializa
    ini_robot
End Sub



Private Sub MSComm1_Click()

End Sub


Private Sub guardar_Click()
 ' Si ocurre un error ejecutar ManipularErrorGuardar
  On Error GoTo ManipularErrorGuardar
  ' Filtros
  CommonDialog1.Filter = "Archivos Rhino (*.rh)|" & _
                "*.rh|Todos los Archivos (*.*)|*.*"
  ' Filtro por defecto
  'CommonDialog1.FilterIndex = 1
  ' Visualizar la caja de diálogo
  CommonDialog1.ShowSave

  ' CommonDialog1.FileTitle contiene el nombre del
  ' Archivo elegido. Escribir el texto en este Archivo.
  Dim NumArchivo As Integer
  ' Obtener un número libre de Archivo
  NumArchivo = FreeFile
  ' Abrir el Archivo para escribir
  Open CommonDialog1.FileTitle For Output As NumArchivo
  ' Guardar el texto en el Archivo
  Print #NumArchivo, FormRhino!PresentArchivo.Text
  ' Cerrar el Archivo
  Close NumArchivo

SalirGuardar:
  Exit Sub

ManipularErrorGuardar:
  ' Manipular el error
  MsgBox Err.Description
  Resume SalirGuardar
```

```
End Sub

Private Sub Image1_Click()

End Sub

Private Sub MueRobot1_Click()
   If IsNumeric(Text1.Text) Then
      var1 = CDbl(Text1.Text)
      Text15.Text = var1 / 10
   ' Manda a la funcion para mover var1 pasos
      ba (var1)
   Else
      Text15.Text = " "
      MsgBox (" Sólo deben ser Números")
      Text1.Text = " "
   End If
End Sub

Private Sub MueRobot2_Click()
   If IsNumeric(Text4.Text) Then
      var1 = CDbl(Text4.Text)
      Text16.Text = var1 / 10
    'mueve E var1 pasos
      br (var1)
   Else
      Text16.Text = " "
      MsgBox (" Sólo deben ser Números")
      Text4.Text = " "
   End If
End Sub

Private Sub MueRobot3_Click()
   If IsNumeric(Text6.Text) Then
      var1 = CDbl(Text6.Text)
      Text17.Text = var1 / 10
   'mueve D var1 pasos
      ho (var1)
   Else
      Text17.Text = " "
      MsgBox (" Sólo deben ser Números")
      Text6.Text = " "
   End If
End Sub

Private Sub MueRobot4_Click()
   If IsNumeric(Text8.Text) Then
```

```
            var1 = CDbl(Text8.Text)
            Text18.Text = var1 / 10
          'mueve el motor C var1 pulsos
            co (var1)
          Else
            Text18.Text = " "
            MsgBox (" Sólo deben ser Números")
            Text8.Text = " "
          End If
        End Sub

        Private Sub MueRobot5_Click()
          If IsNumeric(Text10.Text) Then
            var1 = CDbl(Text10.Text)
            Text19.Text = var1 / 10
            ' mueve el motor B var1 pasos
            mu (var1)
          Else
            Text19.Text = " "
            MsgBox (" Sólo deben ser Números")
            Text10.Text = " "
          End If
        End Sub

Private Sub MueUnoAOtro_Click()
        Dim q1, q2, q3, q4, q5, q6, tq1, tq2, t4, t5, t1, factor
        factor = 0.015707963
        If IsNumeric(Text11.Text) Then
          If IsNumeric(Text12.Text) Then
            If IsNumeric(Text13.Text) Then
              Px = CSng(Text11.Text)
              Py = CSng(Text12.Text)
              Pz = CSng(Text13.Text)
              l1 = 26
              l2 = 23
              l3 = 23
              l4 = 9
              tq1 = Atn(Py / Px)
              q1 = tq1 / factor
              t1 = (Px ^ 2 + Py ^ 2 + Pz ^ 2 - l2 ^ 2 - l3 ^ 2) / (2 * l2 * l3)
              If Abs(t1) > 1 Then
                tq3 = Atn((Sqr(t1 ^ 2 - 1)) / t1)
              Else
                tq3 = Atn((Sqr(1 - t1 ^ 2)) / t1)
              End If
              t4 = (l4 - Px) / Px
              t5 = Atn(Py / Px)
              q4 = Atn(t4) / factor
```

```
            q5 = t5 / factor
            q3 = tq3 / factor
            tq2 = Atn(Pz / Sqr(Px ^ 2 + Py ^ 2)) - Atn((l3 * Sin(q3)) / (l2 + l3 *
        Cos(q3)))
            q2 = tq2 / factor
            'Nota: q1 es para la base, q2 para el brazo, q3 para el hombro, q4 para
        codo y q5 muñeca
            ba (q1)
            br (q2)
            ho (q3)
            co (q4)
            mu (q5)
            MsgBox ("El robot se movio en base, brazo, hombro de la forma
        siguiente")
            Text15.Text = q1
            Text16.Text = q2
            Text17.Text = q3
            Text18.Text = q4
            Text19.Text = q5
          Else
          MsgBox ("En Z son valores numericos")
          End If
        Else
        MsgBox ("En Y son valores numericos")
        End If
      Else
      MsgBox ("En X son valores numericos")
      End If
End Sub

    Private Sub Nuevo_Click()
      PresentArchivo.Visible = True
      Trabajar.Visible = True
      cerrar.Enabled = True
      guardar.Enabled = True
      cargar.Enabled = False
      PresentArchivo.Text = " "
    End Sub

    Private Sub salir_Click()
      MSComm1.PortOpen = False
      Unload Me
      End
    End Sub


    Private Sub SSCommand1_Click()
      If IsNumeric(Text1.Text) Then
```

```
      var1 = CDbl(Text1.Text)
      Text15.Text = var1 / 10
    ' Manda a la funcion para mover var1 pasos
      ba (var1)
    Else
      Text15.Text = " "
      MsgBox (" Sólo deben ser Números")
      Text1.Text = " "
    End If
End Sub

Private Sub SSCommand2_Click()
    If IsNumeric(Text4.Text) Then
      var1 = CDbl(Text4.Text)
      Text16.Text = var1 / 10
    'mueve E var1 pasos
      br (var1)
    Else
      Text16.Text = " "
      MsgBox (" Sólo deben ser Números")
      Text4.Text = " "
    End If

End Sub

Private Sub SSCommand3_Click()
    If IsNumeric(Text6.Text) Then
      var1 = CDbl(Text6.Text)
      Text17.Text = var1 / 10
    'mueve D var1 pasos
      ho (var1)
    Else
      Text17.Text = " "
      MsgBox (" Sólo deben ser Números")
      Text6.Text = " "
    End If
End Sub

Private Sub SSCommand4_Click()
    If IsNumeric(Text8.Text) Then
      var1 = CDbl(Text8.Text)
      Text18.Text = var1 / 10
    'mueve el motor C var1 pulsos
      co (var1)
    Else
      Text18.Text = " "
      MsgBox (" Sólo deben ser Números")
      Text8.Text = " "
```

```
    End If
End Sub

Private Sub SSCommand5_Click()
    If IsNumeric(Text10.Text) Then
        var1 = CDbl(Text10.Text)
        Text19.Text = var1 / 10
        ' mueve el motor B var1 pasos
        mu (var1)
    Else
        Text19.Text = " "
        MsgBox (" Sólo deben ser Números")
        Text10.Text = " "
    End If
End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
  If IsNumeric(Text1.Text) Then
        var1 = CDbl(Text1.Text)
        Text15.Text = var1 / 10
        ' Manda a la funcion para mover var1 pasos
        ba (var1)
    Else
        Text15.Text = " "
        MsgBox (" Sólo deben ser Números")
        Text1.Text = " "
    End If
End If
End Sub


Private Sub Text10_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
  If IsNumeric(Text10.Text) Then
    var1 = CDbl(Text10.Text)
    Text19.Text = var1 / 10
  mueve el motor B var1 pasos
    mu (var1)
    Else
        Text19.Text = " "
        MsgBox (" Sólo deben ser Números")
        Text10.Text = " "
    End If
End If
End Sub
```

```
Private Sub Text4_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
 If IsNumeric(Text4.Text) Then
   var1 = CDbl(Text4.Text)
   Text16.Text = var1 / 10
 'mueve E var1 pasos
   br (var1)
   Else
      Text16.Text = " "
      MsgBox (" Sólo deben ser Números")
      Text4.Text = " "
   End If
End If
End Sub


Private Sub Text6_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
 If IsNumeric(Text6.Text) Then
   var1 = CDbl(Text6.Text)
   Text17.Text = var1 / 10
'mueve D var1 pasos
   ho (var1)
   Else
      Text17.Text = " "
      MsgBox (" Sólo deben ser Números")
      Text6.Text = " "
   End If
End If
End Sub


Private Sub Text8_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
 If IsNumeric(Text8.Text) Then
   var1 = CDbl(Text8.Text)
   Text18.Text = var1 / 10
 'mueve el motor C var1 pasos
   co (var1)
   Else
      Text18.Text = " "
      MsgBox (" Sólo deben ser Números")
      Text8.Text = " "
   End If
End If
End Sub
```

```
Private Sub Trabajar_Click()
  Dim c As Variant
  ' Si ocurre un error ejecutar ManipularErrorAbrir
  On Error GoTo ManipularErrorTrabajar

   'Abre el archivo y lee una linea a la vez
  Open nombreArchivo For Input As #1
  While Not EOF(1)
    Line Input #1, c
    compara (c)
  Wend

SalirTrabajar:
  MsgBox ("Se ha terminado de trabajar con el archivo")
  Close #1
  Exit Sub

ManipularErrorTrabajar:
  Dim Msg As String
  ' Manipular el error
  If Err.Number = 7 Then
    Msg = "El Archivo es demasiado grande"
  ElseIf Err.Number = 53 Then
    Msg = "El Archivo no existe"
  Else
    Msg = Err.Description
  End If
  MsgBox Msg, vbExclamation, "Editor"
  Close
  Resume SalirTrabajar

End Sub

Private Sub usoSoftware_Click()
    CommonDialog1.HelpFile = "robot.hlp"
    CommonDialog1.HelpCommand = cdlHelpContents
    CommonDialog1.ShowHelp
End Sub
```

Del Modulo en Basic

```
Public puerto As Integer
```

BIBLIOGRAFÍA

- [1,3,6]Data Communications For Microcomputers with practical applications and experiments
  Elizabeth A. Nichols, Joseph C. Nichols
  Pretince – Hall

- [2,4,5,7]Data Communications, Computer Networks and Open System
  Fred Halsall
  Addison Wesley

- [8,9,10,11,12,14,15,16,22]Control Robótico
  P.M. Taylor
  Ediciones CEAC, España

- [13,17,18,19,20,22]Robotica: Control, Detección, Visión e inteligencia
  K.S. Fu, R.C. González, C.S.G. Lee
  Mc Graw-Hill.

- [22]Curso de Robótica
  Ángulo Usategui José Ma. y Avilés González Rafael
  Paraninfo, Madrid España. 1989.

- [16,18,19,21,22]Robótica Practica Tecnología y Aplicaciones
  Ángulo Usategui José María.
  Paraninfo, Madrid España. 1986 2da. Edición.