Ej. 1

UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERIA Y ARQUITECTURA
ESCUELA DE INGENIERIA ELECTRICA

"DISEÑO Y CONSTRUCCION DE UN BRAZO MECANICO (ROBOT) A CONTROL REMOTO MANEJADO POR UNA COMPUTADORA PERSONAL"

TRABAJO DE GRADUACION PRESENTADO POR:

RICARDO ERNESTO CIENFUEGOS ROSALES
MORRIS WILLIAM DIAZ SARAVIA

PARA OPTAR AL TITULO DE

INGENIERO ELECTRICISTA

AGOSTO DE 1994.

SAN SALVADOR, EL SALVADOR, CENTROAMERICA

UNIVERSIDAD DE EL SALVADOR

RECTOR:

DR. FABIO CASTILLO FIGUEROA

SECRETARIO GENERAL:

LIC. MIRNA ANTONIETA PERLA DE ANAYA

FACULTAD DE INGENIERIA Y ARQUITECTURA

DECANO:

ING. JOAQUIN VANEGAS AGUILAR

SECRETARIO:

ING. JOSE RIGOBERTO MURILLO CAMPOS

ESCUELA DE INGENIERIA ELECTRICA

DIRECTOR:

ING. SALVADOR DE JESUS GERMAN

UNIVERSIDAD DE EL SALVADOR
FACULTAD DE INGENIERIA Y ARQUITECTURA
ESCUELA DE INGENIERIA ELECTRICA

TRABAJO DE GRADUACION PREVIO A LA OPCION AL GRADO DE:

INGENIERO ELECTRICISTA

TITULO:

"DISEÑO Y CONSTRUCCION DE UN BRAZO MECANICO (ROBOT) A CONTROL REMOTO MANEJADO POR UNA COMPUTADORA PERSONAL"

PRESENTADO POR:

RICARDO ERNESTO CIENFUEGOS ROSALES
MORRIS WILLIAM DIAZ SARAVIA

COORDINADOR:

ING. RICARDO ERNESTO CORTEZ

ASESORES:

ING. LUIS RAMON PORTILLO

ING. JOSE FRANCISCO ZULETA

TRABAJO DEDICADO A:

MIS PADRES: Gilberto Cienfuegos y Martha Rosales, que me guiaron, que me apoyaron en todo momento para lograr el objetivo de alcanzar la meta de titularme; y que me decían a su manera, **"Los premios de la vida se encuentran al fin de cada jornada, y no cerca del comienzo, y no nos corresponde el saber cuantos sacrificios son necesarios a fin de alcanzar la meta".**

MIS HERMANOS: José Roberto Y Carlos Enrique que me alentaron en esos momentos de flaqueza.

MI ESPOSA: Milagrito Villatoro, una bella dama, quien me enseño el verdadero significado de la paciencia y la fortaleza.

MIS COMPAÑEROS DE TRABAJO: Que brindaron sugerencias, a veces buenas y otras un tanto descabelladas..., pero todas con la mejor intención de ayudar.

MIS COMPAÑEROS DE ESTUDIO: que colaboraron con ideas y con su tiempo desinteresadamente, para el desarrollo de este trabajo.

MIS AMIGOS : que no acabaría de enumerar...

Y ante todo, a ese misterioso "SER SUPREMO", amigo, maestro, creador de todas las cosas, y hacedor de grandes cambios, le doy las gracias por la vida, que me ha dado tanto...

A ELLOS DEDICO ESTE TRIUNFO

Ricardo Ernesto Cienfuegos Rosales

TRABAJO DEDICADO A:

DIOS TODOPODEROSO: Por iluminar y guiar mi vida.

MIS PADRES: José Efraín Díaz y María Gloria Saravia, por brindarme siempre su apoyo y comprensión. Por el esfuerzo de ellos culmino hoy esta carrera.

MI NOVIA: Delmy Sánchez, por su comprensión y apoyo sin límites, aún en los momentos más difíciles.

MIS COMPAÑEROS DE TRABAJO: que nos ayudaron a franquear las diversas dificultades que aparecieron en el camino.

MIS COMPAÑEROS DE ESTUDIO: que colaboraron con ideas y con su tiempo desinteresadamente, para el desarrollo de este trabajo.

MI COMPAÑERO DE TRABAJO Y AMIGO: Marvin Hernández, que brindó su colaboración desinteresada en el desarrollo de este trabajo.

A ELLOS DEDICO ESTE TRIUNFO

Morris William Díaz Saravia

ACTA DE CONSTANCIA DE NOTA Y DEFENSA FINAL

En esta fecha, 28 de Julio de 199 4,
en el local de Sala de Lectura de la E.I.E.
a las 16:00 horas, con la presencia de las siguientes autoridades de la Escuela de Ingeniería Eléctrica de la Universidad de El Salvador:

1- Ing. Salvador de J. German
Director E.I.E.

2- _____

3- _____

Firma

ESCUELA DE INGENIERIA ELECTRICA
FACULTAD DE INGENIERIA
Y ARQUITECTURA
Universidad de El Salvador

Y con el Honorable Jurado de evaluación integrado por las personas siguientes:

1- Ing. René Naúm Clímaco Cortez

2- Ing. Carlos Fernando Ascencio

3- Ing. José Roberto Ramos

4- Ing. Ramón Portillo

5- Ing. Francisco Zuleta

6- _____

Se efectuó la defensa final reglamentaria del Trabajo de Graduación: "DISEÑO Y CONSTRUCCION DE UN BRAZO MECANICO (ROBOT) A CONTROL REMOTO MANEJADO POR UNA COMPUTADORA PERSONAL"

a cargo del (los) Br(es): CIENFUEGOS ROSALES, RICARDO ERNESTO y
DIAZ SARAVIA, MORRIS WILLIAM

Habiendo obtenido el presente trabajo una nota final, global de 8.5
(Ocho Punto Cinco)

PREFACIO

El presente trabajo se justifica en base a las necesidad existente de incursionar en el área de la robótica y el tratar de crear una herramienta que pueda ser usada en un ambiente industrial.

El objetivo fundamental de este trabajo es el diseñar y construir un brazo mecánico (robot) controlado por computador, capaz de realizar movimientos en modo manual y automático. Para ello, en un inicio se plantearon los siguientes objetivos:

1- Diseñar y construir un brazo robot con capacidad de movimiento, de sensor (recibir y emitir señales) y de agarre (tomar y dejar objetos).

2- Diseñar y construir en circuito impreso: circuito de comunicación entre computador PC AT/XT y robot, circuito manejador de los motores y el circuito de los sensores.

3- Desarrollar un software entre usuario y robot, con facilidades para la programación de aplicaciones.

Estos objetivos han sido cumplidos satisfactoriamente.

El primer objetivo se cumple, ya que se deja el brazo mecánico armado y funcionando.

El segundo objetivo de construir las interfases de comunicación, controlador de motores y de sensoreo, fue logrado satisfactoriamente. Las tres interfases están en circuito impreso y son controladas desde el computador PC AT/XT, para manipular el brazo robot.

El tercer objetivo que hace referencia al diseño del programa usuario-robot, es desarrollado en PASCAL (Turbo Pascal) y Ensamblador, este programa queda en disco, tanto el código fuente como los programas ejecutables.

Debido a la facilidad de operación del brazo robótico desde el teclado del computador PC AT/XT, al costo y a la importancia del sistema, creemos que el brazo representa una alternativa real para nuestro medio.

Es necesario el apoyar e incentivar cualquier proyecto o trabajo de graduación en el área de robótica, en vista de la enorme cantidad de conocimiento involucrado y del desarrollo de la tecnología salvadoreña.

RESUMEN DEL TRABAJO

Si asociamos la capacidad de control y procesamiento de las computadoras a la capacidad de movimiento de dispositivos electromecánicos, la robótica se presenta hoy como un ramo de realizaciones sofisticadas con posibilidades y desafíos constantes que retan la imaginación del humano en la necesidad de satisfacer soluciones a algunas tareas peligrosas de la industria.

La robótica es un campo nuevo, no desarrollado en El Salvador, por ser una actividad compleja, multidisciplinaria y de costo excesivo, siendo viable solamente en aquellas industrias con una infraestructura muy grande.

Conocimientos de electrónica, informática, mecánica y control automático constituyen exigencia principal para el desarrollo de un sistema robótico. El diseñador y constructor deben interactuar en estas áreas y manejarlas a fondo, para llevar a cabo un buen trabajo.

El propósito en este documento es dejar sentada las bases para el diseño y construcción de un brazo robot que sirva de prototipo a sistemas más grandes. Si bien la capacidad de carga será baja, el sistema contendrá todos los elementos de un sistema complejo: programación en línea, programación fuera de línea, ordenes a control remoto, movimiento que considera sus características dinámicas, etc.

Por lo tanto, el brazo a construirse tendrá la mayoría de cualidades que de un brazo robótico se desean, siendo un dispositivo de consulta y a su vez, el punto de partida para otros proyectos con aplicaciones específicas más complejas en cualquier área de la industria.

En el capítulo I, se da una introducción teórica al campo de la robótica, se plantea un poco de historia sobre la robótica, luego se define robot e inteligencia artificial. Este capítulo se termina definiendo la estructura de un brazo robot, y clasificando los diferentes robots existentes.

En el capitulo II, se definen los requerimiento que debe cumplir el prototipo del brazo robot; se definen ciertos objetivos que satisfacen dichos requerimientos y por último se describen las características globales del prototipo a construirse.

En el capitulo III se plantea el diseño mecánico. El proceso de diseño es el siguiente: se plantean las diferentes alternativas para cada elemento del brazo robot, se toman las

consideraciones y criterios necesarios y luego se selecciona la mejor alternativa de acuerdo a las condiciones especificadas. Además, en este capítulo se dan las dimensiones reales del brazo robot, como lo son las dimensiones de los engranajes, poleas, y de todos los elementos físicos del brazo robot.

En el capitulo IV se desarrolla el sistema eléctrico, se definen los motores a utilizar, el sistema de control, los sensores y la interfase del computador hacia el brazo robot.

En el capitulo V se definen los conceptos de programación y lenguaje de robot, así como las diferentes instrucciones o comandos de un programa de robot, también se desarrolla los diferentes movimientos que puede tener un brazo robot.

En el capitulo VI se diseña el software del brazo. Se describe el sistema operativo diseñado especialmente para el brazo; además, se dan diferentes algoritmos de las instrucciones, del programa monitor, traductor y de las ordenes del lenguaje de robot.

Finalmente, el documento, contiene una sección de anexos en el cual se han reunido el cálculo del peso del brazo robot, las hojas de datos del IC SAA1027, el circuito completo del sistema, comandos básicos del Sidekick y el plano mecánico de la estructura del brazo.

TABLA DE CONTENIDOS

LISTA DE TABLAS

LISTA DE FIGURAS

1.1 MARCO HISTORICO

Desde un punto de vista histórico, los orígenes del robot industrial se pueden vincular al desarrollo continuado de maquinaria automatizada, que se inició con la Revolución Industrial en la década de 1760. En la primera oleada de la mecanización, la atención estaba puesta en la manufactura de piezas con la inteligencia y el control del ser humano.

A medida que el desarrollo industrial prosiguió en el siglo XX, el objetivo de la automatización cambió hacia la tecnología de manufactura guiada por la inteligencia humana con control numérico.

La palabra "robot" (del checo robota: trabajo) fue acuñada en 1920 por el escritor checo Karel Capek en su obra teatral R. U. R. (Rossum's universal robots: Los robots universales de Rossum) para denominar a un androide creado por un científico y capaz de llevar a cabo trabajos realizados tradicionalmente por un hombre. El término fue acogido con gran entusiasmo por los escritores de ciencia ficción. A pesar de los numerosos relatos novelescos que narran los poderes de los robots, éstos no son más que una extensión electromecánica del ordenador, con todas las limitaciones e imperfecciones propias de estas máquinas.

Sus orígenes se remontan a los talleres de maquinaria de los años cincuenta, en los que se aplicó por primera vez la teoría del control numérico a las máquinas-herramienta. Estos primeros esfuerzos fueron, previsiblemente, muy elementales: máquinas controladas por cinta de papel de cinco agujeros (del tipo de las que utilizan las máquinas de télex) que, en el mejor de los casos, sólo podían mover una herramienta fija de un punto a otro alrededor del objeto sobre el cual estaban trabajando.

El siguiente paso de su desarrollo fue conseguir que pudieran cambiar de herramienta en el transcurso de la faena. Esto se logró mediante la utilización de un "carrusel" o soporte de herramientas rotatorio; todas ellas tenían fijaciones idénticas, que se podían seleccionar y ajustar al soporte de herramientas bajo el control del programa.

Incluso con esta refinación, una máquina determinada sólo era capaz de realizar un único tipo de tarea: un torno seguía siendo un torno, aun cuando pudiera hacer todos los trabajos de tornería requeridos para un proceso determinado. Por esa misma época se estaban desarrollando manos y brazos accionados por control remoto para trabajar en medios peligrosos: bajo el océano, por ejemplo, o en laboratorios donde se manipulaban elementos radiactivos. Estos dispositivos manipuladores eran meras extensiones de las manos del operario, pero enseguida se comenzaron a utilizar ordenadores para controlarlos

directamente. Los robots que se han desarrollado después son, aplicando un término más preciso, "brazos robot", pues consisten en un soporte de herramientas montado sobre un brazo extensible o articulado.

Realidad nacional

Actualmente es necesario incursionar en el campo de la robótica, para mejorar o permitir que la industria salvadoreña modernice sus sistemas productivos.

Es primordial y menester que la Universidad de El Salvador (U.E.S.) y la Escuela de Ingeniería Eléctrica en particular. se involucre en la búsqueda de alternativas de solución para la industria salvadoreña. Hoy en día es de vital importancia involucrarse en el campo de la robótica para facilitar la adquisición de dicha tecnología, a un costo más accesible para el industrial salvadoreño.

Teñiendo en cuenta la desventaja de lo oneroso que sale el adquirir está clase de tecnología en el extranjero, se observa la necesidad de plantear, una salida viable que facilite la obtención de estos autómatas.

Para ello, se plantea desarrollar un brazo robot prototipo que sea capaz de recibir ordenes desde una P.C., o compatible, esto quiere decir que debe interpretar esas ordenes de acuerdo a la tarea industrial a resolver en un ambiente industrial.

La solución que se ofrece para el industrial es un brazo mecánico controlado desde una P.C..

1.2 LOS ROBOTS Y LA INTELIGENCIA ARTIFICIAL.

La palabra robot, en la mayoría de las personas, produce una asociación en sus mentes con la imagen de una máquina con forma humana, con cabeza y extremidades, dispuestos a servir al hombre. Esta asociación es el resultado de la influencia del género de ciencia ficción, que ya sea en películas o literatura, muestran al robot en forma humana, y con capacidades que generalmente son pura ficción.

Definición de Robot

Una definición aceptable es la utilizada por la RIA (Robotics Industries Association), la cual precisa que "un robot es un manipulador o unidad reprogramable multifuncional diseñada para mover materiales, piezas, herramientas o dispositivos especializados, a través de movimientos variables programables para la realización de diferentes trabajos." Sintetizando un poco la definición de la RIA, podemos afirmar que un robot es un manipulador reprogramable de uso general con sensores

externos que puede efectuar diferentes tareas de montaje.

En la actualidad, limitado por los avances tecnológicos, la finalidad de la construcción de robots se ha orientado a su intervención en los procesos de fabricación.

Estos robots, que no tienen forma humana en absoluto, son los encargados de realizar trabajos repetitivos en las cadenas de proceso de fabricación, como por ejemplo: pintar al spray, moldear a inyección, montar carrocerías de automóviles, etc.

En un futuro, cuando los avances tecnológicos hayan superado muchas limitantes actuales, los robots ya no solo aportarán fuerza o habilidad, sino que también proveerán al hombre de un conocimiento artificial del mundo, es decir se creará el robot "inteligente".

Inteligencia Artificial.

La Inteligencia Artificial consiste en la asimilación de los procesos inductivos y deductivos del cerebro humano. Este intento de imitación se enfrenta en la actualidad a duras restricciones físicas: la complejidad neurológica del cerebro está muy lejos de ser igualada, así como de ser entendidos completamente los complejos procesos que en él se dan.

La inteligencia artificial acepta el reto de la imitación de los procesos del cerebro aplicando mucho ingenio para aprovechar los medios de que se dispone y que se elaboran.

La Inteligencia Artificial se sustenta sobre dos elementos: el primero es el de las estrategias de comportamiento inteligente; se conjuga en la disposición de reglas para formular buenas inferencias o conjeturas y, también, en su uso para la búsqueda de una solución a la cuestión o tarea planteada.

El segundo elemento es el saber, y el cual varía en cada caso específico.

La inteligencia artificial no opera sobre realidades concretas, y recoge en su seno varios aspectos fundamentales:

- Sistemas Expertos;
- Robots;
- Procesamiento de lenguaje natural;
- Modelos de conocimiento y
- Visión Artificial.

Cada uno de los campos señalados posee sus propios objetivos, y requiere de realización especializada, pero es común a todos ellos el objetivo general de la Inteligencia Artificial.

El desarrollo de la Inteligencia Artificial exige de crear computadores, que se clasifican en lo que se ha denominado de quinta generación, los cuales están asociados a un salto cuantitativo, pero sobre todo, cualitativo del tratamiento de la información.

A diferencia de las anteriores generaciones, cuya tarea se reduce al tratamiento de datos, con la quinta generación se pretende alcanzar un grado superior.

El grado superior consiste en adquirir información y, a partir de los materiales y estructuras de que se dispone, elaborar conocimiento, es decir, el despliegue de un tratamiento inteligente.

Brazo Robot

Consiste en un ingenio que reproduce mecánicamente el brazo y la mano humana, realizando operaciones muy precisas y a veces complejas.

Encaja con la definición de unidad reprogramable multifuncional, y por lo tanto es equivalente a decir robot. También es conocido como robot industrial, ya que su mayor aplicación es en ésta.

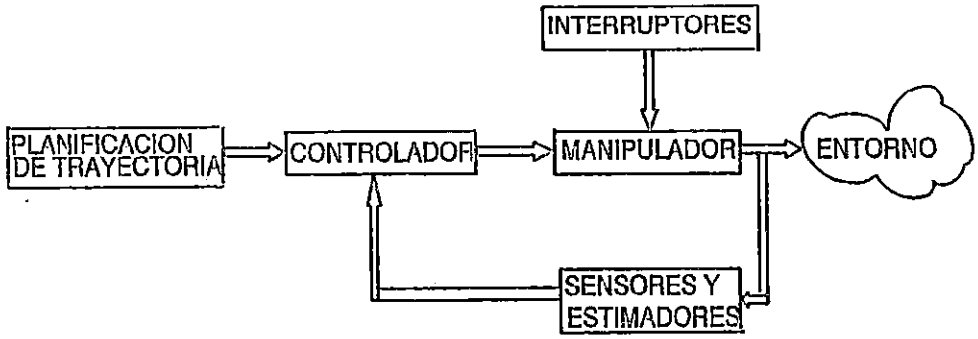


Figura 1.1 Configuración básica de un robot.

1.3 ESTRUCTURA DE UN ROBOT

Es un error común el considerar que un robot no es más que un brazo mecánico articulado.

El error radica en que éste es solamente un componente de lo que técnicamente se considera un sistema robótico.

En la figura 1.1 puede observarse la configuración básica de un robot.

Un sistema robótico consta de las siguientes partes:

1. Manipulador o brazo mecánico.
2. Controladores.
3. Elementos motrices o actuadores.
4. Elementos terminales: Herramientas o aprehensores.
5. Sensores de información en los robots inteligentes.

1.4 EL MANIPULADOR

La mayoría de los brazos robots utilizados en las fábricas actuales están montados sobre una base que está sujeta al suelo. El cuerpo está unido a la base por el hombro y el conjunto del brazo está unido al cuerpo por el codo. Al final del brazo esta la muñeca. La muñeca está constituida por varios componentes que le permiten orientarse en varias posiciones. Los movimientos relativos entre los diversos componentes del cuerpo, brazo y muñeca son proporcionados por una serie de articulaciones. Estos movimientos de las articulaciones suelen implicar deslizamientos o giros. El conjunto de la muñeca, el brazo y el cuerpo se denomina, a veces, el manipulador -brazo-.

Los brazos robots pueden agruparse conforme a su configuración mecánica. En la Figura 1.2 se muestran cuatro configuraciones básicas de brazos robots.

1.4.1 TIPOS DE MANIPULADORES

Cada uno de estos manipuladores presentan ventajas y desventajas, ya sea a nivel de programación, estabilidad mecánica, robustez, alcance de más puntos de trabajo, etc.

a. Manipulador **polar o esférico** que gira en torno a un eje vertical (lineal), a un ángulo de elevación (radialmente) y a un eje horizontal (alcance).
   Ventaja: Un eje lineal, dos ejes rotacionales, gran alcance horizontal.
   Desventaja: No puede alcanzar alrededor de obstáculos. Tiene corto alcance vertical.

b. Manipulador **cilíndrico** que gira en torno a un eje vertical (base), tiene movimiento de elevación (altura), tiene un eje horizontal (alcance).
   Ventaja: Tiene dos ejes lineales, y un eje rotacional, puede alcanzar todo su entorno.
   Los ejes de elevación y alcance son rígidos.
   Desventaja: No puede alcanzar por arriba de sí mismo.
   El eje rotacional de la base es menos rígido que un eje lineal. No puede alcanzar alrededor de obstáculos. El movimiento es circular.

c. Manipulador **cartesiano o x-y-z** (tres ejes lineales),

desplazamiento de la base, alcance y altura.

Ventaja: Tres ejes lineales, fácil de visualizar, estructura rígida, fácil de programar off-line sus ejes son lineales.

Desventaja: Solo puede alcanzar enfrente de sí mismo. Requiere de un gran espacio para su área de trabajo.

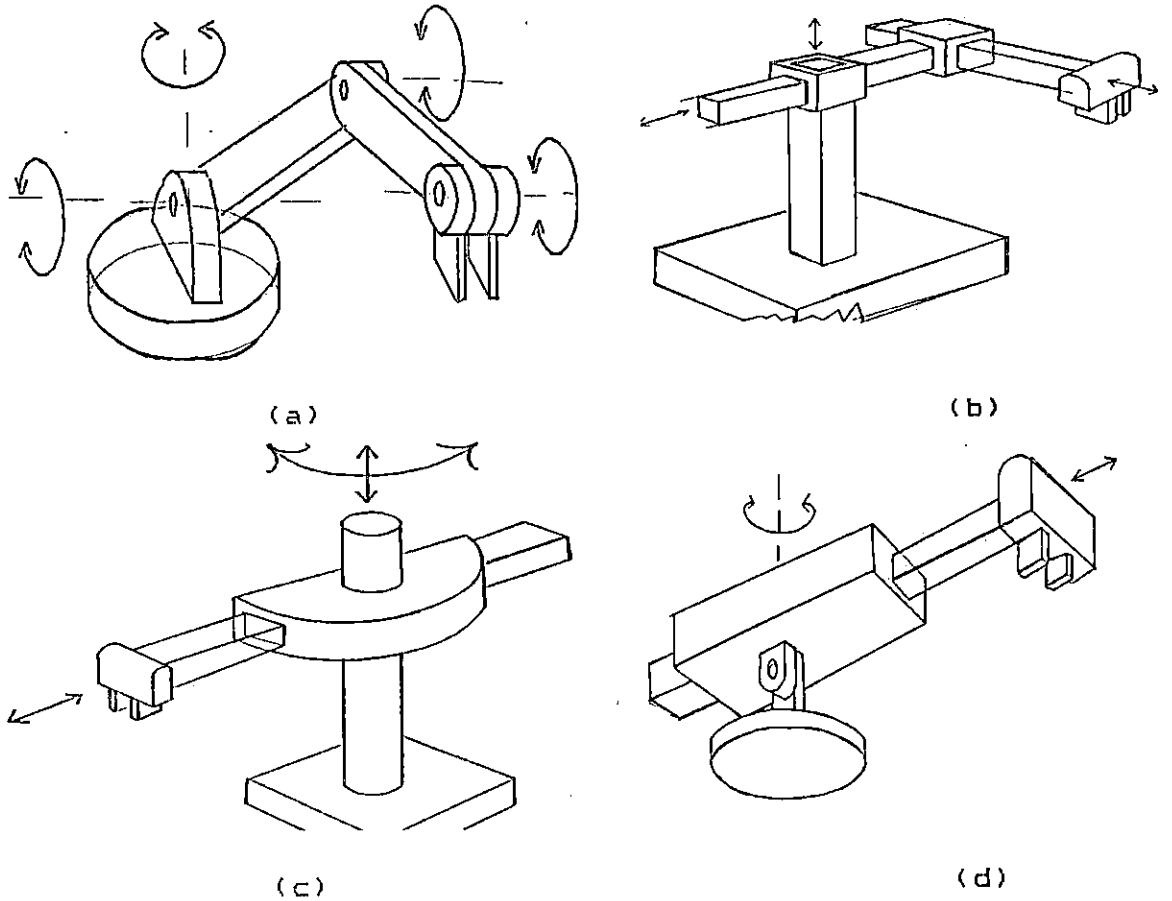


Figura 1.2: Configuraciones de manipuladores: a) brazo articulado, b) brazo cartesiano ; c) brazo cilíndrico; d) esférico

d. Manipulador **articulado** que gira en torno a un eje vertical (base), y tres ejes horizontales (ángulo de elevación y ángulo de alcance).

Ventaja: Cuatro ejes rotacionales. Puede alcanzar por arriba o por bajo de obstáculos. Tiene la mayor área de trabajo en el menor espacio ocupado.

Desventaja: Es difícil de programar off-line. Dos o cuatro formas de alcanzar un punto. Es el más complejo manipulador.

1.5 EL CONTROLADOR.

Es el dispositivo que se encarga de regular el movimiento de los elementos del manipulador y todo tipo de acciones, cálculos y procesamiento de información que se realiza.

El controlador de la mayoría de robots industriales es una microcomputadora. La habilidad de un robot para ser reprogramado para distintas tareas le confiere flexibilidad y lo separa de otras formas de automatización en la fábrica.

Los elementos básicos de un controlador incluyen:

1. Alguna provisión para entradas a partir de los sensores,

2. Entradas para las instrucciones programadas que el robot debe llevar a cabo y;

3. Salidas para las señales destinadas a manejar los actuadores y efectores.

Dos formas de memoria deben estar disponibles; una para almacenar señales de control hasta que éstas puedan ser analizadas, y otra para almacenar los programas de control.

La complejidad de un controlador varía según los parámetros que se gobiernen, pudiendo existir las siguientes categorías:

a) Controlador de posición
Solo intervienen en el control de la posición del elemento terminal.
Puede actuar en modo punto a punto (PIP), o bien, en modo continuo, en cuyo caso recibe el nombre de control continuo de trayectoria (CP).

b) Controlador dinámico.
Se tienen en cuenta, las propiedades dinámicas del manipulador, motores y elementos asociados.

c) Controlador adaptativo.
Además de lo indicado en los anteriores controles, también se considera la variación de las características del manipulador al variar la posición.

Refiriéndose a otro aspecto, el control puede llevarse a cabo en ciclo abierto o en ciclo cerrado.

En el caso del control en ciclo abierto, se produce una señal que determina el movimiento, pero no se analiza si se ha realizado con exactitud o se ha producido un error. El control en ciclo abierto tiene muchas causas de error (inercia, interferencias, fricciones, desplazamientos, etc.)

y si bien es muy simple y económico, no se admite en las aplicaciones industriales, donde es fundamental la exactitud en la repetibilidad de los movimientos. Sin embargo, en robots dedicados a la enseñanza y el entrenamiento, este tipo de control está muy extendido.

La mayoría de los sistemas de robots industriales poseen un control en ciclo cerrado, con realimentación. Este control hace uso de un transductor o sensor de la posición real de la articulación o del elemento terminal, cuya información se compara con el valor de la señal de mando, que indica la posición actual deseada. El error entre estas dos magnitudes, se trata de diversas maneras para obtener una señal final, que aplicada a los elementos motrices, varíe la posición real hasta hacerla coincidir con la deseada. Dependiendo del tratamiento que se le de a la señal de error obtenida, podemos tener controladores de tipo proporcional, integral o derivativo; o una combinación de ellos, denominada PID (proporcional-integral-derivativo).

1.6 LOS ELEMENTOS MOTRICES O ACTUADORES

La capacidad del robot para desplazar su cuerpo, brazo y muñeca se proporciona por el sistema de impulsión utilizado para accionar el robot.

El sistema impulsor determina la velocidad de los movimientos del brazo, la resistencia mecánica del robot y su rendimiento dinámico. En cierta medida el sistema impulsor determina las clases de aplicaciones que puede realizar el robot.

Los elementos motrices son los encargados de producir la energía necesaria para la aceleración o desaceleración de las articulaciones, valiéndose para esto de poleas, cables, cadenas, etc.

Los robots industriales, existentes en el mercado, están accionados por uno de tres tipos de sistemas de impulsión (motores primarios). La selección del sistema impulsor más apropiado depende de la carga útil, velocidad, control, y costo requerido. Estos tres sistemas son:

a. Impulsión hidráulica.
b. Impulsión eléctrica.
c. Impulsión neumática.

Los actuadores **hidráulicos** son idóneos para aplicaciones de momento torsional alto (gran capacidad de carga), y baja velocidad, con elevado factor de trabajo y una salida de fuerza estable. Sus inconvenientes radican en que suelen ser propensos a fugas de aceite y por lo general son voluminosos, sensibles a la temperatura (se requiere precalentamiento) y

son muy costosos.

Los actuadores **eléctricos o motores de c.c.** son la mejor elección por su fácil y preciso control.

Por ser pequeños, sus aplicaciones tienden a los trabajos de mayor precisión ya que estos tienen una alta resonancia (fácil de controlar), bajo momento torsional (par), alta velocidad y casi constante, con factor (ciclo) de trabajo reducido. Sus desventajas consisten en baja relación empuje/peso, no pueden mantener carga estable.

Los robots de impulsión eléctrica son accionados por motores paso a paso (PAP) o servomotores de c.c.

Los actuadores **neumáticos** emplean el aire comprimido como fuente de energía y son adecuados en el control de movimientos rápidos, en aplicaciones de bajo costo que se pueden implementar sin necesidad de control de posición, por servodispositivo o sea de precisión limitada. Suele reservarse para los robots más pequeños que tienen menos grados de libertad (movimientos de dos a cuatro articulaciones).

Estos robots generalmente se limitan a simples operaciones de "tomar y poner" con ciclos rápidos.

1.7 EFECTOR FINAL

El efector final o garra (gripper) son los nombres que se le da en la industria robótica a la mano del robot. Por lo general, el elemento terminal ha de soportar una elevada capacidad de carga y al mismo tiempo conviene que tenga reducido peso y tamaño.

Por la amplia variedad de tareas a las que se destinan los brazos robots, el elemento terminal adopta formas diversas. En muchas ocasiones es necesario diseñar el elemento terminal a medida de la operación que se aplique.

Algunos robots viene equipados con varios efectores finales diferentes los cuales pueden intercambiarse para incrementar la versatilidad del brazo robótico.

Los efectores finales, al igual que los brazos robóticos, pueden ser accionados con energía eléctrica, hidráulica o neumática.

Si se usa energía eléctrica, el motor puede localizarse en la mano para lograr un accionamiento directo, o puede situarse en otra parte del robot para un accionamiento remoto. Cables, cadenas, o fajas pueden usarse para mover los elementos de la

mano si se escoge un accionamiento remoto.

1.8 SENSORES DE INFORMACION

Los robots con capacidad sensorial constituyen la última generación de estas máquinas; ya que pueden relacionarse con el entorno y tomar decisiones en tiempo real. La información que reciben los hace autoprogramables, o sea, alteran su actuación en función de la situación exterior, lo que supone disponer de un cierto grado de inteligencia artificial.

Las informaciones de los sensores hacen referencia a la posición, velocidad, aceleración, fuerzas, pares, dimensiones y contorno de objetos, temperatura, etc. Para cuantificar los valores correspondientes a estos parámetros, existen en el mercado sensores de tipo mecánico, óptico, térmico, ultrasónico, eléctrico, etc.

Los sensores utilizados en robótica comprenden las categorías generales siguientes:

a) **Sensores táctiles.** Se trata de sensores que responden a fuerzas de contacto con otro objeto. Algunos de estos dispositivos son capaces de medir el nivel de fuerza implicada.

b) **Sensores de proximidad y alcance.** Un sensor de proximidad es un dispositivo que indica cuando un objeto está próximo a otro objeto, pero antes de que se produzca el contacto. Cuando puede detectarse la distancia entre los objetos, el dispositivo se denomina un sensor de alcance.

c) **Tipos diversos.** La categoría de diversos incluye las clases restantes de sensores que se utilizan en robótica. Se incluyen los sensores para temperatura, presión, posición, y otras variables.

d) **Visión artificial.** Un sistema de visión artificial es capaz de "visionar" el espacio de trabajo y dar una interpretación a lo que ve. Estos sistemas se emplean en robótica para realizar tareas de inspección, reconocimiento de piezas en tareas de montaje y otras similares.

1.9 CLASIFICACION DE LOS ROBOTS

La maquinaria para la automatización rígida dio paso al robot con el desarrollo de controladores rápidos, basados en el microprocesador, así como el empleo de servos en lazo cerrado, que permiten establecer con bastante exactitud la posición

real de los elementos del robot y establecer el error respecto a la posición actual deseada. Esta evolución ha dado origen a una serie de robots, que se citan a continuación:

1. **Manipuladores.**

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

   a) **Manual.** Cuando el operador controla directamente la tarea del manipulador.

   b) **De secuencia fija.** Cuando se repite, de forma invariable, el proceso preparado previamente.

   c) **De secuencia variable.** Se puede alterar algunas características del ciclo de trabajo.

2. **Robots de repetición o aprendizaje.**

   Son manipuladores que permiten repetir una secuencia de movimientos, previamente ejecutada por un operador humano, haciendo uso de un controlador manual o 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.

   Esta clase de robots son los más conocidos en los ambientes industriales. A este tipo de programación que incorporan se le da el nombre de "gestual".

3. **Robots con control por 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 computador dispone de un lenguaje especifico, compuesto por varias instrucciones adaptadas al robot, con las que se puede escribir un programa de aplicación utilizando solo el terminal del computador, no el brazo. A esta programación se le llama textual y se crea "off-line", es decir, sin la intervención del manipulador.

4. **Robots inteligentes.**

   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 artificial, la síntesis de voz y la inteligencia artificial, son las ciencias que más se están estudiando para su aplicación en los robots inteligentes.

5. **Micro-robots.**

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

CONCLUSIONES DEL CAPITULO I

La principal conclusión que se puede obtener, es que el robot no es un androide que desplazará al hombre en su trabajo, sino por el contrario es una extensión de la fuerza y habilidad del hombre que permitirá mejorar la calidad del trabajo de éste.

En nuestro país, el desarrollo de este campo es casi nulo, y podríamos suponer que esto es debido a la cantidad de recursos económicos y de personal especializado requerido en su desarrollo. En los países latinoamericanos, hay necesidades más apremiantes a las cuales se desvían esos recursos, y hasta que no sea superada esta situación, no se puede esperar un empuje fuerte en lo que es el campo de la robótica.

REFERENCIAS BIBLIOGRAFICAS

[1] Andeen, Gerry B. Robot Design Handbook. Edit McGraw Hill, España, 1a. Edición, 1988.

[2] Engelber, Joseph F. Robótica Industrial. Edit. McGraw Hill, España, 1a. Edición, 1989.

[3] Grupo del Buch Engineering Co. Inc. Robotics Trainings Systems, Concepts and Aplications.

[4] Peralta, Juan y Portillo, Marcos. Diseño del Sistema Locomotor y Sensor De Un Microrobot. Universidad de El Salvador, proyecto. 1992.

CAPITULO II

PROTOTIPO DE BRAZO ROBOTICO.

Introducción

En este capítulo se dan los lineamientos generales que se utilizarán de base para el diseño y construcción del brazo robot. Se plantea las especificaciones que contendrá el prototipo, con las cuales, en los siguientes capítulos se procederá al diseño completo del sistema robótico.

2.1 PLANTEAMIENTO DEL PROBLEMA DESDE EL PUNTO DE VISTA DE LA INDUSTRIA

La problemática existente en la fábrica o industria es la de carecer de recursos viables que beneficien al obrero y al empresario en la producción industrial. Hay ciertas tareas, de carácter peligroso e insalubre, que hasta ahora son realizadas por el hombre y que pueden ser realizadas por robots.

La asignación de tareas a robots, que antes realizaba el hombre, lo beneficiarán -al hombre- en estos aspectos:

a.- La ejecución de trabajos repetitivos (comodidad).
b.- Realización de procesos peligrosos e insalubres (seguridad)
c.- Mejora la calidad del trabajo humano (supervisión).

Con la utilización de robots en los procesos de la industria, el empresario se beneficia en los siguientes aspectos.

a.- Mayor productividad.
b.- Ahorro de materia prima y energía.
c.- Mejores beneficios económicos.

2.2 OBJETIVOS DE LA SOLUCION

Para proporcionar una solución al problema anteriormente planteado, definiremos ciertos objetivos que limitarán el ámbito de la solución propuesta. Estos objetivos concuerdan con los objetivos del trabajo global, y los podemos dividir en objetivos generales y objetivos específicos.

2.2.1 OBJETIVOS GENERALES

A. Dejar sentadas las bases y criterios para el diseño y construcción de un brazo robot.

B. Estudiar a fondo los requerimientos y especificaciones que debe contener un sistema robótico (brazo robótico).

C. Contribuir a la industria salvadoreña con soluciones en el campo de la robótica (brazos robóticos) en aplicaciones peligrosas, insalubres, y repetitivas.

D. Proyectar a la Universidad De El Salvador y Escuela De Ingeniería Eléctrica en el ambiente Industrial Salvadoreño.

2.2.2 OBJETIVOS ESPECIFICOS

A. Construir un brazo robot con capacidad de movimiento, de sensar (retroalimentación), y de agarre (tomar y dejar objetos).

B. Desarrollar un software de fácil comprensión entre usuario-robot (amigable), este programa interactuará con todas las partes del brazo robótico (motores eléctricos de c.c., computador, e interfases).

C. El sistema robótico quedará para uso didáctico (uso general), así en futuras investigaciones se pueda ampliar sus capacidades e implementar a partir de él sistemas autómatas industriales más complejos.

D. Se dejara un manual de operación del brazo robótico para el usuario.

2.3 PROPUESTA DE SOLUCION

En base a lo anteriormente planteado, se persigue desarrollar un brazo mecánico industrial que reúna los siguientes requerimientos.

1. Ser operado en un lugar remoto al lugar de trabajo. Esto le da flexibilidad al robot, ya que podrá ser controlado desde un lugar apartado al ambiente de trabajo; donde, posiblemente se desarrolle una tarea peligrosa, y que para el humano puede ser dañino a su salud.

2. Capaz de entender ordenes vía computador. El robot industrial (brazo robótico) reconocerá las instrucciones u ordenes que estén en el programa. Este es el ambiente idóneo para que el usuario se pueda comunicar con el

brazo robótico; pero antes que lleguen las instrucciones al brazo robot es necesario que sean acondicionadas por el computador, porque así el sistema responderá a lo que el usuario-programa quieran que desarrolle.

3. Desarrollar tareas repetitivas y peligrosas, si es necesario en un ambiente no idóneo para el ser humano.

   El brazo robot será capaz de realizar tareas de índole repetitiva, estas tareas para los humanos se vuelven en un momento dado monótonas y cansadas, lo que conlleva a incurrir en errores, afectando en pérdidas de dinero para el empresario.

   La tarea peligrosa se presenta cuando el humano tiene que realizar un trabajo en un ambiente tóxico e insalubre; esto provoca la baja de productividad del obrero así como el deterioro de su salud.

   El obrero en este ambiente puede presentar síntomas de agotamiento, enfermedad y hasta llegar a la muerte.

4. Capacidad de Servocontrol. A través del sensoreo de la posición del brazo, al sistema se le dará una mayor precisión y eficiencia.

   La precisión está relacionado con el punto al cual se desea llegar, y la eficiencia con las características dinámicas del sistema, que serán deducidas a partir de la posición real y la posición esperada.

2.4. PROTOTIPO DEL BRAZO ROBOTICO

El brazo robot a diseñar debe cumplir con las siguientes características:

1. Debe ser controlado por un computador de uso general

2. Desde el centro de mando (computador) se le dará ordenes al brazo en un lugar remoto.

3. El brazo robot tendrá un servomecanismo en lazo cerrado.

En vista de presentar una solución al problema planteado, se han generado ciertas especificaciones que tendrá el brazo robot a construirse; éstas se detallan a continuación.

2.4.1 ESPECIFICACIONES GENERALES

De acuerdo a lo planteado en los antecedentes, el robot a construirse será controlado por un computador, obedeciendo a

un lenguaje altamente específico - exclusivo del ambiente del robot - al cual se le denomina lenguaje 'textual'.

El control del robot se podrá llevar a cabo en dos modos diferentes:

1. **Modo Programado:** este se ejercerá mediante el uso del lenguaje textual (lenguaje orientado al brazo), el cual tendrá una serie de operaciones -programa- que el robot ejecutará consecutivamente hasta finalizar el programa. En este caso, el usuario creará un programa, que se almacenará en la memoria del computador, y podrá ser ejecutado repetidamente sin intervención de elementos externos.

2. **Modo Manual:** o modo interactivo. En este modo, el usuario podrá ejercer control directo de las partes del brazo mediante teclas específicas del computador. Por ejemplo, para cerrar la garra y abrirla podrá presionar la tecla 'P', y el brazo responderá de inmediato al presionarse la tecla. En este caso, cada acción que ejecuta el brazo, es respuesta a determinada tecla que el usuario presiona.

2.4.1.1 MANIPULADOR

Por el tipo de manipulador o brazo, se utilizará una configuración articulada - parecido al brazo humano -, el cual le proporcionará al brazo robot ciertas características:

1. Es el brazo que tiene la mayor área de trabajo posible por menor espacio ocupado.
2. Su campo de acción lo constituye una esfera con centro en la base del brazo. En caso de que la base esté a nivel del piso, el área de trabajo se limita a una semiesfera.
3. Puede alcanzar por arriba o debajo de obstáculos.
4. Es muy difícil de programar
5. Es el manipulador más complejo

2.4.1.2 CONTROLADOR

El robot será controlado a través de un computador PC compatible de uso general.

A grandes rasgos, la PC se encargará de la interacción con el usuario y el brazo, a partir de las ordenes dadas por un programa desde la PC.

Las funciones que la PC tendrá a su cargo se describen a continuación:

1. Proporcionar al usuario un ambiente amigable para

desarrollar sus aplicaciones, lo cual se logra con un programa que proporciona menúes explicativos, mediante los cuales el usuario será "guiado de la mano" por el software para realizar la tarea deseada. Además, existirá un lenguaje idóneo para la programación del robot. En este lenguaje, el usuario tendrá a su disposición poderosos procedimientos con los cuales no se preocupará de detalles de hardware.

2. Procesar las instrucciones dadas por el usuario, utilizando complejos algoritmos de cálculo geométrico. A partir de estos cálculos se obtendrán como resultado las señales que se deben aplicar a cada motor.

   Por ejemplo, el usuario mediante una instrucción simple, estará indicando que la garra -efector final- llegué a un punto de referencia u origen; el software en la PC debe, a partir de la posición actual, y la posición futura calcular las señales que se deben aplicar en cada motor del brazo, para que éste efectivamente ejecute la función deseada.

3. Tendrá a su cargo el movimiento de cada articulación, así como el conjunto en total, lo cual requiere de la aplicación de señales complejas para tener en consideración las propiedades dinámicas del sistema.

   Estás señales, usando lógica digital, sería en la realidad muy difícil de implementarse, por lo cual se recurre a la capacidad del microprocesador de realizar algoritmos complejos para que genere las señales de cada motor. Para la ejecución del movimiento como un todo, el microsistema tendrá algoritmos que considerarán las características dinámicas y geométricas del brazo.

4. Procesar las señales de retroalimentación recibidas de los sensores de posición y de las líneas de puerto disponibles para el usuario.

El controlador funcionará como un servomecanismo en lazo cerrado. Mediante el uso de sensores de posición - entre ellos optoacopladores o potenciómetros -, la PC obtendrá señales que le indicarán si el brazo ha llegado a la posición deseada, para actuar en consecuencia.

2.4.1.3 SISTEMA DE IMPULSION

Para producir el movimiento de las articulaciones, se utilizará un sistema de impulsión eléctrica, cuyas características fundamentales son: bajo momento torsional, velocidad casi constante, poca carga útil y control preciso. Como primera alternativa se ha pensado en motores eléctricos

paso a paso, debido al control preciso - con poca lógica externa - que se puede ejercer sobre su movimiento angular, y además se tienen a la mano.

Los motores eléctricos son utilizados ampliamente en aplicaciones que requieren de una precisión elevada, como lo es el montaje.

2.4.2 ESPECIFICACIONES DE SOFTWARE

El software será el elemento que le dará vida a todo el sistema físico. Debido a las múltiples funciones que se le han asignado, así como la complejidad de éstas, el software a desarrollarse tiene un nivel de dificultad muy grande.

Las funciones específicas de este software se detallan a continuación:

1. Interfase Usuario-Sistema Robótico.

Para obtener lo mejor de la PC compatible, su función principal radica en proporcionar al usuario una poderosa herramienta de comunicación entre él y el brazo robot, de tal forma que el usuario se sienta en la capacidad de ejecutar cualquier aplicación por muy compleja que esta parezca, sin involucrar mucho esfuerzo de parte de él.

En todo momento, el usuario se sentirá en un ambiente amigable. Lo anterior se logra a través de dos características del software: la primera es la disponibilidad de un lenguaje de alto nivel orientado hacia el usuario y la segunda, la existencia de rutinas de comunicación directa con el usuario (por ejemplo, rutinas de visualización de mensajes sobre la acción que el robot ejecuta, el porcentaje de su movimiento, etc.).

El software se implementará con rutinas en lenguaje Ensamblador del microprocesador 8086 y rutinas en lenguaje PASCAL, aprovechando al máximo sus características de lenguajes de alto nivel.

Al final del trabajo de graduación se proporcionará un disco con el software, tanto con el código fuente como los programas ejecutables.

2. Lenguaje de Programación.

La segunda función, y no por eso menos importante, es la disponibilidad de un lenguaje altamente orientado a la programación de robots. El usuario podrá crear, a través del uso de este lenguaje, programas de aplicación que puedan ser almacenados en disco, y de ahí ejecutarse miles y miles de

veces, obteniendo del brazo robot siempre los mismos resultados.

Hay que enfatizar que los requisitos del usuario que programará las aplicaciones, son relativamente bajos, únicamente debe tener conocimientos básicos de programación; no es necesario que conozca algún lenguajes específico como Pascal, Basic, Ensamblador o que tenga conocimiento de control por microcomputador, ya que todas estas son inherentes al sistema.

Un lenguaje se define por sus reglas sintácticas y el conjunto de instrucciones que lo componen. Para cada instrucción, habrá una rutina correspondiente en lenguaje Pascal, que realizará lo indicado por la instrucción. Algunas instrucciones o subrutinas se codificarán en lenguaje ensamblador para los movimientos en tiempo real.

En un lenguaje orientado al control de un brazo robot, las instrucciones que poseerá se pueden dividir de la siguiente manera:

a. Control de movimiento: es el requisito principal de cualquier lenguaje de programación de robots. Con estas instrucciones, el usuario podrá manipular las articulaciones del brazo a su antojo. Inherentes al software -no visibles por el usuario-, estas instrucciones provocarán una serie de operaciones, entre las que tenemos: el cálculo de la rutas del movimiento, determinación de parámetros para el movimiento efectivo, etc.

b. Sensoreo: el lenguaje será capaz de obtener datos de los sensores de posición acoplados en cada articulación, con lo cual el brazo puede controlar el proceso en su entorno y responder a cambios del medio.

c. Toma de decisiones y lógica: estas instrucciones incluyen instrucciones de repetición, ramificación, etc. Una condición de toma de decisiones y lógica podrá ser generada internamente por el programa u obtenida a través de las instrucciones de sensoreo.

d. Procesamiento de datos: incluye las funciones de lectura/escritura de datos del almacenamiento, visualización de mensajes para operadores, instrucciones aritméticas, etc.

e. Capacidad de diagnóstico integrada: estas instrucciones le proporcionaran al lenguaje la característica de seguridad, ya que mediante estas se pueden establecer límites de tolerancia en el software para la ubicación física de cada pieza del brazo. Estos límites de tolerancia, están determinados por las dimensiones físicas del robot, así como por sus características dinámicas.

3. Modos de operación.

El software permitirá el control del robot de diferentes formas. La primera ya se describió, mediante el uso de un lenguaje específico se crearán programas que se podrán ejecutar cuántas veces se desee. Se podría llamar a esta forma Modo Automático de control o control Off-Line.

Otra forma de controlar el brazo es directamente desde el teclado del computador. Por ejemplo, cuando se mantenga presionada la tecla **Cursor Arriba**, la garra se moverá en una línea vertical hacia arriba, y similar función se le podría asignar a las otras cursoras. Es evidente, que en este modo, el usuario observa inmediatamente la acción del brazo. A este modo de operación le denominaremos Modo Manual de control o control On-Line.

Usando el modo manual, se pierde la característica de repetición que deseamos que tenga el brazo. Su finalidad se verá aclarada a continuación. Mediante rutinas de software, estando en modo manual, se puede grabar en disco las instrucciones que responden a cada movimiento que es ejecutado manualmente. Para realizar lo anterior, se seguirían los siguientes pasos:

a. Se activa el modo manual.

b. Se activa Grabar Instrucciones.

c. Se realiza manualmente - a través del teclado - la operación que el brazo ejecutará. Por cada acción sobre el teclado se generará una instrucción que es almacenada en memoria.

d. Se cierra Grabar instrucciones, y el programa en memoria es traslado a un archivo en el disco para su posterior ejecución en forma automática.

Es evidente en este punto la utilidad que presenta el modo manual. El proceso es ejecutado manualmente una vez por el operario, y el computador crea un programa, el cual posteriormente se puede ejecutar en modo automático cuantas veces sea necesario.

4. Control de las articulaciones

Esta función es la que relaciona el control directo de las articulaciones a través de una interfase de potencia.

A partir de las instrucciones generadas en la PC, se calculan los parámetros para ejecutar el movimiento.

Para realizar este movimiento, el software debe tener en

cuenta, las características dinámicas que el brazo posee y así poder manipularlo en forma confiable y eficiente. Por ejemplo, una característica que toma en cuenta la dinámica del movimiento, es que al inicio del movimiento comienza con una velocidad baja, acelerando y luego desacelerando hasta llegar a la posición final de reposo. Al realizar un movimiento a velocidad constante, se presentaría problemas debido a la inercia del sistema y el sistema no respondería como se espera.

5. Recepción e interpretación de los Sensores.

El programa captará la información de los sensores de posición, para tratar la señal obtenida a nivel local, con rutinas especiales de atención.

CONCLUSIONES DEL CAPITULO II

La conclusión principal que se puede obtener, es que la industria debe invertir en la investigación y desarrollo de sistemas del tipo planteado, ya que redunda en mayores beneficios para el empresario, así como en un mejoramiento de la calidad de trabajo del ser humano.

Otra conclusión, es que si bien el sistema a desarrollar no tendrá una capacidad para movilizar piezas de gran peso, la generalidad con que se ha especificado permitiría retomar diferentes módulos para construir un sistema de uso específico en la industria, ya sea nacional o extranjera. En este trabajo, en el proceso de diseño mismo se pretende plantear un algoritmo que ayude a futuros investigadores a desarrollar su propia aplicación robótica.

Podemos decir sin equivocarnos, que los sistemas robóticos ya no siguen siendo una novedad en el mundo, pero en nuestro país todavía lo son. Estos sistemas ya han probado que son lo suficiente capaces de ayudar a la diversidad de procesos industriales a aumentar la productividad.

Las enormes posibilidades tecnológicas que ofrecen los robots son evidentes y éstas no harán sino aumentar en un futuro cercano. Pero su repercusión en el ámbito laboral no deja de plantear incógnitas o cuanto menos, inquietudes; debemos ser conscientes y aceptar la realidad del impacto laboral ya que los trabajadores requerirán de un esfuerzo mayor que los obligará a capacitarse continuamente, o quizás lleguen a ser reemplazados completamente o ser reducidos a un mínimo operante.

La industria sin la necesaria reconversión tecnológica no podrá competir con el mercado foráneo y esto redunda en perjuicio económico.

Por los costos que involucra obtener en el exterior un robot con capacidades bien definidas en la elaboración de un proceso determinado; nos vemos en la necesidad de aportar desarrollando un robot prototipo que reúna las características típicas de un brazo robot industrial.

El hardware se simplificará en la lógica de control de cada motor; redundando al mismo tiempo, en que el software a utilizarse será de elevada complejidad.

REFERENCIAS BIBLIOGRAFICAS

[1] Andeen, Gerry B. Robot Design Handbook. Edit McGraw Hill, España, 1a. Edición, 1988.

[2] C. S. G Lee, K. S. FU y R. C. Gonzales. Robótica. Control, Detección, Visión e Inteligencia. McGraw Hill, 1a. Edición. 1988.

[3] Engelber, Joseph F. Robótica Industrial. Edit. McGraw Hill, España, 1a. Edición, 1989.

[4] Grupo del Buch Engineering Co. Inc. Robotics Trainings Systems, Concepts and Aplications.

[5] Peralta, Juan y Portillo, Marcos. Diseño del Sistema Locomotor y Sensor De Un Microrobot. Universidad de El Salvador, proyecto. 1992.

CAPITULO III

DISEÑO MECANICO DEL BRAZO ROBOT

Introducción

El procedimiento de diseño mecánico se ha dividido para efectos prácticos en dos subprocesos: el diseño mecánico propiamente dicho y el procedimiento de cálculo.

En este capítulo se han incluido, en una primera parte, las consideraciones y alternativas relativas al manipulador físico, como lo son el tipo de configuración, tipo de actuadores necesarios, ubicación de los actuadores, medios de transmisión del torque, contrapeso; en suma lo que es la estructura del brazo robot.

En la segunda parte de este capítulo se procede al diseño, cálculo (engranes, poleas y piñones) y dimensionamiento de la estructura del brazo, así como la selección de los materiales a utilizarse.

3.1 SELECCION DEL TIPO DE CONFIGURACION

Las diferentes configuraciones existentes de brazos mecánicos son: polar o esférico, cilíndrico, cartesiano o x-y-z y el articulado. Cada uno de ellos presenta ventajas y desventajas, que pueden referirse a la programación, estabilidad mecánica, robustez, etc.

Las consideraciones al elegir un tipo de configuración específica son relativas al tipo de tarea que se realizará. Para este caso, el brazo robot será para ejecutar un trabajo donde se requiere poca capacidad de carga, pero a su vez, dentro de estos límites, que su uso sea de carácter general. De las 4 configuraciones descritas anteriormente, la opción escogida es la del brazo articulado -parecido al brazo humano- ya que es el que presenta mayor flexibilidad en su movimiento. La desventaja principal que presenta es un alto nivel de complejidad estructural y de programación.

Como ya se dijo, esta configuración es modelada en base al brazo humano. En la parte inferior hay un cuerpo rotacional articulado en la base. Una articulación en el hombro -base- con movimiento rotacional, mueve el antebrazo hacia arriba y

abajo. En el otro extremo del antebrazo se encuentra otra articulación (codo) que conecta al brazo y en el extremo de este último se encuentra la garra.

Según estudios realizados en brazos robots con este tipo de configuración, la máxima cantidad de trabajo -o flexibilidad de movimiento del brazo- se logra cuando la longitud del antebrazo y el brazo son similares o casi iguales.

3.2 SELECCION DEL TIPO DE ACTUADOR

Los actuadores o sistemas de impulsión para robótica, deben satisfacer múltiples requerimiento, que muchas veces están en conflicto. En las aplicaciones comunes, de los actuadores se requiere el mejor funcionamiento en uno o dos parámetros; en cambio para aplicaciones robóticas se desea que la mayoría de parámetros tengan las mejores características de funcionamiento.

Los tipos de actuadores disponibles son: eléctricos, hidráulico y neumáticos. En motores eléctricos hay una gran variedad, desde motores de paso hasta motores dc servos, los cuales poseen un excelente control. Entre las cualidades de los motores eléctricos están la simplicidad, buen control y bajo torque de arranque. Los sistemas hidráulicos y neumáticos se caracterizan por un elevado torque, tanto en reposo como en movimiento, y bajo control -entre mejor control se desea, mayor es la complejidad-.

La mejor selección para la aplicación a desarrollar son los motores eléctricos, lo cual reduce la complejidad del control y también porque está orientado al área de Ingeniería Eléctrica.

3.2.1 PARAMETROS DE FUNCIONAMIENTO DE LOS ACTUADORES

A continuación se describen una serie de parámetros que se deben considerar al elegir un motor para aplicaciones robóticas. Además se describe lo que se desea de dicho parámetro en aplicaciones robóticas.

Curva Torque/Velocidad

Esta es el más fundamental parámetro de funcionamiento de un motor. La máxima velocidad en vacío -sin carga- esta íntimamente relacionada con la máxima velocidad requerida por el robot. El máximo torque con el rotor bloqueado se relaciona con la máxima aceleración y carga requerida por el robot.

En la curva Torque/velocidad de los motores de paso, con frecuencia se encuentran picos y saltos repentinos, lo cual

indica que hay una velocidad de resonancia. Para que el motor funcione como se debe, lo mejor es mantener la máxima velocidad por debajo de estas velocidades de resonancia, en las cuales el motor no responde adecuadamente, es decir que por cada pulso de entrada ya no avanza un ángulo constante, denominado paso.

Inercia del Rotor

La inercia del rotor afecta directamente a la aceleración que puede proporcionar el motor. Por lo tanto una baja inercia es deseada, ya que aumentará la aceleración en la articulación del robot.

Eficiencia

A diferencia de los motores industriales, la eficiencia no es un parámetro importante ya que los motores para aplicaciones robóticas consumen poca potencia. A medida que los robots se expandan en la industria, y se incremente el costo de la energía, este parámetro adquirirá mayor importancia.

Peso

Los motores pueden ser ubicados en la base o cerca de la articulación que moverán -en el interior de la estructura del robot-. Lo deseable es la segunda opción, lo cual incrementa el peso de la estructura. Por lo tanto un bajo peso es deseable si el motor se coloca en la estructura del brazo.

Escobillas

En motores que usan escobillas, el cambio de estas es frecuente, por lo tanto se desea que el mantenimiento de éstas no requiera el desarmar todo el robot. En síntesis se desea un motor de fácil mantenimiento.

Potencia en el eje

En los motores industriales, se desea una salida máxima de potencia continua. En cambio, en aplicaciones robóticas no es así, existiendo demandas de elevados torques instantáneos, picos de inercia o picos de peso. Lo anterior requiere de un incremento de la potencia de salida, lo cual se puede lograr con un buen sistema de ventilación de los motores.

3.2.2 TIPOS DE MOTORES

Varios tipos de motores eléctricos pueden ser utilizados en aplicaciones robóticas. En los siguiente párrafos se describe brevemente las características de los motores DC servos, motores DC sin escobillas y motores de paso.

Servomotores DC

Los servomotores DC, o motores DC servocontrolados tienen un buen torque, velocidad y una buena salida de potencia continua. Son relativamente fáciles de usar en aplicaciones servocontroladas, y tiene un comportamiento satisfactorio con torque o velocidad casi cero. La desventaja principal de estos motores es la elevada inercia del rotor, lo cual redunda en una aceleración baja. Este efecto puede ser despreciable si la inercia de la carga en el eje es mucho mayor que la inercia del rotor, pero aún así la aceleración es menor que en el caso que el rotor presentara menor inercia.

En términos generales, el torque es proporcional a la corriente de rotor cuando la corriente de estator es constante; y a carga cero, la velocidad del motor es proporcional al voltaje del rotor.

Motores DC sin escobillas

Este motor es realmente un sistema motor DC sin escobillas, que consta del motor y un controlador electrónico del motor. El rotor es un magneto permanente y posee un estator ranurado. Además posee un circuito codificador -parte integral del motor-, el cual se encarga de conmutar la corriente al estator en la secuencia apropiada para producir la rotación del rotor.

La ventaja principal de estos motores es que no poseen escobillas que redunda en la reducción del mantenimiento y se evita el arco de las escobillas. Su desventaja es que son más caros y aún siguen teniendo un rotor cuya inercia es elevada.

Motores de Paso

Los motores de paso -stepper motor- son únicos, ya que el movimiento del rotor es determinado por las señales de entradas del motor. Las señales de entradas son una serie de pulsos; y el rotor avanza un paso por cada pulso. El tamaño del paso puede variar desde una fracción de grado hasta 15°.

Ya que existe una relación directa entre las señales de entrada y el movimiento del rotor, los motores de paso no requieren sistema de servo-control ni codificador para lograr el posicionamiento deseado en aplicaciones robóticas.

Desafortunadamente, estos motores tienen varias desventajas. Entre estas se encuentran:

a. Para un torque y potencia dada, el peso de los motores de paso es mayor que otros tipos de motores usados para robots.

b. Inercia del rotor es mayor que otros tipos de motores.

c. Si la carga del motor es excedida, el motor no responderá adecuadamente, y puede perder o ganar pasos, con lo cual la posición se pierde, y el sistema no se ha enterado.

d. La velocidad máxima de estos motores no es muy alta. Un valor máximo encontrado en motores de paso es de 80 rev/min.

Para efectos de diseño, el diseñador debe estar consciente de estos aspectos, que pueden redundar en un mal funcionamiento de su sistema.

3.2.3 SELECCION DEL ACTUADOR

Puestas todas las alternativas, se utilizarán motores de paso, lo que nos beneficiará en la simplificación del control, pero nos dificultará el diseño físico del manipulador.

Sin embargo, al haber escogido motores de paso, nos limita muchas alternativas en cuanto a la estructura del manipulador.

En primer lugar, la estructura debe tener un bajo peso, dimensionada para que la carga del motor no exceda los límites de respuesta adecuada.

Los motores no pueden estar en el interior de la estructura, ya que aumentaría la carga en el eje del motor. Lo anterior nos obliga a concentrar los motores en la base y utilizar un sistema de trasmisión de torque hacia las articulaciones, donde es requerido el torque.
Una amplia descripción de los motores de paso la puede encontrar en el capitulo IV.

3.2.4 UBICACION DE LOS ACTUADORES

La primera consideración al ubicar los motores, es colocarlos lo más cerca posible de la base y lejos de la garra; lo anterior disminuirá la inercia del manipulador y maximizará la carga y velocidad que pueda alcanzar el manipulador.

En el diseño planteado, los motores serán colocados en la base del manipulador, aprovechando eficientemente la capacidad de carga de los motores de paso.

Lo anterior hace necesario un sistema de transmisión, que transmita el movimiento de los actuadores (motores) a las articulaciones del brazo y antebrazo.

Para facilitar el mantenimiento de los actuadores, y el sistema de transmisión serán montados en un lugar accesible.

3.3 SISTEMA DE TRANSMISION

El torque y la velocidad disponibles en los actuadores es diferente a la necesitada para controlar el movimiento de la articulación.

La transmisión, aparte de llevar el movimiento a la articulación desde el actuador, permite multiplicar el torque a costa de disminuir la velocidad.

Al escoger el tipo de sistema de transmisión, se evaluaron dos posibilidades: 1o) usando engranajes en combinación de correas o fajas dentadas, y 2o) la utilización de engranajes, poleas y cables.

La primera opción fue desechada por la razón de que no hay en el mercado fajas dentadas de las medidas que se deseaban, y era posible conseguirlas en el extranjero pero a precios elevados.

Se escogió la segunda opción: cada motor transmite su torque a un engranaje, al cual se ha unido a una polea, de tal forma que al girar el engranaje, la polea gira con él. De la polea sale un cable, trasmitiéndose la fuerza hacia la polea final, de esta polea salen los extremos del cable a sujetarse a la parte del enlace (antebrazo, brazo o garra) que se moverá.

En vista que se usarán cables, se hace necesario un sistema para ajustar la tensión. Los sistemas existentes se muestran en la figura 3.1 y 3.2. El método utilizado en el diseño es el mostrado en la figura 3.1, siendo el más sencillo de los tres métodos presentados.

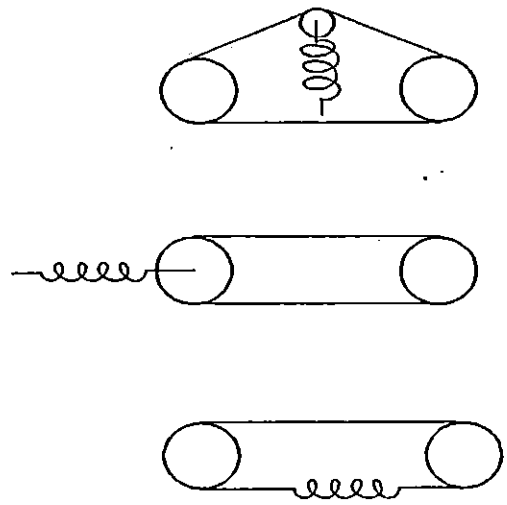


Figura 3.1 Métodos para ajustar tensión en cables

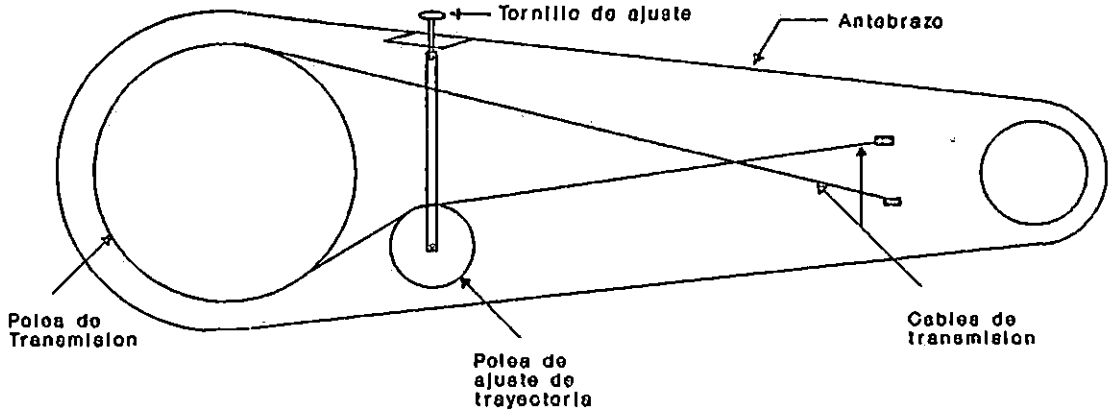


Figura 3.2 Mecanismo utilizado en el brazo robot para el ajuste de la tensión.

En la figura 3.2 se detalla la forma de implementar el mecanismo de ajuste de tensión, y que se logra enroscando o desenroscando el tornillo de ajuste señalado en la figura. Es de hacer notar que el tornillo está sujeto a una pestaña que sale de la estructura del brazo.

El usuario ajusta la tensión girando el tornillo mostrado. Cuando el tornillo gira, la polea conectada a él sube o baja. El propósito de esta polea es incrementar la trayectoria del cable (cuando sube) o disminuirla (cuando baja); en el primer caso, al incrementarse la trayectoria del cable, se está aumentando la tensión en el cable; en el caso contrario, al disminuir la trayectoria la tensión del cable disminuye.

3.4 CONTRAPESO

El torque requerido para el movimiento del robot es crítico. Una manera de reducir éste, es colocando contrapesos que eliminen el peso de la estructura del robot, como se muestra en la figura 3.3.

El uso de contrapesos puede reducir el tamaño del motor, pero la inercia es incrementada y en forma contraproducente, la aceleración máxima alcanzada por el brazo será menor.

Para nuestro diseño desecharemos la utilización de contrapeso, ya que el peso del brazo se minimizará utilizando materiales livianos, por lo cual el torque requerido para moverlo es bajo; si se introdujera contrapeso, el sistema tendería a ser más lento en sus movimientos al disminuir la aceleración máxima.

Considerando el hecho, que a través del sistema de transmisión se redujo la velocidad para incrementar el torque, colocar

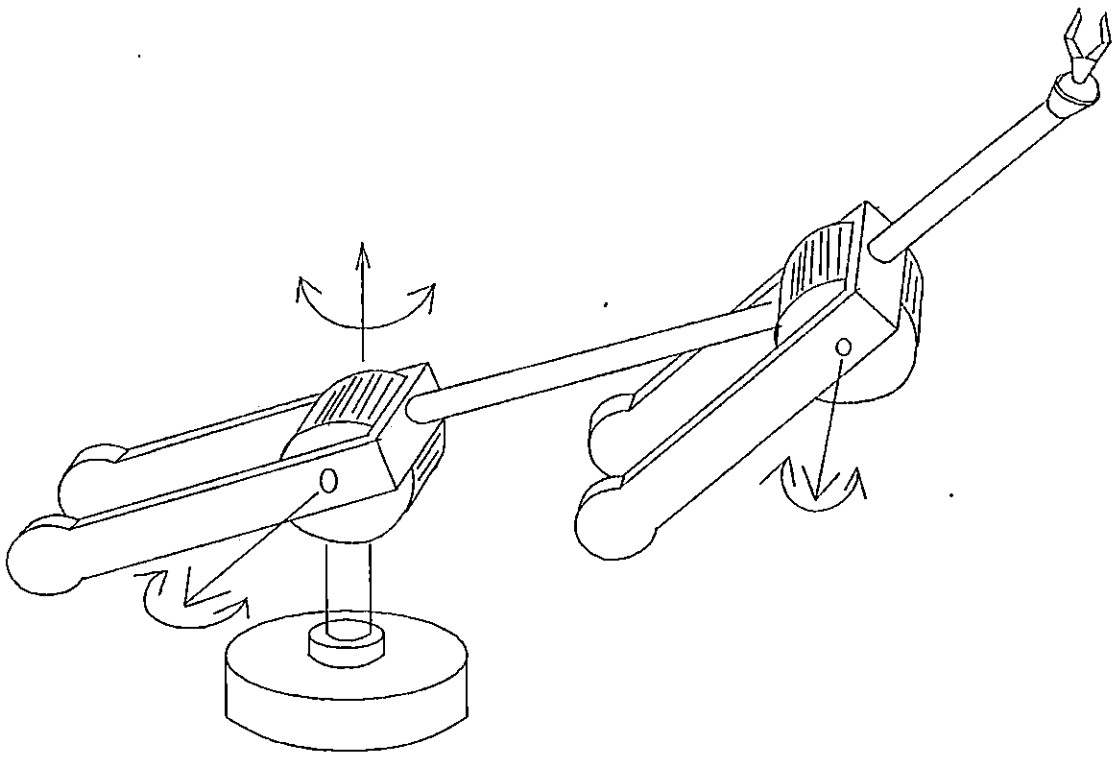


Figura 3.3 Configuración de manipulador con contrapeso

contrapesos redundaría en una disminución mayor de la velocidad, a niveles no permisibles.

3.5 INTERFASE DE LA GARRA

Los diversos usos de los brazos robots, requieren de diferentes tipos de garras, y actualmente la industria -de países desarrollados- ve la necesidad de estandarizar una interfase, que permita adaptar cualquier tipo de garra, de tal forma que el mismo robot permita múltiple aplicaciones, y al mismo tiempo que las garras sean compatible entre robots de diferentes fabricantes.

En las grandes industrias, los robots son cada vez más complejos y realizan múltiples tareas; lo anterior hace necesario que el intercambio de garra sea automático, sin la necesidad de la intervención humana.

Este es uno de los objetivos que se siguen en la construcción de una interfase universal para la garra.

Para este caso, se ha escogido un tipo de garra que consta de dos dedos planos que se cierran para agarrar el objeto.
Se seleccionó este tipo de garra por su simplicidad, y porque únicamente utiliza un motor para su implementación. Entre las aplicaciones de este tipo de garra, se encuentra la transferencia de materiales, operaciones en las cuales el

objetivo principal es mover una pieza de una posición a otra. Otra aplicación es el recubrimiento al spray para pintar objetos de metal, madera,. etc.

Las garras, entre más complejas sean, requieren de una mayor cantidad de actuadores, que incrementa el costo global del brazo robot.

3.6 SELECCION DE MATERIALES A UTILIZAR

El brazo robot constará de varios elementos, como lo son los enlaces, poleas, engranes, cables, ejes y otras piezas para soporte, uniones, etc. Los criterios principales para elegir el tipo de materiales a utilizar son su costo y peso.

El primer criterio no requiere mayor análisis, entre menor sea el costo, el costo global se reducirá. Se desea que sean de bajo peso, ya que de esta manera la carga útil que podrá soportar será mayor. Lo anterior se debe a que el motor debe levantar la carga así como el peso del brazo, y la suma de ambos define un valor máximo constante (dada por la capacidad máxima del motor). De esta manera, para obtener una mayor carga útil debemos buscar minimizar el peso del brazo.

El material usado para la estructura mecánica es un metal no ferroso: el aluminio, que reúne características sobresalientes tales como: liviano en comparación con su volumen (densidad 0.10 $lbs/pulg^3$), de fácil mantenimiento, resistencia a la corrosión, alta fuerza tensil (13 kpsi) y fácil de moldear o fabricar.

Comercialmente se dispone de aluminio en forma de placas, barras, laminas o planchas, hojas, varillas y tubos y en perfiles estructurales. El aluminio a usar es del tipo plancha con espesor aproximado de 2 mm.

Los engranes y poleas generalmente se fabrican de acero, hierro fundido, aluminio, bronce, resinas fenólicas como: nylon, teflón, pvc, etc. Recientemente se ha usado con éxito nylon, teflón, pvc, titanio, aluminio. La gran variedad de materiales existentes da la oportunidad de obtener el óptimo para cualquier necesidad en particular, ya sea que se trate alta resistencia mecánica, larga duración al desgaste, operación silenciosa, alta confiabilidad y/o capacidad de obtener peso reducido.

Los engranes no metálicos se conectan con engranes de aluminio, bronce, acero y/o hierro fundido para obtener la máxima capacidad de carga. Para lograr buena resistencia al desgaste, el metálico debe tener una buena dureza de por lo

menos 300 BHN[1]. Uno no metálico no soportaría casi tanta carga como uno de hierro fundido o de acero maleable, aún cuando la resistencia del material sea mucho menor por el bajo módulo de elasticidad[2]. Este permite al no metálico absorber los efectos de los errores en los dientes, de manera que no se origina carga dinámica[3]. Un engrane no metálico tiene también la ventaja de trabajar bien con lubricación marginal.

Las fibras fenólicas como: el nylon, PVC y/o teflón son materiales de engranes y poleas que han dado resultados excelentes.

El material idóneo para el diseño de los engranes y poleas es el cloruro de polivinil o comúnmente llamado PVC, ya que reúne características o ventajas importantes como: bajo costo, alta fuerza tensil, fácil de moldear o fabricación de piezas, fácil de mantenimiento, fácil de destruir -no contamina el ambiente- inodoro e insaboro, liviano, alta resistencia a la tracción de 30 a 50 N/mm^2, se rompe difícilmente y mantiene forma estable hasta los 60ºC.

Los otros tipos de fibras fenólicas también reúnen esas características o ventajas, pero lo que obliga a que nos decidamos por el PVC es el bajo costo (13 colones/cm) con un diámetro de 2.5", en comparación el costo de los otros tipos de fibras es superior a los 70 colones/cm con diámetro de 2.5".

También será necesario utilizar aluminio para la fabricación o construcción del engrane que soportara toda la estructura del brazo. La decisión de usar este material para este caso se debe a su facilidad de moldearlo, al bajo peso que se tiene en proporción a su volumen y al hecho de que fue donado. En este caso, el uso de fibra fenólica sería idóneo, pero en el mercado no hay fibras fenólicas de diámetro de 6", que es el diámetro requerido para soportar toda la estructura (ver análisis estático de la base).

3.7 ESPECIFICACIONES DE DISEÑO MECANICO

En el diseño mecánico se definen ciertas especificaciones que debe cumplir el brazo robot, y a partir de ellas se obtienen y calculan las dimensiones físicas de la estructura del robot.

---

[1] Ver página 187. Diseño en ingeniería Mecánica. Cuarta Edición. JOSEPH E. SHIGLEY.- LARRY D. MITCHEL.

[2] Ver página 42. OP. CIT.

[3] Ver Página 52. OP. CIT.

En el proceso de diseño, las especificaciones de funcionamiento del brazo robot son el punto de partida para definir los demás parámetros de construcción. Para el diseño desarrollado, las especificaciones corresponden a tres parámetros: la resolución del efector final, la máxima carga útil a levantar y la máxima velocidad lineal que el brazo puede alcanzar.

La resolución espacial de un robot se define como el más pequeño incremento de movimiento en el que el robot puede dividir su volumen de trabajo. Entre más pequeño es este incremento, la precisión del movimiento es mayor. Por tanto, un valor bajo de este incremento es deseable, y entre más bajo sea, se dice que tiene mayor resolución. La resolución que se encuentran en robots impulsados eléctricamente, por lo general oscila entre 0.5 mm y 5 mm.

La máxima carga útil es la capacidad de transporte de carga del robot -peso máximo a levantar-. Esta capacidad de carga debe especificarse bajo la condición de que el brazo del robot esté en su posición más débil. En el caso de una configuración de brazo articulado, esto significaría que el brazo del robot esté en la extensión máxima y ubicado horizontalmente. Lo mismo que en el caso de un ser humano, es más difícil elevar una carga pesada con brazos completamente extendidos que cuando los brazos están cercanos al cuerpo. En este caso, un valor mayor de carga útil es deseable. Los pequeños robots de montaje, impulsados eléctricamente tienen capacidades de transporte de peso en el orden de 1 libra hasta 5 libras (Brazo Robot Maker 110).

La velocidad lineal del brazo es la rapidez de la garra medida en unidades de longitud por segundo. En un brazo articulado, la máxima velocidad lineal se obtiene cuando el brazo está extendido, es decir la garra se encuentra a la distancia máxima del eje vertical del robot. Es deseable un valor alto de velocidad, ya que de esta manera se minimiza el tiempo para realizar un ciclo de trabajo.

Definidos los términos anteriores , lo ideal sería que cada uno de ellos tuviese el mejor valor deseable. Esto no es posible, en vista que hay diversas limitantes. Entre éstas, se encuentra el hecho que para mejorar un parámetro se sacrifica otro. Por ejemplo, no se podría esperar mover cargas grandes a la misma velocidad que se mueven las cargas livianas. A mayor peso, la velocidad de movimiento es menor. Lo mismo sucede entre el peso y la precisión, entre mayor sea el peso máximo a mover, la precisión en la ubicación de la garra se pierde.

Entre otras limitantes, está la potencia y velocidad máxima entregada por los motores de paso que hay disponibles en el mercado local.

Por tanto, se deben definir las especificaciones teniendo en cuenta estas limitantes. A continuación se da una tabla con las especificaciones del brazo robot desarrollado. Estas especificaciones fueron obtenidas a partir de modelos desarrollados en el campo de la robótica, y se adaptan para brazos robóticos pequeños impulsados eléctricamente[43].

Las especificaciones dadas en el proceso diseño a desarrollar son las siguientes:

Resolución Espacial ≤ 5 mm
Máxima Carga útil ≥ 5 lbs
Velocidad máxima en la garra ≥ 10 cms/seg

3.8 CALCULO DE PARAMETROS DEL DISEÑO MECANICO

En esta sección, a partir de las especificaciones dadas, se definirán ciertos parámetros de diseño, que se utilizarán para dimensionar la estructura mecánica del brazo.

El proceso de diseño debería partir de las especificaciones dadas y de las dimensiones deseadas del brazo, obteniéndose mediante cálculo la potencia necesaria de los motores a utilizar. En este caso particular se variará ligeramente el proceso, debido principalmente a que los motores de paso ya se tienen -donación-, lo cual nos limita las dimensiones del brazo.

Las dimensiones de las piezas de la estructura mecánica a construir están limitados por el torque máximo obtenido de los motores, ya que no podemos diseñar una estructura con piezas del tamaño del ser humano porque carecemos de los recursos idóneos como los motores de paso con capacidad de mover grandes pesos. El sistema robótico usará cuatro motores de paso; un motor de paso será usado para mover toda la estructura del brazo -base-, un segundo motor de paso moverá el antebrazo, un tercero moverá el brazo y el último se encargara de abrir y cerrar la garra. Los parámetros de estos motores, necesarios para el diseño se describen a continuación:

---

[43] Engelber, Joseph F. Robótica Industrial. Edit. McGraw Hill, España, 1a. Edición, 1989.

TABLA 3.1 Características de motores de paso usados

MOTOR DE PASO	POTEN. [Watt]	VOLTAJE [Volt]	IMPEDANCIA [Ω]	PASO [°]	VELOC. [rpm]
ANTEBRAZO	4	12	36	7.5	48
BRAZO	4	12	36	7.5	48
GARRA	2.8	14	70	7.5	48
BASE	2.89	9	28	7.5	48

La velocidad que aparece descrita, es la velocidad máxima a la cual se trabajará cada motor. El valor máximo típico de la velocidad en los motores de paso oscila entre 60 y 80 rpm. Si se intenta obtener una mayor velocidad, el motor ya no responde avanzando un paso por cada pulso de entrada, y puede perder o ganar pasos en relación a la cantidad de pulsos que lleguen. En el caso presente, la velocidad máxima de los motores de paso es desconocida; pero para motores de igual potencia y tamaño se ha encontrado -en material bibliográfico- que el valor máximo de velocidad es 60 rpm. Para introducir un margen de seguridad se tomará un valor que corresponde al 80% de ésta, es decir 48 rpm.

A fin de definir los parámetros de diseño, se hará un análisis estático de cada enlace, el cual considerará las fuerzas que se ven involucradas en el diseño de las partes del brazo robot.

3.8.1 ANALISIS ESTATICO DEL ANTEBRAZO

Con este análisis se persigue que a partir de los elementos que se poseen -motores de paso con potencia definida-, dar dimensiones a los elementos que constituyen la estructura física del brazo: los enlaces -elementos rígidos que unen dos articulaciones- y el sistema de transmisión.

La nomenclatura siguiente se emplea para representar las variables involucradas en el análisis estático del antebrazo.

Pent = Potencia entregada por el motor en [Watts].
Nm = Velocidad angular del motor [rpm].
Nh = Velocidad angular del antebrazo [rpm]
Th = Par de entrada en el hombro [Nt-mt]
Tm = Par de entrada dado por el motor [Nt-mt]
V = Velocidad lineal [mt/seg].
W = Carga útil a levantar [lbs] o [Nt].
Wa = Peso del antebrazo [lbs] o [nt].
Wb = Peso del brazo [lbs] o [nt].
Wg = Peso de la garra [lbs] o [nt].
S = Resolución o desplazamiento del extremo del brazo [cm].

θm = Es el paso del motor dado en [grados].
θs = Es el paso de salida [grados, radianes o revoluciones].
θ1 = Angulo del antebrazo en grados.
θ2 = Angulo del brazo en grados
La = Longitud del antebrazo [cm].
Lb = Longitud del brazo [cm].
Lg = Longitud de la garra [cm].

En la figura 3.4 se detallan las variables anteriores.

El siguiente análisis tiene como objetivo obtener funciones que definan el peso máximo W_{max}, la resolución S y la velocidad máxima lineal que se desea V_{max}. Cuando se hayan obtenido dichas funciones, se evalúan las especificaciones dadas, con lo cual se obtienen los parámetros necesarios que definen las dimensiones de la estructura del brazo.

Calculo del Peso Máximo W_{max}

Al analizar este enlace (antebrazo) se tienen en cuenta fuerzas gravitatorias, ejercidas por el peso a levantar, el peso del antebrazo, brazo y garra, cómo se indica en la figura 3.4. A continuación se obtienen las diferentes ecuaciones del análisis estático.

Haciendo suma de momentos en el hombro [h], encontramos el par de salida del hombro que relaciona longitudes y pesos.

$$Th = Tc + Wa(La/2)\cos(\theta 1) \qquad (3.0)$$

$$Tc = Wb[La*\cos(\theta 1)+(Lb/2)\cos(\theta 2)]+ (Wg+W)[La*\cos(\theta 1)+(Lb+Lg)\cos(\theta 2)] \qquad (3.1)$$

sustituyendo (3.1.) en (3.0.)

$$Th = \cos(\theta 1)[La((Wa/2)+Wb+Wg)+W(La)]+ \cos(\theta 2)[Lb((Wb/2)+Wg)+Wg(Lg)+W(Lb+Lg)] \qquad (3.2)$$

Considerando el peor caso, ocurre cuando el brazo esta extendido horizontalmente, θ1 = θ2 = 0 grados.

$$\text{Entonces} \quad Th = La(Wa/2+Wb+Wg) + Lb(Wb/2+wg) + Wg(Lg) + W(La+Lb+Lg) \qquad (3.3)$$

$$\text{y el } W = \frac{Th - La(Wa/2+Wb+Wg) - Lb(Wb/2+Wg) - Wg(Lg)}{La+Lb+Lg}$$

Según la sección 3.1, el mayor volumen de trabajo en un brazo articulado se obtiene cuando las longitudes del antebrazo y brazo son iguales o casi similares. Según este criterio, asumiremos La = Lb. Reduciendo y agrupando términos obtenemos:

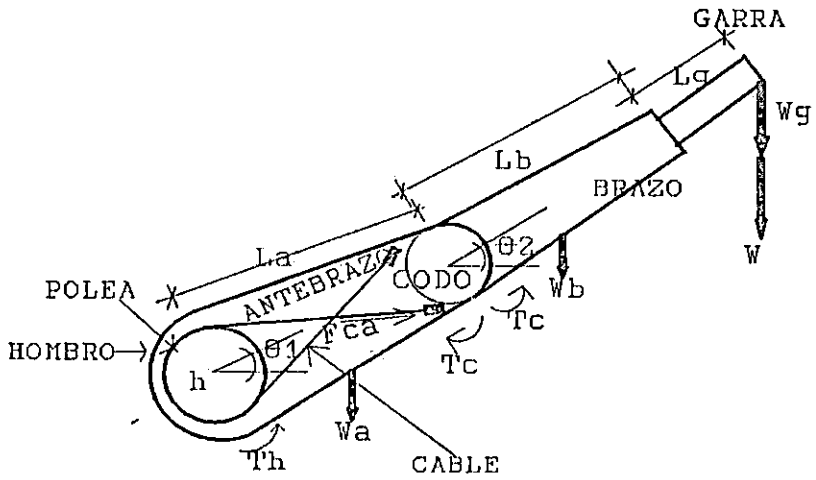


Figura 3.4 Fuerzas y pares del antebrazo

$$W = \frac{Th}{2La+Lg} - \left(\frac{La(Wa/2+3Wb/2)}{2La+Lg} + Wg\right) \qquad (3.3)$$

De la Ec. 3.3, se observa que el peso el cual levantará el brazo está determinado por el torque entregado por la fuente de impulsión (Th) menos el efecto producido por el peso de la estructura del brazo.

Para obtener una función del peso de mayor utilidad, se pueden realizar las siguientes simplificaciones:

1) En primer lugar, el peso de la garra se puede despreciar en comparación con el peso global de la estructura. Al final hay que tener presente, que la carga útil máxima obtenida W debe incluir el peso de la garra.

2) Se puede establecer una relación entre las longitudes, de tal forma que La, Lb y Lg, sean función de la longitud total del brazo, es decir La+Lb+Lg. Según lo anterior, una relación aceptable sería:

$$La = 0.4L \quad \text{(longitud del antebrazo)}$$

$$Lb = 0.4L \quad \text{(longitud del brazo)}$$

$$Lg = 0.2L \quad \text{(longitud de la garra)}$$

donde $$L = La + Lb + Lg$$

En base a las simplificaciones anteriores, el peso que se podría levantar, quedaría expresado como:

$$W = \frac{Th}{L} - \frac{0.4L(Wa/2+3Wb/2)}{L}$$

Simplificando aún más nos quedaría:

$$W = \frac{Th}{L} - 0.4\ (Wa/2+3Wb/2) \qquad (3.4)$$

donde L es la longitud total desde el hombro hasta el extremo de la garra.

El par Th tiene una relación directa con la potencia del motor que impulsa el antebrazo. El torque del motor Tm guarda una relación directa con el torque Th, de tal forma que se puede establecer:

$$\frac{Th}{Tm} = f$$

donde f lo conoceremos como el factor de amplificación del hombro, ya que amplifica el torque del motor hasta llegar al torque del hombro, y está definido por la relación geométrica de los diámetros de poleas y engranes que transmiten el torque desde el motor hasta el hombro. Hay que notar que f es un factor adimensional.

El factor f también define las siguientes relaciones:

$$f = \frac{Nm}{Nh} = \frac{\theta m}{\theta s}$$

Donde: Nm y θm son la velocidad y paso del motor
Nh y θs son la velocidad y paso en el antebrazo.

Calculando el torque en el hombro Th a partir de la potencia entregada por el motor del antebrazo:

$$Th = f \cdot Tm = f \cdot (Pent/\omega_m) \qquad (3.5)$$

De la tabla 3.1, Pent = 4 Watts y la ω_m (velocidad del motor en rad/seg) sería:

$$\omega_m = Nm \cdot 2\pi \frac{rad}{rev} \cdot \frac{1\ min}{60\ seg} = 0.10472 \cdot Nm \qquad (3.6)$$

Sustituyendo la Pent y ω_m de la Ec. 3.6 en la Ec. 3.5, obtenemos:

$$Th = f \cdot 4/(0.10472 \cdot Nm)$$

$$Th = 38.2 \cdot f/Nm$$

El valor de Nm es la velocidad del motor en rpm. El torque Th se debe calcular para el peor de los casos, es decir cuando la velocidad del motor sea máxima (ya que en este momento el torque entregado es mínimo), la expresión final para Th nos quedaría: (para Nm = 48rpm)

$$Th = 38.2 \cdot f/48 = 0.8 \cdot f \qquad (3.7)$$

Sustituyendo el valor de Th en la Ec. 3.4, el W_{max} que el brazo extendido levantaría es:

$$W_{max} = \frac{0.8 \cdot f}{L} - 0.4\ (Wa/2+3Wb/2) \qquad (3.8)$$

Cálculo de la Resolución S.

La resolución se calculará cuando el brazo se encuentra extendido, ya que en ese punto el incremento por cada paso del motor es mayor. Esto se debe a que la resolución es proporcional a la distancia entre el centro del brazo (hombro) y la posición de la garra.

El cálculo de la resolución involucra las variables de la longitud total del brazo L y el paso del motor θm. El motor de paso se caracteriza porque cada vez que gira, lo hace en múltiplos de un ángulo, denominado paso del motor, el cual está dado en grados; este valor limita la resolución. El paso del motor es θm = 7.5°, es decir que gira en pasos de 7.5°; en el hombro este paso se ve afectado por el factor de amplificación que se da entre los elementos de transmisión entre el motor y el hombro. Dicha relación sería:

$$f = \theta m / \theta s$$

$$\theta s = \theta m / f$$

Como en el extremo del brazo, la resolución corresponde a un arco recorrido por el ángulo θs con un radio L, podemos establecer:

$$S = \theta s \cdot L$$

Donde el paso del motor θs esta dado en radianes. Relacionando ambas ecuaciones, obtenemos:

$$S = (\theta m/f) \cdot L = L \cdot \theta m/f$$

Hay que recordar que θm se encuentra en radianes. Evaluando la Ec. para θm, obtenemos:

$$S = L \cdot (7.5° \cdot \pi / 180)/f$$

La relación final obtenida es:

$$S = 0.1309 \cdot \frac{L}{f} \qquad (3.9)$$

Cálculo de la Velocidad Lineal.

La velocidad se calculará en función del desplazamiento al extremo de la garra cuando recorre un arco con el brazo extendido. La relación sería:

$$V = X / t \qquad (3.10)$$

X en metros y t en segundos.

Ya que X es un arco, cumple la siguiente relación:

$$X = \theta s \cdot L \qquad \theta s \text{ en radianes} \qquad (3.11)$$

El valor de t se puede obtener a partir de los parámetros del motor:

$$t = \frac{\theta m}{\omega_m} = \frac{\theta m}{(2\pi/60)\ Nm} \qquad (3.12)$$

donde θm está en radianes y Nm en rpm.

De las ecuaciones 3.10, 3.11 y 3.12, obtenemos:

$$V = \frac{\theta s \cdot L}{\frac{\theta m}{(2\pi/60)\ Nm}}$$

Simplificando, y recordando que f=θm/θs (θm y θs en rad)

$$V = \frac{2\pi \cdot Nm \cdot L}{60 \cdot f}$$

La velocidad máxima lineal se obtiene cuando la velocidad del motor es máxima, es decir Nm = 48 rpm. Evaluando para este caso, obtenemos:

$$V = 5.02656 \cdot (L/f) \qquad (3.13)$$

3.8.2 INTERPRETACION DEL ANALISIS ESTATICO DEL ANTEBRAZO

En la sección anterior se definieron el peso, la resolución y velocidad lineal en función de ciertos parámetros. Recordaremos dichas ecuaciones:

$$W = \frac{0.8 \cdot f}{L} - 0.4\ (Wa/2+3Wb/2) \qquad (3.8)$$

$$S = 0.1309 \cdot \frac{L}{f} \qquad (3.9)$$

$$V = 5.02656 \cdot (L/f) \qquad (3.13)$$

Hay que enfatizar la importancia de los parámetros comunes en las tres ecuaciones.

El primer parámetro, f, se definió como la amplificación que sufre el torque desde el motor hasta el punto de aplicación, el hombro.

El segundo parámetro, L, es la longitud global del brazo, desde el hombro hasta el extremo de la garra.

Existen dos parámetros que no son comunes, Wa y Wb, y que representan el efecto del peso estructural del brazo. El peso de la estructura del brazo robot consta de dos partes: el peso de cada miembro o enlace y el peso del resto de elementos, como lo son engranes, poleas, cables, ejes, sensores, pernos, etc. El peso de cada miembro es proporcional a la longitud global del brazo, pero en el caso del resto de elementos no hay una relación entre longitud y peso, no pudiendo establecer una relación adecuada del peso Wa y Wb en función de L.

Para solventar dicho problema, en el Anexo A se hace un análisis que establece una relación entre la longitud y el peso Wa y Wb, con lo cual obtenemos una función cuadrática en L. Basados en dicho análisis, asumiremos que para un rango de longitud de $0 \leq L \leq 0.5$ mts, el peso máximo que puede llegar a tener la estructura es de 2 lbs, distribuido de la siguiente manera:

$$Wa = 1\ lb = 4.45\ Newton$$

$$Wb = 1\ lb = 4.45\ Newton$$

Sustituyendo dichos valores en la Ec. 3.8, obtenemos la

expresión del peso:

$$W = \frac{0.8 \cdot f}{L} - 3.56 \qquad (3.14)$$

Retomando las especificaciones dadas en la sección 3.7, se pueden replantear las Ecs. 3.14, 3.9 y 3.13 como inecuaciones:

$$W = \frac{0.8 \cdot f}{L} - 3.56 \geq 22.25 \text{ N} \quad [5 \text{ lbs.}] \qquad (3.14.a)$$

$$S = 0.1309 \cdot \frac{L}{f} \leq 0.005 \text{ mts} \quad [5 \text{ mms}] \qquad (3.9.a)$$

$$V = 5.02656 \cdot (L/f) \geq 0.1 \text{ m/seg} \quad [10\text{cm/seg}] \qquad (3.13.a)$$

Despejando el factor f/L en las Ecs. 3.14.a, 3.9.a. y 3.13.a, y siguiendo las reglas de las inecuaciones, llegamos a obtener para cada ecuación, las siguientes relaciones:

$$f/L \geq 32.3 \quad [\text{dada por } W_{max}] \qquad (3.14.b)$$

$$f/L \geq 26.2 \quad [\text{dada por } S_{max}] \qquad (3.9.b)$$

y

$$f/L \leq 50.3 \quad [\text{dada por } V_{min}] \qquad (3.13.b)$$

A partir de estas inecuaciones, definimos un rango de diseño, el cual es:

$$32.3 \leq \frac{f}{L} \leq 50.3$$

El límite inferior del rango lo define el peso máximo a levantar el brazo, y se constituye en uno de los factores más críticos. Este límite puede variar en la práctica por las consideraciones realizadas al calcular el peso. Si el peso de la estructura es mayor al asumido en el análisis, este límite tendería a aumentar, disminuyendo el rango de diseño obtenido.

El límite superior está definido por la velocidad lineal del brazo en su extremo. Si deseáramos una velocidad mayor a la especificada, el valor de este límite tendería a reducirse, disminuyendo el rango de diseño obtenido.

En suma, la tendencia será alejarnos del límite inferior del rango definido, ya que si el peso máximo es sobrepasado, el motor no entregaría el torque necesario y el sistema se bloquearía o respondería en forma errónea.

En cambio, si nos acercamos al límite superior, la repercusión

en el sistema es que el brazo se movería a una menor velocidad a la especificada.

Aparentemente podríamos darle cualquier longitud L al brazo, en tanto que se aumentase el factor f. En la práctica no es posible, ya que la longitud está limitada principalmente por el peso de la estructura. A mayor longitud, el peso de la estructura sería tal, que los motores no podrían levantar nada más que el brazo.

Si tenemos una longitud de L = 35 cms = 0.35 mts, el valor de f podría variar de la siguiente manera:

$$11.27 \leq f \leq 17.61$$

con un valor de f = 17, obtendríamos:

$$f/L = 48.57$$

el cual es un valor aceptable y que cumple la característica de estar en el rango de diseño y alejarse del límite inferior. Con tales parámetros, las dimensiones del brazo están definidas. Las longitudes obtenidas son:

La = 0.4L = 14 cms (longitud del antebrazo)

Lb = 0.4L = 14 cms (longitud del brazo)

Lg = 0.2L = 7 cms (longitud de la garra)

El peso, la resolución y la velocidad que el sistema tendría se calculan a partir de las Ecs. 3.14.a, 3.9.a y 3.13.a, y dan los siguiente resultados:

W = 35.3 Newton > 22.25 N [5 lbs.]

S = 0.0027 mts < 0.005 mts [5 mms]

V = 0.103 mt/seg > 0.1 m/seg [10cm/seg]

Con el valor de f obtenido, el sistema de transmisión se puede definir, de tal forma que tenga la limitante para el valor de f. El sistema de transmisión desarrollado en la práctica, se muestra en la figura 3.5.

El factor f se define por la relación de amplificación proporcionada por el sistema de transmisión. Cuando es una polea la involucrada, el valor necesario para obtener la amplificación es el radio; no así en los piñones, donde el valor que se utiliza es el número de dientes que posee. De la

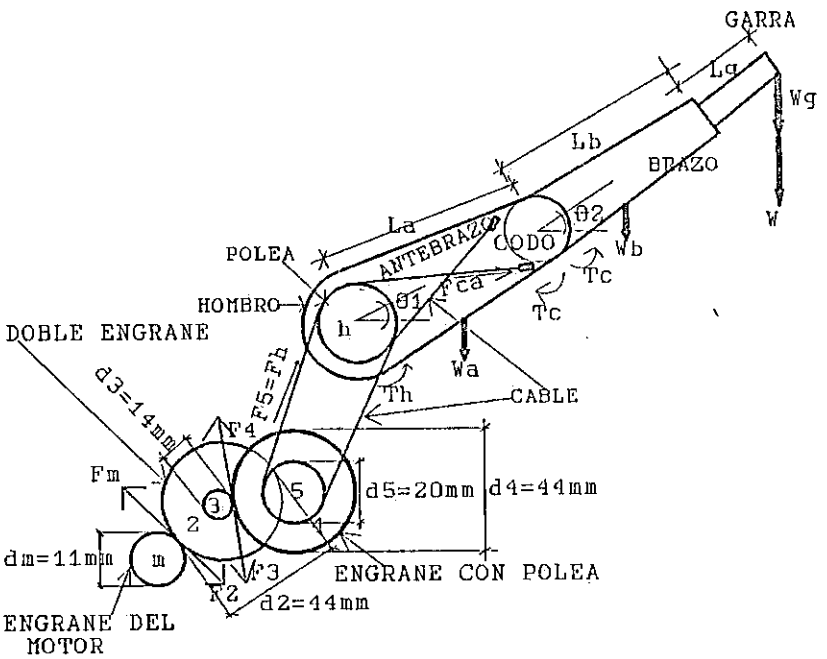


Figura 3.5 Sistema de transmisión de torque del motor al hombro

figura 3.5, el factor de amplificación f sería:

$$f = \frac{Z_2 \cdot Z_4 \cdot r_h}{Z_m \cdot Z_3 \cdot r_5} \qquad (3.15)$$

Donde los valores son:

Z_2 = 86, número de dientes del piñón 2.

Z_4 = 86, número de dientes del piñón 4.

r_h = 12mm, diámetro de la polea del hombro (h)

$Z_m = Z_1$ = 20, número de dientes del piñón del motor

Z_3 = 26, número de dientes del piñón 3.

r_5 = 10mm, diámetro de la polea 5.

En siguientes secciones de este capítulo, se obtendrán las relaciones para el cálculo del número de dientes de los piñones de la expresión anterior. Sustituyendo estos valores en dicha expresión (Ec. 3.15), el valor de f obtenido es:

$$f = 17.06769 \approx 17$$

Cuyo valor cumple los requerimientos de diseño especificados inicialmente.

3.8.3. ANALISIS ESTATICO DEL BRAZO.

El procedimiento es similar al caso anterior. Al analizar este enlace (brazo) se desea obtener las dimensiones del sistema de transmisión que controlará el brazo. La dimensión del brazo ya fue definida en la sección anterior, y se llegó a que la longitud del brazo era Lb = 14 cms.

Para facilitar el análisis, se procederá estableciendo como punto de referencia la articulación (codo) que une el brazo y antebrazo.

A continuación se obtienen funciones que definen el peso máximo W_{max}, la resolución S y la velocidad máxima lineal que se desea V_{max}, en función de los parámetro del brazo. Luego, se evaluarán las funciones con las especificaciones dadas, y se obtienen los parámetros necesarios que definen las dimensiones del sistema de transmisión del brazo.

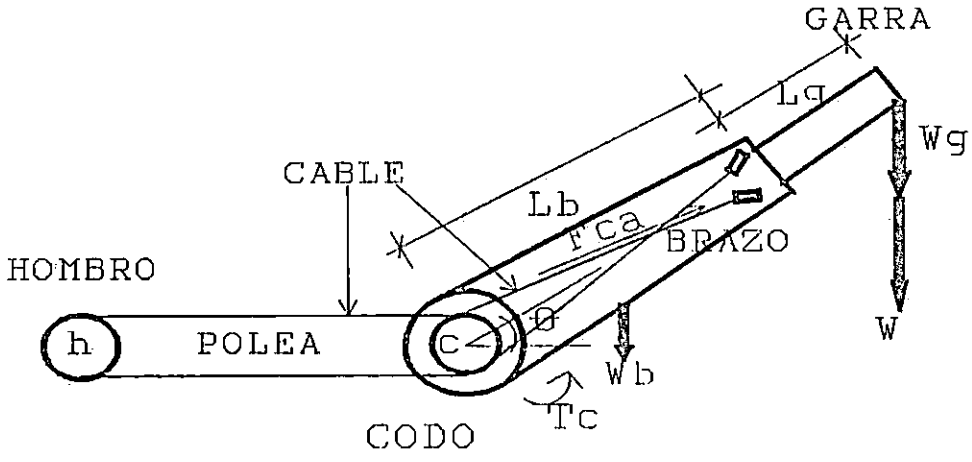


Figura 3.6 Diagrama del brazo para el análisis estático

Cálculo del peso.

En base a la figura 3.6, se hace suma de momentos en el codo (c), encontramos el par de salida del codo que relaciona longitudes y pesos.

$$Tc = Wb \cdot (Lb/2) \cdot \cos(\theta) + (Wg+W) \cdot (Lb+Lg) \cdot \cos(\theta)$$

Al considerar el peor caso, este ocurre cuando el brazo esta extendido horizontalmente, $\theta = 0°$. Entonces

$$Tc = Wb \cdot (Lb/2) + Wg \cdot (Lb+Lg) + W \cdot (Lb+Lg) \qquad (3.15)$$

y el

$$W = \frac{Tc - Lb \cdot Wb/2 - Wg \cdot (Lg+Lg)}{Lb+Lg}$$

agrupando, nos queda:

$$W = \frac{Tc - Lb \cdot Wb/2}{Lb+Lg} - Wg$$

Al igual que en el caso del antebrazo, el peso de la garra se puede despreciar en comparación con el peso global de la estructura. Al final hay que tener presente, que la carga útil máxima obtenida W debe incluir el peso de la garra.

$$W = \frac{Tc}{Lb+Lg} - \frac{Lb \cdot Wb/2}{Lb+Lg}$$

Las longitudes Lb y Lg se determinaron en la sección anterior, así como el peso Wb ya se había definido:

Wb = 1 lb = 4.45 N,

Lb = 14 cms

Lg = 7 cms.

Evaluando en la ecuación anterior con los valores dados, la expresión final del peso W obtenida es:

$$W = 4.762 \cdot Tc - 1.483 \qquad (3.16)$$

El par Tc tiene una relación directa con la potencia del motor que impulsa el antebrazo. El torque del motor Tm guarda una relación directa con el torque Tc, de tal forma que se puede establecer:

$$\frac{Tc}{Tm} = f'$$

donde f' lo conoceremos como el factor de amplificación del brazo, ya que amplifica el torque del motor hasta llegar al torque del codo, y está definido por la relación geométrica de los diámetros de poleas y engranes que transmiten el torque desde el motor hasta el codo. Hay que notar que f' es un

factor adimensional.

El factor f' también define las siguientes relaciones:

$$f' = \frac{Nm}{Nc} = \frac{\theta m}{\theta c}$$

Donde: Nm y θm son la velocidad y ángulo del motor
Nc y θc son la velocidad y ángulo en el brazo.

Considerando que la potencia del motor del brazo es idéntica a la del antebrazo, y las velocidades angulares máximas consideradas para ambos motores son iguales, el torque en el hombro Tc a partir de la potencia entregada por el motor del antebrazo satisface la Ec. 3.7, obtenida en el análisis del antebrazo:

$$Tc = 38.2 \cdot f'/48 = 0.8 \cdot f' \qquad (3.17)$$

Sustituyendo el valor de Tc en la Ec. 3.16, el W que el brazo levantaría es:

$$W = 3.8095 \cdot f' - 1.483 \qquad (3.18)$$

Cálculo de la Resolución S.

El cálculo de la resolución involucra las variables de la longitud del brazo y la garra, Lb+Lg, y el paso del motor θm. El paso del motor es θm = 7.5°; en el codo este paso se ve afectado por el factor de amplificación que se da entre los elementos de transmisión entre el motor y el hombro. Dicha relación sería:

$$f' = \theta m / \theta c$$

$$\theta c = \theta m / f'$$

En el caso del antebrazo, las condiciones que se tenían era un paso θm = 7.5°, y una longitud total L. En nuestro caso el paso es igual, y la longitud es Lb+Lg.
La Ec. 3.9 del análisis estático del antebrazo es válida con una leve modificación:

$$S = 0.1309 \cdot \frac{Lb+Lg}{f'}$$

Sustituyendo Lb y Lg, obtendríamos:

$$S = 0.0275/f' \qquad (3.19)$$

Cálculo de la Velocidad Lineal.

La velocidad se calculará en función del desplazamiento de la muñeca, cuando recorre un arco θm. La relación sería:

$$V = X / t \qquad (3.20)$$

X en metros y t en segundos.

Ya que X es un arco, cumple la siguiente relación:

$$X = \theta c \cdot (Lb+Lg) \quad \theta c \text{ en radianes} \qquad (3.21)$$

El valor de t se puede obtener a partir de los parámetros del motor:

$$t = \frac{\theta m}{\omega_m} = \frac{\theta m}{(2\pi/60)\ Nm} \qquad (3.22)$$

donde θm está en radianes y Nm en rpm. De las ecuaciones 3.20, 3.21 y 3.22, obtenemos:

$$V = \frac{\theta c \cdot (Lb+Lg)}{\frac{\theta m}{(2\pi/60)\ Nm}}$$

Simplificando, y recordando que f'=θm/θc (θm y θc en rad)

$$V = \frac{2\pi \cdot Nm \cdot (Lb+Lg)}{60 \cdot f'}$$

La velocidad máxima lineal se obtiene cuando la velocidad del motor es máxima, es decir Nm = 48 rpm. Evaluando para este caso, obtenemos:

$$V = 1.0556 / f' \qquad (3.23)$$

3.8.4 INTERPRETACION DEL ANALISIS ESTATICO DEL BRAZO

Las Ecs. 3.18, 3.19 y 3.23 describen el peso, resolución y velocidad que tendrá el brazo en función del parámetro f', que determina la amplificación que debe proporcionar el sistema de transmisión. Estas ecuaciones se replantean a continuación como inecuaciones en función de las especificaciones dadas en la sección 3.7:

$$W = 3.8095 \cdot f' - 1.483 \geq 22.25\ N \quad [5\ lbs.] \qquad (3.18.a)$$

$$S = 0.0275 / f' \qquad \leq \ 0.005 \text{ mts [5 mms]} \qquad (3.19.a)$$

$$V = 1.0556 / f' \qquad \geq \ 0.1 \text{ m/seg [10cm/seg]} \qquad (3.23.a)$$

Despejando el factor f' en las Ecs. 3.18.a, 3.19.a. y 3.23.a, y siguiendo las reglas de las inecuaciones, llegamos a obtener para cada ecuación, las siguientes relaciones:

$$f' \geq 6.23 \quad [\text{dada por } W_{max}] \qquad (3.18.b)$$

$$f' \geq 5.50 \quad [\text{dada por } S_{max}] \qquad (3.19.b)$$

y

$$f' \leq 10.56 \quad [\text{dada por } V_{min}] \qquad (3.23.b)$$

A partir de estas inecuaciones, definimos un rango de diseño, el cual es:

$$6.23 \leq f' \leq 10.56$$

El límite inferior del rango lo define el peso máximo a levantar el brazo, y se constituye en uno de los factores más críticos. Este límite puede variar en la práctica por las consideraciones realizadas al calcular el peso. Si el peso de la estructura es mayor al asumido en el análisis, este límite tendería a aumentar, disminuyendo el rango de diseño obtenido.

El límite superior está definido por la velocidad lineal del brazo en su extremo. Si deseáramos una velocidad mayor a la especificada, el valor de este límite tendería a reducirse, disminuyendo el rango de diseño obtenido.

En suma, la tendencia será alejarnos del límite inferior, ya que si el peso máximo es sobrepasado, los motores no entregarían el torque necesario y el sistema se bloquearía o respondería en forma errónea.

Con el rango de f' obtenido, el sistema de transmisión se puede definir, de tal forma que cumpla la limitante para el valor de f'. El sistema de transmisión desarrollado en la práctica, se muestra en la figura 3.7.

El factor f' define la relación de amplificación proporcionada por el sistema de transmisión, y se constituye en la relación de diametros -radios- y número de dientes de las poleas y piñones encontrados en el sistema de transmisión. De la figura 3.7, el factor f' se define como:

$$f' = \frac{Z_2 \cdot r_c}{Z_m \cdot r_3} \qquad (3.24)$$

Donde los valores son:

Z_2 = 86, número de dientes del piñon 2.

r_c = 21mm, diámetro de la polea del codo (c)

$Z_m = Z_1$ = 20, número de dientes del piñon del motor

r_3 = 10mm, diámetro de la polea 3

Evaluando en Ec. 3.24, el valor obtenido es f' = 9.03, el cual es un valor aceptable ya que cumple la caracteristica de estar en el rango de diseño y alejarse del límite inferior.

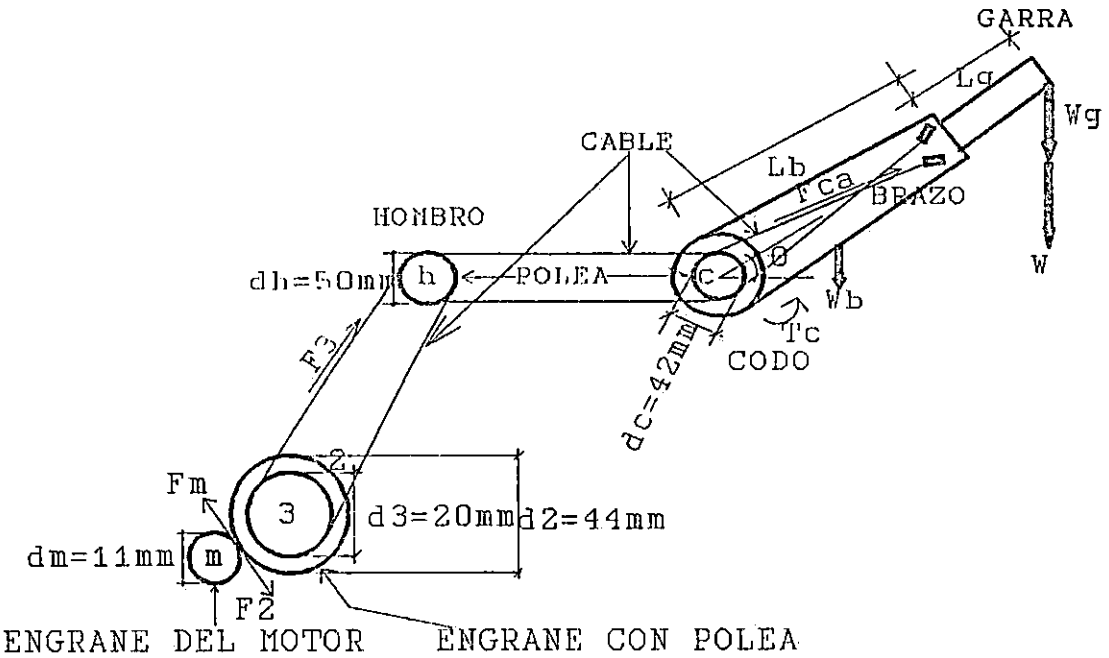


Figura 3.7 Sistema de transmisión de torque del motor al brazo

El peso, la resolución y la velocidad que el sistema tendría se calculan a partir de las Ecs. 3.18.a, 3.19.a y 3.23.a, y dan los siguiente resultados:

W =	32.9 Newton	>	22.25 N	[5 lbs.]
S =	0.0030 mts	<	0.005 mts	[5 mms]
V =	0.117 mt/seg	>	0.1 m/seg	[10cm/seg]

El cálculo del número de dientes de cada piñon se vera en secciones posteriores.

3.8.5 ANALISIS ESTATICO DE LA BASE

La base soportará toda la estructura del brazo, como lo son los motores, poleas, piñones, pesos de los enlaces, carga, planchas de aluminio, etc. En el movimiento de la base no existen cargas gravitacionales involucradas, ya que únicamente se gira el brazo alrededor de un eje vertical. Descrito de tal manera, el motor que impulse la base debe poseer un torque que venza la inercia total del brazo. La configuración que tendrá la base se muestra en la figura 3.8.

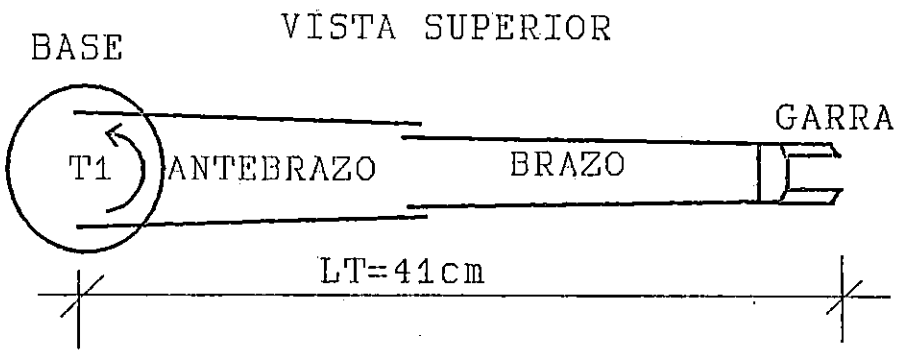


Figura 3.8. Configuración de la base y el brazo.

Calculo del Torque.

El siguiente análisis tiene como objetivo determinar el torque necesario en la base para que todo el sistema gire sobre un eje vertical. Luego de ellos, considerando el torque máximo entregado por el motor, se calcula la amplificación que debe sufrir el torque del motor, de tal forma que rote la base a una aceleración especificada.

El problema fundamental al realizar este cálculo, es que, la inercia del sistema global se calcula a partir de la masa del brazo -que incluye la masa de la estructura y de la carga que está transportando-, y la distancia que hay entre ésta y el punto sobre el que rota el brazo. En el caso del brazo robot, la masa y su distancia al centro varía, dependiendo de la posición de la carga. Por lo tanto el calculo dará valores distintos para diferentes posiciones del brazo. Una forma de resolver este problema es considerando el peor de los casos,

el cual se da cuando el motor debe manejar la máxima inercia o sea cuando el brazo esta en su máxima extensión.

La relación que existe entre la aceleración angular $d^2\theta/dt^2$ y la aceleración lineal d^2X/dt^2 a una distancia r del centro, se define de la siguiente manera:

$$\frac{d^2X}{dt^2} = r \frac{d^2\theta}{dt^2} \qquad (3.25)$$

El momento inercial de una carga-puntual está definida de la siguiente manera:

$$J = m \cdot r^2 \qquad (3.26)$$

donde m: masa de la carga puntual

r: distancia de la masa al eje de rotación

J: inercia rotacional que presenta la masa m

Si consideramos despreciable la inercia que presenta la estructura, la única inercia que el motor manejará es la debida a la masa de la carga y la garra.

En este caso, la máxima inercia que el motor debe manejar es cuando el brazo se encuentra totalmente extendido, siendo éste el peor caso.

El cálculo del torque para impulsar la inercia J, se obtiene de:

$$T_1 = J_T \cdot \frac{d^2X}{dt^2} \cdot \frac{1}{L_T} \qquad (3.27)$$

Donde J_T es la inercia total del sistema, d^2X/dt^2 es la aceleración máxima que puede tener la carga y garra (se considerará una aceleración máxima de 4 m/s^2) y L_T es la longitud total del brazo.

Si se observa, L_T es diferente al valor de la longitud L (L_T = 41 cms y L = 35 cms). El valor L corresponde a la longitud desde la articulación del hombro hasta el extremo de la garra, en cambio L_T es la longitud desde el centro del eje sobre el que gira la base hasta el extremo de la garra.

Para el cálculo de la inercia se hace uso de las dimensiones del brazo, así como la masa que tiene. En este caso se han estimados masas concentradas y radios a partir de la dimensión del brazo, calculada en las secciones anteriores. Así, para los enlaces del antebrazo y brazo , en el análisis estatico de

ambos se asumió 1 lbm, y la ubicaremos en el centro del enlace, a 0.13 y 0.27mts respectivamente. Los motores se tomarán con 4 lbm, a una distancia de 0.06 mts del centro. La garra y la carga tienen 5 lbm -especificaciones dadas- concentradas en el extremo, es decir 0.41 mts. Por último, la estructura de la base -soportes, piñones cocentrados, etc- se considerará con una masa de 2 lbm. El resumen del cálculo de la inercia se observa en la Tabla 3.2:

TABLA 3.2 Cálculo de la inercia del brazo robot

	Masa (lbm)	Masa (Kg)	Longitud (mts)	Inercia J (Kg·m2)
Enlace del brazo	1.000	0.455	0.130	0.008
Enlace del antebrazo	1.000	0.455	0.270	0.033
Motores	4.000	1.818	0.060	0.007
Garra + Carga	5.000	2.273	0.410	0.382
Base	2.000	0.909	0.060	0.003
Inercia total del sistema: J_T =				0.433

Todas las masas se han trasladado a Kg, para realizar el cálculo en sistema MKS. La columna de la inercia contiene las inercias parciales para cada elemento, así como la inercia total del sistema, $J_T = 0.433\ Kg \cdot m^2$.

Es de hacer notar que la inercia de la garra y carga a tomar representa el 88% de la inercia total.

Evaluando T_1 en 3.27, obtenemos:

$$T_1 = (0.433\ Kg \cdot m^2)(4\ m/s^2)(1/0.41mt)$$

$$T_1 = 4.22\ N \cdot m$$

Por tanto, en la articulación de la base se necesita una torque de 4.22 N·m para impulsar la garra más la carga con una aceleración de 4 m/s^2. El torque máximo que puede entregar el motor de la base está dado por la fórmula:

$$T_m = \frac{30 \cdot Pm}{\pi \cdot Nm} \qquad (3.28)$$

Sustituyendo Pm (potencia del motor, 2.89 W), Nm (velocidad máxima del motor, 48 rpm):

$$T_m = \frac{30 \cdot 2.89}{\pi \cdot 48} = 0.575 \text{ N} \cdot \text{m}$$

En este punto definiremos el factor de amplificación de torque f", como:

$$f'' = \frac{T_i}{T_m} = \frac{4.22}{0.575} = 7.34$$

Por tanto, al diseñar el sistema de transmisión se buscará un factor de transmisión $f'' \geq 7.34$

Cálculo de la Resolución.

La expresión de la resolución sería:

$$S = \theta b \cdot L_T$$

donde $\theta b = \theta m / f''$ y $L_T = 41$ cms. θm es el paso del motor en radianes

El paso del motor es:

$$\theta m = 7.5° \cdot (\pi/180°) = 0.1309 \text{ rads}$$

Sustituyendo en la Ec. de S:

$$S = (\theta m / f'') \cdot 0.41$$

$$S = 0.0537 / f'' \qquad (3.29)$$

La resolución debe cumplir la especificación $S \leq 5$ mms, por tanto se puede expresar:

$$0.0537/f'' \leq 0.005 \text{ mts}$$

Es decir: $f'' \geq 10.74$

Cálculo de la Velocidad Lineal.

El cálculo de la velocidad es idéntico a los casos del antebrazo y brazo. La expresión obtenida fue:

$$V = 5.02656 \cdot (L_T / f'') \qquad (3.30)$$

Sustiyendo el valor de L_T y planteándola como una inecuación obtenemos:

$$V = 2.061 / f'' \geq 0.1 \text{ mt/seg}$$

Es decir: $f'' \leq 20.61$

Cálculo del Sistema de Transmisión.

En la práctica, el diseño de transmisión se construyo a partir de un piñon de diámetro 152 mm con 192 dientes (Z_b) y el piñon del motor, con un diámetro de 13 mm con 14 dientes (Z_a). El sistema de transmisión se observa en la figura 3.9.

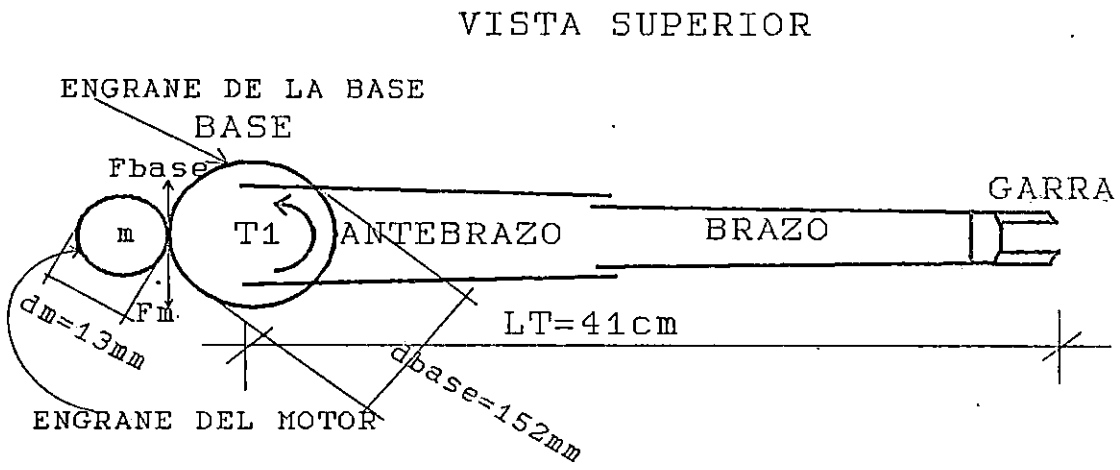


Figura 3.9. Sistema de Transmisión de la base.

El factor f" está definido por la relación de dientes de los piñones:

$$f'' = 192 / 14 = 13.71$$

Este valor f" satisface todos las condiciones que debe cumplir f":

$f'' \geq 7.34$ (dada por la inercia)

$f'' \geq 10.74$ (dada por la resolución)

$f'' \leq 20.61$ (dada por la velocidad max.)

El factor más crítico, por las estimaciones realizadas, es la inercia del sistema, cuyo valor en la práctica puede variar. Respecto a la limitante dada por la inercia ($f'' \geq 7.34$), el valor de f" obtenido es 1.8 (13.71/7.34) veces superior al valor mínimo deseado, suficiente para soportar cambios en la inercia -hasta el 80%- del sistema obtenido en la práctica.

3.8.6 ANALISIS ESTATICO DE LA GARRA

Definiciones y operaciones básicas.

Una pinza mecánica es un efector final que utiliza dedos mecánicos impulsados por un mecanismo para agarrar una pieza. Los dedos, algunas veces llamados uñas, son los accesorios de la pinza que están en contacto con la pieza.

Además, los dedos están unidos al mecanismo o son una parte integral del mismo.

En la mayoría de aplicaciones dos dedos son suficientes para sostener la pieza u otro objeto. Las pinzas con tres o más dedos son menos frecuentes.

La función del mecanismo de pinza es trasladar algo a partir del suministro de energía que origina una acción de agarre de los dedos sobre la pieza.

El mecanismo debe ser capaz de abrir y cerrar los dedos y de aplicar la fuerza suficiente contra la pieza para sostenerlo de forma segura cuando se cierren los dedos.

Existen dos formas de sostener al objeto dentro de los dedos. La primera es comprimiendo la pieza con los dedos.

En este método los dedos encierran a la pieza hasta alguna posición, limitando el movimiento de la pieza.

La figura 3.10 ilustra este método.

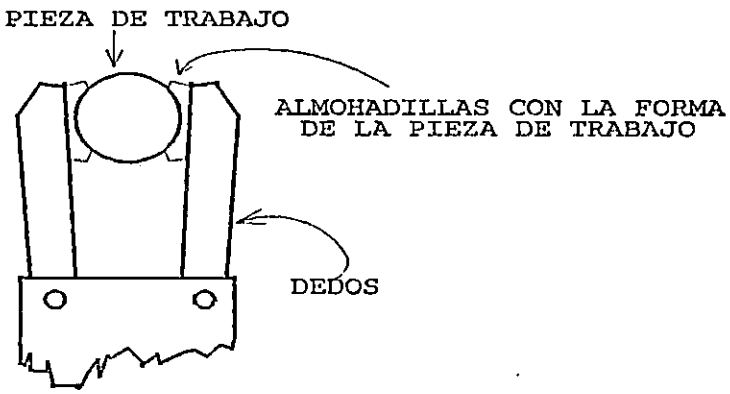


Figura 3.10. Método de encerrar la pieza.

La segunda forma de sujección de la pieza es mediante el rozamiento entre los dedos y la pieza. Con este método los dedos deben aplicar una fuerza que proporcione un rozamiento suficiente para retener la pieza en contra de la gravedad en el ciclo de trabajo. Las almohadillas o cojinetes unidos a los dedos que hacen contacto con la pieza suelen ser fabricados de

un material que es relativamente blando para permitir que la pieza se amolde al cojinete. Este se encarga de aumentar el coeficiente de rozamiento entre la pieza y la superficie de contacto de los dedos. También sirve para proteger la superficie de la pieza de posibles arañazos u otros daños. El método de rozamiento para sujetar la pieza se convierte en el diseño de dedos más simple e incluso más barato y tiende a adaptarse con gran facilidad a una gran variedad de piezas, por esta razón se elige este metodo para el diseño de la garra. La figura 3.11 ilustra este metodo. Más adelante se analizarán las fuerzas de este metodo de rozamiento.

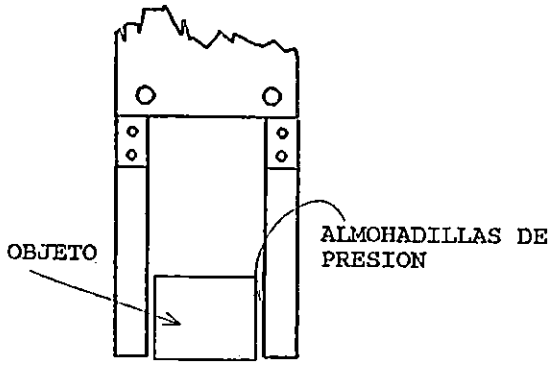


Figura 3.11. Metodo de rozamiento de los dedos contra la pieza.

Análisis de fuerzas del sistema de la garra.

La garra del sistema brazo robot tiene que ejercer una fuerza para tomar o dejar la pieza a mover, entonces es vital saber que fuerza es la requerida por los dedos o pinzas de la garra para hacer esta operación. El peso máximo que deberá sujetar los dedos es de 5 lbs, partiendo de esta limitante se encuentran las diferentes fuerzas involucradas en dicho proceso.

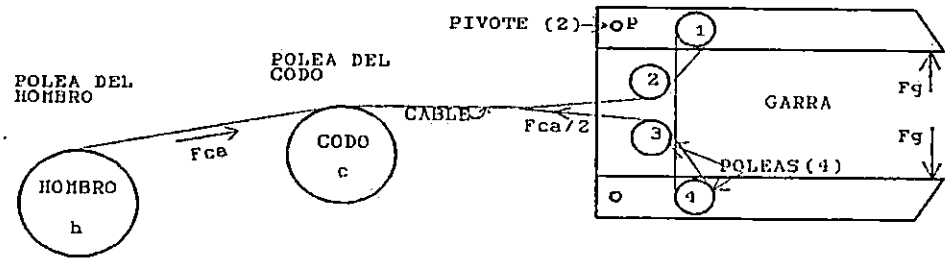


Figura 3.12. Fuerzas de contacto Fg y de tensión Fca.

Estas fuerzas corresponden, ver figura 3.12, a la fuerza de

tensión necesaria para cerrar los dedos de la garra Fca, y la fuerza de contacto (debida a la fricción) entre la pieza a sujetar y los dedos, Fg.

Al observar la garra se muestra como la simetría de los dedos puede utilizarse de forma ventajosa de modo que solo se tenga que considerar la mitad del mecanismo ver figura 3.13.

Hallando expresión de la fuerza ejercida por el cable necesaria para agarrar la pieza, Fca, referirse a la figura 3.13.

Haciendo momento en el pivote P.

$$(Fca/2)\ \cos(\theta) \cdot X = Fg \cdot Lx + T1 \cdot X$$

despejando la Fca.

$$Fca = \frac{2 \cdot (Fg \cdot Lx + T1 \cdot X)}{\cos(\theta) \cdot X} \qquad (3.31)$$

$$(Fca/2)\ \cos(\theta) \cdot X = Fg \cdot Lx + T1 \cdot X$$

despejando la Fca.

$$Fca = \frac{2 \cdot (Fg \cdot Lx + T1 \cdot X)}{\cos(\theta) \cdot X} \qquad (3.31)$$

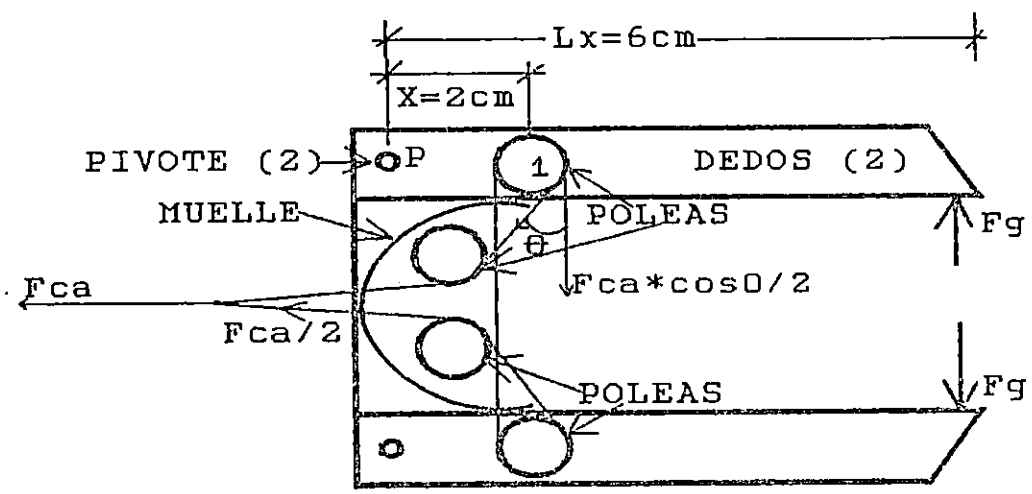


Figura 3.13. Simetría de la garra.

Donde:

T1 : es la fuerza que ejerce el muelle, su valor según pruebas es 5 libras.
X : es la distancia de la polea 1 al pivote y su valor es 2 cm.
Lx : es la distancia desde el extremo del dedo hasta el pivote y su valor es 6 cm.
θ : es el menor ángulo entre la magnitud de la fuerza del cable y la vertical, su valor es 45°.
Fg : es la fuerza de contacto entre los dedos y la pieza, también conocida como fuerza de rozamiento.
Ff : fuerza de fricción entre la pieza y los dedos.

Para determinar la magnitud de la fuerza de los dedos Fg segun la figura 3.14, se deben de considerar los 2 dedos (Nd) de la garra, el peso de la pieza (W = 5 libras) y el coeficiente de rozamiento para el peor caso se da cuando ($\mu s = 0.2$[5]) entre la superficie de la pieza y la superficie de los dedos.

La expresión para la fuerza de contacto Fg es:

$$W = Ff = \mu s \cdot Nd \cdot Fg$$

Despejando Fg:

$$Fg^{[6]} = \frac{W}{\mu s \cdot Nd} \qquad (3.32)$$

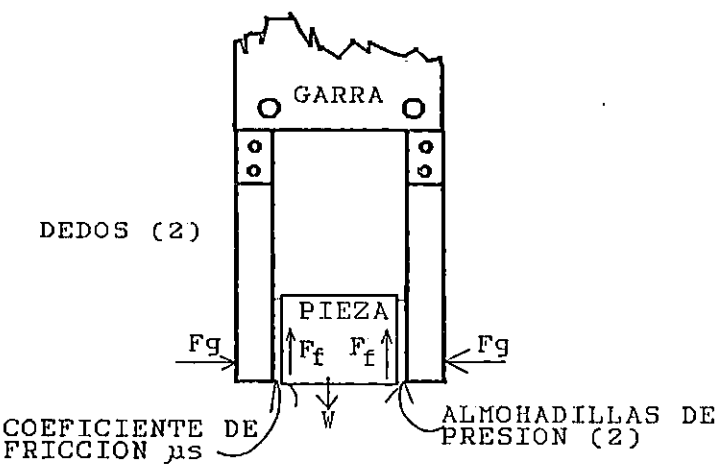


Figura 3.14. Fuerza de contacto contra la pieza.

---

[5] Engelber, Joseph F. Robótica Industrial. Edit. McGraw Hill, España, 1a. Edición, 1989.

[6] OP. CIT.

Ahora, sustituyendo ecuación (3.32) en (3.31), la expresión de la fuerza del cable es:

$$Fca = \frac{2 \cdot W \cdot Lx}{\cos\theta \cdot X \cdot \mu s \cdot Nd} + \frac{2 \cdot T1}{\cos\theta} \qquad (3.33)$$

Los valores de la Ec. 3.33 son:

Lx	= 0.06 mt	X	= 0.02 mt
μs	= 0.25	Nd	= 2 dedos.
(θ)	= 45°	W	= 5 lbs. = 22.24 Nt.
T1	= 5 lbs. = 22.24 Nt.		

Con los valores de los parametros de la ecuación anterior 3.33 conocidos, la fuerza del cable necesaria para levantar el peso es:

$$Fca = \frac{2 \cdot 22.24 \cdot 0.06}{\cos 45° \cdot 0.02 \cdot 0.25 \cdot 2} + \frac{2 \cdot 22.24}{\cos 45°}$$

$$Fca = 440.33 \text{ Nt.}$$

Cálculo de la fuerza del motor, Fm.

El motor de paso debe ejercer una fuerza que es trasmitida al cable de la garra para cerrar los dedos. Esta fuerza se define como:

$$Fm^{[7]} = \frac{60 \cdot Pm}{3.1416 \cdot dm \cdot Nm} \qquad (3.34)$$

Donde:

Potencia eléctrica $Pm = (14v)^2/70 = 2.8$ [watts].

Velocidad del motor $Nm = 48$ [rpm].

El paso del motor $\theta m = 7.5°$.

El diámetro del piñón acoplado al eje del motor (este viene acoplado al eje de fabrica), $dm = 0.011$ mt.

Ahora, evaluando la ecuación 3.34, la fuerza del motor es:

$$Fm = \frac{60 \cdot 2.8}{3.1416 \cdot 0.011 \cdot 48}$$

$$Fm = 101.3 \text{ Nt.}$$

---

[7] OP. CIT.

Factor de amplificación f_3.

La fuerza necesaria en el cable que cierra y abre la garra, Fca, es 440.33 N; la fuerza máxima que puede entregar el motor, Fm, es 101.3 N.
El sistema de transmisión debe proporcionar una amplificación que incremente el valor de la fuerza del motor.

Este factor de amplificación se llamara f_3 y para encontrarlo, relacionamos la fuerza del cable entre la fuerza del motor así:

$$f_3 = \frac{Fca}{Fm}$$

Al evaluar este factor f:

$$f_3 = \frac{440.33}{101.3}$$

La amplificación debe ser:

$$f_3 \geq 4.35$$

El factor de amplificación f_3 encontrado corresponde a la relación del juego de engranes y polea requeridos para amplificar la fuerza del motor a un valor igual al valor de la fuerza del cable.

En la figura 3.15 se muestra el sistema de transmisión obtenida en la práctica para la garra.

Los parámetros obtenidos para los piñones y poleas son:

Piñon 2: r2 = 25 mm Z_2 = 98 dientes.

Piñon 3: r3 = 7 mm Z_3 = 26 dientes

Piñon 4: r4 = 22 mm Z_4 = 86 dientes.

Polea 5: r5 = 10 mm

El factor f_3 está definido por la relación:

$$f_3 = \frac{Z_2 \cdot r4}{Z_3 \cdot r5} = \frac{98 \cdot 22}{26 \cdot 10} = 8.29$$

El valor f_3 = 8.29 nos da un margen de seguridad de 1.9 con respecto al valor 4.35, que es el valor mínimo permitido para f_3.

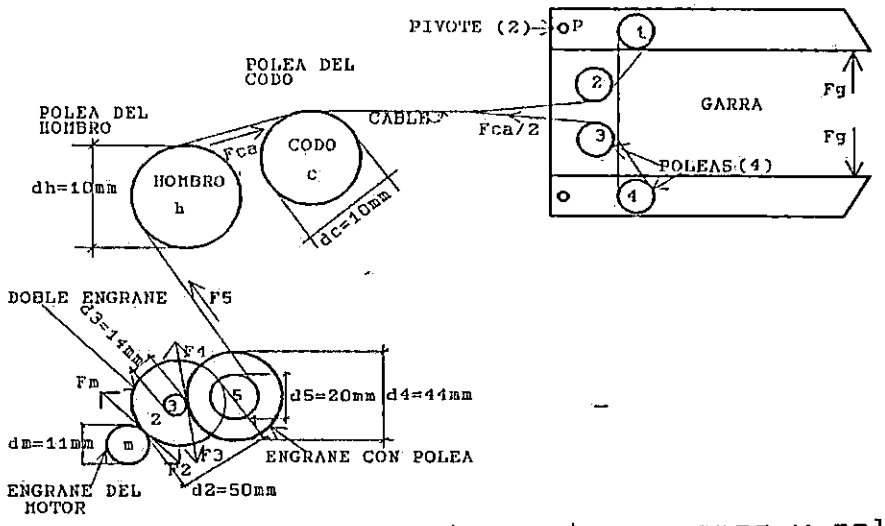


Figura 3.15. Ubicación de juego de engranes y polea.

Al hacer el análisis de la garra existe un supuesto implícito en los cálculos, y es que se supone que la garra agarra el objeto por su centro de masas y no existe ningún momento que haga que el objeto tienda a girar. El factor de seguridad de 1.9 ayudaría a compensar el problema potencial del objeto agarrado por otra posición distinta al centro de masas.

3.9. UBICACION DE LAS PIEZAS DEL BRAZO ROBOT.

En los siguientes apartados se explicará la ubicación de las piezas del antebrazo, brazo, garra y base.

3.9.1. UBICACION DEL ANTEBRAZO

El antebrazo se ensamblará entre dos puntos de apoyo, el codo y los soportes del hombro, teniendo el antebrazo una extensión de 160 mm de largo y 140 mm de eje a eje y un espesor de 2 mm aproximadamente. Son dos piezas a construir de la misma medida y estarán separadas entre sí una distancia aproximada de 65 mm como se observa en la figura 3.16. Será necesario darle estabilidad a estas piezas, esto se logra al colocar dos ejes de 3/16" de diámetro en sus puntos de apoyo. Para evitar que las piezas se salgan de su lugar se fijarán con seguros tipo E o chavetas, si fuera necesario se colocarán ángulos tipo c entre las dos piezas del antebrazo para dar mayor estabilidad.

3.9.2. UBICACION DEL BRAZO

Explicación del ensamblado de las piezas del brazo, éste estará ubicado entre dos puntos de apoyo la garra y el codo, teniendo una extensión de 150 mm de largo y 140 mm desde el

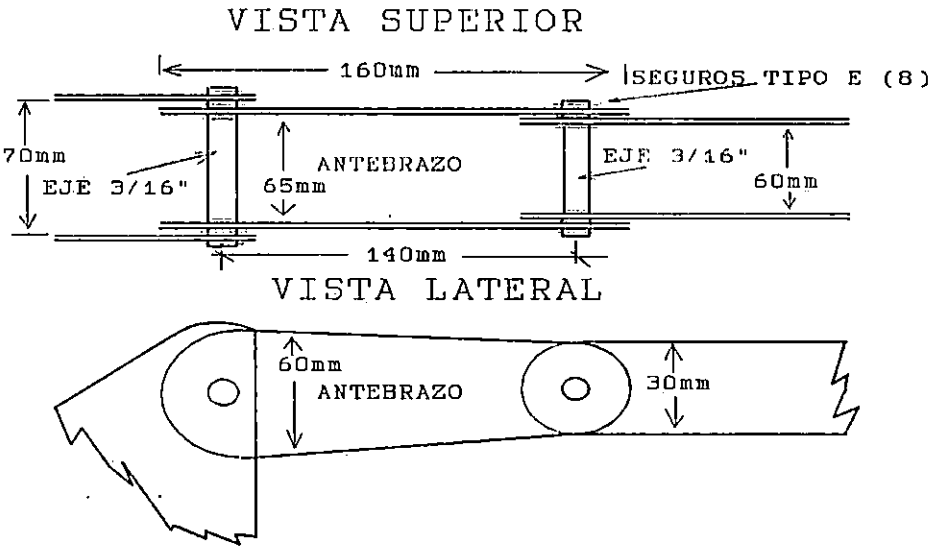


Figura 3.16. Ubicación de las piezas del antebrazo.

eje del codo hasta el empalme con la garra y un espesor de 2mm, como se observa en la figura 3.17. Se cortarán dos piezas de la misma medida y se colocaran a una distancia entre si de unos 60 mm aproximadamente. Para asegurar estas dos piezas es necesario ubicar un eje de acero inoxidable de un diámetro de 3/16 pulgadas en el codo. Al final del brazo o sea en el empalme con la garra se colocará un ángulo tipo "c" para darle mayor fijeza a la estructura. Será necesario usar seguros tipo E o chavetas para evitar que las piezas se salgan de su posición en el punto de apoyo del codo.

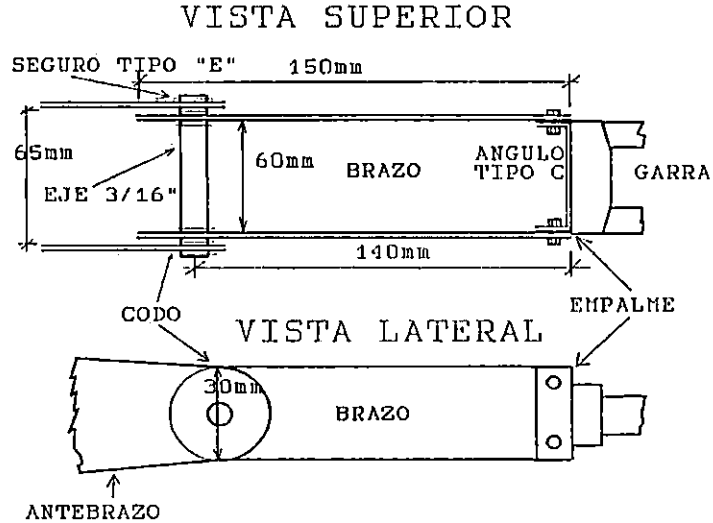


Figura 3.17. Ubicación de las piezas del brazo.

3.9.3 UBICACION DE LA GARRA

Esta pieza consta de varios elementos ensamblados de tal manera que conformen la estructura de la garra como se indica en la figura 3.18. El sistema tiene dos dedos o pinzas capaces de sujetar cualquier objeto no mayor de cinco libras, que ajuste en la máxima abertura de la garra (siete centímetros), estos dedos están sujetos a un ángulo C, teniendo un pin pasado por cada dedo que sirve de pivote al abrir o cerrar la garra, este ángulo C esta sujeto a la estructura final del brazo, también se cuenta con cuatro poleas plásticas o de cualquier otro material que no representen mucho peso con sus respectivos pines o ejes pasados ya que por ellos deslizara el cable que hará tensión a los dos dedos de la garra. Para evitar deslizamiento en la punta de los dedos al sostener un objeto es necesario colocar zapatas de hule. Se ajustara una laminita o muelle con cierta tensión para mantener abierta la garra cuando no actúa el motor. Las piezas de aluminio tendrán un espesor de 2 mm máximo y las medidas se aprecian en la figura 3.18.

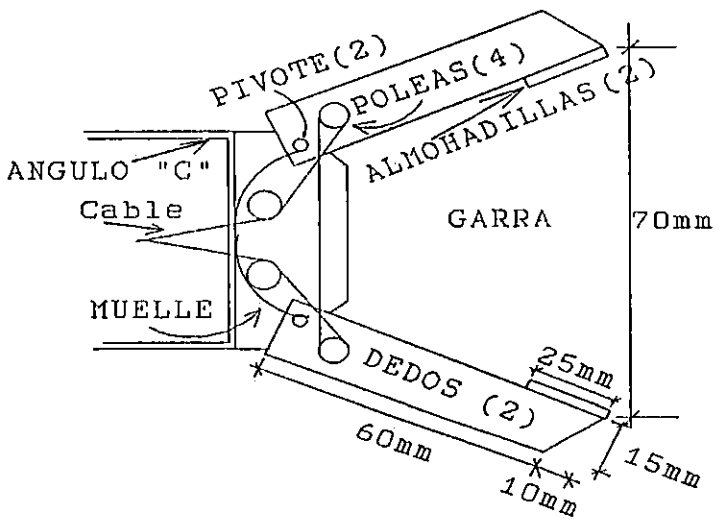


Figura 3.18. Ubicación de las piezas de la garra.

3.9.4 UBICACION DE LA BASE

La base será giratoria con capacidad de soportar la estructura mecánica del brazo y los motores de paso, ésta se moverá gracias a dos piñones ubicados de la siguiente manera: el primero que corresponde al de menor diámetro será ensamblado en el eje del motor y el segundo, el de mayor diámetro soportará la estructura mecánica. La base será conformada por una plancha de aluminio de 180 mm por 210 mm con espesor de 2 mm como se aprecia en la figura (3.19). A está plancha se acoplará el engranaje de mayor diámetro y el conjunto estará

montado en un mástil o varilla con balero que permite que gire libremente sin presentar rozamiento considerable. La base tiene también dos planchas de aluminio que sirven de soporte al hombro, a los motores y al juego de engranes con poleas.

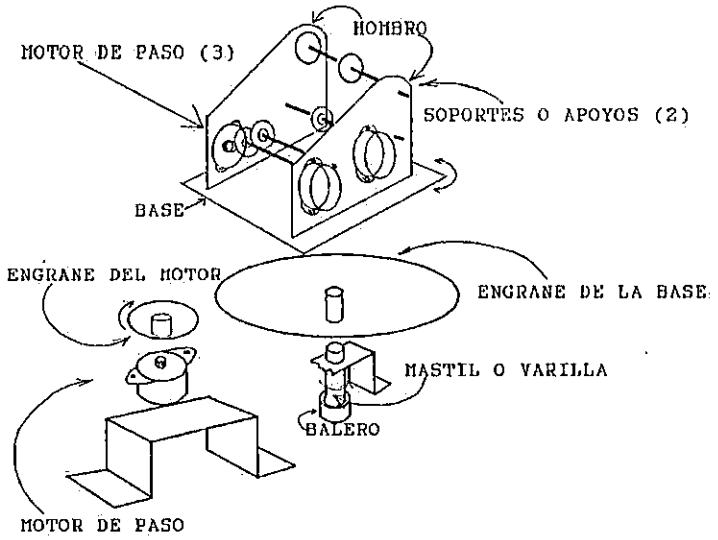


Figura 3.19 Ubicación de las piezas de la base.

3.10 EXPLICACION DE FUNCIONAMIENTO DEL SISTEMA

El brazo mecánico en su posición extendida tendrá una longitud aproximada de 350 mm, para llevar a cabo la extensión del brazo será necesario ubicar en el sistema cuatro motores de paso, cada uno actuará independiente del otro. Los motores de paso recibirán la señal eléctrica y ejecutarán un movimiento ya prefijado por el programa que se diseñará más adelante. Cada motor tiene su propio cable transmisor, menos el caso de la base, además cada motor tiene su propio juego de engranes y poleas.

La descripción detallada del funcionamiento del sistema mecánico robótico lo dividiremos en cuatro apartados.

a) Descripción de funcionamiento del antebrazo.
a) Descripción de funcionamiento del brazo.
c) Descripción de funcionamiento de la garra.
d) Descripción de funcionamiento de la base.

3.10.1 DESCRIPCION DE FUNCIONAMIENTO DEL ANTEBRAZO

Después que la controladora envía su señal eléctrica al motor se genera el movimiento del antebrazo gracias a las siguientes partes de la estructura, al juego de cuatro engranes, al juego

de dos poleas y al cable que se encarga de transmitir el movimiento desde la polea que se encuentra acoplada al engrane cuatro cómo se indica en la figura 3.20, el cable pasa por la polea que se ubica en el hombro y los extremos del cable se sujetan a una de las dos piezas de aluminio del antebrazo. El motor de paso esta acoplado a uno de los dos apoyos (plancha de aluminio de la izquierda) de la base, cuando el motor gira en cualquier sentido el antebrazo comienza a moverse por efecto de la tensión en el cable.

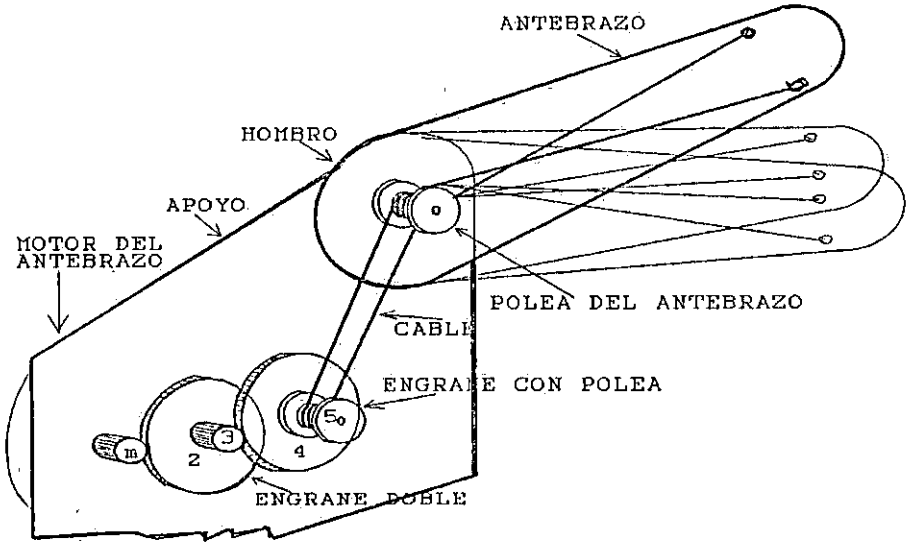


Figura 3.20. Movimiento del antebrazo por cable.

3.10.2 DESCRIPCION DE FUNCIONAMIENTO DEL BRAZO

La filosofía de movimiento es similar al caso anterior; el actuador del brazo es independiente del actuador del antebrazo.

Después que la controladora envía su señal eléctrica al motor se genera el movimiento del brazo gracias a las siguientes partes de la estructura: el juego de dos engranes, el juego de tres poleas y el cable que transmite el movimiento desde la polea que se encuentra acoplada al engrane "dos" como se indica en la figura 3.21, el cable que viene de la polea acoplada al engrane "dos" pasa por la polea del hombro y polea ubicada en el codo, los extremos del cable se sujetan a una de las dos piezas (del lado derecho) de aluminio del brazo. El actuador, motor de paso, está acoplado a uno de los dos apoyos (plancha de aluminio de la derecha) de la base, cuando el motor gira en cualquier sentido el brazo comienza a moverse por efecto de la tensión en el cable.

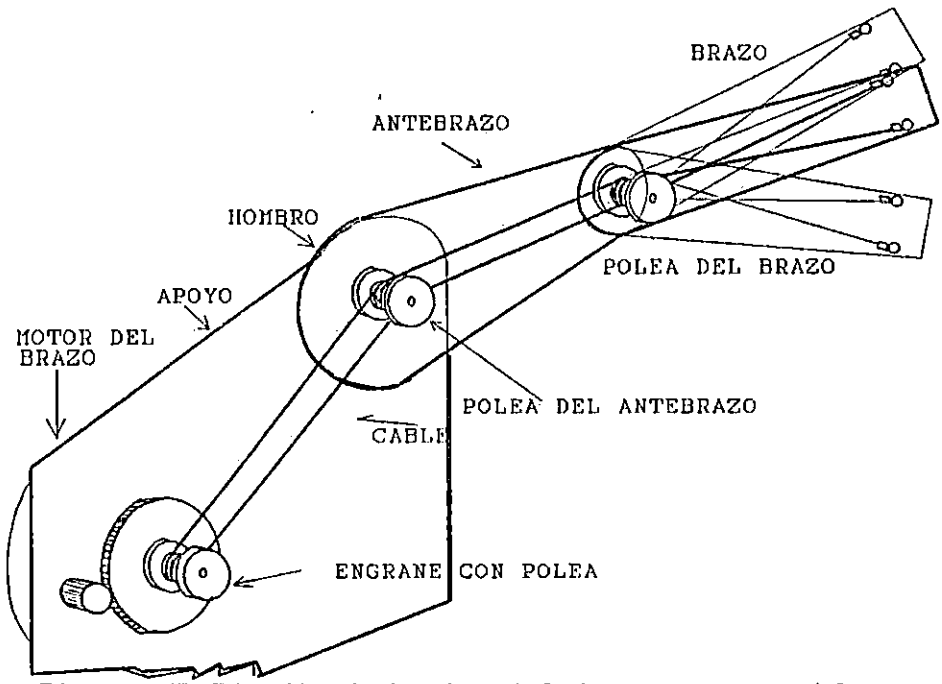


Figura 3.21. Movimiento del brazo por cable.

3.10.3 DESCRIPCION DE FUNCIONAMIENTO DE LA BASE

El actuador (motor de paso) al recibir la señal eléctrica de la controladora comenzará a girar el engrane acoplado en su eje, luego el engrane que soporta toda la estructura comienza a girar en el sentido contrario al del engrane del motor como se aprecia en la figura 3.19. Podemos decir que el conjunto de engranes serán capaces de girar hasta 180 grados en la dirección horaria o antihoraria.

3.10.4 DESCRIPCION DE FUNCIONAMIENTO DE LA GARRA

El actuador (motor de paso) de la garra recibe la señal eléctrica de la controladora para abrir o cerrar los dedos de la garra. Para realizar la abertura y cierre de los dedos es necesario acoplar un juego de cuatro piñones, tres poleas y un cable, ver figura 3.22. La transmisión de la fuerza requerida para abrir o cerrar la garra se realiza por el cable, a partir de la polea "cinco" acoplada al piñón "cuatro", este cable pasa por la polea del "hombro" y la polea del "codo" hasta los dedos de la garra, ver figura 3.22. El mecanismo de la garra comienza a trabajar cuando la tensión del cable ocurre y vence la fuerza que ejerce el muelle.

Es de hacer notar que el tipo de garra que se utilizará es bastante simple, ya que consta de dos dedos que cierran o abren, y no pueden girar sobre el eje axial del brazo, ni tampoco acomodarse fuera del plano en el que se ensamblarán.

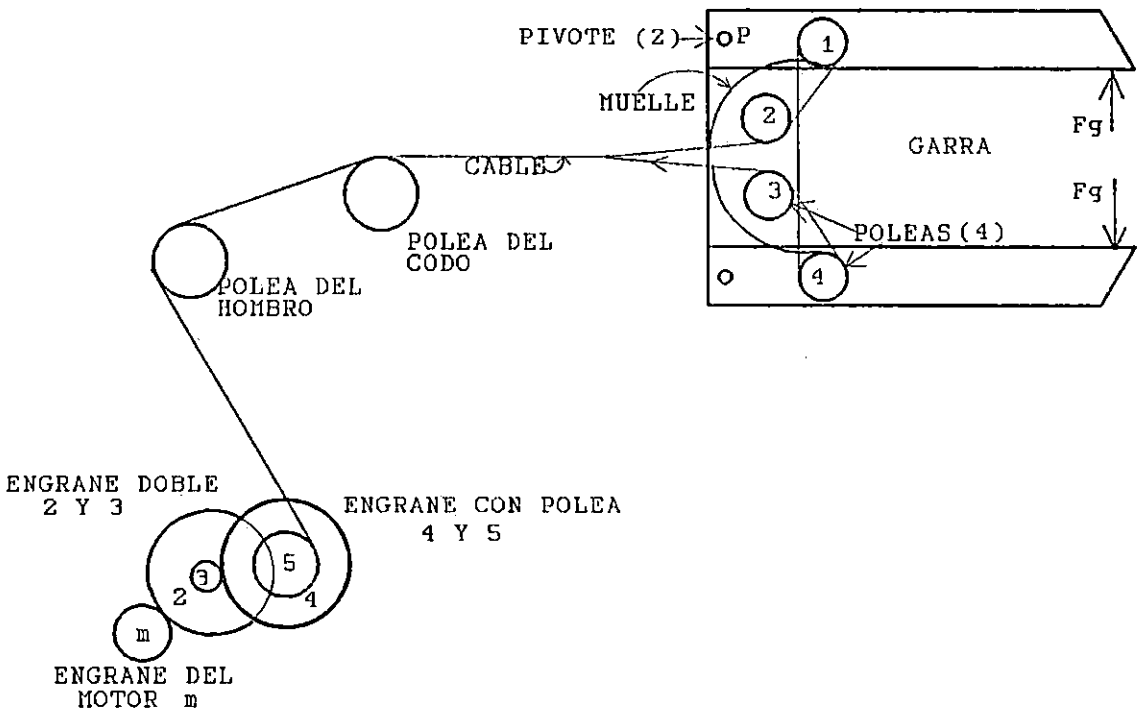


Figura 3.22. Mecanismo de la garra.

Cuando se evaluó el tipo de garra a utilizar, se pensó en usar una garra que hiciera todos los movimiento descritos anteriormente, y para ello se necesitaban de 3 engranajes cónicos para el juego mecánico de los cables y el movimiento de la garra. Si bien, se tenía la buena intención de realizar este tipo de garra, el problema se tuvo cuando se fue al taller para que los hicieran.

La problemática es la siguiente: en el taller de mecánica de la Facultad de Ingeniería y Arquitectura de la UES no tienen el equipo necesario para construir este tipo de engranaje cónicos, y en los talleres externos -empresa privada-, la construcción de cada uno de los engranajes tiene un costo ¢300, con un costo total de ¢ 900, únicamente en los engranajes cónicos.

Es obvia la razón por la cual se simplificó la garra.

3.11 DISEÑO Y CONSTRUCCION DE RUEDAS DENTADAS Y POLEAS

En los siguientes párrafos se desarrolla detalladamente la construcción de engranes de dientes rectos, y la construcción de las rondanas o poleas.

3.11.1 ENGRANES Y POLEAS

Las ruedas dentadas se conocen como engranajes o piñones, cuya finalidad es transmitir movimiento de rotación entre dos ejes o árboles. Se le denomina engranaje al conjunto de dos o mas ruedas dentadas, una de las cuales hace girar a las demás. Existen diferentes tipos de engranajes, cada uno de los cuales posee características y aplicaciones específicas. Entre estos se tiene: engranes rectos, helicoidales, cónicos, tornillos sinfin, etc. El engrane adecuado que cumple con las características idóneas para el sistema a desarrollar, es el piñón de dientes rectos, ya que el movimiento de rotación requerido debe de ser recto o axial al brazo.

Las poleas o rondanas, son ruedas ranuradas en su superficie curva, tienen la finalidad de transmitir movimiento de rotación como los engranes, pero con la diferencia, que las poleas se valen de cables para transmitir dicho movimiento.

Para la construcción de las ruedas ranuradas no se requiere conocer tantas relaciones matemáticas de diseño cómo en los engranes de dientes rectos, solo es necesario saber las dimensiones del diámetro exterior, diámetro interno y los espesores internos y externos de la polea. En secciones posteriores se detallarán los cálculos.

3.11.2 ENGRANES DE DIENTES RECTOS

Los engranes de dientes rectos o engranajes rectos se emplean para transmitir movimiento de rotación entre ejes paralelos. Su contorno es de forma cilíndrica circular y sus dientes son paralelos al eje de rotación.

La nomenclatura de los engranajes de dientes rectos se ilustra en la figura 3.23. Las definiciones de dichos términos son las siguientes:

CIRCUNFERENCIA PRIMITIVA O DE PASO. Es aquella según la cual se verifica la tangencia del piñón.

PASO CIRCULAR. Longitud del arco del círculo de paso medida desde un punto de un diente al mismo punto del diente adyacente. Es igual a la suma del grueso del diente y el ancho del espacio entre dos dientes consecutivos.

PASO DIAMETRAL O MODULO. Número de dientes del engrane que hay en cada pulgada del diámetro de paso. Es la relación entre el diámetro de paso y el número de dientes. El módulo o paso diametral es el índice del tamaño del diente.

ADENDO. Es la distancia entre el tope del diente y la circunferencia de paso.

DEDENDO. Es la distancia radial desde la circunferencia de dedendo hasta la circunferencia de paso.

ALTURA DEL DIENTE. Es la suma del adendo y del dedendo.

CIRCUNFERENCIA DE HOLGURA. Es la circunferencia tangente a la de adendo del engrane conectado.

HOLGURA O CLARO. Es la diferencia del espacio entre dos dientes consecutivos y el grueso del diente del otro engrane, medidos sobre la circunferencia de paso.

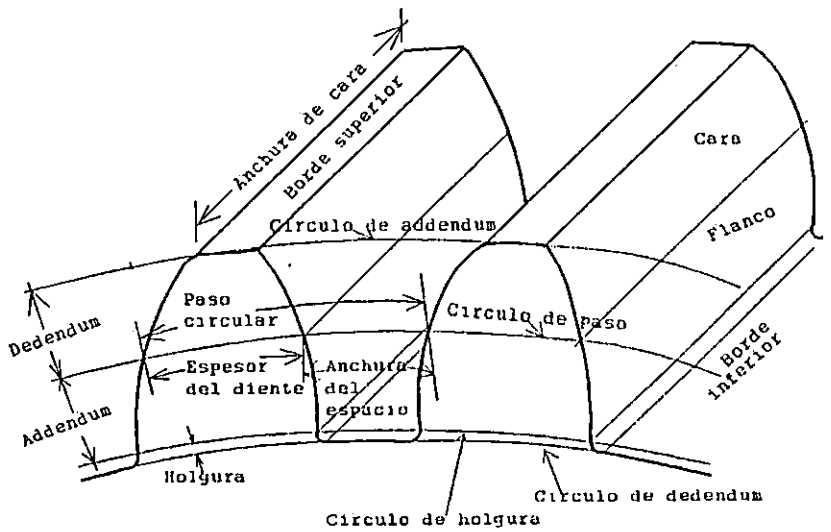


Figura 3.23. Nomenclatura de los engranes de dientes rectos.

3.11.3 RELACIONES FUNDAMENTALES PARA ENGRANAJES RECTOS

Los símbolos que siguen se emplean para representar los términos de nomenclatura de los dientes de los engranes.

P = Paso diametral o módulo.
D = Diámetro de paso.
d = Diámetro exterior.
di= Diámetro interno.
Z = Número de dientes del engrane.
a = Adendo.
b = Dedendo.
h = Altura del diente.
m = Paso circular = π/P.
t = Ancho de la cara
c = Holgura o claro.

Ecuaciones aplicables a cualquier engrane recto.

$P = Z/D$ (3.35)
$d = (Z + 2)/P$ (3.36)
$t = 10/P$ (3.37)
$h = 2.25/P$ (3.38)
$m = \pi/P$ (3.39)

3.11.4 TRENES DE ENGRANAJES

En el caso de los engranes rectos, los sentidos de giro corresponden a los de la mano derecha y se considera positivo o negativo según el sentido de rotación ya sea contrario o igual al de reloj. El tren de engranes que se ilustra en la figura 3.24, está formado por cinco elementos. La velocidad del engrane 6 es:

$$n_6 = \frac{z_2\, z_3\, z_5}{z_3\, z_4\, z6}\, n_2 \qquad (3.46)$$

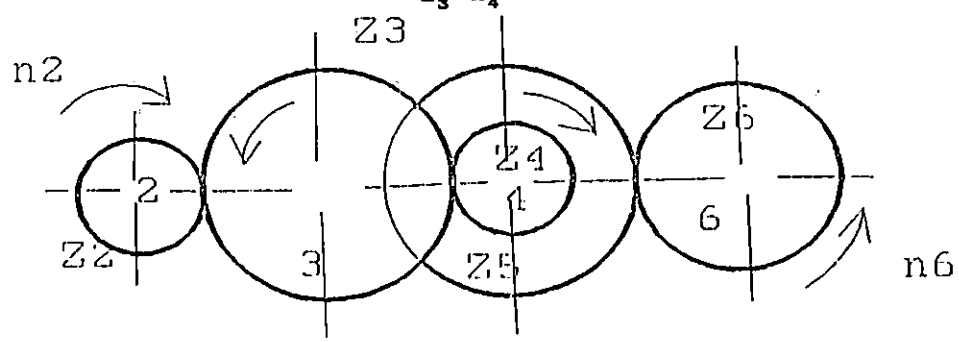


Figura 3.24. Tren de engranajes.

En este caso se observa que el engrane 3 es un engrane libre o loco o sea que se comporta cómo un engrane conductor y conducido a la vez, su número de dientes se cancela y que, por lo tanto, sólo afecta el sentido de giro del engrane 6. Además, los engranes 2, y 5 son conductores, mientras que los 4 y 6 son conducidos.

3.12 CALCULO DEL NUMERO DE DIENTES DE LOS PIÑONES Y DIMENSIONES DE LAS POLEAS DEL BRAZO ROBOT.

Es importante que cada motor de paso lleve un juego de engranes capaces de crear una reducción de velocidad para mejorar el torque que moverá la estructura mecánica y los objetos a sostener; obviamente por las limitaciones del proyecto el peso a levantar no será muy elevado.

Se iniciará el trabajo de diseño a partir de tres piñones conocidos, estos corresponden a dos engranes impulsores que están acoplados a los ejes de los motores de paso del antebrazo y brazo, y un tercer engrane impulsor acoplado al eje del motor de la base. Las dimensiones, diámetros internos de las poleas y diámetros externos de los engranes se obtienen del análisis estático.

3.12.1 JUEGO DE PIÑONES Y POLEAS DEL ANTEBRAZO.

Tomando de referencia la figura 3.25 y sabiendo que el piñón impulsor o conductor del motor del antebrazo tiene Zm = 20 dientes rectos, un espesor de diente de t = 9mm y un diámetro exterior de dm = 11 mm, este engrane del motor viene acoplado de fábrica.

Usando la fórmula 3.36, calcular el paso diametral del engrane impulsor o conductor del motor del antebrazo.

$$P = (Zm + 2)/dm = 2 \text{ mm. paso diametral.}$$

Calculando los dientes del piñón conducido "dos", su diámetro exterior es d2 = 44 [mm] y paso diametral P = 2.

$$Z_2 = P*d2 - 2 = 86 \text{ dientes.}$$

Calculando dientes del piñón conductor "tres", y su diámetro exterior d3 = 14 mm, el paso diametral se mantiene.

$$Z_3 = P*d_3 - 2 = 26 \text{ dientes.}$$

Calculando los dientes del piñón conducido "cuatro", su diámetro exterior es d4 = 44 mm, el paso diametral se mantiene.

$$Z_4 = P*d4 - 2 = 86 \text{ dientes.}$$

El espesor de los dientes en los engranes de acuerdo a la formula 3.37, se mantiene constante para los demás casos (brazo y garra) ya que el paso diametral es el mismo, entonces el espesor del dentado tendrá 5 mm. Si es mayor que 5 mm el ancho del diente el sistema de transmisión no se vería afectado, sino que mejoraría la superficie de contacto axial con el eje.

El espesor de la polea "cinco" acoplada al engrane "cuatro" y el espesor de la polea del "hombro" corresponden, cómo se indica en la figura 3.25, al espesor interno y los espesores externos, el espesor interno depende del diámetro del cable. El diámetro del cable es de 1 mm y por esta razón el espesor interno es de 3 mm, los espesores externos son de 1mm, teniendo una superficie de contacto axial de 5 mm con el eje, la profundidad de la ranura es 2,5 mm para evitar que se salga

el cable.

Para dar estabilidad a los engranes y poleas se ubicarán una o dos bases de la forma cómo se indica en la figura 3.20, estas bases serán de un espesor de 4 mm, en conjunto toda la pieza (polea) tendrá una superficie axial de contacto de 13 mm con el eje.

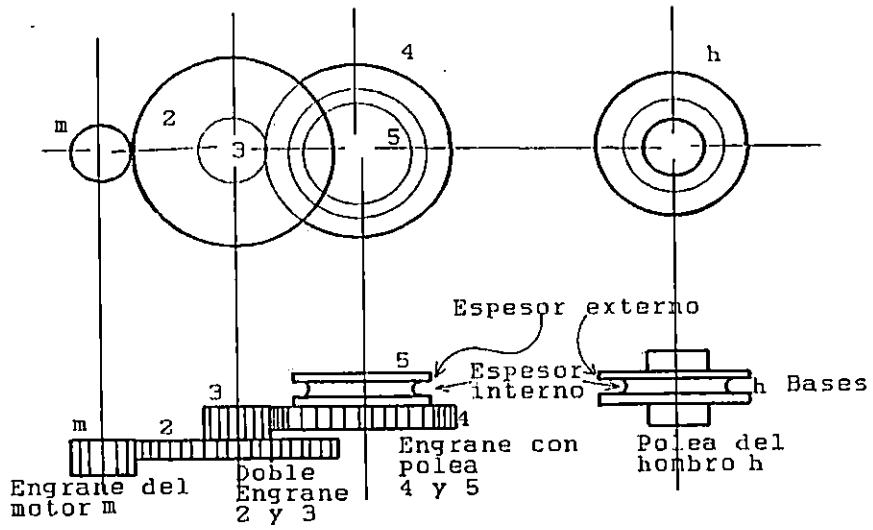


Figura 3.25. Juego de piñones y poleas del antebrazo.

Para todos los engranes se cumple esto:

P = 2 mm. paso diametral o módulo.
h = 2.25/P = 1.125 mm. altura o profundidad del diente.
m = π/P = 1.57 mm. paso circular.

3.12.2 JUEGO DE PIÑONES Y POLEAS DEL BRAZO

Tomando de referencia la figura 3.26 y sabiendo que el piñón impulsor o conductor del motor del brazo tiene Zm = 20 dientes rectos, un espesor de diente de t = 9mm y un diámetro exterior de dm = 11 mm, este engrane del motor viene acoplado de fábrica.

Usando la fórmula (3.36), calcular el paso diametral del engrane impulsor o conductor del motor del brazo.

$$P = (Zm + 2)/dm = 2 \text{ mm. paso diametral.}$$

Calculando los dientes del piñón conducido "dos", su diámetro exterior es d2 = 44 [mm] y paso diametral P = 2.

$$Z_2 = P*d2 - 2 = 86 \text{ dientes.}$$

Al evaluar la ecuación (3.37) el espesor del dentado es 5 mm y es como se dijo en el caso anterior del antebrazo.

Los espesores de: la polea "tres" acoplada al engrane "dos", la polea del "hombro" y la polea del "codo" corresponden cómo se indica en la figura 3.26, al espesor interno y a los espesores externos, el espesor interno depende del diámetro del cable. El diámetro del cable será de 1 mm y por esta razón el espesor interno es de 3 mm, los espesores externos son de 1mm, teniendo una superficie de contacto axial de 5 mm con el eje. La profundidad de la ranura es 2.5 mm.

Para dar estabilidad a los engranes y poleas se ubicarán una o dos bases de la forma cómo se indica en la figura 3.26, estas bases serán de un espesor de 4 mm, en conjunto toda la pieza (polea) tendrá una superficie axial de contacto de 13 mm con el eje.

Para todos los engranes se cumple esto:

P = 2 mm. paso diametral o módulo.
h = 2.25/P = 1.125 mm. altura o profundidad del diente.
m = π/P = 1.57 mm. paso circular.

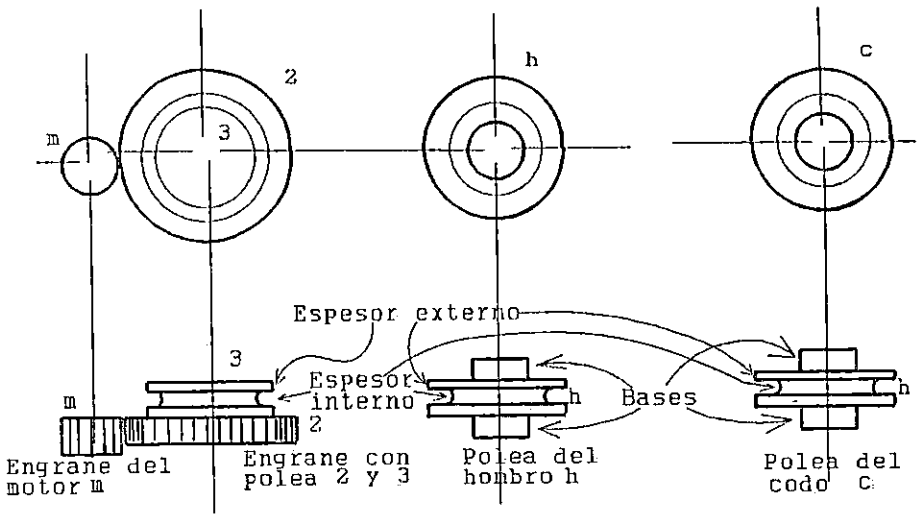


Figura 3.26. Juego de piñones y poleas del brazo.

3.12.3 JUEGO DE PIÑONES Y POLEAS DE LA GARRA.

Tomando de referencia la figura 3.27 y sabiendo que el piñón impulsor o conductor del motor tendrá Zm = 20 dientes rectos, un espesor del diente de t = 9mm y un diámetro exterior de dm=11 mm, estos datos se toman de las dimensiones de cualquiera de los dos piñones acoplados al motor del antebrazo

o brazo. Usando la fórmula 3.36, calcular el paso diametral del engrane impulsor o conductor del motor de la garra.

$$P = (Zm + 2)/dm = 2 \text{ mm. paso diametral.}$$

Calculando los dientes del piñón conducido "dos", el diámetro es d2 = 50 [mm] y se mantiene el paso diametral.

$$Z_2 = P*d2 - 2 = 98 \text{ dientes.}$$

Calculando dientes del piñón conductor "tres", su diámetro exterior es d3 = 14 mm, el paso diametral se mantiene.

$$Z_3 = P*d_3 - 2 = 26 \text{ dientes.}$$

Calculando los dientes del piñón conducido "cuatro", su diámetro exterior es d4 = 44 mm, el paso diametral se mantiene.

$$Z_4 = P*d4 - 2 = 86 \text{ dientes.}$$

El espesor de los dientes en los engranes de acuerdo a la fórmula 3.37, es 5 mm.

Los espesores de la polea "cinco" acoplada al engrane "cuatro", de la polea del "hombro", y de la polea del "codo" corresponden cómo se indica en la figura 3.27, al espesor interno y a los espesores externos, el espesor interno depende del diámetro del cable. El diámetro del cable 1 mm y por esta razón el espesor interno es de 3 mm, los espesores externos son de 1 mm, teniendo una superficie de contacto axial de 5 mm con el eje. La profundidad de la ranura es 2.5 mm.

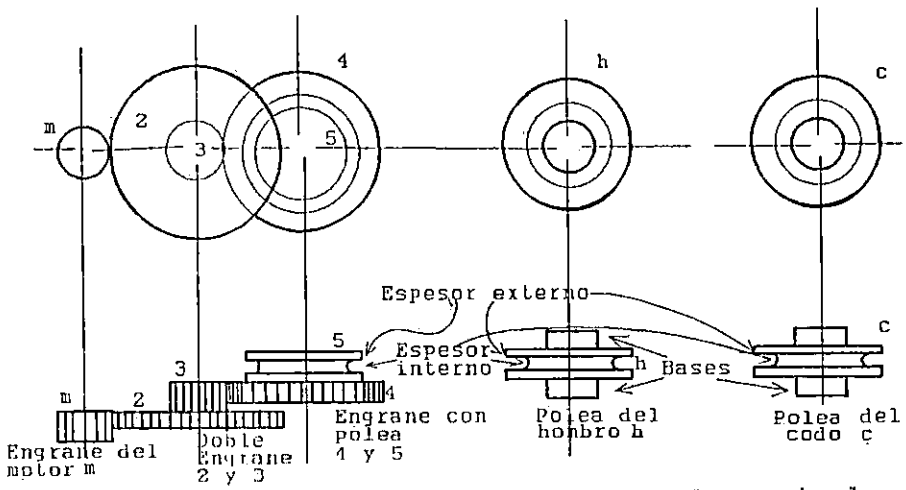


Figura 3.27. Juego de piñones y poleas de la garra.

Para dar estabilidad a las poleas se ubicarán una o dos bases

de la forma cómo se indica en la figura 3.27, estas bases serán de un espesor de 4 mm, en conjunto toda la pieza (polea) tendrá una superficie axial de contacto de 13 mm con el eje.

Para todos los engranes se cumple esto:

P = 2 mm. paso diametral o módulo.
h = 2.25/P = 1.125 mm. altura o profundidad del diente.
m = π/P = 1.57 mm. paso circular.

3.12.4 JUEGO DE PIÑONES DE LA BASE

Tomando de referencia la figura 3.28 y conociendo que el piñón impulsor del motor tiene 14 dientes, un espesor de 9 mm. y un diámetro exterior aproximado de 13 mm.

Usando formula 3.36, el paso diametral que se ha obtenido es 1.2613784.

Calculando el número de dientes del engrane impulsado "dos" de la base. Se conoce diámetro exterior del piñón impulsado d2 = 152 mm. Usando formula 3.36, el número de dientes del piñón impulsado es:

$$Z_2 = P*d2 - 2 = 192 \text{ dientes.}$$

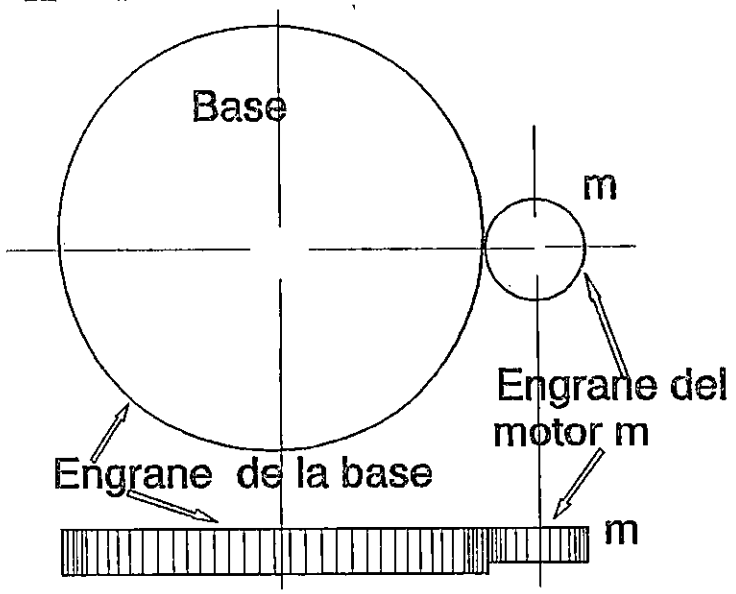


Figura 3.28 Juego de piñones de la base.

CONCLUSIONES DEL CAPITULO III

En este capítulo se ha desarrollado un proceso de diseño mecánico, que a nuestro juicio, es muy completo, ya que abarca la mayoría de aspectos involucrados en el diseño mecánico del brazo robot.

Hay que hacer notar, que una de las mayores dificultades que se nos presentó es el hecho de no tener el conocimiento de ingenieros mecánicos, lo cual representó una verdadera desventaja al principio del diseño, y nos obligó a investigar e involucrarnos en muchos conceptos de la ingeniería mecánica, significando un gasto de tiempo valiosísimo.

Con el diseño mostrado acá, ya se está listo para comenzar a construir sus partes y ensamblar el manipulador.

Son muchas las consideraciones a la hora de diseñar, y en especial, un brazo robot, requiere de un estudio profundo de las alternativas, ya que sus requerimiento son muy especiales – hay muchos aspectos críticos –.

Un aspecto importante, que se mencionó, pero deseamos recalcar, es el hecho que el brazo robot ya terminado, debe dársele mantenimiento, y entre más fácil, en menores costos se incurrirán. No sería práctico que un fabricante construya robots que tienen gran capacidad de carga, velocidades elevadas, etc. si para darle mantenimiento, sea necesario desensamblarlo por su gran complejidad.

REFERENCIAS BIBLIOGRAFICAS

[1] Andeen, Gerry B. Robot Design Handbook. Edit McGraw Hill, España, 1a. Edición, 1988.

[2] C. S. G Lee, K. S. FU y R. C. Gonzales. Robótica. Control, Detección, Visión e Inteligencia. McGraw Hill, 1a. Edición. 1988.

[3] Engelber, Joseph F. Robótica Industrial. Edit. McGraw Hill, España, 1a. Edición, 1989.

[4] Beer, Ferdinand P. y Johnton, E. Russel. Mecánica Vectorial para Ingenieros: Dinámica. Edit. McGraw Hill, España, 3a. Edición, 1979.

[5] Beer, Ferdinand P. y Johnton, E. Russel. Mecánica Vectorial para Ingenieros: Estática. Edit. McGraw Hill, España, 3a. Edición, 1979.

[6] Shigley, Joseph E y Matchel, Larry D. Diseño en Ingeniería Mecánica. Cuarta Edición.

CAPITULO IV

DISEÑO ELECTRICO DEL BRAZO ROBOT

Introducción

En este capítulo se procederá al diseño del sistema eléctrico del brazo robot. Este constará del diseño de la interfase de comunicación -entre computador y brazo-, la cual llevará las señales de control hacia el robot, y, en sentido inverso, las señales de sensoreo del robot hacia el computador.

Además, se diseñará la etapa de potencia, la cual amplificará la potencia de las señales recibidas del computador -a través de la interfase-, a niveles aceptables para manejar los motores de paso.

Para cerrar el ciclo, se diseñan los sensores de posición, los cuales enviarán la posición del brazo a la computadora, mediante el uso de la interfase de comunicación.

En este capítulo se presenta la teoría de los motores de paso, que será de mucha ayuda en el diseño global.

4.1 SISTEMA DE CONTROL

En la sección 1.5, se definió el controlador como: "el dispositivo que se encarga de regular el movimiento de los elementos del manipulador y todo tipo de acciones, cálculos y procesamiento de información que se realizan". En esta sección se ampliará dicho concepto.

Realmente, el controlador y los actuadores se incluyen dentro de los componentes del sistema de control. La función del controlador es comparar la salida real del sistema con la orden de entrada para proporcionar una señal de control que proporcionará una señal de actuadora que reducirá el error a cero o tan cerca de cero como sea posible.

El sistema de control es el responsable de proporcionar estabilidad al sistema global. En robótica, la estabilidad se suele definir como una medida de las oscilaciones que se producen en el brazo durante el movimiento desde una posición a la siguiente. Un robot con buena estabilidad presentará pocas o ninguna oscilación durante el movimiento o fin del

movimiento del brazo.

En el diseño del sistema de control se desea que el robot tenga una buena estabilidad. El diseño se complica, ya que una buena estabilidad reduce la velocidad de respuesta del brazo. La velocidad de respuesta es la capacidad del robot para desplazarse a la siguiente posición en un breve período de tiempo, y es deseable que tenga el máximo valor posible.

Si se incorporan elementos amortiguadores, la estabilidad se mejora, pero al mismo tiempo, la velocidad de respuesta disminuye. Por tanto, la solución tiene que estar comprometida en buscar la estabilidad del brazo y operar a velocidades altas.

El sistema de control se compone del controlador y los actuadores. El actuador se utiliza para convertir la acción de control en un movimiento físico del brazo.

La figura 4.1 muestra la configuración típica, en diagrama de bloques, de un sistema de control para una articulación basado en microprocesador.

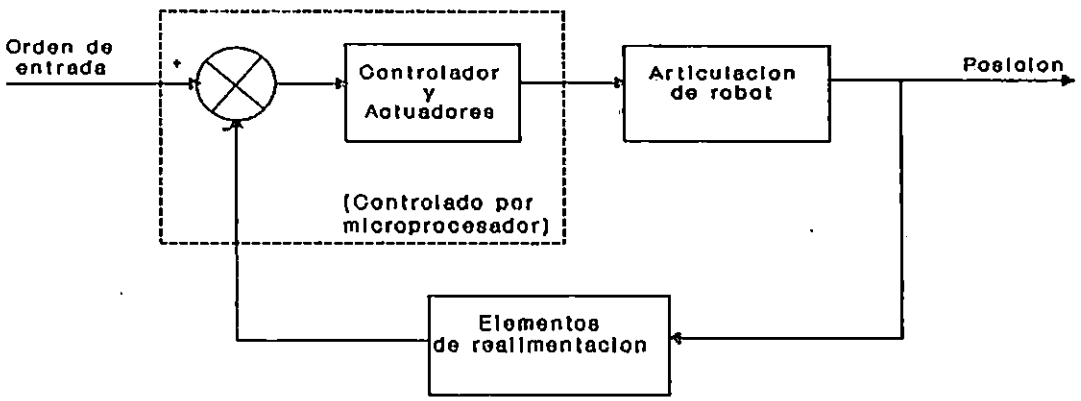


Figura 4.1. Diagrama de bloques de un sistema de control basado en microprocesadores.

De la figura 4.1, el controlador consta de una unión de suma donde se comparan las señales de entrada y salida. El dispositivo de control determina la acción de control necesaria; esta acción de control es aplicada a los amplificadores de potencia que impulsan los motores, los cuales mueven la articulación. En el lazo de retroalimentación, la posición de la articulación es realimentada al controlador mediante el uso de sensores de posición.

El controlador consta de los siguientes elementos:

1) Servocontrolador de articulación. Es el encargado de impulsar a su correspondiente actuador de articulación por medio de la etapa de amplificación. Es un sistema servocontrolado ya que recibe realimentación de la salida del sistema, verificando si se ha obtenido la salida esperada.

2) Amplificadores de potencia. La salida del servocontrolador no tiene la potencia necesaria para impulsar los actuadores -motores-, por lo cual se requiere de una etapa amplificadora de potencia que manipule los motores directamente.

3) Memoria de programa y dispositivos de entrada/salida. El controlador ejecuta las órdenes de movimiento a partir de dos posibles orígenes: entrada del operador o memoria de programa. En el primer, caso un operador introduce órdenes en el sistema utilizando un dispositivo de entrada -por ejemplo un teclado-. En el segundo caso, el conjunto de órdenes se ha programado con anterioridad en la memoria utilizando los dispositivos de entrada.

4) Procesador ejecutivo y matemático. El procesador ejecutivo coordina las actividades del sistema y el procesador matemático se encarga de los cálculos matemáticos que requiere el sistema. Para cada orden de movimiento, el procesador ejecutivo informa al procesador matemático de los cálculos de las transformaciones de coordenadas que se deben realizar. Cuando los cálculos de transformación se completan, el procesador ejecutivo los carga, enviando los resultados a los controladores de la articulación como órdenes de posición. En la práctica, ambos procesadores pueden ser sustituidos por un procesador de alta velocidad, que ejecuta ambas funciones.

Los microprocesadores se suelen utilizar en varios de los componentes de un controlador de robot moderno. Entre estos componentes se incluyen el procesador matemático, el procesador ejecutivo, los servocontroladores y el dispositivo de entrada. En aplicaciones robóticas, se utilizan los microprocesadores de uso general, como lo son el 8088, 8086, 80286, MC68000 y otros microprocesadores especializados con palabra de 16 bits o más.

Otro tipo de controladores utilizan circuitos lógicos digitales de propósito especial, denominados ASIC (aplication-specific integrated circuit) diseñados específicamente para usarse en robot. La ventaja de estos, es que son más veloces que los microcomputadores de propósito general. Otra alternativa es el uso de arreglos lógicos programables (PAL), que es una alternativa mucho más económica que el uso de ASIC, pero más compleja en su implementación.

Para el diseño a desarrollar, el controlador se basará en un microcomputador PC AT -de uso general- compatible, que posee un microprocesador 80286. En vista de que las interfases y programas para AT funcionan y corren también en sistemas 386, 486 y Pentium, estamos asegurando que tanto el controlador como el programa serán compatibles con dichos sistemas.

El microcomputador PC abarcará las funciones del procesador ejecutivo y matemático, servocontrolador, la memoria y entrada/salida del sistema.

La acción de control la ejecutará el computador mediante un programa apropiado.
El controlador -la PC y el programa-, se utilizarán como un controlador dinámico, interviniendo en el control de la posición del elemento terminal.

El control de los motores implica el desarrollo de un servomecanismo de posición, en el cual además de considerar los motores en sí, debe tomarse en cuenta la forma de detectar la posición a fin de determinar si se ha alcanzado el punto deseado.

Para transmitir la acción de control a los actuadores, así como captar las señales de realimentación, el microcomputador requiere de una interfase de entrada/salida, la cual debe ser diseñada.

En las siguientes secciones se desarrollan los procesos:

1) Selección de los actuadores.

2) Diseño de la etapa amplificadora.

3) Diseño del sistema de retroalimentación, sensores.

4) Comunicación del sistema con la computadora.

4.2 SELECCION DE LOS ACTUADORES

En el capítulo III se conocieron las diferentes alternativas de actuadores, entre ellos los motores DC y motores de paso. Mediante una serie de criterios se llegó a la selección de motores de paso como actuadores para nuestro diseño.

En esta sección describiremos las característica de estos motores en mayor detalle.

Motores Paso a Paso.

Los motores de paso - o stepper motor-, son un tipo especial de motores que permiten el avance de su eje en ángulos muy

precisos y por pasos en las dos posibles direcciones de movimiento, horario o antihorario. Aplicando a ellos una determinada secuencia de señales digitales, avanzan por pasos hacia un lado u otro y se detienen exactamente en una determinada posición.

Cada paso tiene un ángulo muy preciso -constante-, determinado por la construcción del motor, lo que permite realizar movimiento exactos sin necesidad de un sistema de control por lazo cerrado.

A un motor de paso se le puede ordenar, por medio del control, que avance cinco o diez pasos en sentido horario, luego un determinado número de pasos en sentido antihorario, o simplemente que no gire. Este sistema ha simplificado enormemente la implementación de automatismos y las aplicaciones de la robótica.

Los motores de paso presentan grandes ventajas con respecto a la utilización de servomotores debido a que se pueden manejar digitalmente sin realimentación, su velocidad se puede controlar fácilmente, tienen una larga vida, son de bajo costo, la interfase es sencilla y su mantenimiento es mínimo debido a que no tienen escobillas.

Los motores de paso generalmente se utilizan en ciclo abierto, ya que se asume que el motor gira un número de pasos controlados; sin embargo en muchas situaciones esta asunción da lugar a errores de posicionamiento ya que no siempre se puede trabajar en un ambiente totalmente controlado.

Quizás el control en ciclo abierto mediante motores de pasos funcione correctamente en dispositivos tales como impresores, máquinas de escribir o disqueteras, ya que estos aparatos han sido diseñados para condiciones precisas y la carga nunca presenta variaciones; en cambio, en un sistema robótico, debido al movimiento de la carga en el extremo del brazo, el motor de paso ve una carga variable, que puede exceder el límite del motor y no responder como debe.

Funcionamiento de los motores de Paso.

La operación de los motores de paso se basa en las fuerzas de atracción y repulsión ejercidas entre los polos magnéticos.

Si por ejemplo se tuviese un motor de paso con un estator de 4 polos, tal como el de la figura 4.2., con un cambio en la polaridad de estos por medio de un control externo, el rotor giraría en sentido contrario a las agujas del reloj, con incrementos de 90°. En suma, para girar este tipo de motores aplicamos pulsos en una de dos secuencias apropiadas, lo cual hará que gire en una u otra dirección.

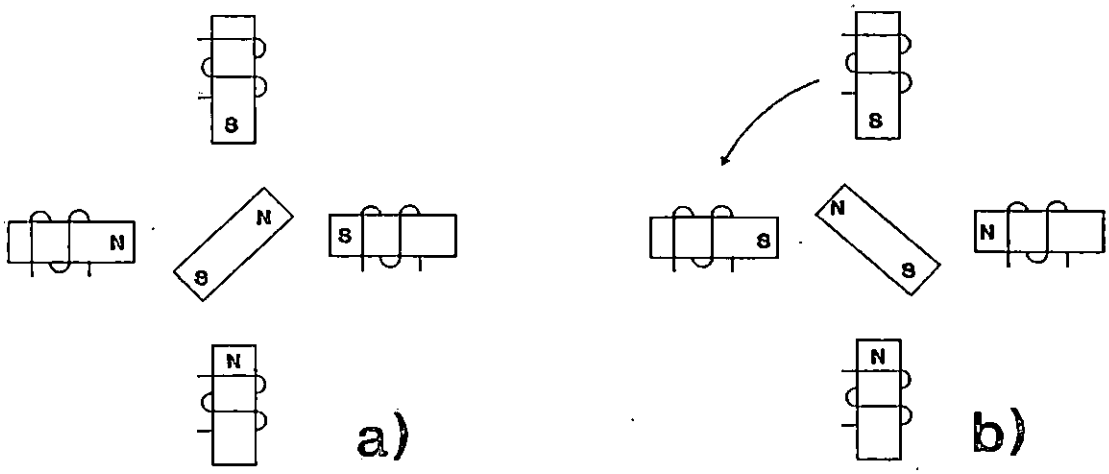


Figura 4.2. Principio de funcionamiento de un motor de paso.

Para lograr un movimiento mucho más suave, los motores de paso se fabrican aumentando el número de polos y se les practican una serie de ranuras tanto en el rotor como en el estator.

Así se logran movimientos que van hasta 1.8º por paso. Los grados de avance por paso es uno de los parámetros más importantes en este tipo de motores.

Para calcular el valor del paso del motor aplicamos la siguiente fórmula:

$$x = \frac{360}{f \cdot n}$$

Donde x es el valor del paso en grados;

f es el número de fases;

n es el número de dientes del rotor.

Tipos de motores de paso.

Según su construcción, hay tres tipo de motores de paso: de imán permanente, de reluctancia variable e híbridos.

En los primero, su rotor es un imán permanente que está ranurado en toda su longitud y el estator está formado por una serie de bobinas enrolladas alrededor de un núcleo o polo.

En los de reluctancia variable el rotor está fabricado por un

cilindro dentado de hierro y el estator está formado por dos bobinas que crean los polos magnéticos.

Como este tipo no tiene un imán permanente, su rotor gira libremente cuando las bobinas no tienen corriente lo que puede ser inconveniente en un momento dado si hay una carga que presione el eje.
Este tipo puede trabajar a mayor velocidad que el anterior.

Los híbridos combinan las dos características anteriores lográndose un alto rendimiento a buena velocidad.

En cuanto a la forma de conexión y excitación de las bobinas del estator, los motores de paso se dividen en dos tipos: unipolares y bipolares.
En los unipolares hay dos bobinas y tienen toma media, en decir tienen seis terminales.

Los bipolares tienen dos bobinas sin toma media, es decir, tienen cuatro terminales.

Curva Torque/Velocidad.

En la figura 4.3 se puede observar la curva torque/velocidad característica de los motores de paso:

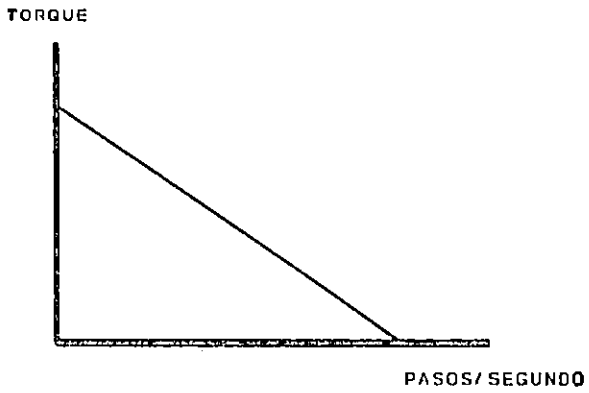


Figura 4.3. Curva Torque/Velocidad para motores de paso.

De la figura, se obtienen las siguientes ideas: si estando en reposo, se le aplica una muy alta velocidad, el torque será tan pequeño que no podrá vencer la inercia del rotor y la carga conectada. De la misma forma, si deseáramos pararlos cuando va a una velocidad elevada, la misma inercia lo impediría.

En ambos casos habrá pérdidas o aumentos de pasos, y por lo tanto la posición final se desconocería.

La forma de evitar el anterior inconveniente, en un movimiento completo, es acelerar al inicio desde velocidad 0 para vencer la inercia y al final desacelerar para parar.

4.3 DISEÑO DE LA ETAPA AMPLIFICADORA

La acción de control del servocontrolador maneja la velocidad y posición de la articulación, y para ello genera una secuencia de impulsos eléctricos apropiada para aplicarse a las bobinas del motor.

En vista que el controlador seleccionado es una computadora, ésta resulta ideal para enviar una secuencia de señales digitales ordenadas por un programa.

Existen dos maneras en que una computadora puede controlar un motor de paso. En la primera, cada bobina del motor es manejado por un bit independiente en un puerto de salida. En la segunda, se utilizan circuitos integrados especializados que requiere de pocas líneas de control por parte del microcomputador.

4.3.1 CONTROL DIRECTO DE LAS BOBINAS

En la figura 4.4 se puede observar como cada bit controla cada bobina del motor de paso, a través del circuito ULN2003, que es un buffer amplificador de potencia media que puede manejar una buena variedad de motores de paso.

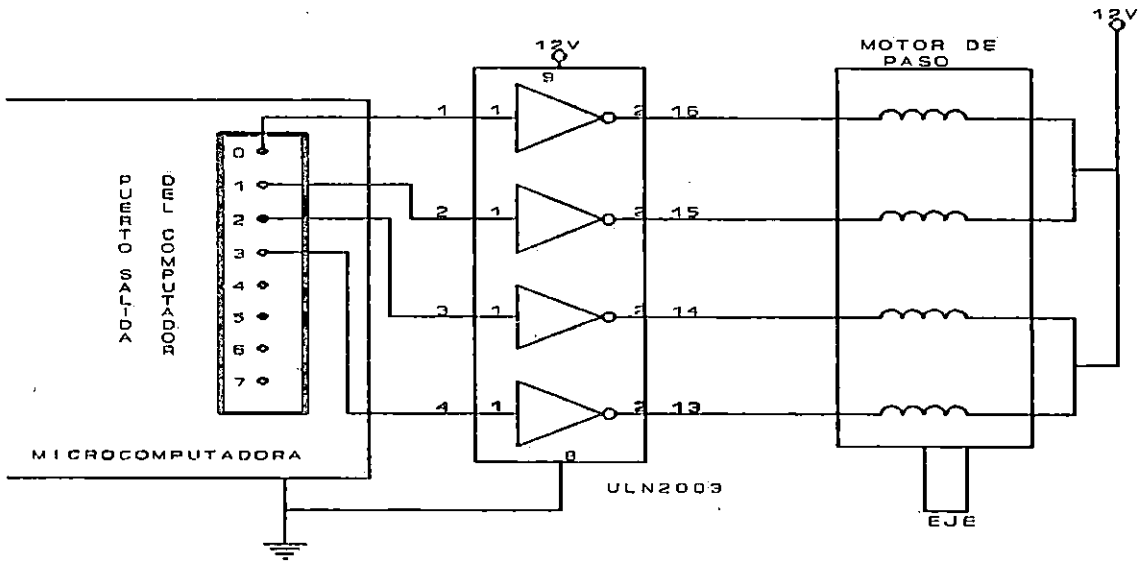


Figura 4.4. Control de un motor de paso a través de un microcomputador.

Cada uno de los buffers puede manejar cargas hasta de 500mA. Si un motor necesitara más corriente, es necesario utilizar transistores de potencia, preferiblemente de tipo Darlington.

El microcomputador debe generar la secuencia adecuada para que el motor gire en un sentido u otro.

4.3.2. CONTROL CON IC ESPECIALIZADOS.

En el mercado existen circuitos integrados específicos para el control de motores de pasos. La ventaja principal de esta alternativa, es que las líneas de control del microprocesador hacia la interfase se reducen, pudiendo manejar mayor cantidad de motores por puerto.

Uno de los circuitos especializados más utilizados para el control de motores de paso es el IC Philips SAA1027, del cual se da una descripción.

El integrado SAA1027.

El SAA1027 es proyectado para controlar motores paso a paso de 4 fases con dos estatores. El diagrama interno de este IC, lo puede observar en la figura 4.5. El circuito básicamente consiste de un contador direccional de 4 estados y un conversor de código capaz de excitar cuatro salidas en la secuencia necesaria para el funcionamiento de un motor de paso.

El SAA1027 puede excitar directamente motores paso a paso con tensiones de trabajo de 9.5 a 18V y corrientes de hasta un valor de 500 mA.

Los únicos componentes externos son elementos cuyos valores dependen del tipo de motor usado. Para el circuito de alimentación tenemos una división en dos sectores siendo uno de corriente elevada para el motor y el otro de baja corriente con filtrado adicional dada por R1 y C1.

El resistor Rx conectado al circuito de alta corriente determina la máxima corriente que se aplicará al motor de paso y su valor está dado por la fórmula:

$$Rx = \frac{4 \times V}{I - 60}$$

Donde

Rx es el valor del resistor en ohm.
V es la tensión de alimentación en voltios.
I es la corriente deseada en cada fase del motor en Ma.

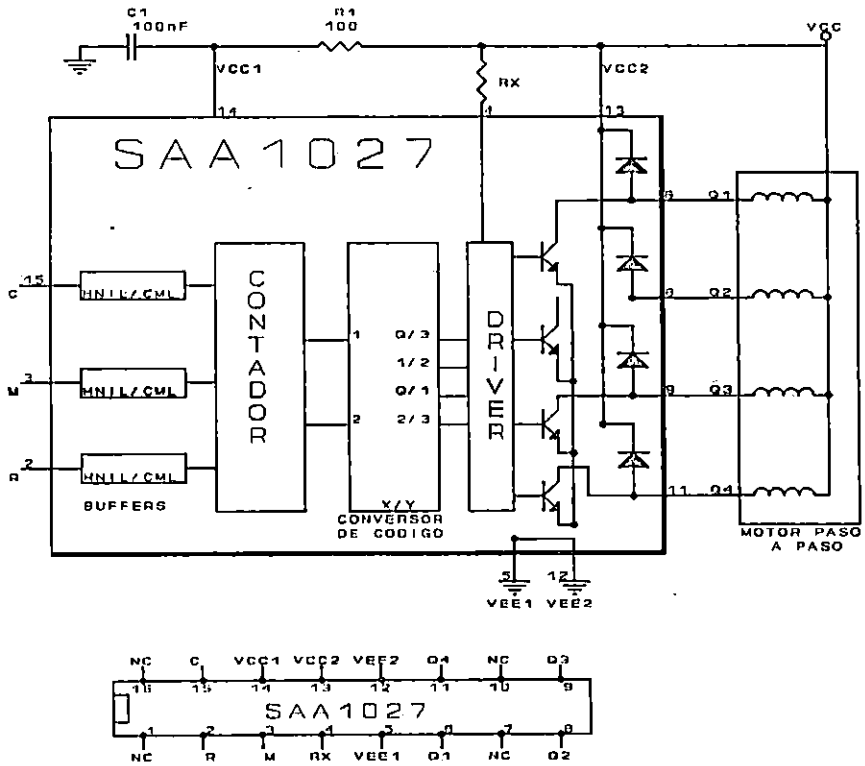


Figura 4.5. IC SAA1027 y su diagrama de bloque interno.

Los terminales de control funcionan de la siguiente manera:

La entrada RESET (pin 2) es mantenida en el nivel alto, para que a cada transición positiva del terminal COUNT (pin 15) tengamos una alteración del estado en las salidas del integrado. Esta secuencia dependerá del nivel aplicado a la entrada MODE (pin 3) que determina el sentido de la rotación del motor.

Tenemos entonces dos secuencias posibles que corresponden a la rotación del motor en el sentido directo o en el sentido inverso.

Si llevamos la salida RESET al nivel bajo, el contador llevará las salidas a la posición correspondiente al 0 de la secuencia de conteo.

El circuito integrado es del tipo colector abierto en las salidas. Para evitar problemas de sobretensión en las mismas cuando se produce la conmutación de una carga inductiva como la representada por los bobinados del motor, se usan diodos de protección. En el circuito equivalente observamos la

colocación de estos diodos. Las especificaciones principales de este intégrado SAA1027 se resumen en la tabla B.1 y sus especificaciones generales están en la tabla B.2, del ANEXO B.

4.3.3. DISEÑO DE LA ETAPA DE AMPLIFICACION.

Planteadas las alternativas anteriores, se observa claramente que la opción que simplifica tanto el hardware como el software es la utilización del IC SAA1027.

Utilizando este IC, únicamente se requieren de dos líneas de un puerto del computador para controlar un motor de paso, a diferencia de la otra alternativa que requiere de 4 líneas.

La característica anterior es de vital importancia, en vista de que se manejarán 4 motores de paso. Para la primera alternativa (usando el ULN2003) se requeriría de 16 líneas de puerto, 4 líneas por cada motor; mientras que usando el SAA1027 únicamente se requieren de 8 líneas, 2 líneas por motor. Hay que hacer notar que además de estas señales de control, el computador debe estar manejando las señales de los siete sensores, lo cual vuelve aún más crítico el reducir la cantidad de bits de puerto a usarse para el control de los motores.

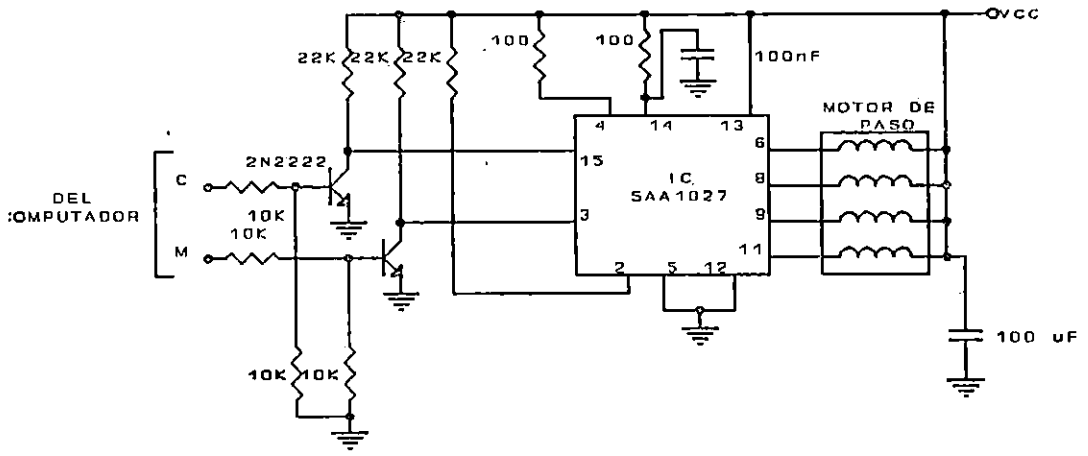


Figura 4.6. Etapa amplificadora para impulsar un motor de paso.

Una interfase para el control de los motores de paso, utilizando el SAA1027, se muestra en la figura 4.6

Esta interfase es conectada a un puerto de salida (I/O) de la computadora. Los niveles altos aplicados en la base de los transistores, corresponden a niveles bajos en las entradas

correspondientes del integrado SAA1027, ya que la configuración emisor común invierte los niveles lógicos.

Optoaislamiento.

Los motores de paso que se utilizarán poseen diferentes niveles de voltaje (9V, 12V, 14V). Los motores consumen potencia elevada, lo cual puede afectar la regulación de su fuente, por lo cual se ha optado por un aislamiento entre el circuito del motor paso a paso y la computadora. Para ello, cada bit del puerto de la computadora será transmitida a la etapa amplificadora de potencia por un opto-aislador (aislador óptico), que aislará completamente ambos circuitos. En la figura 4.7 se muestra el opto-aislador a utilizarse.

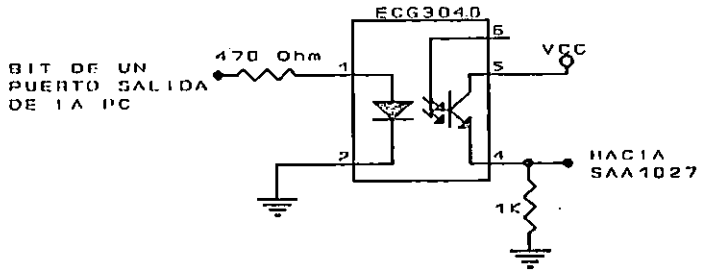


Figura 4.7. Configuración típica del opto-aislador ECG3040.

En este circuito, el bit del puerto de la computadora se conecta directamente para impulsar el led, usándose una resistencia limitadora de corriente de 470 Ω.

En el lado derecho, la fuente Vcc corresponde a un valor de 9V, y corresponde al circuito de los motores. Notemos que la tierra del diodo es de la PC; y la tierra del transistor corresponde al circuito del motor.

Generación de señales.

La interfase mostrada en la figura 4.6 requiere de la aplicación de señales digitales en sus pines C (count) y M (modo). El pin R (RESET) permanecerá siempre en alto. Con el pin MODO se determina el sentido de giro del motor. Con el pin COUNT el motor gira un paso por cada transición positiva que llega. Con RESET habilitamos el SAA1027.

La serie de pulsos que provienen del puerto del computador hacia el pin C (count) son muy angostos en el tiempo y andan en el orden de los nanosegundos. Estos pulsos son filtrados por la etapa de aislamiento (Optoaislador), la cual funciona como filtro pasa-baja. Para aumentar el ancho del pulso que llega al pin C (count), se hace uso de una emisión simple. La señal obtenida de la emisión simple puede ser transmitida a

través del optoaislador sin que sufra filtrado. El tiempo que permanece activa la emisión simple es ajustado por su constante de tiempo, que depende de los valores de resistencia y capacitancia externa a ella. La constante de tiempo depende de la máxima frecuencia que el optoaislador permite transmitir.

Las dos señales (pulsos) del puerto que vienen del computador C y M, pasan, una de ellas C por la emisión simple y luego al optoaislador; la otra, M se dirige directamente hacia el optoaislador. Los pulsos que salen del optoaislador, debido al efecto de filtrado que le introduce el optoaislador, no son señales cuadradas perfectas. En vista que las señales C y M del SAA1027 trabajan con transiciones, la señal que les llegue debe ser perfectamente cuadrada, y para ello, la señal del optoaislador llega a una compuerta inversora scmitt-trigger (74LS14), la cual la convierte en señal cuadrada perfecta.

El circuito completo que maneja un motor queda descrito en la figura 4.8.

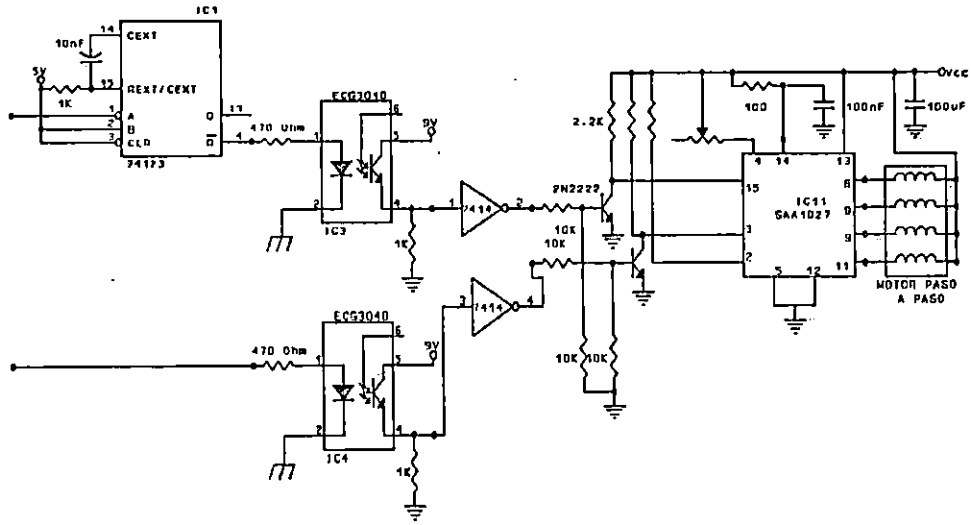


Figura 4.8. Circuito manejador de un motor.

Descripción de la emisión simple IC 74123.

Este componente 74123 contiene dos emisiones simples empaquetados en un solo chip. Por cada motor se requerirá una emisión simple, en total se utilizarán 2 chips 74123 o mejor dicho, una emisión simple por cada interfase de manejo de los motores de paso.

Para este caso la emisión simple puede trabajar en transición positiva o negativa, según la combinación de niveles de

voltaje que tengan los pines de entrada A, B y CLR. Cuando las entradas A, B y CLR se conectan como se muestra en la figura 4.9 (B = 1, CLR = 1, A = línea del computador), la emisión se activará con una transición negativa en A, pasando la salida Q (-Q) a alto (bajo). Q (-Q) se mantendrá en uno (cero) lógico por un período de tiempo T prefijado por la red RC externa acoplada en los pines CEXT y REXT/CEXT.

Descripción del IC scmmitt-trigger 7414.

Este componente contiene seis inversores de lógica positiva Y=-A empaquetados en un solo integrado. Por cada motor se requiere dos inversoras scmmitt-trigger, en total son ocho inversores, por tanto se hacen uso de dos componentes 7414 para tal propósito. Si a una inversora llega un uno (cero) lógico a su salida habrá cero (uno) lógico.

4.3.4 ETAPA GLOBAL DE AMPLIFICACION

La etapa global de amplificación consta de cuatro circuitos, estos son: circuito del motor del antebrazo, circuito del motor del brazo, circuito del motor de la base y circuito del motor de la garra. La etapa global se muestra a nivel de bloques en la figura 4.9.

El circuito de cada motor se muestran en la figura 4.8. El diagrama eléctrico completo se muestra en el APENDICE C.

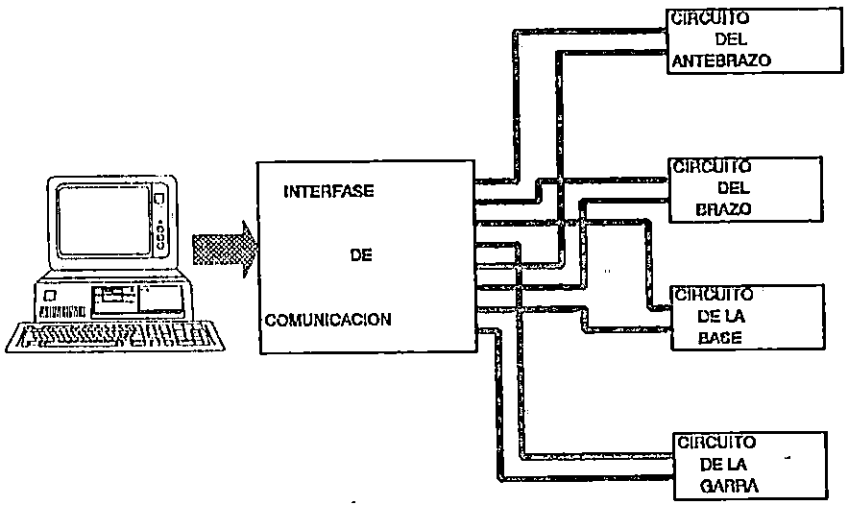


Figura 4.9. Etapa de amplificación global.

4.4 DISEÑO DEL SISTEMA DE SENSORES

En la teoría de los motores de paso, cuando se describía su característica Torque/Velocidad, se dijo que bajo ciertas

circunstancia, el efecto inercial podría hacer que el motor de paso perdiese o ganase pasos indeseables. En un sistema robótico, debido al movimiento -impredecible- de la carga en el extremo del brazo -redundando en una variación del torque aplicado al eje del motor-, este efecto es aún más crítico.

La razón de ello es que, si en un sistema de ciclo abierto, un paso es perdido o ganado -en forma indeseable-, no sabríamos la posición real de la carga, y por tanto el sistema fallaría al ubicarla.

Lo anterior exige que a pesar de usar motores de paso, se debe tener una manera de saber la posición real de la carga, y corregirla hasta hacerla llegar al punto deseado. Para ello se necesita de un transductor, que transforme la posición en un variable eléctrica que pueda ser digitalizada y enviada al computador.

4.4.1 SELECCION DEL SENSOR DE POSICION

Existen distintos tipos de transductores de posición, entre ellos los codificadores ópticos, codificadores ópticos láser, potenciómetros, codificadores magnéticos, synchros, LVDT (lineal variable differencial transformer) y RVDT (rotary variable differential transformer). Los más empleados en aplicaciones robóticas son los primeros tres.

La idea básica con los potenciómetros es obtener un voltaje proporcional a la posición del elemento que se desea controlar. Su ventaja principal es el bajo costo. Entre sus desventajas, se pueden mencionar la baja precisión que poseen, con un tiempo de vida muy bajo cuando son expuestos a movimiento continuo.

Los codificadores ópticos consisten de un disco acoplado al eje del motor. Este disco tiene zonas opacas y transparentes que cortan o dejan pasar un haz luminoso que va desde un emisor hacia un detector de luz. La resolución de los codificadores ópticos depende de la cantidad de ranuras que posea el disco. Son muy útiles a bajas y altas velocidades, ya que el elemento detector -la luz- es muy rápida. Su precisión es excelente.

En cuanto a los codificadores ópticos láser, su principio es idéntico a los codificadores ópticos, y la diferencia radica en el emisor, el cual es un diodo láser.

Agregando un cuarto transductor de posición a esta lista, los microswitch de fin de carrera, estos al cerrar sus contactos envían un pulso por medio de un optoaislador hacia el software del computador, acá se procesa el pulso y se corrige la posición real de los dedos de la garra (motor de la garra).

El uso de potenciómetros como sensores de posición en el brazo robot, exigiría de ellos un continuo movimiento, el cual provocaría su daño a corto plazo. Lo anterior redunda en un mantenimiento continuo, que contradice una de las características deseables en un sistema robótico, y es el fácil y poco mantenimiento de sus partes. En cuanto a los codificadores ópticos láser, el problema principal radica en conseguir el diodo láser.

La alternativa seleccionada son los codificadores ópticos, con un led infrarrojo como emisor y un fototransistor como detector. Una mejor alternativa es trabajar con optoacopladores, los cuales integran el diodo infrarrojo y el fototransistor. Uno de estos optoacopladores se obtiene fácilmente en el mercado y corresponde al ECG3100. Sus especificaciones se muestran a continuación:

ECG3100	
Potencia de disipación total	250mW
LED (especificaciones máximas)	
corriente en directa:	60 mA
voltaje en reversa:	6 V
TRANSISTOR (especificac. máximas)	
V_{ce}:	5 V
I_c:	100mA

En la construcción, por cada articulación del brazo se hace uso de dos optoacopladores para detectar la posición del brazo. Estos dos optoacopladores están ajustados de tal manera que las señales que emiten tienen un desfase de 90 grados. Este ajuste de fase proporciona información sobre el sentido de giro, es decir, si la señal A se adelanta en fase a la señal B en 90 grados, significa que el disco ranurado gira en un sentido (y por tanto la articulación se mueve en ese sentido). Si B se adelanta en fase a la señal A, significa entonces que el disco ranurado gira en sentido contrario. Veáse la generación de señales en la Fig. 4.10.

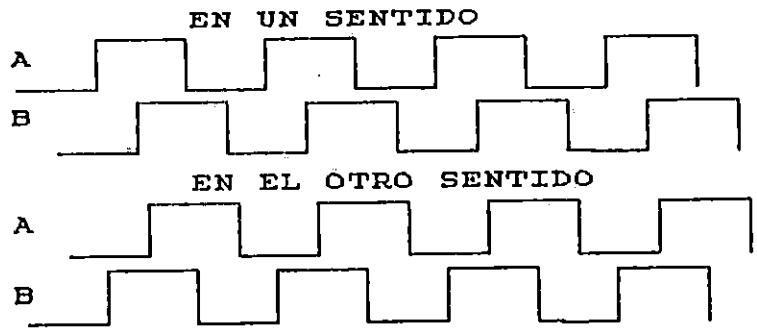


Figura 4.10. Sentido de giro.

Ambas señales son transmitidas a dos puerto de entrada de la PC (de 1 bit cada uno), y el software lleva el control del número de pulsos, calculando a partir de ellos la posición real del eje en cada instante.

Por facilidad solo se muestra un optoacoplador en la figura 4.11., donde se puede observar la conexión del optoacoplador configurado para enviar la información al computador. El inversor Schmmit-Trigger se utiliza para acelerar las transiciones, de manera que sean interpretadas como tales por el puerto de la computadora.

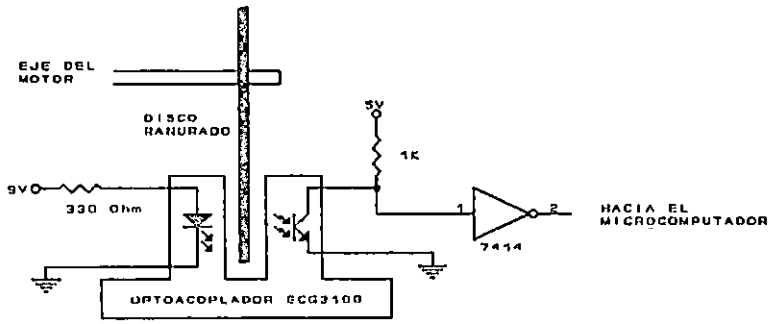


Figura 4.11. Circuito para detectar la posición de las articulaciones en el brazo robot.

El funcionamiento del circuito es como sigue: si el fototransistor recibe radiación del fotodiodo, el fototransistor se satura y el voltaje de colector baja casi a 0V, con lo cual se está aplicando un 0 lógico al inversor 7414. A la salida del inversor habrá un 1 lógico.

Si por el contrario, entre el fotodiodo y el fototransistor la trayectoria está obstruida, este último entra en corte, y el voltaje de colector es aproximado a Vcc. Con ello, se aplica un 1 lógico al inversor, el cual lo convierte en 0 lógico.

De esta manera, el fototransistor generará una serie de pulsos - uno por cada paso que gire el motor - que a través de un Schmit-Trigger son realimentados a un puerto de la computadora, a partir de los cuales el programa procesa y calcula la velocidad y posición del motor.

4.4.2 UBICACION DE LOS SENSORES DE POSICION

Dado que los cuatro motores utilizados tienen engranajes reductores de velocidad, el sensor de posición puede colocarse en cualquier punto del tren de engranajes. Sin embargo, el punto más conveniente para colocarlo es en la articulación, que origina el cambio de posición, pues siempre se da un desfase que puede dar lugar a errores de posicionamiento.

De esta forma, cuatro sensores de posición (optoacopladores)

se colocarán en las articulaciones - ejes - donde se origina el movimiento, o sea dos sensores por articulación.

4.4.3 SENSOR DE LA GARRA

La garra del brazo robótico tiene como función principal tomar y transportar algún objeto. Para ello, el computador primero activa la garra hasta que agarre el objeto -cierre- y ejerza la fuerza de aprehensión necesaria sobre el objeto. Para que el computador sepa cuando la garra ha sujetado el objeto, se colocará un sensor de contacto. La PC recibe la señal de retroalimentación del sensor de contacto, la cual se procesa por el computador, el cual decide si se sigue presionando o no. Además, se colocó otro sensor de contacto en la garra, el cual permite saber si la garra está en su máxima abertura.

Una alternativa simple de implementar estos dos sensores es usando microswitch de carrera, el cual posee dos niveles de información, cerrado o abierto.

Uno de los switchs está normalmente abierto y el otro normalmente cerrado. El que esta normalmente abierto se usa para el cerrado de la garra y el normalmente cerrado para la abertura de la garra.

La fuerza de aprehensión que tiene la garra depende del resorte del micro-switch, el cual se puede cambiar fácilmente.

La misma figura 4.12 sirve para los dos casos de microswitch de abertura y cerrado de la garra. El switch esta abierto, con lo cual el puerto de la PC recibe un 0 lógico - note que se usa un CI7414 para que la transición de estados sea fácilmente detectada por el computador .

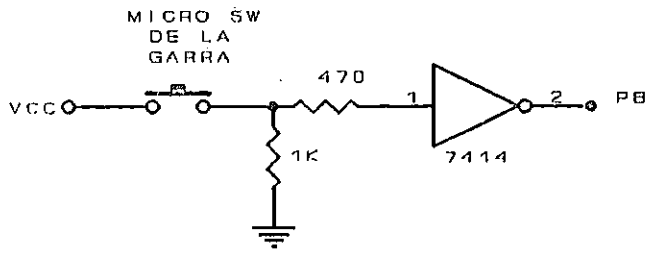


Figura 4.12. Configuración del microswitch de la garra.

Cuando el dedo ejerce fuerza de aprehensión sobre un objeto, después de sujetado, el switch se cerrará y transmitirá un 1 lógico al computador.

El lugar más adecuado para la ubicación del sensor de la garra o microswitchs es en el dedo de la garra. Uno de los sensores

se colocará en el punto de contacto entre el dedo y el objeto a transportar, el otro se coloca en la terminación de la muñeca del enlace del brazo con el dedo de la garra.

4.5 DISEÑO DE LA INTERFASE DE COMUNICACION ENTRE LA COMPUTADORA Y EL PERIFERICO BRAZO ROBOT

En este mismo capítulo se estableció que el sistema será programado desde una computadora. El sistema robótico será visto como un periférico conectado a un puerto de la computadora. Cada periférico presenta características propias, y por lo tanto un problema particular. Los dispositivos periféricos se pueden clasificar, en cuanto a su velocidad en:

- Dispositivos lentos
- Dispositivos de velocidad media
- Dispositivos de alta velocidad.

Existen dos tipos de comunicación con una computadora:

- Comunicación paralelo.
- Comunicación Serie

Describiremos brevemente ambas formas de comunicación para luego seleccionar la más adecuada.

4.5.1 COMUNICACION PARALELO

Con esta comunicación se necesita un número de líneas que depende del tamaño en bits de los datos.

En este método de comunicación un byte entero de información puede transmitirse a la vez, debido a que líneas separadas transmiten un bit al mismo tiempo. Este tipo de manejo de la información es fácil y además es perfectamente entendible.

En el mercado existe un puerto paralelo estandarizado, es la interfase Centronics, utilizada para manejar el impresor. Aunque su forma de manejar la comunicación es en paralelo, presenta el gran inconveniente de que no es bidireccional el flujo de datos.

El puerto paralelo Centronics es un puerto de salida exclusivamente, no de entrada.

Existen diferentes IC especializados que pueden ser utilizados para la construcción de interfases para la comunicación paralelo. Entre ellos tenemos:

- La PIA: contiene puerto de entrada/salida y algunas líneas de control.

- La VIA: incluye dos puertos de entrada/salida, cuatro líneas de control, dos contadores de 16 bits y un registro de desplazamiento de 8 bits.

- La PPI: incluye 4 puertos paralelos. 2 de 8 bits y 2 de 4 bits. Presenta una gran flexibilidad de programación.

4.5.2 COMUNICACION SERIE

En este modo de comunicación, los datos y la información de control se transmite un bit a la vez, utilizando un solo cable más la tierra eléctrica.

Uno de los sistemas de comunicación más populares en formato serie es la interfase RS-232. Cualquier computadora PC o XT de IBM tiene incorporado este sistema.

Arquitectónicamente, la interfase de comunicación RS-232 está basada en el IC-8250. Este circuito integrado convierte los datos paralelos que llegan del microprocesador a datos serie.

En esta interfase, el uno lógico es de -3V hasta -12V y el cero lógico es de 3V hasta 12V.

El uso de la interfase RS-232 reduce la cantidad de líneas a utilizarse, pero incrementa la complejidad de su recepción así como del software a utilizarse.

Ya que la información es enviada un bit a la vez, es necesario en el punto de recepción y transmisión un UART (Universal asyncrhronous receiver transmitter) para realizar esta conversión. Si se utiliza este método de comunicación se debe construir una interfase muy compleja a partir del UART, la cual presentaría poca versatilidad para otras aplicaciones.

4.5.3 SELECCION DEL TIPO DE COMUNICACION

En este punto se definirá el modo de comunicación de la computadora con el sistema. Si se utilizará la interfase paralelo, implícitamente queda establecido lo siguiente:

- Se debe construir la interfase paralelo, ya que la Centronics no nos es útil.

- La cantidad de líneas dependerá de la cantidad de bits a comunicar simultáneamente.

- El software del sistema será relativamente sencillo.

Si por otro lado se considera la comunicación serie, se debe

tener en cuenta lo siguiente:

- La interfase RS-232 se encuentra en la mayoría de PC.
- Habría que adecuar los niveles lógicos del puerto a niveles lógicos utilizado en el sistema robótico.
- Hay que diseñar y construir la interfase receptora en el brazo robótica, a partir de un UART.
- Se debe diseñar el software de recepción; siendo mucho más complicado que el de la comunicación paralelo.

En suma, en ambos casos debemos diseñar y construir una interfase, y en la comunicación serie el software se complica más. Es evidente que la mejor opción es la utilización de una interfase de comunicación paralelo.

4.5.4 DISEÑO DE LA INTERFASE DE COMUNICACION PARALELO

Los criterios de la sección 4.5.3 conducen al diseño de una interfase paralela para PC. El diseño final se obtuvo usando la PPI 82C55, de Intel, por su gran versatilidad y disponibilidad.

4.5.4.1 DESCRIPCION GENERAL

El diagrama de bloques de este dispositivo aparece en la figura 4.13. A continuación se describe cada uno de los bloques y señales de entrada y salida que componen la PPI 82C55A.

DATA BUS BUFFER.

Es un buffer bidireccional de tres estados de 8 bits, que es usado para interfazar la CPU con el 8255.

LOGICA DE LECTURA Y ESCRITURA.

La función de este bloque es la de manejar la transferencia interna y externa de los datos y control o palabras de estado.

CS (CHIP SELECT).

Un cero en esta entrada habilita la comunicación entre el 8255 y la CPU.

RD (READ).

Un cero en esta entrada habilita al 8255 a emitir datos o información de estado a la CPU sobre el bus de datos. En

esencia esta señal permite a la CPU leer desde el 8255.

WR (WRITE).

Un cero en este pin de entrada habilita a la CPU a escribir datos o palabras de control en el 8255.

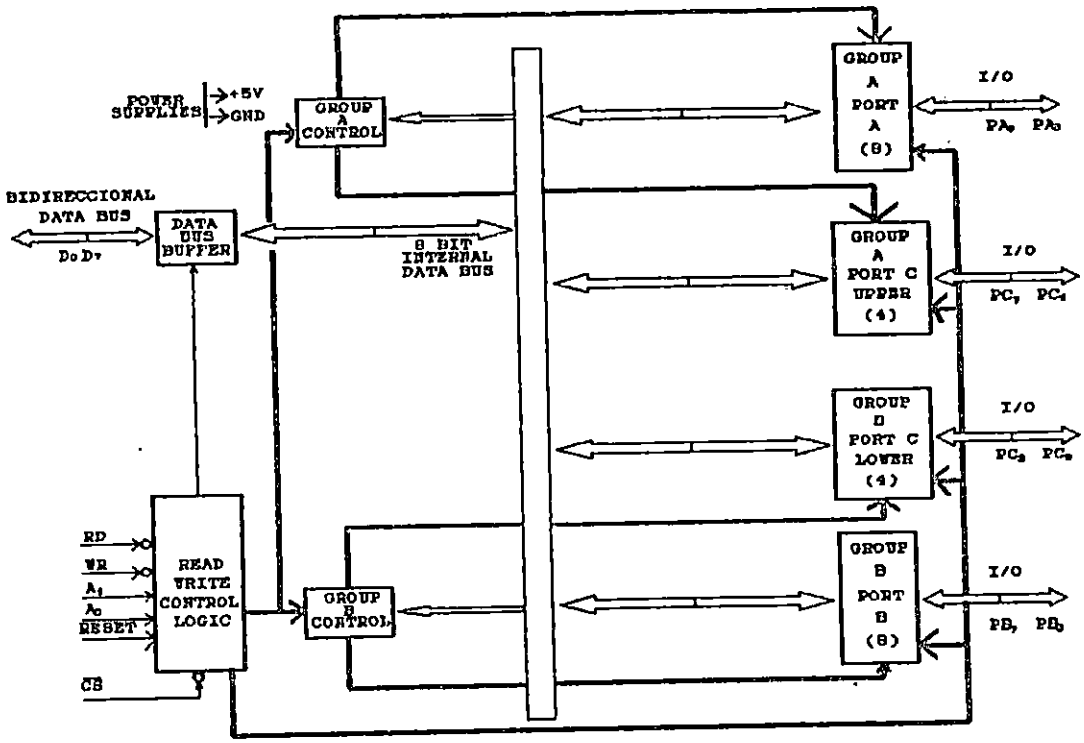


Figura 4.13. Diagrama de bloques de la PPI 82C55A.

AO Y A1 (PORT SELEC 0 Y PORT SELEC 1).

Estas señales de entrada en conjunto con RD y WR, controlan la selección de uno de los tres puertos o registros de la palabra de control y su significado es como sigue:

A1	AO	RD	WR	CS	OPERACION DE ENTRADA	
0	0	0	1	0	PA	BUS DATOS
0	1	0	1	0	PB	BUS DATOS
1	0	0	1	0	PC	BUS DATOS
					OPERACION DE SALIDA	
0	0	1	0	0	PA	BUS DATOS
0	1	1	0	0	PB	BUS DATOS
1	0	1	0	0	PC	BUS DATOS
1	1	1	0	0	CONTROL	BUS DATOS

					FUNCION DE DESHABILITACION
X	X	X	X	1	3 - STA BUS DATOS
1	1	0	1	0	CONDICION ILEGAL
X	X	1	1	0	3 - STA BUS DATOS

RESET.

Un uno (1) en esta entrada limpia los registros de control y todos los puertos (A, B y C) son colocados en modo de entrada.

CONTROLES DEL GRUPO A Y B.

La configuración funcional de cada puerto es programada por el software del sistema. La CPU escribe una palabra de control al 8255. La palabra de control contiene información tal como "modo". "bit set", etc., que inicializa la configuración funcional del 8255.

CONTROL GRUPO A.

Puerto A y líneas más significativas del puerto C.

CONTROL GRUPO B.

Puerto B y líneas inferiores del puerto C.

PUERTOS A, B Y C.

El 8255 contiene 3 puertos de 8 bits (A, B y C). Todos pueden ser configurados en una amplia variedad de características funcionales por el software del sistema, pero cada uno teniendo sus propia característica.

PUERTO A.

Un buffer/latch de salida de 8 bits y un buffer de entrada de 8 bits.

PUERTO C.

Un latch/buffer de salida de 8 bits y buffer de entrada de 8 bits. Este puerto puede ser dividido en dos puertos de dos de 4 bits por medio de la palabra del modo del control. Cada puerto de 4 bits contiene un latch y este puede ser usado para el control de señales de salida y estado de señales de entrada en conjunto con los puertos A y B.

4.5.4.2. DESCRIPCION OPERACIONAL DEL 8255.

SELECCION DEL MODO.

Hay tres modos básicos de operación que pueden ser

seleccionados por el software del sistema:

MODO 0: Entrada/Salida básica.
MODO 1: Entrada/Salida con Handsheking.
MODO 2: Bus bidirireccional con Handsheking.

La figura 4.14, muestra como debe ser el formato de la palabra de definición del modo operación.

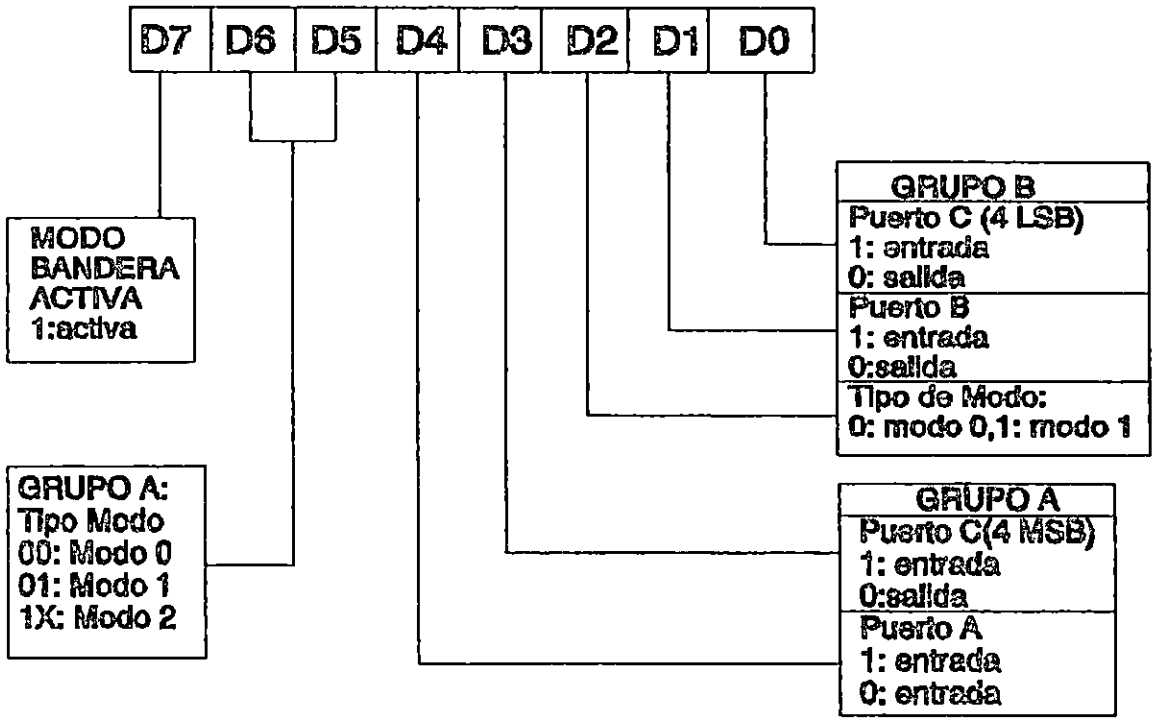


Figura 4.14. Formato de definición del modo.

CARACTERISTICA SET/RESET DE UN SOLO BIT.

Cualquiera de los ocho bits del puerto C puede ser SET o RESET. Usando una sola instrucción de salida. Esta característica reduce requerimientos de software en aplicaciones basadas en control.

La figura 4.15, ilustra el formato de definición para funciones de bit SET/RESET del puerto C.

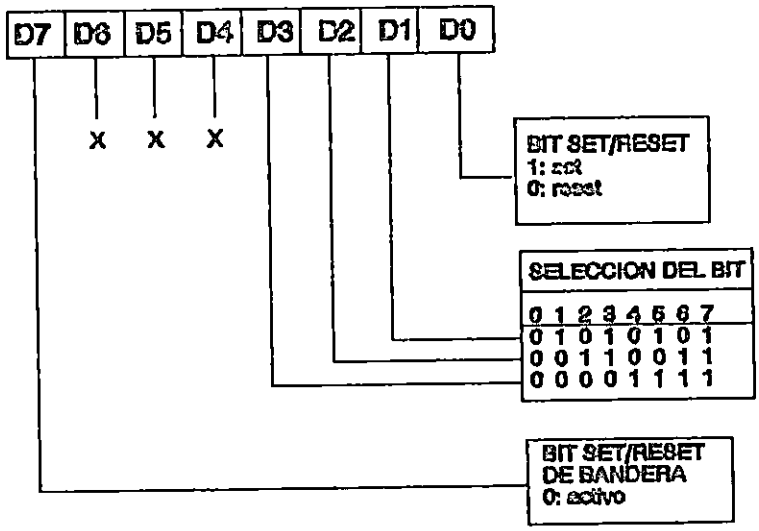


Figura 4.15. Formato Bit SET/RESET.

4.5.4.3 MODOS DE OPERACION

MODO 0.

Esta configuración provee operaciones de E/S simple para cada uno de los tres puertos. No se requiere "handshaking", los datos son simplemente leídos o escritos desde un puerto específico.

DEFINICIONES FUNCIONALES BASICAS EN MODO 0:

- Dos puertos de 8 bits y dos de 4 bits.
- Cualquier puerto puede ser E/S.
- Las salidas son enclavadas (latched).
- 16 configuraciones de E/S son posibles en este modo.

MODO 1.

Esta configuración provee un medio para transferir datos de E/S desde un puerto específico en conjunto con señales strobes o "handshaking". En modo 1 el puerto A y el puerto B usas las líneas del puerto C para generar o aceptar estas señales de "handshaking".

DEFINICIONES FUNCIONALES BASICAS EN MODO 1:

- Dos grupos de 12 líneas (Grupo A y Grupo B).
- Cada grupo contiene un puerto de 8 bits y un puerto de control de datos de 4 bits.
- El puerto de 8 bits puede ser de entrada o salida;

ambas salidas y entradas son enclavadas.
- El puerto de 4 bits es usado para control de un puerto de 8 bits.

DEFINICIONES DE LAS SEÑALES DE CONTROL DE ENTRADA.

STB (STROBE INPUT).

Un nivel bajo en esta entrada, carga de datos en el "latch" de entrada.

IBF (INPUT BUFFER FULL F/F).

Un nivel alto en esta salida indica que los datos han sido cargados en latch de entrada.

INTR (INTERRUP REQUEST).

Un nivel alto en esta salida puede ser usado para interrumpir a la CPU cuando un dispositivo de entrada esta solicitando servicio. INTR es vuelto a su estado inicial por la transición negativa de RD. Este procedimiento permite a un dispositivo de entrada, solicitar servicio desde la CPU por simple "strobing".

INTEA.
Es controlado por el bit SET/RESET de PC4.

INTEB.
Es controlada por el bit SET/RESET de PC2.

4.5.4.4 DESCRIPCION DEL SLOT DE LA PC

La interfase de comunicación del perifrico (Brazo Robot) con la computadora requiere de las ranuras de expansión de los canales I/O del BUS PC AT o compatibles.

A continuación se da una breve información de este BUS PC (SLOT).

El BUS de la PC tiene 62 contactos tipo borde de tarjeta (SLOT), 31 por cada cara (A/B), espaciados 0.1 pulgadas, ver figura 4.16. Las señales disponibles en el BUS de la PC son:

CLOCK (S): Es el reloj del sistema. Su frecuencia depende del tipo de aparato. Suele ser 4.7, 6, 8, 12, 16 y 33 MHz.

RESET (DRV) (S): Inicializa el sistema al encender la máquina.

SD0-SD7 (E/S): Bus de datos de 8 líneas de Entrada/Salida.

SA0-SA19 (E/S): Bus de direcciones de 20 líneas que determinan el máximo de memoria direccionable = 1 MByte.

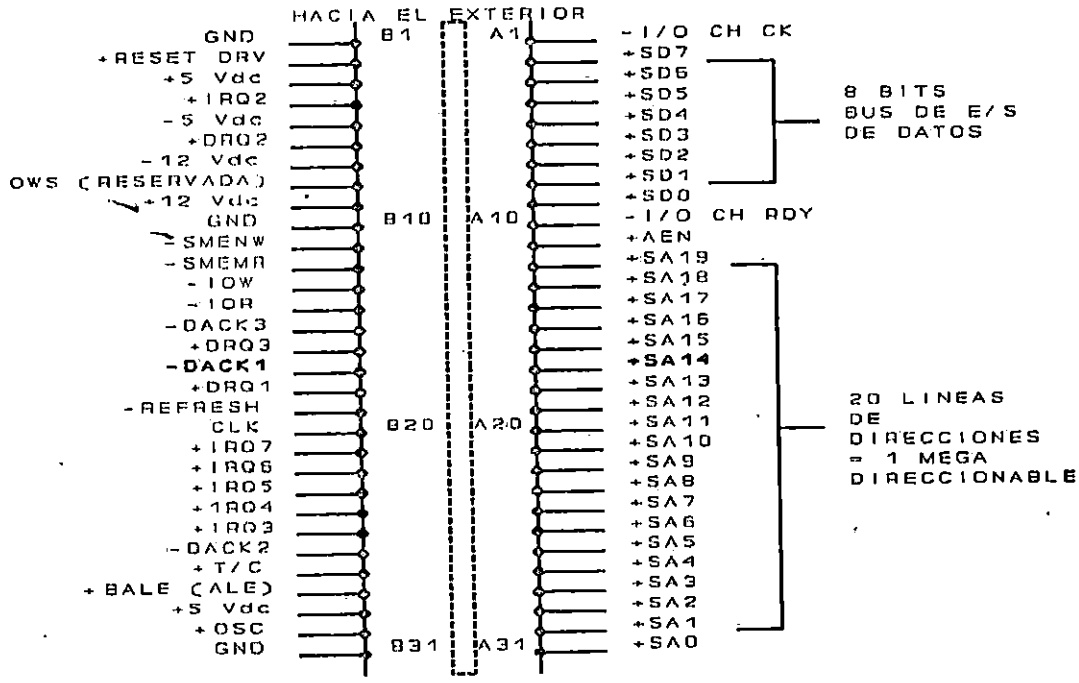


Figura 4.16. Bus de la PC (SLOT).

SA0 es el bit LSB y SA19 es el bit MSB.
Señales de salida generadas por el procesador o por el controlador de DMA cuando ste toma el control. Contactos SA31-SA12 del BUS.

IRQ2-IRQ7 (E): Petición de interrupción.
Son 6 líneas que se utilizan para indicar al procesador que algún perifrico requiere su atención.
IRQ2 es la señal más prioritaria y IRQ7 la menos prioritaria.

4 LINEAS DE CONTROL DE LECTURA/ESCRITURA:

2 Líneas para acceso a memoria:

-SMEMR (S): Indica a la memoria que el dato situado en el BUS ha sido leído. El signo - significa que es activa a nivel bajo. Es activa solo para rangos menores de 1 MBYTE.

-SMEMW (S): Indica a la memoria que guarde el dato situado en el BUS. Controlada por el CPU o el controlador DMA. Activa a nivel bajo. Es activa solo para rangos menores de 1 MBYTE.

2 Líneas de acceso a dispositivos externos o puertos:

-IOR (E/S): Indica a los perifricos la lectura del dato situado en el BUS. Controlada por el procesador o por el controlàdor de DMA o similares presentes en el canal entrada/salida. Activa en nivel bajo.

-IOW (E/S): Indica a los perifricos la escritura de un dato situado en el BUS. Controlada por el procesador o por el DMA. Activa en el nivel bajo.

6 Líneas para acceso directo a memoria (DMA):

DRQ1-DRQ3 (E): Petición de DMA por los perifricos. DRQ1 es la de mayor prioridad y DRQ3 la de menor prioridad. Se mantienen en alto hasta que el correspondiente DACK se activa.

DACK1-DACK3 (S): Reconocimientos de DMA. Activas nivel bajo.

AEN (S): Cuando es activa a nivel alto, el DMA controla el BUS de direcciones, BUS de datos y líneas de lectura/escritura.

T/C (S): Proporciona un pulso alto cuando es alcanzada la cuenta final o el ciclo es terminado por cualquier canal DMA.

-I/O CH CK (E): Proporciona al sistema información de error de paridad en memoria o dispositivos en el canal de E/S. Es activa en nivel bajo.

-I/O CH RDY (E): Activa en nivel bajo. Indica que el perifrico no esta listo, permitiendo prolongar los ciclos de E/S. Es utilizado por sistemas lentos.

OSC (S): Señal de reloj de 14,31818 MHz, no sincronizada.

-REFRESH (E/S): Activa baja. Puede controlar un procesador en canal E/S.

BALE (ALE) (S): Buffered Address Latch Enable. Esta señal proviene del controlador del bus (82288) y es utilizada para enclavar direcciones válidas del microprocesador. Se utiliza como indicador de direcciones válidas del DMA o CPU.

4 Niveles distintos de tensión de alimentación: + 5, - 5, -12 y +12 voltios de corriente continua.

4.5.5 INTERFASE PARALELO CON PPI 82C55

La interfase requerida para enviar y recibir la información desde el computador hacia el brazo robot será para comunicación paralelo.

Esta interfase posee 3 elementos: PPI 82C55A, BUFFER MM74C245 (CMOS) de 8 bits y el circuito de decodificación.

Estos componentes asociados con la PPI responderán a las interrupciones, transferencia y recepción de datos (información), etc., la interfase de comunicación paralelo se describe a continuación y se muestra en la figura 4.17.

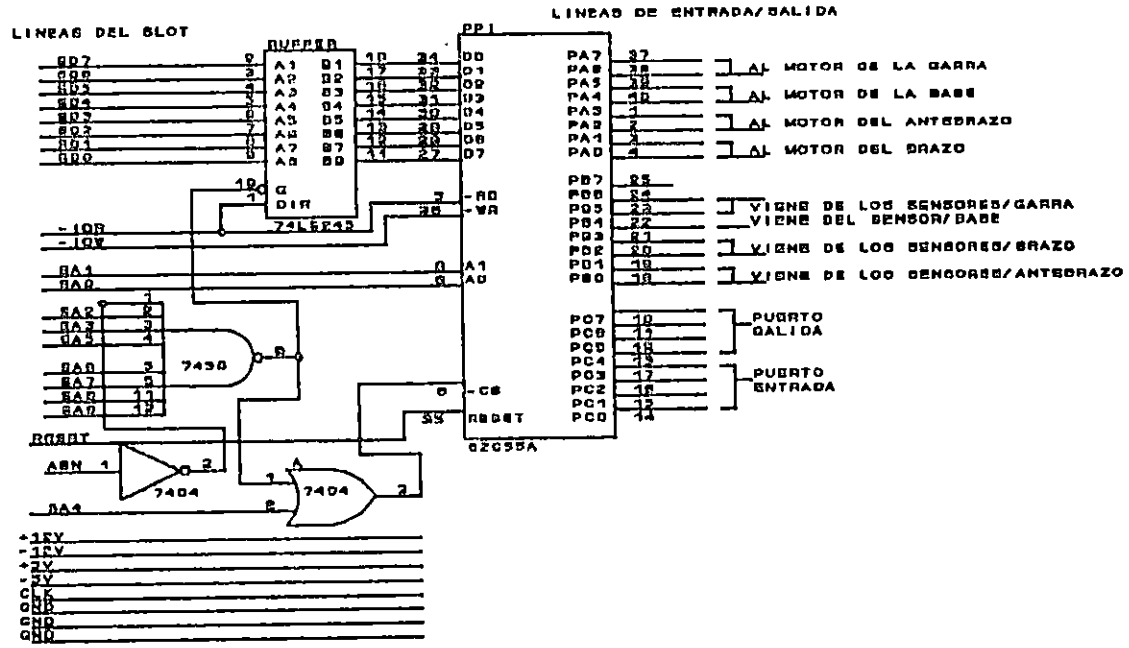


Figura 4.17. Interfase paralelo basada en PPI 82C55

Buffer 74245.

El buffer MM74C245 sirve para aislar los datos SD0-SD7 (provenientes del slot) de los circuitos externos o periféricos (circuitos de manejo para cada motor de paso del sistema robótico), y el buffer es habilitado por la compuerta NAND de 8 entradas en nivel bajo. La entrada -IOR habilita el sentido del flujo de datos del buffer.

Circuito de decodificación.

Como circuitos de decodificación se utilizará la compuerta NAND MM74C30 de 8 entradas a las cuales se conectan las líneas de dirección SA2, SA3, SA5, SA6, SA7, SA8, SA9 (del slot de la PC) y la señal negada de AEN. Esto último evita que la interfase sea acceda por el controlador del DMA en alguna transferencia por canales DMA.

La decodificación se complementa con una compuerta OR MM74C32 de dos entradas, a las cuales se conecta SA4 y la salida de la NAND de 8 entradas. La salida de esta OR habilita en bajo el 8255A. La salida de la NAND de 8 entradas habilita además al buffer MM74C245 cuando esta en bajo.

En general los circuitos de decodificación en la interfase paralelo sirven para habilitar o deshabilitar el buffer y la comunicación entre la CPU, PPI 8255 y periférico.

PPI 82C55A.

Es uno de los dispositivos principales del diseño y proporciona 3 puertos de 8 bits programables como entradas o salidas. Utiliza 1 dirección para cada puerto y otra adicional para el control de puertos.

Su función principal es la comunicación entre la CPU y los dispositivos periféricos, el circuito integrado 82C55A es un componente programable de entradas o salidas paralelas de propósito general. Contiene 3 puertos paralelos de entrada/salida de 8 bits cada uno: puerto A, puerto B y puerto C. Puede configurarse para operar con señales de protocolo.

Cuando se inicializa el 82C55A (al energizar la máquina), sus tres puertos quedan configurados como puertos de entrada. El 82C55A utiliza una dirección para cada puerto y otra adicional para el control de puertos o sea que se requieren 4 direcciones para accesar los puertos de este dispositivo.

4.6 CONEXION DEL SISTEMA ROBOTICO A LA INTERFASE

La interfase con la PPI nos proporciona 3 puertos de 8 bits cada uno. Para el control del sistema robótico se han utilizado todas las líneas, en la manera que se describe a continuación:

Puerto A:

Se ha configurado como puerto de salida, y los ocho bits controlan los cuatro motores del brazo robot. Por cada motor se utilizan 2 bits.

Puerto B:

Se ha configurado como puerto de entrada, y se utiliza para recibir la información de los sensores. Los sensores de la articulación del brazo y antebrazo utiliza 4 bits, los dos sensores de la garra usan 2 bits. Un séptimo bit es utilizado para un optoacoplador de la base, el cual le servirá para referencia inicial al motor de la base.

Puerto C:

Los 4 bits menos significativos de este puerto se han configurado como salida, y los 4 bits más

significativos como entrada. Este puerto se ha dejado para las necesidades del usuario, el cual puede manipularlos mediante el software.

En la figura 4.17 se detalla la conexión de los diferentes puertos.

CONCLUSIONES DEL CAPITULO IV

Podemos decir, que con el capitulo III y IV el diseño de las partes del hardware quedan completamente definido.

En el desarrollo del sistema eléctrico se hizo énfasis en la simplicidad de sus componentes, con lo cual se facilita su construcción y depuración.

La forma en que se ha descrito el diseño permite construirla y depurarla en una forma modular, es decir etapa por etapa, lo cual redundará en menor tiempo perdido en la depuración global del sistema, así como asegurar el buen funcionamiento final.

El diseño eléctrico desarrollado puede ser usado para cualquier brazo robot cuyos actuadores sean motores de paso.

REFERENCIAS BIBLIOGRAFICAS

[1] Andeen, Gerry B. Robot Desing Handbook. Ed. Mac Graw Hill, 1a Edición, 1988.

[2] Del Toro, Vincent. Fundamentos de Ingeniería Eléctrica. Prenctice Hall Hispanoamericana, S. A. 2a. Edic. 1988.

[3] Grupo del Buch Engineering Co. Inc. Robotics Trainings Systems, Concepts and Aplications.

[4] Intel Corporation. Microprocessor and Peripheral Handbook 8086, 8088, 80286 & 80386. Santa Clara, Cal.: Intel Corporation, 1987.

[5] Peralta, Juan y Portillo, Marcos. Diseño del Sistema Locomotor y Sensor de un Microrobot. Universidad de El Salvador, proyecto. 1992.

[6] Platero, Marco y Escalante, Herberth. "Diseño y Construccion de un Dispositivos de Distorsión Armónica en Redes de Baja Tension". Universidad de El Salvador, tesis. 1991.

[7] Texas Instrument. The TTL Data Book for Design Engineers. 2a. Edición. Houston, Texas: Texas Instrument, 1981.

CAPITULO V

PROGRAMACION Y LENGUAJES DE ROBOT.

Introducción

El objetivo de este capitulo es introducir los conceptos relacionados con la programación de brazos robot, así como introducir la teoría general de los lenguajes existentes actualmente para el control de brazos robot.

Inicialmente se introduce el concepto de programa desde el punto de vista robótico, así como los diferentes métodos de programación existentes. Se continua describiendo los tipos de movimiento que puede ejercer el brazo, así como la interpolación del movimiento.

Posteriormente se detallan la estructura del lenguaje orientado a la programación de robots. Para finalizar se introduce una serie de instrucciones que actualmente poseen la mayoría de lenguajes de robot de aplicaciones industriales.

5.1 CONCEPTO DE PROGRAMA DE ROBOT

Desde el punto de vista de la informática, un programa es: "un conjunto de instrucciones secuenciales, correspondiente a un algoritmo escrito en algún lenguaje de programación, con las que se puede realizar un trabajo determinado mediante la ejecución de tales instrucciones por parte de la computadora".

En robótica, como una primera aproximación, un programa de robot se puede definir como una trayectoria en el espacio a través de la cual se ordena al manipulador que se desplace, y este movimiento también incluye el control del efector final o garra.

En realidad, un programa de robot es más que eso, ya que un robot puede hacer mucho más que mover su brazo a lo largo de una serie de puntos dentro de un espacio. Los robots actuales pueden aceptar datos de entrada procedentes de sensores y otros dispositivos; pueden enviar señales a elementos del equipo que operan con ellos; pueden tomar decisiones y otras muchas acciones que requieren.

En suma, un programa de robot es una serie de instrucciones especializadas, cuyo objetivo principal es indicar al manipulador que movimiento debe realizar, y además, le permiten interactuar con el medio y tomar decisiones en base a las condiciones presentes.

Un programa define una trayectoria en el espacio a lo largo de la cual el robot mueve el efector final. Puesto que el robot consta de varias articulaciones (ejes) unidas, la definición de la trayectoria en un espacio requiere que el robot mueva sus ejes hacia varias posiciones con el objeto de seguir esa trayectoria. Para un robot de tres ejes, cada punto de la trayectoria consta de tres valores de coordenadas.

Estas coordenadas pueden estar en coordenadas cartesianas (en robótica también llamadas coordenadas universales X,Y,Z) o en coordenadas de articulación. Se definen las coordenadas de articulación como la posición de cada una de las articulaciones, y habrá tantas coordenadas de articulación como ejes tenga el brazo. Para un brazo articulado, dichas coordenadas suelen darse en medidas angulares (grados, radianes, etc.), y describen la posición angular de cada uno de los enlaces respecto a una referencia dada.

Hay que hacer notar, que un punto dado en coordenadas universales, puede corresponde a más de un conjunto posible de valores de coordenadas de la articulación que se pueden utilizar para que el robot alcance ese punto. Por ejemplo, existen dos configuraciones de ejes alternativas que pueden ser utilizadas por el brazo de la figura 5.1.

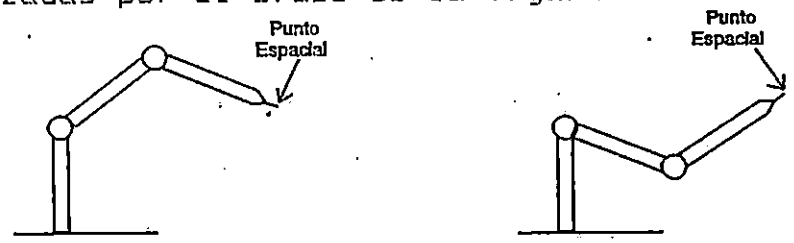


Figura 5.1. Dos configuraciones de ejes alternativas con efectores finales localizados en un mismo punto.

Partiendo de ello, la especificación de un punto en un espacio no define, de forma unívoca, las coordenadas de las articulaciones del robot. Por ello, se redefinirá un programa de robot, como aquel que describe una secuencia de posiciones coordenadas de la articulación en vez de una trayectoria en el espacio.

5.2. METODOS DE PROGRAMACION DEL ROBOT

La programación del robot se puede realizar de varias formas. En la práctica industrial actual, existen dos tipos básicos: el método de aprendizaje directo y los lenguajes textuales del robot.

Método de Aprendizaje Directo

En la programación de aprendizaje directo se mueve el robot a

lo largo de la trayectoria del movimiento deseado con el objeto de almacenar esta información en la memoria del controlador. Hay dos modos de realizar la programación de aprendizaje directo:

1. Aprendizaje directo motorizado,
2. Aprendizaje directo manual.

En el aprendizaje directo motorizado se hace uso de un control de mandos, el cual controla los distintos motores de las articulaciones y la garra a través de una serie de puntos en el espacio. El control de mandos, también llamado dispositivo suspendido de enseñanza o control manual, suele ser un dispositivo por teclas, botones, pulsadores o joystick, mediante los cuales se regulan los movimientos físicos del robot y las capacidades de programación. El modo de aprendizaje directo motorizado es el más utilizado en la actualidad por la robótica industrial, y se limita a los movimientos punto a punto, ya que el control de mando no se presta para definir trayectorias complejas. Entre las aplicaciones que requieren movimientos de punto a punto, se encuentran las tareas de transferencia de piezas, carga y descarga de máquina y soldadura por puntos.

En el modo motorizado, la forma de programar el robot, es ejecutando los movimientos del robot mediante el control de mandos, el cual tiene control directo e independiente sobre cada articulación. Es necesario mencionar que los dispositivos de enseñanza modernos permiten no solo controlar los movimientos, sino que también pueden generar funciones auxiliares como: selección de velocidades, generación de retardos y funciones especiales.

El procedimiento de programar se lleva normalmente con los siguientes pasos: 1) Dirigiendo al robot con un movimiento lento utilizando el control manual para realizar la tarea completa y grabando los ángulos del movimiento del robot en la memoria del controlador; 2) Reproduciendo y repitiendo el movimiento enseñado; 3) si el movimiento enseñado es correcto, entonces se hace funcionar al robot en el modo repetitivo.

En el método de aprendizaje directo manual, el programador agarra físicamente el brazo del robot y la garra, y los mueve manualmente a lo largo del ciclo de movimiento deseado. Si el brazo es demasiado grande, se suele utilizar un modelo a escala del brazo, el cual también es denominado sistema maestro-esclavo. Este modelo tiene la misma geometría que el brazo, y a diferencia de éste, permite manipularse fácilmente durante la programación. Los pasos en la programación usando el método manual son similares a los tres pasos descritos anteriormente. La diferencia fundamental radica en el paso 1, en el cual el programador no usa un control de mandos para realizar la tarea, sino que lo manipula físicamente el brazo, o por medio de un sistema maestro-esclavo.

El método de aprendizaje directo manual se utiliza con más frecuencia en aplicaciones donde el brazo requiere complejos movimientos curvilíneos. Una aplicación común de este tipo es la pintura por pulverización, en el cual la muñeca del robot debe efectuar un movimiento que sigue un patrón regular y uniforme, con objeto de aplicar la pintura uniformemente sobre la superficie total a recubrir. Otra aplicación es la soldadura de arco continuo.

En los métodos de aprendizaje directo, el usuario no necesita conocer ningún lenguaje de programación, simplemente debe habituarse al empleo de los elementos que constituyen el dispositivo de enseñanza. Es de hacer notar que en el proceso de programación se necesita al propio robot para la confección del programa.

Lenguajes Textuales de Robot.

La programación de robot con lenguajes textuales se realiza, en cierto modo, lo mismo que la programación de computadoras. El programador introduce por teclado el programa en un monitor utilizando un lenguaje de alto nivel orientado a robots. Este procedimiento se suele completar con la utilización de técnicas de aprendizaje directo para señalar al robot la posición de los puntos dentro del espacio de trabajo. En este método de programación, el programador debe tener conocimientos generales de programación, y en algunos casos de técnicas de programación estructurada. En la sección 5.5 se ampliará el método de programación por lenguajes textuales.

5.3. TIPOS DE MOVIMIENTOS DEL ROBOT

Cuando se programa un robot, no sólo se está interesado por los puntos finales alcanzados por la articulación del robot, sino también por la trayectoria seguida por el brazo en el recorrido desde un punto a otro en el espacio de trabajo.

Los tipos de movimiento más comunes que un manipulador puede realizar durante el recorrido desde un punto a otro son: movimiento de articulación, movimiento de coordenadas XYZ, movimiento de sesgo, movimiento interpolado de articulación Y movimiento interpolado de línea recta.

En los movimientos de articulación y de coordenadas XYZ, se hace uso de un control de mandos para ejecutar el movimiento.

Movimiento de Articulación

Este es el movimiento más básico, y se refiere al movimiento de cada una de la articulaciones en una u otra de sus direcciones, hasta que el efector final haya alcanzado la posición deseada. Para controlar el movimiento de cada articulación, se hace uso de un control de mandos, el cual

permite el control independiente de cada articulación.

Movimiento de Coordenadas XYZ.

El movimiento de coordenadas XYZ, mediante el uso de un control de mandos permite al programador desplazar la garra en movimientos paralelos a los ejes XYZ. Para ello, el controlador define un sistema convencional de coordenadas cartesianas con origen en alguna posición en el cuerpo del robot. Para el robot de brazo articulado, el controlador debe resolver un conjunto de ecuaciones matemáticas para convertir los movimientos de la articulación rotacional del robot en el sistema de coordenadas cartesianas. Estas conversiones son realizadas de tal manera que el programador no tiene que preocuparse por los cálculos importante, que son desarrollados por el controlador.

Movimiento de Sesgo

Los movimientos de sesgo representan el tipo de movimiento más simple. Cuando el robot recibe órdenes para realizar un recorrido desde un punto A hacia un punto B, cada eje del manipulador se traslada a la máxima velocidad, desde su respectiva posición inicial a su posición final requerida. Por lo tanto, todos los ejes comienzan su movimiento al mismo tiempo, pero no terminan en el mismo instante. La articulación que recorre la menor distancia termina antes que las demás. El movimiento de sesgo suele dar lugar a un desgaste innecesario en las articulaciones (ya que se mueve a su máxima velocidad) y suele llevar a resultados imprevistos en lo que respecta a la trayectoria tomada por el manipulador.

Por ejemplo, para un manipulador de tres ejes, que debe recorrer las siguientes distancias utilizando un movimiento de sesgo: la articulación 1, 30°; la articulación 2, 60° y la articulación 3, 90°. Si todas las articulaciones se desplazan a una velocidad rotacional máxima de 30°/segundo, los tiempos para cada articulación se describen a continuación:

La articulación 1, realizará su movimiento en 30°/(30°/segundo) = 1 segundo,

La articulación 2, lo hará en 60°/(30°/segundo) = 2 segundos

La articulación 3, en 90°/(30°/segundo) = 3 segundos.

Las tres articulaciones iniciarían el movimiento en t = 0, pero terminarían en t = 1 seg, t = 2 seg y t=3 para la articulación 1,2 y 3, respectivamente.

Movimiento Interpolado de Articulación.

El movimiento interpolado de la articulación exige al controlador del robot calcular el tiempo que necesitará cada

articulación para alcanzar su destino a la velocidad requerida. Luego, selecciona, entre ellos, el tiempo máximo y lo utiliza como el tiempo para todos los ejes. Esto significa que para cada uno de los ejes, se calcula una velocidad por separado . La ventaja del movimiento interpolado de la articulación con respecto al movimiento de sesgo es que las articulaciones suelen ser impulsadas a una velocidad más baja que sus respectivas velocidades máximas, con lo que se reduce los problemas de mantenimiento para el robot. Además, la trayectoria seguida por el manipulador se puede repetir y predecir sin considerar el tiempo total y la velocidad requerida. Para ilustrar este tipo de movimiento, se da un ejemplo.

Un brazo de tres ejes, se desea que recorra las siguientes distancias con movimiento interpolado de las articulaciones: la articulación 1, 30°; la articulación 2, 60° y la articulación 3, 90°. La articulación 3 tiene la distancia más grande para recorrer y necesitará el tiempo máximo para realizar su movimiento. Si la velocidad máxima rotacional es de 30°/s, la articulación 3 recorrería los 90° en un tiempo:

$$90°/(30°/\text{segundo}) = 3 \text{ segundos}$$

La velocidad de la articulación 2 es:

$$60°/\ 3 \text{ segundos} = 20°/\text{segundo}$$

y para la articulación 1 es:

$$30°/\ 3 \text{ segundos} = 10°/\text{segundo}$$

Si bien, las tres articulaciones iniciarán el movimiento en t=0 y lo terminarán en t= 3 seg, las velocidades serán diferentes. Respectivamente la articulación 1, 2 y 3 se moverán a una velocidad 10°/seg, 20°/seg y 30°/seg. Se concluye, que el desgaste será mayor para la articulación que recorrerá mayor distancia.

Movimiento Interpolado de Línea Recta.

El movimiento de interpolación en línea recta exige que el extremo del manipulador (o efector final) se desplace a lo largo de una trayectoria rectilínea definida en coordenadas cartesianas. Este es el tipo de movimiento más exigente para el controlador de un brazo articulado. Para manipuladores con articulaciones rotacionales, la mayoría de los movimientos en línea recta no son naturales y el controlador calcula el camino en línea recta entre los dos puntos y desarrolla la secuencia de puntos direccionables a lo largo del camino a través del cual pasa el robot.

Para ordenar al brazo que se desplace desde el punto A al punto B a lo largo de una trayectoria lineal, se debe calcular

la serie de transformaciones intermedias -entre coordenadas de articulación y cartesianas- a lo largo de la trayectoria en línea recta (recordemos que las coordenadas de articulación corresponden con la posición angular de los motores). Estas transformaciones están separadas cierta distancia, determinada por el tiempo que el controlador del brazo ejecute las transformaciones y resuelva la cinemática del brazo.

En este punto, hay un choque entre la precisión y la velocidad del movimiento. Para que el efector final describa una línea recta entre dos puntos, se hace necesario definir la mayor cantidad de puntos posibles que se acerquen a dicha trayectoria. Pero, entre mayor sea la cantidad de puntos definidos, el tiempo de cálculo será mayor y afectará la velocidad de movimiento del brazo. Se puede establecer la siguiente conclusión: la precisión en el movimiento de línea recta dependerá de una velocidad de cálculo elevada, así como de la máxima velocidad deseada en el movimiento del efector.

5.4 CONSIDERACIONES DINAMICAS EN EL CONTROL DE VELOCIDAD

La mayoría de los robots permiten que su velocidad de movimiento sea regulada durante la ejecución del programa. En el método de aprendizaje directo, la velocidad puede ser modificada con el control de mandos. Cuando se utiliza un lenguaje textual, las instrucciones de movimientos deben controlar la velocidad del manipulador. En este caso, el programador no tiene control directo sobre la velocidad en todo el movimiento del brazo, únicamente puede establecer al inicio del movimiento la velocidad máxima que puede alcanzar en el transcurso de éste.

En los robots programados por los métodos de aprendizaje directo, la velocidad no se suele dar como velocidad lineal en el extremo del efector final por varias razones. En primer lugar, la velocidad lineal del robot en el efector final depende de cuántos sean los ejes que se mueven al mismo tiempo y de qué ejes se trate. En segundo lugar, la velocidad del robot depende de la configuración actual de sus ejes. Para finalizar, la velocidad del robot será afectada por la carga que transporte debido a las fuerzas de aceleración y desaceleración.

En los motores de paso, la velocidad angular se puede expresar en pasos/seg. En la figura 5.2 se muestra la característica torque/velocidad de los motores de paso.

En la sección 4.2 se mencionó que los motores de paso giraban un paso -ángulo constante- por cada impulso eléctrico que les llegaba. De esta manera, la velocidad angular de este tipo de motores es controlada por la frecuencia del tren de pulsos que lo exciten.

De la figura 5.2, si estando en reposo el motor, se desea hacer girar a una velocidad alta (enviándole un tren de pulsos de frecuencia elevada), el torque sería pequeño y no vencería la inercia del rotor y el mecanismo conectado a su eje. En forma similar, si el eje del motor gira velozmente, y se para bruscamente -eliminando el tren de pulsos que se le envían-, la misma inercia haría que el motor siguiese girando, sin responder a la excitación eléctrica.

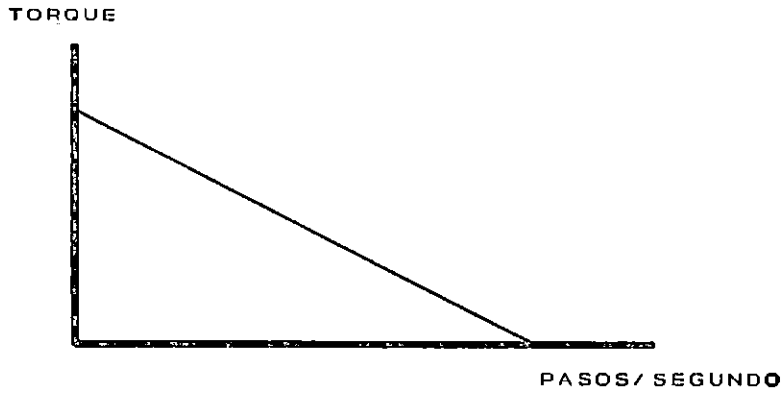


Figura 5.2. Curva Torque/Velocidad de un motor de paso.

Se concluye, que si bien las señales eléctricas aplicadas al motor pueden variar bruscamente, el incremento en la velocidad del motor estará limitada por la inercia del sistema. Así, ante un incremento brusco en los pulsos que excitan el motor, la velocidad del motor no se incrementaría bruscamente por la oposición de la inercia. De la misma manera, si el motor gira a cierta velocidad, y hubiese un decremento brusco en los pulsos aplicados, el eje del motor no bajaría su velocidad tan bruscamente por la oposición de la inercia del sistema.

Ante tal situación, el software debe tomar en consideración el efecto de la inercia, de tal manera que al inicio del movimiento la velocidad se incremente desde el reposo hasta una velocidad máxima y al final del movimiento, la velocidad se decremente hasta llegar a 0, según se muestra en la figura 5.3.

En la figura 5.3 se observa que para distancias cortas, el motor no llega a alcanzar la velocidad máxima. Debido a problemas de aceleración y desaceleración, un robot es capaz de desplazarse en una distancia larga en menos tiempo que una secuencia de distancias cortas, cuya suma sea igual a la distancia larga. Esto es porque, en distancias cortas, puede no permitir al robot alcanzar la velocidad máxima programada.

Otro factor que influye sobre la velocidad es el peso del objeto desplazado. Objetos más pesados significa mayor inercia y cantidad de movimiento y el robot debe accionar con más

lentitud para tratar con seguridad estos factores.

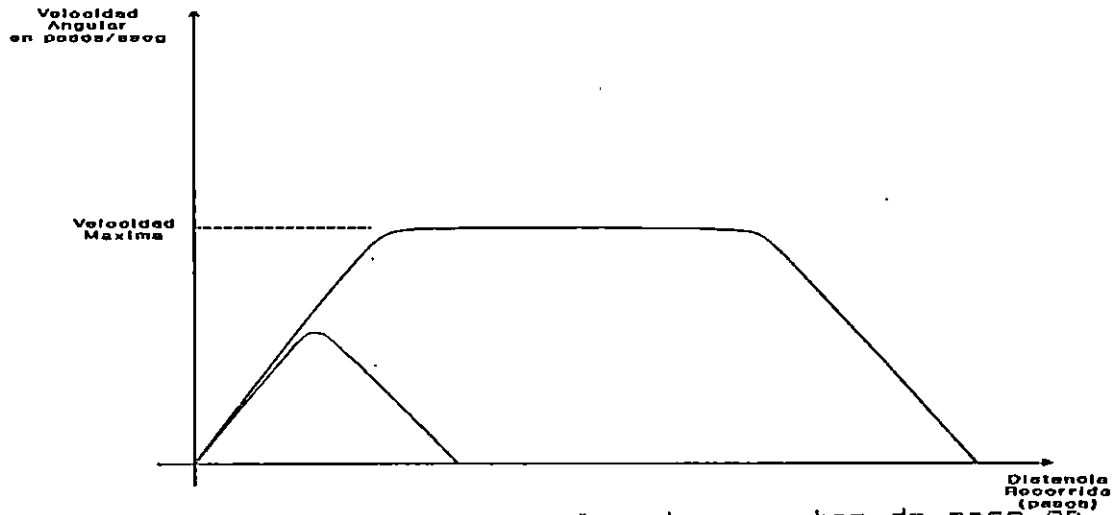


Figura 5.3. Velocidad angular de un motor de paso en función de su distancia angular recorrida.

5.5 LENGUAJES DE PROGRAMACION DE ROBOT

El principal obstáculo cuando se usan manipuladores es la dificultad de una comunicación eficiente y adecuada entre el usuario y el sistema robot, de tal forma que el usuario pueda dirigir al manipulador para cumplir la tarea dada. Las dos maneras comúnmente utilizada para comunicarse con un robot son la enseñanza y repetición (aprendizaje directo) y los lenguajes de programación de alto nivel orientado a robots. El primero de ellos ya fue tratado en secciones anteriores.

Los lenguajes de programación de alto nivel suministran una solución más general para resolver el problema de comunicación hombre-robot. En los sistemas basados en microprocesadores, el lenguaje de alto nivel es el medio que el hombre emplea para gobernar su funcionamiento. Por tanto, el lenguaje orientado a robot debe proporcionar la correcta adaptación con la tarea a realizar y la sencillez de manejo, que se convierten en factores determinante en el rendimiento de los robots.

En las aplicaciones como soldadura o pintura con spray, el método de aprendizaje directo es adecuado porque no hay interacción entre el robot y su entorno y pueden ser fácilmente programados. Sin embargo, la utilización de robots para llevar a cabo tareas complejas requieren técnicas de programación en lenguajes de alto nivel ya que el robot suele confiar en la realimentación de los sensores y este tipo de interacción sólo puede ser mantenida por métodos de programación que contengan condiciones.

La programación de robots es bastante diferente de la programación tradicional. Podemos definir varias consideraciones que debemos mantener para cualquier método de programación de robots: los objetos a manipular por un robot son objetos tridimensionales que tiene numerosas propiedades físicas; los robots trabajan en un espacio especialmente complicado; la descripción y representación de los objetos tridimensionales en un computador es imprecisa y la información de los sensores tiene que ser controlada, manejada y utilizada de forma adecuada.

La solución más común que se toma en el lenguaje del nivel de robot es la de extender un lenguaje de alto nivel ya existente para que cumpla las especificaciones de la programación del robot. Por ejemplo, con el lenguaje Pascal se pueden implementar funciones y procedimientos almacenados en archivos externos denominados unidades; de esa manera, se puede implementar en Pascal una unidad que reúna todos los procedimientos de control del robot, y de esta manera se aprovechan las demás característica que posee Pascal.

5.5.1 GENERACIONES DE LOS LENGUAJES ORIENTADOS A ROBOTS

Los lenguajes textuales de robots actuales poseen una gran variedad de estructura y capacidades, y constantemente se están actualizando con mejores técnicas. En la actualidad se reconocen dos generaciones de lenguajes.

Lenguajes de Primera Generación

La primera generación de lenguajes utiliza una combinación de órdenes y procedimientos del control manual para el desarrollo de los programas de robot. Fueron especialmente desarrollados para implantar un control de movimiento con un lenguaje de programación textual y, por consiguiente, algunas veces se les llama lenguajes de "nivel de movimiento". Entre las características típicas se incluyen la posibilidad de definir los movimiento del manipulador (empleando las órdenes para definir la secuencia de movimiento y el control manual para definir la localización de los puntos), la interpolación de línea recta, la bifurcación y las órdenes elementales de sensor que implica el uso de señales binarias. Pueden utilizarse para definir la secuencia de movimientos del manipulador, tienen las capacidades de entrada/salida y se pueden utilizar para escribir subrutinas.

Entre las limitaciones más comunes de los lenguajes de la primera generación se incluyen la incapacidad para realizar cálculos aritméticos complejos para su uso durante la ejecución del programa, la incapacidad para hacer uso de sensores complejos (por ejemplo, detección de imágenes) y la capacidad limitada para comunicarse con otros computadores.

Lenguajes de la segunda generación

La segunda generación de lenguajes salva algunas de las limitaciones de los lenguajes de la primera generación y aumentan estas capacidades mediante la incorporación de característica que hacen que el robot parezca más inteligente. Activan al robot para que realice tareas más complejas. Estos lenguajes han llamados lenguajes de programación estructurada porque poseen las construcciones de control estructuradas utilizadas en los lenguajes de programación de la computadora. La programación en estos lenguajes es muy similar a la programación de computadores. Esto podría considerarse como una desventaja puesto que se requieren las capacidades de los programadores de computadoras para realizar la programación. Los lenguajes de la segunda generación a menudo hacen uso del control manual para definir las posiciones en los espacios de trabajo.

Las características y capacidades de los lenguajes de la segunda generación se listan a continuación:

1. Control de movimiento. Esta característica es básicamente la misma que para los lenguajes de la primera generación. Los tipos de movimientos se han expandido; aparte del movimiento en línea recta del efector, se han agregado movimiento circulares, cilíndricos y otros.

2. Capacidades de sensor avanzadas. Entre las mejoras de los lenguajes de la segunda generación se incluyen la capacidad para tratar con más que señales binarias y la capacidad para controlar los dispositivos por medio de los datos del sensor. Tienen la capacidad para utilizar señales analógicas y binarias, y además pueden comunicarse con otros dispositivos controlados por esas señales.

3. Inteligencia limitada. Esta es la capacidad que utiliza la información recibida sobre el entorno de trabajo para modificar el comportamiento del sistema de una manera programada. Con esta característica, el robot tiene la capacidad para tratar con sucesos irregulares que suceden durante el ciclo de trabajo de modo que parezca inteligente. El robot da la apariencia de conducirse de un modo inteligente, pero funciona bajos los algoritmos que han sido programados dentro del controlador.

4. Comunicaciones y procesamiento de datos. Los lenguajes de la segunda generación suelen tener medios para interactuar con computadoras y bases de datos de computadoras con el propósito de almacenar registros, generar informes y controlar las actividades de la célula de trabajo.

5. Extensibilidad. Esta característica significa que el

lenguaje puede ser extendido o incrementado por el usuario para satisfacer los requerimientos de la aplicaciones futuras, de los futuros dispositivos de detección y de los futuros robots que pueden ser más sofisticados que los de la época en la que el lenguaje se desarrollo inicialmente. También significa que el lenguaje puede ser expandido mediante el desarrollo de ordenes, subrutinas y funciones que no están incluidas en el conjunto inicial de instrucciones.

5.5.2. ESTRUCTURA DE LOS LENGUAJES DE ROBOT

Los lenguajes que funcionan con sistemas robóticos deben ser capaz de soportar la programación del robot, el control del manipulador del robot y la interconexión con los periféricos dentro del espacio de trabajo. En vista que ninguno de los lenguajes de alto nivel de uso general (PASCAL, BASIC, C, etc.) no poseen instrucciones y comandos que emplean los robots, ha obligado a que los investigadores y constructores en robótica diseñen lenguajes propios.

El software utilizado en la programación de brazos robot pueden dividirse en dos áreas:

1. Programas de aplicación.

2. Sistema Operativo

Los programas de aplicación contiene la información para que el robot ejecute las tareas de una manera ordenada. Este es un programa escrito por el usuario en un lenguaje de robot específico, el cual resuelve problemas individuales del mismo. El lenguaje de robot está definido por un conjunto de instrucciones y reglas sintácticas.

El sistema operativo le dice al brazo robot como ejecutar las tareas que tiene que realizar, establece un protocolo para los programa de aplicación; el sistema operativo es el software que soporta el funcionamiento interno del sistema de la computadora, y su propósito es facilitar el funcionamiento del sistema global al usuario obteniendo el máximo rendimiento y eficiencia del sistema robótico.

El sistema operativo surge de la necesidad que el robot debe poseer un ambiente para su propia programación y así poder ejecutar las tareas de una forma fácil. Un sistema operativo consiste de comandos e instrucciones que hacen más fácil de usar un robot. Incluye además el soporte necesario para que el robot se comunique con una computadora externa y otras entradas y salidas del robot. Este sistema operativo, a diferencia del DOS, genera un ambiente para la programación del brazo. Generalmente el sistema operativo dice al robot como ejecutar una acción y puede estar almacenado en memoria

de solo lectura o en discos flexibles. La complejidad y el tamaño de tal sistema depende de la estructura del robot y fundamentalmente de la aplicación.

5.5.3 MODOS DE OPERACION DEL SISTEMA OPERATIVO

Un sistema operativo de lenguaje de robot contiene los tres modos operativos básicos siguientes:

1. Modalidad monitor.

2. Modalidad ejecución.

3. Modalidad edición.

La modalidad monitor se utiliza para realizar el control de supervisión del sistema. En esta modalidad, el usuario puede definir localizaciones en el espacio utilizando un control de mandos, ajustar la velocidad de operación, entrar en los otros dos modos de operación. También puede ejecutar funciones que se relacionan con la supervisión del sistema, procesamiento de datos y comunicaciones.

La modalidad ejecución se utiliza para ejecutar un programa de robot, en el cual el robot efectúa la secuencia de instrucciones que indican el programa de aplicación. Es deseable que el sistema operativo permita al usuario entrar a la modalidad edición o monitor, para efectos de depuración.

La modalidad editor proporciona un conjunto de instrucciones que permiten al usuario escribir nuevos programas o editar programas ya existentes. Entre las funciones de edición se encuentran la escritura de nuevas líneas de instrucciones en secuencia, borrado o realización de cambios en las instrucciones existentes e inserción de nuevas líneas dentro de un programa.

Cuando el sistema operativo procesa un programa de aplicación para su ejecución, lo hace utilizando un intérprete o un compilador. Un intérprete es un programa del sistema operativo que ejecuta una a una, cada una de las instrucciones de un programa fuente. Un compilador es un programa del sistema operativo que pasa a través del programa fuente completo y pretraduce al código de nivel de máquina todas las instrucciones para que puedan ser leídas y ejecutadas por el controlador del robot.

En las secciones siguientes se desarrollarán los elementos y funciones básicas que deberían ser incorporadas al lenguaje para permitir que el robot realice tareas de complejidad media a alta.

5.6 INSTRUCCIONES DE LOS LENGUAJES DE ROBOT

En esta sección se examinarán algunos de los elementos e instrucciones que han sido incorporadas a los lenguajes de robot para permitir que el robot realice tareas de complejidad media a alta.

5.6.1 CONSTANTES, VARIABLES Y PUNTOS

Una constante es un valor utilizado en el programa que no cambia durante la ejecución del programa. Una variable es un símbolo que puede cambiar de valor durante la ejecución del programa. Las constantes y variables pueden ser enteros o reales (positivos o negativos) y cadenas de caracteres encerradas entre corchetes.

La sintaxis exacta para cada constante puede variar de un lenguaje a otro. La generalidad, es que los símbolos y nombres de variables se crean a partir del conjunto de caracteres alfanuméricos (que lo constituyen las letras del alfabeto y los 10 dígitos decimales). Una regla usual, es que el nombre de la variable debe comenzar con un carácter alfabético y que no debe ser igual a una palabra de vocabulario del lenguaje.

Un punto, es un elemento especial de los lenguajes de robot, y se utiliza para especificar los valores de coordenadas de las articulaciones de un robot. Una forma de definir un punto en lenguajes de robot es la siguiente:

```
A1 = POINT(50.5,236.0,14.581)
```

La interpretación común del punto A1, es que los tres valores se refieren al ángulo de rotación de las articulaciones con respecto a la posición neutra de referencia.

5.6.2 ORDENES DE MOVIMIENTO

Una de las funciones más importantes del lenguaje y la característica principal que distingue a los lenguajes de robot de los lenguajes de programación de computadoras, es el control del movimiento del manipulador.

La función básica de movimiento es la instrucción MOVE, cuya sintaxis se puede definir como:

```
        MOVE  A1
o       MOVE  60,55,12
```

La instrucción MOVE produce el movimiento del extremo del brazo partiendo de su presente posición a un punto definido en la instrucción. En el primer caso, el valor A1 es un punto previamente definido mediante POINT. Recordemos que un punto

se define en términos de las posiciones de la articulación de robot, que a su vez define a un punto en el espacio. En el segundo caso, el punto es definido en la instrucción misma, y cada valor representa un valor angular que se desplazaría cada articulación del brazo.

La sentencia MOVE suele hacer que el brazo se mueva con un movimiento interpolado de la articulación. Existen variaciones en la sentencia MOVE. Por ejemplo, la siguiente instrucción podría hacer que el efector final se moviese en línea recta desde el punto actual hasta el punto A1:

```
MOVEL  A1
```

El sufijo L establece la interpolación de línea recta. El controlador calcula una trayectoria rectilínea desde la posición actual al punto A1 y hace que el brazo del robot siga esa trayectoria.

5.6.3 ORDENES DEL EFECTOR FINAL Y DE CONTROL

En la mayoría de los lenguajes de robot existen formas de ejercer el control sobre el funcionamiento del efector final. Las órdenes más elementales son OPEN y CLOSE. La orden OPEN permite que la pinza se abra, y la función CLOSE permite que la pinza se cierre. En algunas garras se tienen sensores táctiles incorporados en los dedos, lo que permite al robot detectar la presencia de un objeto entre sus dedos.

La función CLOSE cerrará los dedos de la garra hasta que entre los dedos haya una distancia específica y constante. En la acción de tomar un objeto, el controlador necesita saber el momento exacto en que el objeto ha sido fuertemente sujetado, para luego proceder a su traslado. En dicho caso, mediante un sensor en los dedos se puede determinar si el objeto ha sido tomado o no, para ello se puede utilizar una variación de la instrucción CLOSE, la cual denominaremos CLOSER.

La función CLOSER cerrará la garra hasta que haya un objeto entre sus dedos y sobre el cual se ejerce cierta presión. Dicha presión puede tener un valor fijo o estar definida en el programa. Por ejemplo, la instrucción:

```
CLOSER 3.0 LB
```

indica el tipo de orden que podría utilizarse para aplicar una fuerza de agarre de 3 libras contra el elemento.

El control de la fuerza de la pinza puede ser notablemente más refinado que la orden anterior. Para una mano adecuadamente instrumentada, la sentencia

```
CENTER
```

proporciona un nivel bastante sofisticado de control para la detección táctil. La llamada a esta orden hace que la pinza se cierre lentamente hasta que uno de los dedos establezca contacto con el objeto. Luego, mientras que los dedos continúan manteniendo el contacto con el objeto, el brazo del robot cambia de posición mientras que el dedo opuesto se cierra poco a poco hasta que también hace contacto contra el objeto. La sentencia CENTER permite al robot centrar su brazo alrededor de un objeto en vez de hacer que el objeto se desplace mediante el cierre de la pinza. Esto podría ser útil en la determinación de la posición de un objeto, cuya localización es conocida sólo de forma aproximada por el robot.

Los robots suelen trabajar con algunos objetos en su espacio de trabajo. En el caso más simple puede ser una pieza que el robot tome, mueva y deje caer durante la ejecución de su ciclo de trabajo. En los casos más complicados, el robot trabajará con otros elementos del equipo de trabajo, debiendo coordinar las actividades de los diversos equipos. Esta situación introduce el tema del control del equipo de trabajo.

Casi todos los lenguajes de robots industriales poseen instrucciones que envían o reciben señales durante la ejecución del programa. Las señales de este tipo suelen ser binarias; es decir, la señal de nivel alto-bajo.

Este tipo de señales normalmente se utilizan para coordinar la acción del robot con otro dispositivos dentro del área de trabajo. Por ejemplo, consideremos un robot cuya tarea es la de descargar una prensa. Es importante inhibir al robot de tener su pinza entre la prensa antes de que la prensa se abra e incluso, de manera más obvia, es importante que el robot extraiga su mano de la prensa antes de cerrarla.

En la mayoría de robot industriales existen diversas líneas de salida y entrada disponibles para el controlador. Dichas señales pueden ser accesadas directamente por el usuario. Para los siguientes ejemplos supondremos que existen cuatro líneas binarias de entrada, numeradas del 1 al 4; y otras cuatro líneas binaria de salida, numeradas del 5 al 8.

Para llevar a cabo esta coordinación, introduciremos dos órdenes que pueden utilizarse durante el programa. La primera orden es SIGNAL que puede utilizarse para activar o desactivar una señal de salida. La sentencia

```
SIGNAL 3,ON
```

haría que la línea 3 sea activada (puesta a 1 lógico) en un punto del programa (donde sea ejecutada la instrucción). Para desactivar dicha línea se utilizaría la sentencia

```
SIGNAL 3,OFF
```

La segunda orden es WAIT, una sentencia usando esta orden se muestra a continuación

```
WAIT 5,ON
```

lo cual le indicaría al programa que espere en dicha instrucción hasta que reciba un uno lógico por la línea de entrada 5. En pocas palabra, el programa se detiene condicionado el estado de la señal de entrada 5: si esta señal tiene un 0 lógico, el programa para; si tiene un 1 lógico, el programa continua con la siguiente instrucción.

Para ejemplificar el uso de ambas señales se muestra un bloque de instrucciones. En la secuencia siguiente, el robot proporciona la energía a algún dispositivo exterior. WAIT se utiliza para verificar que el dispositivo ha sido activado, antes de permitir que continúe el programa. Luego, en el programa, el robot desactiva el dispositivo y el dispositivo señaliza que ha sido desactivado antes de que continúe el programa. Las órdenes pertinentes son las siguientes:

```
SIGNAL 1,ON       El robot activa el dispositivo
WAIT  5,ON        El dispositivo indica que esta activado
    .             Otras
    .             Instrucciones
    .             del programa
SIGNAL 1,OFF      El robot desactiva el dispositivo
WAIT   5,OFF     El dispositivo indica que está desactivado.
```

5.6.4 CONTROL DEL FLUJO DEL PROGRAMA

En la escritura de un programa de robot suele ser necesario ejercer control sobre la secuencia en que se ejecutan las sentencias mediante bifurcaciones y subrutinas. Una diversidad de instrucciones están disponibles en los lenguajes de robot textuales para controlar el flujo lógico del programa y para nombrar y utilizar subrutinas dentro del programa.

Entre las instrucciones posibles que se pueden utilizar para controlar el flujo lógico dentro del programa se encuentra GOTO. La sentencia

```
GOTO ROTULO
```

indica un salto incondicional a la sentencia denominada ROTULO.

La sentencia IF proporciona la oportunidad para una estructura lógica más complicada en el programa, en forma de una sentencia IF...THEN...ELSE...END. Se podría escribir de la siguiente forma:

```
IF (expresión lógica) THEN
```

.
(grupo de instrucciones)
.
ELSE
.
(grupo de instrucciones)
.
END

Si la expresión lógica es verdadera, se ejecutará el grupo de sentencias entre THEN y ELSE. Si la expresión lógica es falsa, se ejecuta el grupo de sentencias entre ELSE y END. El programa continua después de la sentencia END.

Los bucles DO se pueden realizar en los programas de robot. La secuencia de sentencias se leería como sigue:

DO
.
(grupo de instrucciones)
.
UNTIL (expresión lógica)

La sentencia DO indica el comienzo del bucle DO. El grupo de instrucciones, que sigue a la sentencia DO, forman un conjunto lógico cuyos valores variables afectarían a la expresión lógica asociada con la sentencia UNTIL. En cada ejecución del grupo de instrucciones, la expresión lógica es evaluada. Si el resultado es falso, el bucle DO se ejecuta de nuevo; si el resultado es verdadero, entonces el programa continúa en la primera sentencia después del paso UNTIL.

Existen ocasiones, durante el programa, en que es necesario retardar o parar la ejecución antes de ejecutar el paso siguiente. Este retardo podría ser para asegurar que alguna acción ha tenido lugar antes de proseguir la ejecución del programa. En este caso, el robot se programaría para esperar durante una cantidad específica de tiempo. La orden DELAY se puede utilizar para retardar la continuación del programa, durante un tiempo especificado. La sentencia

DELAY 5

indica que el robot debe esperar 5 segundos antes de proseguir con el siguiente paso en el programa.

La orden STOP se utiliza para ordenar al controlador parar inmediatamente la ejecución del programa y el movimiento del manipulador. Esta función está relacionada con las condiciones de inseguridad del sistema. Su sintaxis es la siguiente

STOP

Otra sentencia de parada del programa es la sentencia

PAUSE

la cual detiene la ejecución del programa y devuelve el control al modo operativo de monitor. El programa se puede reanudar por parte del operador utilizando la orden

RESUME

que produce la ejecución del programa para continuar con la sentencia inmediatamente después de PAUSE.

Las instrucciones PAUSE y RESUME se deben utilizar cuando el área de trabajo incluye a un trabajador humano, cuyas acciones son variables y existe la necesidad de regular el ciclo del robot.

5.6.5 SUBRUTINAS

La mayoría de los controladores para robots industriales proporcionan un método para dividir un programa en una o más ramas. Una subrutina es un segmento de programa, el cual puede ejecutarse una o más veces mediante llamadas desde el programa principal.

La subrutina se puede ejecutar por la bifurcación (llamada) en un lugar determinado del programa o mediante la prueba de una línea de señal de entrada. La mayoría de controladores permiten al usuario especificar si la señal de entrada interrumpirá la instrucción que se está ejecutando en ese momento o si esperará hasta que se complete la ejecución de la instrucción.

La capacidad de interrupción suele ser utilizada por las bifurcaciones de error. Una bifurcación de error se llama cuando una señal de entrada indica que algún suceso anormal (por ejemplo una condición operativa insegura) se ha producido. Dependiendo del suceso y del diseño de la subrutina de error, el robot tomará alguna acción correctiva o simplemente finalizará el movimiento del robot y emitirá una señal de petición de asistencia humana.

La instrucciones para declarar una bifurcación se muestran a continuación

```
SUB   PROCED1
      .
END   PROCED1
```

La sentencia SUB PROCED1 indica el inicio de la subrutina denominada PROCED1; la sentencia END PROCED1 indica el final de la subrutina PROCED1.

Para llamar la subrutina durante la ejecución del programa,

simplemente utilizaremos el nombre de la subrutina

PROCED1

Por medio de las subrutinas, se evita la repetición de instrucciones, redundando en una eficiencia significativa en la programación del robot. Además, el proceso de depuración se facilita, ya que el dividir un programa largo en pequeña subtareas, hace más fácil el identificar y corregir errores.

5.6.6 COMUNICACIONES Y PROCESAMIENTO DE DATOS

En esta sección las comunicaciones se refieren al comunicación entre el sistema del robot y el operador.

La capacidad del robot para comunicarse con el operador permite al robot solicitar información o para informar al operador sobre que lo que está haciendo en ese momento. En un proceso repetitivo, el robot podría solicitar al usuario la cantidad de veces que el proceso se repetirá.

El dispositivo normalmente utilizado por el robot para comunicarse con el operador es el TRC o monitor. El programador u operador también interactúa con el sistema con el objeto de preparar y editar programas, introducir datos requeridos por el sistema y controlar el funcionamiento del robot. Los dispositivos comunes utilizados por el operador para comunicarse con el sistema son el control de mandos y el teclado alfanumérico.

El lenguaje textual de robot tiene las sentencias READ y WRITE, empleadas por el programador para comunicación del sistema-a-operador y operador-a-sistemas, respectivamente. La sentencia WRITE se utiliza para escribir mensajes (o archivos) al operador en el TRC y la sentencia READ se usaría para leer las entradas introducidas por el operador al sistema. Las siguientes instrucciones representan un intercambio típico que se puede producir durante el funcionamiento del sistema

```
WRITE 'Entre el número de elementos a procesar'
READ (CANT)
```

El dialogo muestra que el sistema ha pedido al operador que le indique cuantos elementos se procesarán. La sentencia READ se utilizar para establecer que los datos introducidos por el operador a través de la consola han de almacenarse en la variable CANT.

5.7 COMANDOS DEL MODO MONITOR DEL SISTEMA OPERATIVO

La modalidad monitor del sistema operativo se utiliza para funciones tales como introducción de posiciones mediante el

control de mandos y establecimiento de la velocidad. En esta sección describiremos los diferentes comandos que debe tener el sistema operativo.

HERE: Este comando se utiliza para definir la posición de un punto. La sintaxis de esta orden es

HERE P1

Esta orden, dada en el modo monitor, define la variable P1 como un punto cuyo valor contiene la posición actual de la garra.

SPEED: Se utiliza para definir la velocidad máxima para la ejecución de un programa.

EDIT: Este comando permite abrir un archivo de programa o editar uno ya existente. Su sintaxis es:

EDIT nombre-programa

SAVE: se utiliza para guardar un archivo en disco o cinta.

LOAD: lee un archivo a partir del almacenamiento secundarios (disco o cinta) hacia la memoria del controlador.

LIST: muestra en pantalla del monitor el programa que se encuentra actualmente en memoria.

PRINT: imprime al printer el programa en memoria.

EXECUTE: el usuario la utiliza para indicar que el robot ejecute un programa. Su sintaxis es:

EXECUTE nombre-programa

Las diversas órdenes de monitor suelen ser especificas para un sistema operativo de control del robot particular. En esta sección se ha mostrado solo las funciones básicas posibles.

CONCLUSIONES DEL CAPITULO V

En esta sección se han definido una serie de instrucciones y comandos que contiene la mayoría de lenguajes de robot en aplicaciones industriales. Hay que hacer notar que los nombres de las instrucciones pueden variar de un lenguaje a otro, y los nombres utilizados en esta sección son representativos de la función que ejecutan, y no forman necesariamente parte de los lenguajes ya existentes.

Un programa de robot tiene la finalidad de generar ordenes y recibir información del mismo, para ser analizado en tiempo real y tomar acción en el siguiente movimiento a realizar, como se vio un programa de robot es una secuencia de instrucciones capaces de entablar una comunicación con el medio externo (a través de sensores) y el usuario.

Se estudiaron diferentes métodos de programación de robot, tales como aprendizaje directo motorizado y manual, y lenguajes textuales. Cada uno tiene sus ventajas así como desventajas, pero la opción de cual se utilizará depende de la complejidad que se desea del movimiento, así como de la capacidad del usuario.

El método de programación del robot es el modo textual, el tiene la ventaja de poder lleva a cabo aplicaciones de mediana y alta complejidad. En el siguiente capítulo describirá el diseño global del software desarrollado.

REFERENCIAS BIBLIOGRAFICAS

[1] Andeen, Gerry B. Robot Design Handbook. Edit McGraw Hill, España, 1a. Edición, 1988.

[2] C. S. G Lee, K. S. FU y R. C. Gonzales. Robótica. Control, Detección, Visión e Inteligencia. McGraw Hill, 1a. Edición. 1988.

[3] Engelber, Joseph F. Robótica Industrial. Edit. McGraw Hill, España, 1a. Edición, 1989.

[4] Grupo del Buch Engineering Co. Inc. Robotics Trainings Systems, Concepts and Aplications.

[5] Peralta, Juan y Portillo, Marcos. Diseño del Sistema Locomotor y Sensor de un Microrobot. Universidad de El Salvador, proyecto. 1992.

CAPITULO VI

DISEÑO DEL SOFTWARE BRAZO ROBOT

Introducción

En el capitulo anterior se describió los diferentes métodos de programar un brazo robot, y además se describieron varias instrucciones de lenguajes de robot. La mayoría de dichas instrucciones son comunes a los lenguajes de la primera y segunda generación de los lenguajes de robot.

Este capitulo tiene como objetivo detallar los algoritmos generales necesarios para implementar el sistema operativo y un compilador de lenguaje de robot. Además de ello se describirán las estructuras de datos principales utilizadas para dicho objetivo.

El lenguaje escogido para la implementación es el PASCAL apoyado por subrutinas en lenguaje Ensamblador del microprocesador 80286.

El diseño inicia describiendo tanto el programa que tiene las funciones de sistema operativo, así como describiendo las características del lenguajes, como lo son sus reglas sintácticas y conjunto de instrucciones.

Un lenguaje se compone, tanto de un conjunto de reglas sintácticas, como de un conjunto de instrucciones. Los programas de aplicación son los programas creados por el usuario en lenguaje de robot, y son ejecutados a partir de un sistema operativo específico.

6.1. SELECCION DEL LENGUAJE DE PROGRAMACION.

El lenguaje de alto nivel seleccionado para el desarrollo del software es el lenguaje PASCAL. Además de él, hay una serie de procedimientos que se realizaron en lenguaje ensamblador.

Se utilizó el PASCAL por varias razones, entre ellas tenemos:

- Posee poderosas estructuras de datos ideales para el desarrollo de un lenguaje. Entre ellas se destacan las variables tipo puntero.

- Posee un conjunto de procedimientos y funciones muy amplio, que facilitan la programación del computador.

- Es un lenguaje que permite ser ampliado con funciones y procedimientos del usuario, a través del uso de unidades.

En su mayoría, el software ha sido implementado en Pascal, pero hay ciertas rutinas en las cuales se uso el lenguaje ensamblador. Entre estas rutinas, se encuentran las rutinas del siguiente tipo:

- Rutinas con tiempo de ejecución crítico. Hay varios procedimientos que idealmente se desearía que su tiempo de ejecución sea cero, por lo cual se implementaron en lenguaje ensamblador.

- Rutinas de manejo de bits. Hay rutinas que hacen uso del manejo de bits, por ejemplo rotación izquierda o derecha de bits. La rutina correspondiente a los sensores tiene que manejar bits, y al mismo tiempo se requiere que se ejecute a gran velocidad. El implementar esta rutina en PASCAL hubiese requerido un procedimiento grande y lento, por tal razón se desarrolló en Ensamblador.

6.2. DESCRIPCION DEL SISTEMA OPERATIVO

El sistema operativo proporciona al usuario un ambiente para la programación del brazo robot, facilitando el funcionamiento del sistema.

Un sistema operativo suele componerse por un programa interprete de comandos y un conjunto de programas utilitarios. El programa intérprete, al cual le denominaremos programa monitor, se encarga de obtener comandos del usuario y ejecutar la acción correspondiente. Para la ejecución de un programa de aplicación, el programa monitor cede el mando a un programa compilador, el cual toma el programa de aplicación y lo codifica a nivel de la máquina para su posterior ejecución.

En nuestro caso, al programa compilador le denominaremos programa traductor, ya que realmente no traduce el programa de aplicación hasta un archivo en lenguaje de nivel de máquina.

El software desarrollado, comprende el programa monitor así como el programa traductor.

Entre las funciones principales del programa monitor se encuentran: inicialización del sistema, manejo de archivos, creación/edición de puntos y variables, ejecución de comandos que actúan sobre el movimiento del robot, y por supuesto, permite la entrada al modo ejecución.

Las funciones del programa traductor comprende la codificación y ejecución de los programas de aplicación.

A continuación se describirá el programa monitor, y en secciones posteriores se ampliará sobre el programa traductor.

Cuando el usuario hace contacto con el sistema robótico, su primer contacto es con el programa monitor, el cual utiliza un sistema de menúes que le permiten interactuar con el usuario.

Cuando se ejecuta el programa monitor desde el prompt del DOS, en pantalla aparece el siguiente menú:

File	Edit	Run	Options	Command	Quit

Al entrar al programa, sobre la opción File, aparece la barra de menú (video inverso), la cual nos indica la selección actual. La barra de menú se puede desplazar sobre las otras opciones con las teclas cursoras. Cada una de estas opciones tiene un submenú, el cual se detalla a continuación:

File:
- Load
- Directory
- View
- Print

Edit:
- Variables
- Puntos

Options:
- Velocidad
- Desplegar Punto

Command:
- MOVE
- MOVEL
- CLOSE
- OPEN
- SIGNAL

Run:
- Start
- Resume

Quit:
- no tiene opciones.

Describiremos brevemente cada una de las opciones, lo cual nos detallará cada una de las funciones que realiza el programa monitor.

6.2.1. OPCION FILE

La opción File es la encargada del manejo de archivos. Entre las opciones de File tenemos:

Load:

Carga un programa de aplicación del disco a la memoria de la computadora.

Directory:

Visualiza el listado de archivos que posee el directorio actual.

View:

Permite ver el contenido de un archivo.

Print:

Imprime el contenido de un programa que se encuentre en memoria.

6.2.2. OPCION EDIT

Edit se usa para la definición de variables y puntos que definen una posición en el espacio. Entre las opciones de Edit, tenemos:

Variables:

Permite crear, editar y borrar variables que puedan ser usadas por un programa de aplicación.

Puntos:

Esta opción se utiliza para la definición de puntos. Permite crear, editar y borrar puntos de posicionamiento que puedan ser usadas por un programa de aplicación. La forma de definir un punto es posicionando el efector final en el punto deseado, y a continuación a ese punto le asignamos un nombre. En esta modalidad, el robot se puede mover con las teclas cursoras según las indicaciones dadas en pantalla.

Para transferir la información obtenida es a travs de un archivo de puntos y variables. El procedimiento es el siguiente: haciendo uso de la opción Puntos y Variables se definen tantos puntos y variables como deseemos, luego, haciendo uso del programa monitor enviamos esta información a un archivo de texto; a continuación dicho archivo de texto se utiliza como cabecera en el programa de aplicación, el cual podrá utilizar dichas variables y puntos en sus instrucciones.

6.2.3. OPCION OPTIONS

Options permite cambiar variables de ambiente del sistema, como lo son la velocidad máxima de los motores y la forma en que es desplegada la posición del efector final.

Velocidad:

Permite regular la velocidad máxima a la cual se moverán las articulaciones del robot.

Desplegar punto:

En la pantalla aparece la posición del efector final. Esta opción permite cambiar la presentación entre medidas angulares y coordenadas XYZ.

6.2.4. OPCION COMMAND

La opción Command permite el movimiento del robot haciendo uso de menúes. Cuando se accesa esta opción, aparece en pantalla un listado de comandos a partir de los cuales se pueden ejecutar acciones del brazo robot. El listado de dichos comandos es: MOVE, MOVEL, CLOSE, OPEN y SIGNAL.

MOVE:

Permite mover el brazo con movimiento interpolado de articulación.

MOVEL:

Permite mover el brazo con movimiento interpolado en línea recta.

CLOSE:

Cierra los dedos de la garra.

OPEN:

Abre los dedos de la garra.

SIGNAL:

Permite asignar un 0 o 1 lógico en los puertos de salida que posee el brazo robot.

6.2.5. OPCION RUN

Con la opción RUN, el programa monitor llama al programa traductor, el cual toma el programa que se cargó con LOAD, opción File, y lo codifica para su ejecución inmediata. En siguientes secciones se ampliará el funcionamiento del programa traductor, así como su opción Resume.

6.2.6. OPCION QUIT

Esta opción termina de ejecutar el programa monitor y retorna al sistema operativo DOS.

6.3. ALGORITMOS GENERALES DEL PROGRAMA MONITOR

El flujo de ejecución del programa monitor se muestra en el diagrama de flujo de la Fig. 6.1. El primer proceso que se ejecuta es la inicialización del sistema.

El proceso de inicialización consta de varios pasos, entre ellos tenemos:

- Configuración y reconocimiento de los perifricos utilizados para el control del brazo robot.
- Reservación de espacio para variables y arreglos.
- Creación e inicialización de banderas y constantes.

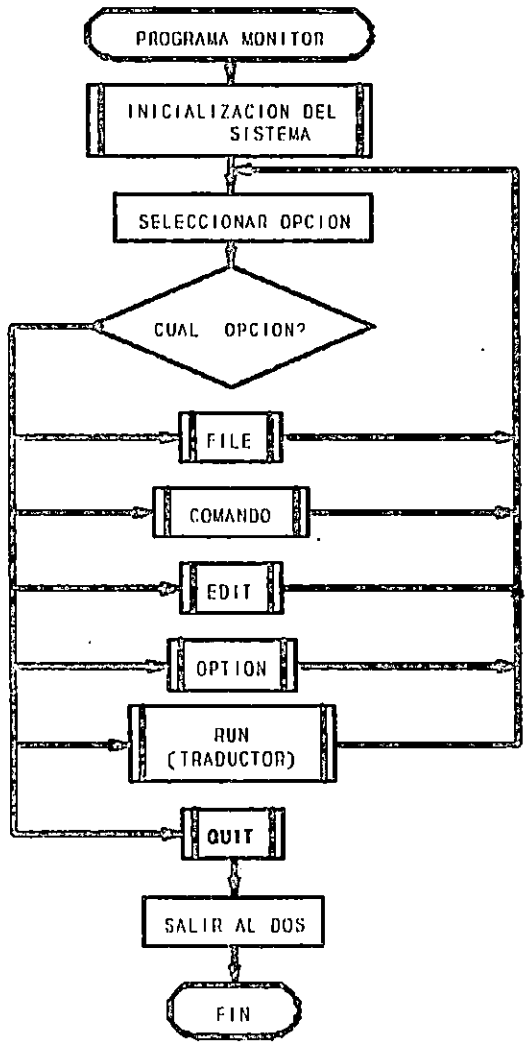


Figura 6.1. Diagrama de Flujo del Programa Monitor

- Colocar el brazo en una posición neutra o de reposo conocida.

Posteriormente a la inicialización, el programa monitor entra en un lazo de selección múltiple, el cual corresponde al primer nivel de menúes que posee. Mediante las cursoras y la tecla ENTER, el usuario selecciona una opción y se ejecuta la acción adecuada, que en este caso aparece como un procedimiento predefinido. Despus de terminar la ejecución de tal procedimiento, el programa retorna al sistema de menúes, solicitando una nueva opción.

La opción que termina el lazo, y por tanto sale del programa monitor al DOS, es la opción QUIT.

Los diagrama de flujo de las Figs. 6.2, 6.3, 6.4 y 6.5 corresponden a los procedimiento predefinidos File, Edit, Command y Option.

Hay que hacer notar su similitud entre ellos y el diagrama de flujo del programa monitor.

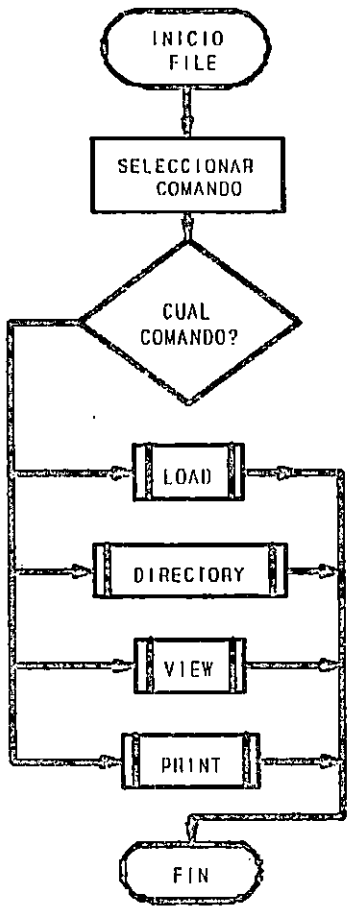


Figura 6.2. Diagrama de Flujo del Procedimiento File

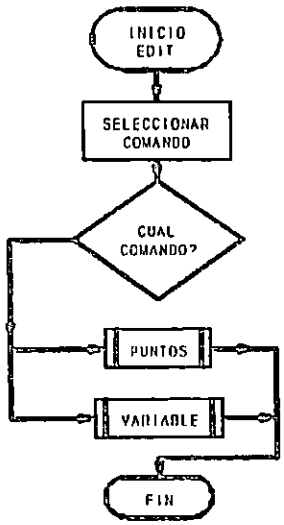


Figura 6.3. Rutina Edit

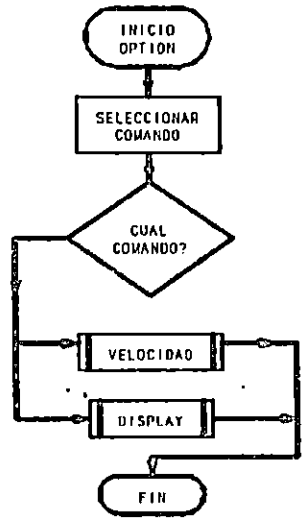


Figura 6.4. Rutina Option

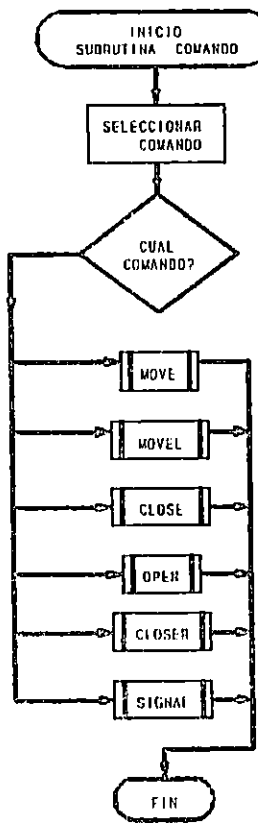


Figura 6.5. Rutina Command

De la figura 6.2, 6.3, 6.4 y 6.5, se concluye que la estructura general de los diagrama de flujo de tales procedimientos es similar, en vista de que en todos ellos hay una opción a escoger en base a la cual se ejecuta una acción correspondiente. A diferencia del programa monitor, el cual tiene un lazo cerrado para seleccionar las opciones, al terminar la acción, el procedimiento termina y el control regresa al programa monitor.

De especial interes es el procedimiento Command, en el cual los comandos proporcionados son realmente instrucciones del lenguaje de robot. Se observará más adelante, que estos mismo comandos (MOVE, MOVEL, etc) son procedimientos que pueden ser ejecutados desde un programa de aplicación.

La ejecución de comandos desde el monitor le da flexibilidad al sistema. En este aspecto, el facilitar estos comandos de esta manera, ayuda al operario con pocos o ningún conocimiento de programación a controlar el brazo robot fácilmente.

La diferencia entre ejecutar estos comandos desde el programa monitor y ejecutarlo en un programa de aplicación, es que en el modo monitor, el computador le indica al usuario los parámetros que debe introducir, y si es posible el significado de ellos. En cambio, en un programa de aplicación, el programador que hace la aplicación debe estar compenetrado del lenguaje y saber la naturaleza de cada instrucción, así como de los parámetros involucrados.

El procedimiento RUN corresponde a la llamada del traductor, y se estudiará en una sección posterior.

6.4. DESCRIPCION DEL LENGUAJE DE ROBOT

El programa traductor equivale a lo que podríamos denominar un programa compilador, ya que toma el programa de aplicación almacenado en memoria, lo traduce y luego lo ejecuta.

Para comprender el funcionamiento del programa traductor, se hace necesario conocer la sintaxis, así como el conjunto de instrucciones que tendrá el lenguaje; además se requiere conocer de las diferentes estructuras de datos que se utilizan para almacenar los diferentes tipos de información que el sistema maneja.

6.4.1. SINTAXIS DEL LENGUAJE DE ROBOT

Un programa en lenguaje de robot está compuesto por una serie de sentencias, donde cada sentencia corresponde con una instrucción simple. Cada sentencia es ubicada en una línea de texto, que tiene como máximo 80 caracteres.

Para el lenguaje desarrollado, habrán tres tipos de sentencias, que se detallan a continuación:

- Sentencias de definición de variables.

- Sentencias de definición de puntos.

- Sentencias de instrucción.

Sentencias de Definición de variables

La definición de variables se hace al inicio del programa, antes de que cualquier instrucción haya sido ejecutado. Una variable se define asignándole un valor numrico. La sentencia de definición de variable tiene la siguiente sintaxis:

```
nombre-variable  =  valor-asignado
```

Donde nombre-variable es el nombre que recibirá la variable, y está compuesto por una cadena de caracteres alfanumricos (letras y número) con una extensión máxima de 10 caracteres (los nombres de variables deben iniciar con un carácter alfabtico); y valor-asignado es un entero entre -32767 y 32768. Es regla que el nombre de la variable inicia en la columna 1 de la línea. Ejemplos de asignaciones de variable son:

```
CONTADOR1   =   1
VARIABLE2   =   300
PUNTOS      =   20
```

Una palabra reservada del lenguaje no puede ser utilizada como variable. Las variables, aunque les fue asignado un valor al inicio del programa, pueden cambiar dicho valor en el transcurso del programa por una instrucción aritmtica.

Sentencia de Definición de puntos

Al igual que la definición de variables, la definición de un punto se hace al inicio del programa. La sintaxis para definir un punto es:

```
nombre-punto POINT valor1,valor2,valor3
```

Donde nombre-punto es el nombre del punto, y cumple las mismas limitaciones que los nombres de variables. POINT es la palabra reservada del lenguaje que indica que se está definiendo un punto. Valor1, valor2 y valor3 son las tres coordenadas de articulación que definen el punto en el espacio. Ejemplos de estas sentencia son:

```
PUNTO1    POINT  230,200,54
ORIGEN    POINT  302,210,98
PNEUTRO   POINT  0,0,0
FINAL     POINT  500,500,500
OTROP     POINT  100,100,100
```

Sentencia de Instrucción

La sentencia que contiene una instrucción, es la de mayor importancia. Este tipo de sentencia se caracteriza por tener una orden (comando) del lenguaje. La sintaxis general de este tipo de sentencia es la siguiente:

nombre-rotulo COMANDO argumentos

Donde nombre-rotulo es un nombre que cumple las reglas de los nombres de variables, y es conocido tambin como etiqueta. Las etiquetas en una línea de instrucción son opcionales y comienzan a escribirse en la columna 1 de la sentencia.

Las etiquetas se utilizan para asociar la sentencia con el nombre de la etiqueta, para una posible referencia posterior de dicha sentencia. Para ejemplo, estudiaremos la siguiente secuencia de instrucciones:

```
LAZO        CLOSE
              .
              .
              .
            GOTO LAZO
```

En este segmento de programa, el flujo de ejecución, al llegar a la línea que contiene la instrucción GOTO, seguiría en la línea denominada LAZO. Queda evidente la utilidad de etiquetar las líneas hacia las cuales se ejecuta un salto desde otra parte del programa.

Los argumentos son parámetros que el comando pueda necesitar para su operación. Los argumentos de un comando pueden ser obligatorios o no, dependiendo del comando que se est ejecutando. Por ejemplo, el comando MOVE necesita argumentos, no así el comando CLOSE.

Cuando hay varios argumentos, por lo general, la separación entre ellos se hace por medio de comas. Los argumentos pueden ser valores numrico, cadenas de caractères, nombres de puntos o nombres de variables.

El COMANDO es el verbo del lenguaje que indica la acción a ejecutarse, y es un elemento obligatorio en este tipo de sentencias. Sobre los diferentes comandos disponibles en el lenguaje implementado se hablará posteriormente.

Cuando una sentencia de instrucción no contenga etiqueta, el verbo o instrucción no puede iniciar en la columna 1 de la sentencia. Entre el verbo y el margen izquierdo debe haber al menos un espacio.

Ejemplos de sentencias de instrucción se muestran a continuación:

```
VINETA    MOVE  200,150,123
          CLOSER
          MOVEL 200,150,200
          OPEN
          MOVE PUNTO1
          GOTO VINETA
```

En el segmento de programa anterior, la primera sentencia es típica, es decir que posee etiqueta, comando y argumentos. En cambio, la segunda sentencia no posee etiqueta ni argumentos.

En la instrucción MOVE PUNTO1, se supone que al inicio se definió un punto denominado PUNTO1, en caso contrario el programa no compilaría.

6.4.2. CONJUNTO DE INSTRUCCIONES

A continuación detallaremos brevemente las instrucciones implementadas, lo cual ayudará a determinar los parámetros necesarios en cada instrucción.

Los parámetros de cada instrucción pueden ser valores numricos, variables, cadenas, etiquetas y puntos. Para distinguir los argumentos se han escrito en itálica. El rango de cada parámetro tambin es descrito en cada instrucción.

MOVE *valor1,valor2,valor3*

Instrucción que mueve las articulaciones del brazo desde la posición actual hasta el punto indicado por los valores. Los tres parámetros corresponden a las coordenadas angulares que definen el punto sobre el cual se desea posicionar la garra. Dichos valores son enteros.

MOVEL *valor1,valor2,valor3*

Instrucción que mueve la garra desde la posición actual hasta la posición indicada por *valor1,valor2,valor3*. En este caso, la garra se mueve en una línea recta desde la posición actual hasta llegar al punto indicado.

CLOSE

Este comando cierra la garra hasta una posición de cierre prefijada. No posee parámetros.

CLOSER

Se utiliza para cerrar la garra hasta que haya sujetado un objeto. No posee parámetros.

OPEN

Este comando abre la garra. No posee parámetros.

SIGNALON *puerto*

Esta instrucción coloca un 1 lógico en una de las líneas de salida del brazo robot. *Puerto* es el número de línea de salida y puede variar de 1 a 4.

SIGNALOFF *puerto*

Esta instrucción coloca un 0 lógico en una de las líneas de salida del brazo robot. *Puerto* es el número de línea de salida y puede variar de 1 a 4.

WAITON *puerto*

Esta instrucción lee la línea de salida indicada por *puerto*, y detiene la ejecución del programa hasta que en dicha línea haya un 1 lógico. Luego continua con la siguiente instrucción. El parámetro *puerto* puede tomar el valor de 4 a 8.

WAITOFF *puerto*

Esta instrucción lee la línea de salida indicada por *puerto*, y detiene la ejecución del programa hasta que en dicha línea haya un 0 lógico. En el momento que haya un 0 lógico, continua con la siguiente instrucción. El parámetro *puerto* puede tomar el valor de 4 a 8.

GOTO *label*

Esta instrucción cambia el flujo del programa, y salta hasta la sentencia indicada por la etiqueta *label*. El parámetro *label* es buscado en la Tabla de Etiquetas, y es sustituida por la dirección de memoria que contiene a la instrucción hacia la cual se salta.

DELAY *time*

Produce un retardo de tiempo equivalente a *time* segundos. El valor *time* puede variar desde 1 hasta 1024 segundos.

STOP

Para la ejecución del programa inmediatamente. No tiene parámetros.

PAUSE

Para la ejecución del programa, pero este puede ser continuado en la instrucción siguiente a PAUSE. No necesita parámetros.

JMPON *puerto,label*

Transfiere el flujo del programa a la línea etiquetada *label* si en la línea de entrada *puerto* hay un 1 lógico. *Label* debe

estar definido en el programa como una etiqueta, y el valor de *puerto* puede variar desde 4 a 8.

JMPOFF *puerto,label*

Transfiere el flujo del programa a la línea etiquetada *label* si en la línea de entrada puerto hay un 0 lógico. *Label* debe estar definido en el programa como una etiqueta, y el valor de *puerto* puede variar desde 4 a 8.

DO
UNTIL condición

Repite el lazo hasta que la condición sea verdadera.

SUB *label*
END *label*

Se utilizan para definir subrutinas. El segmento de programa encerrado entre ambas instrucciones constituye la rutina denominada *label*.

CALL *label*

Se utiliza para llamar una subrutina denominada *label*. Esta tuvo que haber sido definida con SUB y END.

WRITE 'mensaje' o WRITELN 'mensaje'

Escribe en pantalla un mensaje en la pantalla. El parámetro que usa es una cadena de caracteres. Corresponde a los comandos del mismo nombre en PASCAL.

READ *var*

Lee del teclado un número y lo almacena en la variable *var*. Dicha variable tuvo que haber sido inicializada en el programa al principio de ste.

INC *var*

El valor almacenado en la variable *var* es incrementado en 1

DEC *var*

El valor almacenado en la variable *var* es decrementado en 1

6.5. DESCRIPCION DE ESTRUCTURAS DE DATOS

En la implementación del lenguaje se usaron múltiples estructuras de datos para almacenar en memoria la información manejada por el sistema. A continuación se describirán las diferentes estructuras de datos utilizadas, y de que manera son usadas.

Las variables del programa son almacenadas en un vector de registros. Para cada variable se ha utilizado un registro, y en cada registro se almacena el nombre de la variable así como su valor. Una variable se puede identificar internamente por su nombre o por el índice asignado en el vector de registros. En adelante, llamaremos a esta estructura Tabla de Variables.

Los puntos definidos al inicio del programa son almacenados en un vector de registros. En un registro se almacena un punto, y cada registro consta de un campo alfanumrico y tres campos numricos. En el campo alfanumricos se ubica el nombre del punto y en los campos numricos se ubican los valores de coordenadas de articulación del brazo. En adelante, llamaremos a esta estructura Tabla de Puntos.

Las etiquetas distribuidas a lo largo del programa son almacenadas en un vector de registros con dos campos, el primer campo mantiene el nombre de la etiqueta y el segundo campo, la dirección de memoria de la instrucción relacionada con dicha etiqueta. A esta estructura la reconoceremos como Tabla de Etiquetas.

Además, debe existir una Tabla de Procedimientos, implementado con una matriz que contiene en cada registro el nombre del procedimiento, así como la dirección de inicio y fin de ste.

Los programas de aplicación (lenguaje fuente) son cargados a memoria con la instrucción Load del modo monitor. Cuando el comando Load es ejecutado, el programa lee del disco el archivo fuente y lo guarda en memoria del computador. Para guardar el programa fuente, se ha usado un vector de string. Cada elemento del vector, es decir un string, contiene una línea de programa cuya longitud máxima es de 80 caracteres. A esta estructura le denominaremos Tabla de Programa Fuente.

Cuando del modo monitor se pasa al modo ejecución, ejecutando la opción Run, el programa contenido en la Tabla de Programa Fuente es codificado (podría decirse una seudocompilación) y se almacena en una cola FIFO, a partir de la cual se efectúa la ejecución. Llamemos a esta estructura, Programa Ejecutable.

La cola FIFO consiste de una serie de posiciones de memoria, donde cada una recibe el nombre de nodo. Por cada instrucción de programa se crea un nodo, el cual contiene toda la información necesaria para la ejecución de tal instrucción. Se denomina cola FIFO (First Input, First Output), porque la primera instrucción que entra (codificada) es la primera que sale (ejecutada).

En la Fig. 6.6 se observa esquemáticamente una cola FIFO. Existen dos nodos especiales, los cuales no contienen información del programa, sino información relativa a la cola. El primer nodo es el Nodo Frente, y el último es el Nodo Final. Toda nueva instrucción es introducida entre el Nodo

Final, y el nodo de la última instrucción. Cuando el programa es ejecutado, el traductor comienza a ejecutar las instrucciones por el Nodo Frente, hasta que el flujo de ejecución lo lleve al Nodo Final.

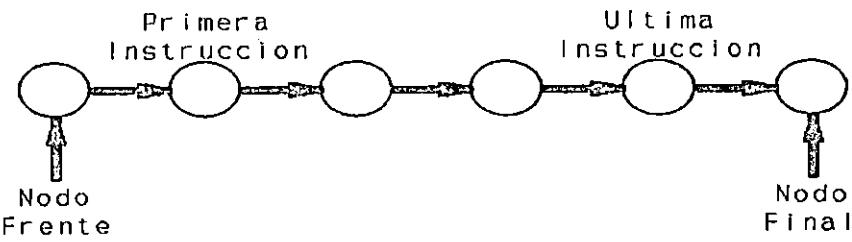


Figura 6.6. Estructura de datos Cola FIFO

Si se observa, de cada nodo sale una flecha hacia el siguiente nodo. Esta flecha indica que cada nodo tiene la dirección del siguiente nodo, lo cual permite desplazarse a travs de toda la cola en un forma secuencial. Si un nodo no tuviese la dirección del siguiente nodo, sería imposible llegar a l y se rompería la estructura

El nodo por lo general es un registro que contiene varios campos de información. Un campo obligatorio es el campo que contiene la dirección del siguiente nodo, el cual es de tipo puntero. Los demás campos del registro contienen la información de la instrucción, y pueden ser de cualquier tipo de datos proporcionados por Pascal: enteros, reales, string, vectores, puntero, etc.

La cantidad y tipo de campos que posea un nodo no afecta la estructura de otro nodo, de tal manera, que en una misma cola pueden existir un nodo con tan sólo un campo entero (más el de dirección) y el siguiente nodo tener diez campos de cualquier tipo.

La ventaja principal de una estructura de nodos con respecto a una estructura matricial, es que la matriz se define al inicio y el espacio memoria es reservado, sea que se use o no. En cambio, en la estructura nodal, un nodo es creado cuando se necesita, y ste puede ser eliminado liberando la memoria que ocupaba. Además, en una estructura nodal, la memoria que puede ser usada es toda la memoria disponible que tiene el computador.

Otra ventaja, es que en una estructura nodal, cada elemento puede variar su estructura. Por ejemplo, un nodo podría tener un campo numrico, y el siguiente en la lista tener dos campos numricos, tres campos de carácter y tres campos de string. Estas variaciones en una matriz es muy dificultoso de manejar.

Otra ventaja, es que si de un nodo -instrucción- saltamos a

otra instrucción que no es la siguiente, lo único que tenemos que poseer es la dirección de dicha instrucción, dicha dirección puede ser almacenada en un campo del nodo.

En base a la sección 6.4.2, se puede definir los diferentes campos que necesita cada instrucción del lenguaje, y por tanto definir el nodo asignado a cada tipo de instrucción.

Cuando se codifica una instrucción, se genera un nodo que contiene la información primordial para la ejecución de la instrucción. Existe un campo denominado Tipo, el cual es tipo char, el cual define a la instrucción univocamente. Es decir, por cada instrucción existe un carácter ASCII que la identifica. Los demás campos de la instrucción corresponden a los diferentes argumentos que ella utiliza. Los campos necesarios por cada instrucción, así como el tipo de dato se detallan en la Tabla 6.1.

Tabla 6.1. Estructura de Datos usadas por cada instrucción

Instrucción	Tipo	Campo 1	Campo 2	Campo 3	
MOVE n,n,n	M	Entero Coord1	Entero Coord2	Entero Coord3	
MOVEL n,n,n	L	Entero Coord1	Entero Coord2	Entero Coord3	
CLOSE	C				
CLOSER	R				
OPEN	O				
SIGNALON p	S	Byte # port			
SIGNALOFF p	F	Byte # port			
WAITON p	W	Byte # port			
WAITOFF p	T	Byte # port			
GOTO d	G	Puntero Dirección de Salto			
DELAY n	D	Entero Tiempo pausa			
JMPON p,d	J	Byte # port	Puntero Dirección de Salto		

Tabla 6.1. Continuación.					
JMPOFF p,d	P	Byte # port	Puntero Dirección de Salto		
STOP	Z				
PAUSE	A				
INC var	I	Byte # var			
DEC var	X	Byte # var			
WRITE m	Q	String Mensaje			
WRITELN m	H	String Mensaje			
READ var	K	Byte # var			
SUB	U				
END	N	Byte # rutina	Puntero Dirección Retorno		
CALL d	V	Byte # rutina	Puntero Inicio Rutina	Puntero Fin rutina	
DO	B	Puntero Dirección Final			
UNTIL p,c	Y	Byte # port	Byte Condicion		

Es de hace notar, que las instrucciones que involucran direcciones de otras instrucciones usan un tipo puntero para almacenar esa información. Una variable puntero, es una variable que contiene la dirección de otra variable, la cual puede ser de cualquier tipo: registro, vector, entero, etc.

6.6. ALGORITMO DE FUNCIONAMIENTO DEL PROGRAMA TRADUCTOR

Habiendo conocido las diferentes estructuras de datos existentes, es posible adentrarnos en el funcionamiento del programa traductor. La función global del programa traductor es tomar un programa en lenguaje de robot y ejecutar las acciones indicadas por 1. El proceso se ha dividido en dos subprocesos: codificación y ejecución.

En el proceso de codificación, el traductor toma cada instrucción dada en lenguaje fuente y la codifica según la Tabla 6.1. Esta información es almacenada en una estructura nodal que hemos denominado Cola FIFO. Qu razones hay para ello?

Si el programa se ejecutará en forma secuencial, no habría necesidad de codificar las instrucciones y podrían ser ejecutadas directamente desde el lenguaje fuente. El problema surge porque el programa no necesariamente tiene un flujo secuencial de ejecución , ya que posee saltos condicionales, incondicionales, hay estructuras de lazo, etc. La estructura a base de punteros se adapta a este tipo de tratamiento, en vista que un nodo puede contener campos que apunten a una dirección diferente de la siguiente instrucción.

El diagrama de flujo del módulo traductor se muestra en la Fig. 6.7.

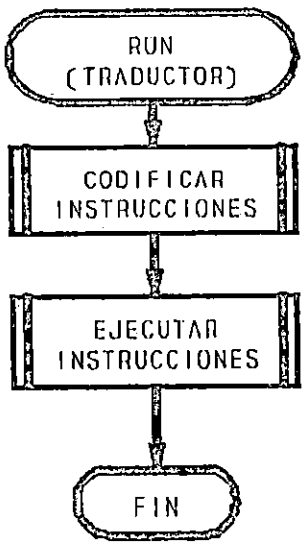


Figura 6.7. Módulo Run (Traductor)

En la Fig. 6.8 tenemos el diagrama de flujo del módulo de Codificación. Hay que hacer notar que al codificar las instrucciones se hace en dos pasos. En el primer paso, a partir del programa se crea la Tabla de Variables, Tabla de Puntos, Tabla de Etiquetas y Tabla de Procedimientos.

En algunas situaciones, existen instrucciones que utilizan etiquetas que aún no han sido definidas, porque están etiquetando sentencias posteriores; en este caso, al hacer la primera pasada aún no se conoce la dirección a la que corresponde la etiqueta. Por ello, se hace necesario hacer una primera pasada de cada sentencia para crear la Tabla de Etiquetas y Procedimientos, así como la Tabla de Puntos y Variables.

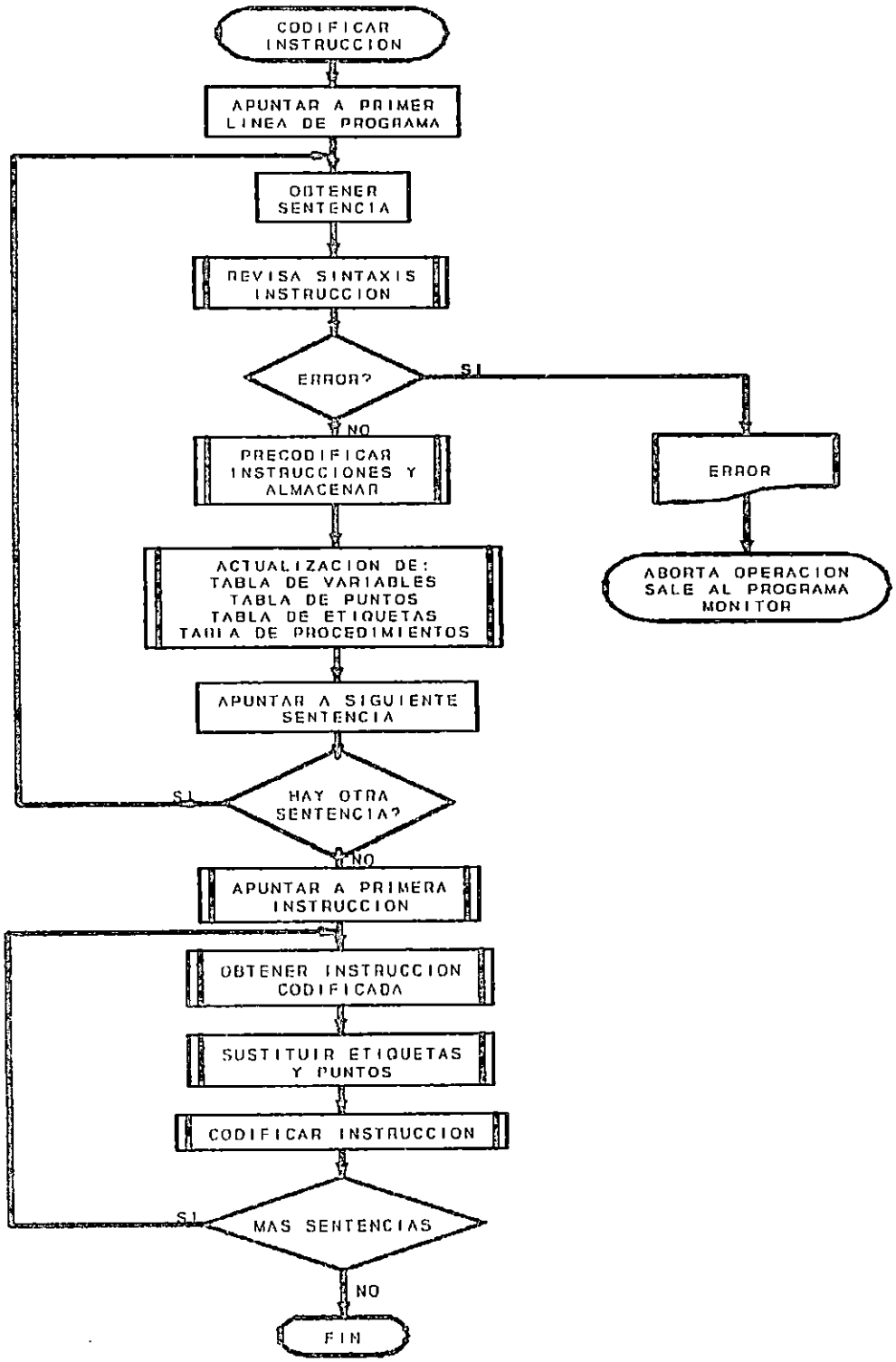


Figura 6.8. Módulo Codificación de Instrucciones.

En la segunda pasada, la Tabla de Variables, Puntos, Etiquetas y Procedimientos se encuentran totalmente definidas. En la primera pasada, la instrucciones que utilizaban etiquetas aún no definidas, se dejaron incompletas. En esta pasada, ya se conocen la dirección de tales etiquetas (de la Tabla de Etiquetas), y se pueden completar dichas instrucciones para su codificación final. De esta forma, en la segunda pasada, se verifica si cada instrucción está incompleta, si lo está, obtiene la dirección de la etiqueta o procedimiento, y lo almacena con la instrucción, despus de lo cual está lista para ejecución.

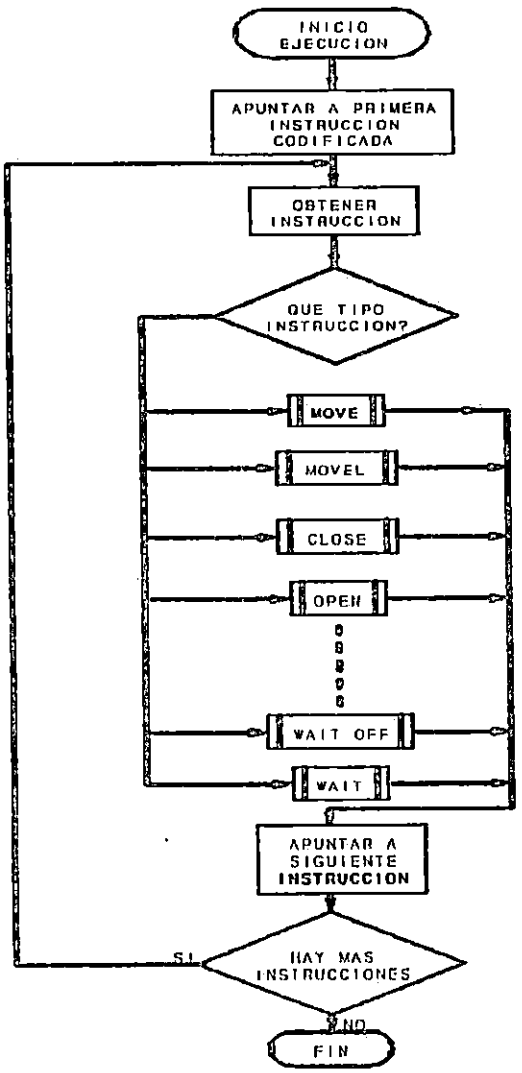


Figura 6.9. Diagrama de flujo del Módulo Ejecución

En la Fig. 6.9 está desarrollado el diagrama de flujo del módulo de ejecución. Este es relativamente sencillo en cuanto

a su estructura. El algoritmo que sigue es tomar la primera instrucción codificada del programa, y mediante una estructura de selección múltiple (CASE), determina la acción a ejecutarse en base al campo tipo que posee la instrucción codificada. Al terminar la ejecución de la instrucción actual, obtiene la siguiente instrucción y la ejecuta de igual manera. Este procedimiento siguiente hasta que ya no hayan instrucciones por ejecutar y sale del modulo Ejecución para regresar al programa monitor.

En este punto, es necesario introducir el concepto de puntero de instrucción (variable program_pointer), el cual apunta a la siguiente instrucción a ejecutarse. Antes de ejecutar la instrucción, el programa toma del nodo actual la dirección de la siguiente instrucción y lo almacena en el program_pointer.

El procedimiento de actualizar el program_pointer se hace antes de ejecutar la acción correspondiente a la instrucción, porque hay varias instrucciones que afectan el program_pointer para cambiar el flujo de ejecución del programa. Si la actualización del program_pointer se hiciese posterior a la ejecución de la instrucción, la modificación que hacen tales instrucciones sobre el program_pointer se invalidaría.

Por cada instrucción del lenguaje existe una opción diferente (así como un procedimiento) en la estructura CASE del módulo ejecución. Este conjunto de procedimientos constituyen el set de instrucciones del lenguaje, y se encuentran almacenados en una unidad denominada ORDENES.TPU.

En las secciones siguiente se describirán brevemente el algoritmo de cada uno de estos procedimientos.

6.7. ALGORITMOS GENERALES DE LAS ORDENES DEL LENGUAJE DE ROBOT

Por cada instrucción disponible en el lenguaje de robot existe un procedimiento en PASCAL que realiza la acción asignada a dicha instrucción. Varios procedimientos son relativamente sencillos y cortos, pero tambin hay procedimientos muy complejos y largos. Este conjunto de procedimientos es la parte medular del lenguaje, y si bien son los últimos en describirse, fueron los primeros en desarrollarse.

La descripción de tales procedimientos, así como de sus algoritmos se hará por bloques de instrucciones que tienen similares funciones. Las instrucciones se han agrupado de la siguiente manera:

Movimiento del brazo:	MOVE, MOVEL
Movimiento de la garra:	CLOSE, CLOSER, OPEN
Control de Puertos:	SIGNALON, SIGNALOFF, WAITON, WAITOFF

Entrada/Salida de consola: WRITE, WRITELN, READ

Flujo de Ejecución: GOTO, JMPON, JMPOFF, PAUSE, DELAY, STOP, DO-UNTIL, SUB-END

Aritméticas: INC, DEC

6.7.1. PROCEDIMIENTOS MOVE, MOVEL

Definición en Pascal:

```
procedure MOVE(coord1,coord2,coord3:integer);
procedure MOVEL(coord1,coord2,coord3:integer);
```

Constituyen los procedimientos fundamentales, que permiten el movimiento de la estructura completa del robot según el deseo del usuario.

MOVE

El diagrama de flujo de la instrucción MOVE se muestra en la Fig. 6.10.

Entre las variables de ambiente del sistema, se encuentra la variable de posición_real y pulsos_posición, ambas referidas a la posición actual del extremo de la garra. La posición_real se refiere a la posición obtenida a partir de los sensores, y pulsos_posición es la posición calcula a partir de los pasos angulares que el motor supuestamente ha realizado. En teoría, ambas posiciones deben ser iguales (el robot no puede estar en dos posiciones al mismo tiempo). Recordemos que los motores, bajo condiciones de carga extrema pueden perder o ganar pasos.

La posición, como se detallo en el capítulo V, se puede dar en coordenadas universales XYZ o en coordenadas de articulación. Para el control de los motores, las coordenadas de articulación son ideales, ya que el cálculo del movimiento angular de cada motor se facilita.

Para este procedimiento, se desea el movimiento desde una posición (actual) hacia otra nueva, dada en la instrucción. El movimiento debe ser con interpolación articulada, lo cual significa que todos los motores comenzarán y terminarán de girar al mismo tiempo, a velocidades distintas.

El primer paso en el procedimiento, es calcular los pulsos necesarios por motor para alcanzar la posición deseada. Luego, se calcula el tiempo de retardo entre la aplicación de los diferentes pulsos, teniendo en cuenta las consideraciones de velocidad dada en el capítulo V. Todo esto se resume en un vector, donde cada elemento contiene el número de motor que se impulsará y cuanto tiempo se esperará para el siguiente pulso de otro motor.

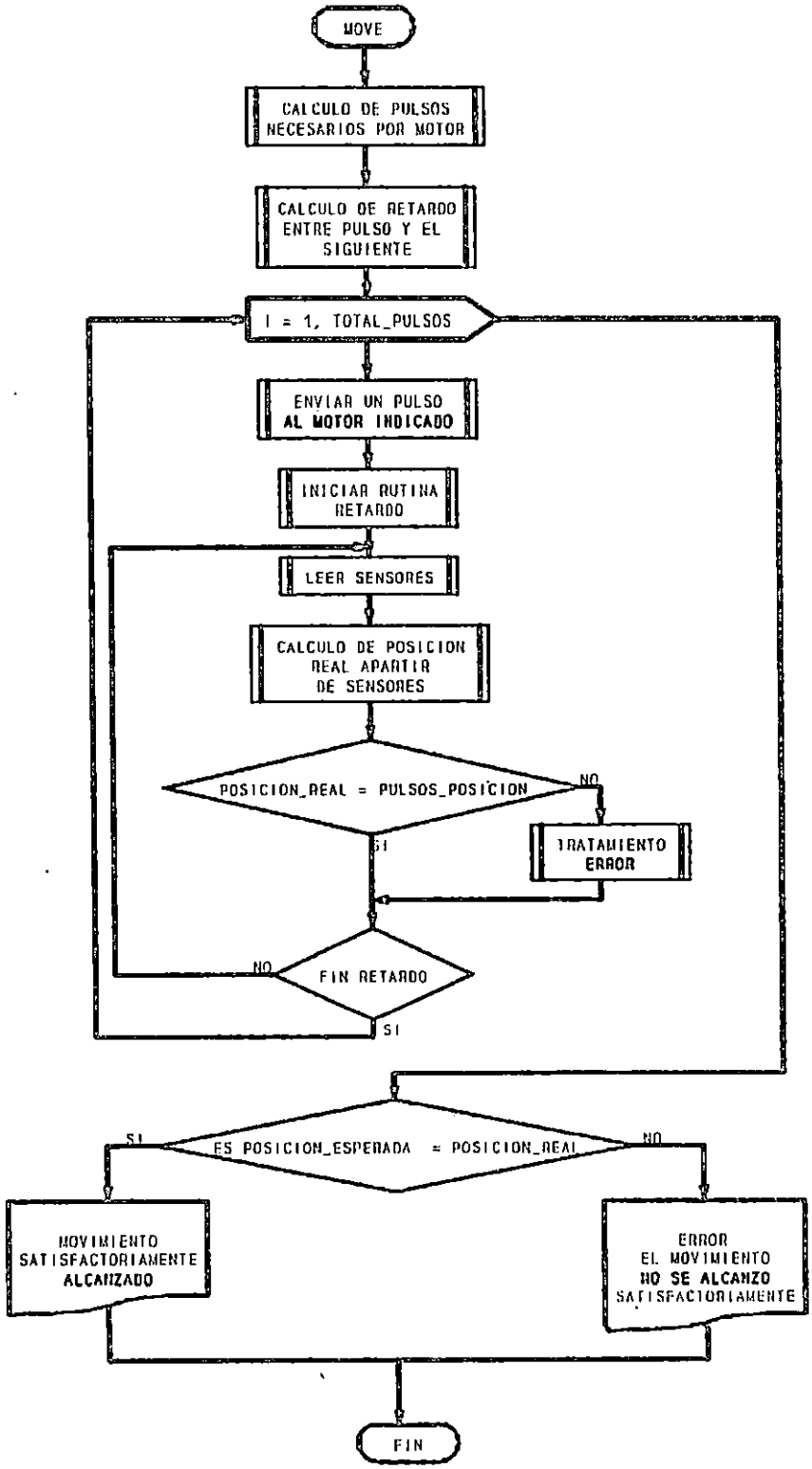


Figura 6.10. Módulo MOVE

Obtenido dicho vector de movimiento, se procede a enviar los pulsos a cada motor mediante un lazo. Este lazo envía el pulso al motor e inicia una rutina de temporización (retardo). Realmente, es más que una rutina de retardo; éste es un procedimiento desarrollado en ensamblador, el cual recibe un tiempo en milisegundos, y que cuando es activado retorna el control al computador, pero activa una temporización interna que es reflejada en una variable booleana denominada TimeOut. Mientras el tiempo de retardo transcurre, la variable TimeOut es falsa; en el instante que el tiempo termina, la variable TimeOut se hace verdadera. De esta manera, el computador puede realizar otras tareas mientras el tiempo de retardo transcurre, y mediante la revisión continua de TimeOut, el computador sabe que el tiempo de retardo ha transcurrido.

Mientras el retardo transcurre, el computador estará determinando la posición real mediante la lectura de los sensores. Además estará verificando si la posición real corresponde con la posición esperada (pulsos_posición), si no corresponde es probable que el motor haya perdido un paso, y pasará a un procedimiento denominado Tratamiento de Error.

El tratamiento del error funciona de la siguiente manera, si un paso es perdido, se corrige enviando un paso extra. Si la posición real coincide con la posición esperada, no ha pasado nada. Pero si después de 10 correcciones, todavía se incurre en el error, el procedimiento aborta la ejecución del programa y envía un mensaje al usuario para que verifique si el brazo no ha sido obstruido, o se ha roto un cable.

Cuando el retardo termina, el siguiente pulso es enviado y entra en la rutina de retardo nuevamente, lee sensores y verifica posición_real y posición esperada. De esta manera continua, hasta que todos los motores hayan desplazado hasta su posición angular deseada.

Como último procedimiento, se verifica si se alcanzó la posición final, y se envía un mensaje adecuado, como se muestra en el diagrama.

MOVEL

El procedimiento MOVEL es más complejo que MOVE, ya que el movimiento del efector final debe ser sobre una línea recta que va desde el punto inicial al punto final. En la figura 6.11 se muestra su respectivo diagrama de flujo.

A partir de la posición actual, y la posición deseada, el brazo tiene que ejecutar un movimiento en línea recta. El algoritmo seguido es el siguiente: se convierte el punto posición actual y el punto posición deseada a coordenadas XYZ. Por medios matemáticos se calcula la recta que va desde un punto hacia el otro y se determinan N puntos sobre dicha línea (estos puntos se obtienen en coordenadas XYZ).

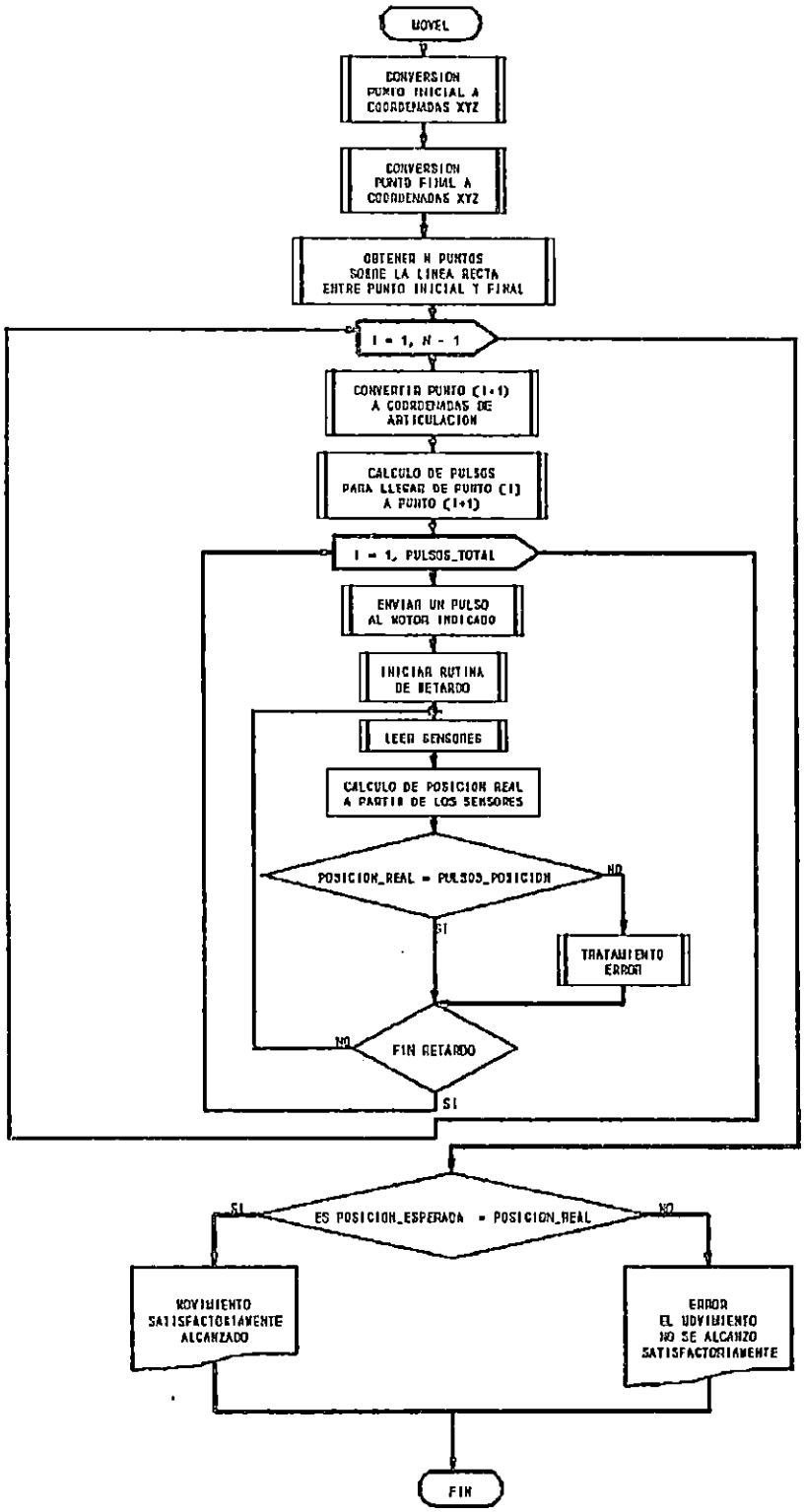


Figura 6.11. Módulo MOVEL

Para ejecutar el movimiento global, se ejecuta una serie de movimientos interpolados de articulación entre dos puntos próximos de los N puntos calculados sobre la línea recta.

El fundamento que se sigue es el siguiente: si el movimiento entre dos puntos próximos se hace interpolado de articulación, en vista de que están tan próximos, este movimiento será cercano a una línea recta, de tal manera que el movimiento total describirá una línea recta.

Las resolución que tenga el movimiento dependerá de cuantos puntos se calculen sobre la línea recta. A mayor cantidad, el movimiento será más cercano al movimiento lineal, pero posiblemente el procesador sea tan rápido para convertir tantos puntos de un sistema coordenadas a otro.

En vista que los puntos calculados sobre la línea recta se obtuvieron en coordenadas XYZ, para el calculo de los pulsos de cada motor se hace necesario la conversión de cada uno de estos puntos a coordenadas de articulación. Si se toman N puntos sobre la línea recta, se harán N conversiones de coordenadas XYZ a coordenadas de articulación.

6.7.2. PROCEDIMIENTOS CLOSE, CLOSER Y OPEN

Definición en Pascal:

```
procedure Close;
procedure Closer;
procedure Open;
```

Estos procedimientos constituyen las ordenes para el control de la garra, permitiendo cerrarla y abrirla.

CLOSER

En los dedos de la garra existe un switch, dispuesto de tal manera que cuando un objeto es presionado por ambos dedos, este switch se cierra, y una señal es enviada al computador. La instrucción CLOSER cierra los dedos hasta que la presión del dedo contra el objeto cierra el switch, entonces para el movimiento de cierre. En la figura 6.12 se muestra el diagrama de flujo de este procedimiento.

El algoritmo que sigue, es que el computador envía pulsos al motor de la garra hasta que sensor_close tenga un 1 lógico. Sensor_close es una variable booleana que indica si el switch está cerrado (1) o abierto (0). Pero hay una limitante del movimiento, si no hay ningún objeto entre la garra, el switch nunca cerrará; para corregir tal situación, el computador enviará pulsos (para cerrar garra) hasta que sensor_close sea 1 y la distancia entre los extremos de los dedos sea mayor que 1,5 cms. Esta última condición se traduce a un número fijo de pulsos que el motor recibe, y la expresión pulsos < max_garra indica esa condición.

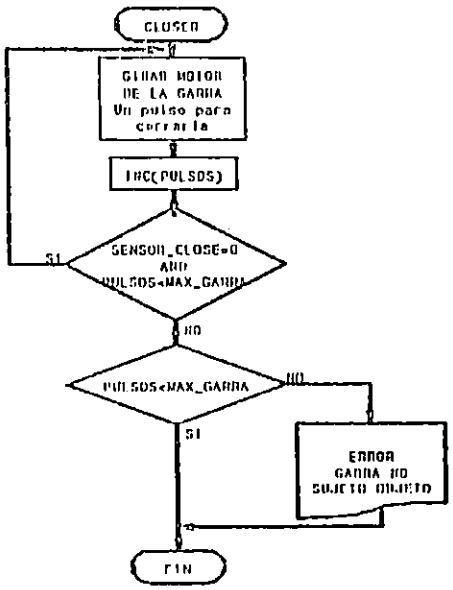


Figura 6.12. Procedimiento CLOSER

CLOSE

Esta instrucción ordena cerrar la garra hasta que entre los extremos de los dedos haya una distancia de 3 cms, ignorando el estado del switch del dedo de la garra. En la Fig 6.13 tenemos el diagrama de flujo de este procedimiento.

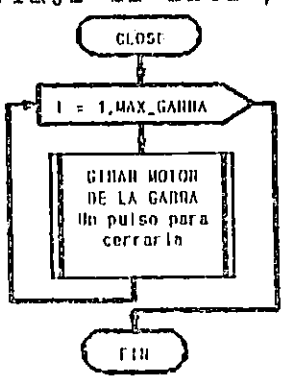


Figura 6.13. Procedimiento CLOSE

Este procedimiento se creó pensando en que hay muchos objetos que no ofrecen una consistencia suficiente para cerrar el switch de los dedos. Por ejemplo, para tomar una esponja, el usuario tiene que usar una instrucción CLOSE, lo cual permite cerrar la garra sin esperar que sensor_close pase a alto.

OPEN

La función OPEN abre los dedos de la garra. En la máxima abertura permisible se ha colocado un switch, de tal manera que no se exceda esta abertura. En Pascal hay una variable booleana correspondiente a este switch, Sensor_Open. Dicha variable es verdadera si la garra está en su máxima abertura,

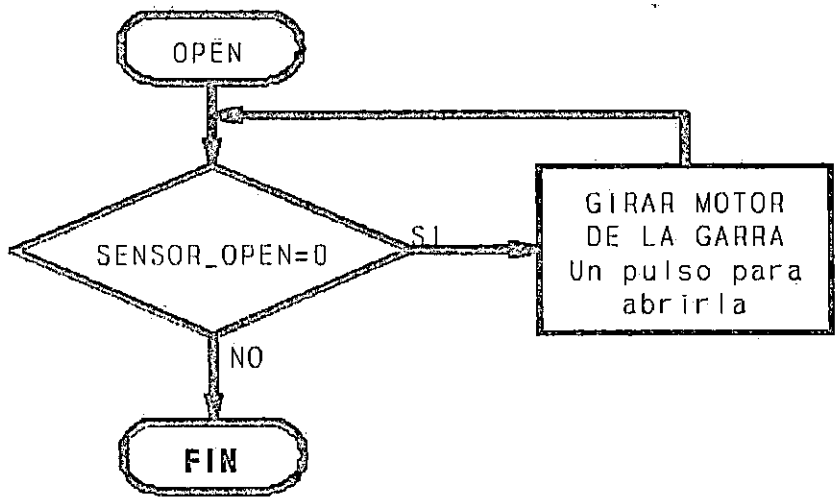


Figura 6.14. Módulo OPEN

y será falsa si está parcialmente o totalmente cerrada.

En la Fig. 6.14 se da el diagrama de este procedimiento. El computador envía pulsos al motor de la garra para que abra los dedos, hasta que Sensor_Open sea 1, lo que indica que la máxima abertura de los dedos ha sido alcanzada.

6.7.3. PROCEDIMIENTOS SIGNALON, SIGNALOFF,WAITON Y WAITOFF

Definición en pascal:

```
procedure SignalOn(num_port: byte);
procedure SignalOff(num_port: byte);
procedure WaitOn(num_port:byte);
procedure WaitOff(num_port:byte);
```

El sistema robótico tiene 8 líneas de puerto de 1 bit cada uno, y las cuales se encuentran a disposición del usuario. Dichas líneas están configuradas de la siguiente manera: de la línea 1 a la 4 son salidas y de la línea 5 a la 8 son entradas.

SIGNALON y SIGNALOFF

Estas dos instrucciones controlan los 4 puertos de salida. El parámetro que recibe cada señal es el número de puerto, puerto al que se envía un 0 con la instrucción SIGNALOFF o un 1 con la instrucción SIGNALON.

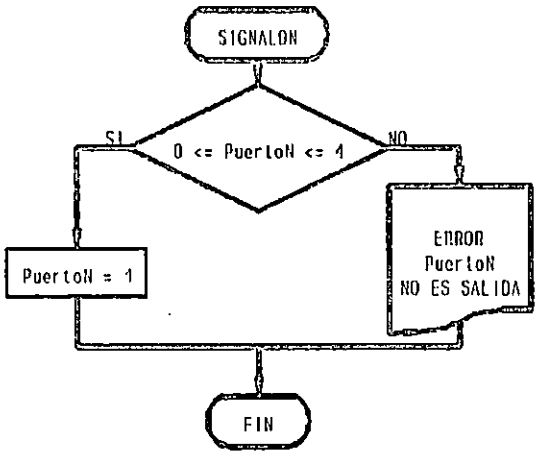


Figura 6.15. Módulo SIGNALON

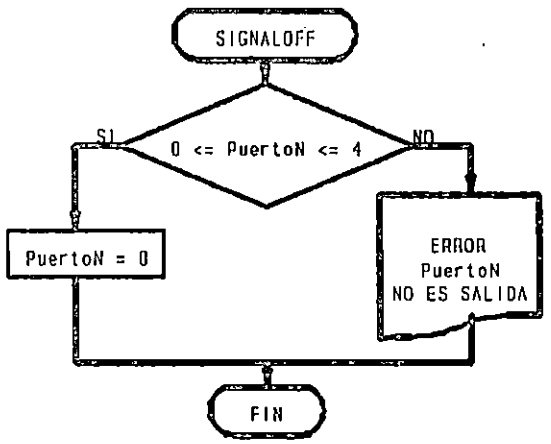


Figura 6.16. Módulo SIGNALOFF

En ambas rutinas se prueba si el número de puerto es realmente de salida, si no es así se genera un mensaje de error al usuario.

WAITON y WAITOFF

Estas dos instrucciones trabajan con los 4 puertos de entrada. El parámetro que reciben es el número de puerto. La lógica que siguen es la siguiente: al ejecutar la instrucción WaitOn, el programa para su ejecución hasta que en el puerto especificado haya un 1. En el instante que haya un 1, se sale de este procedimiento y sigue con las instrucciones restantes del programa en lenguaje de robot.

La misma lógica se aplica a la señal WaitOff, pero con lógica negativa, es decir que espera por un 0. Los diagramas de flujo de ambas instrucciones se muestran en las Figs. 6.17 y 6.18.

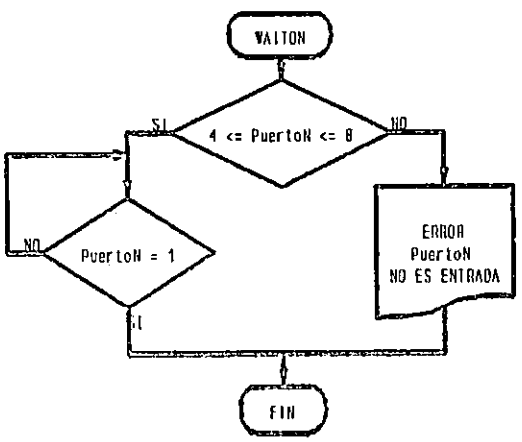


Figura 6.17. Módulo WaitOn

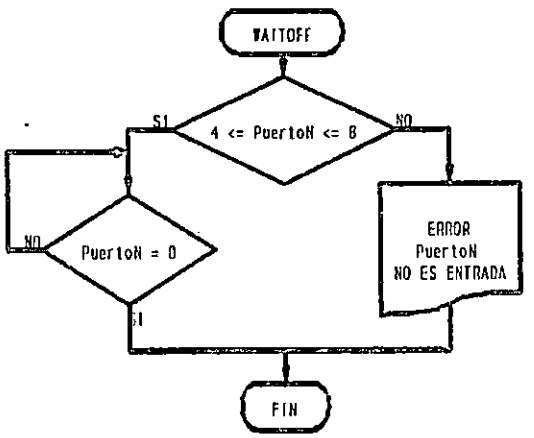


Figura 6.18. Módulo WaitOff

6.7.4. PROCEDIMIENTOS WRITE, WRITELN Y READ

Estas tres instrucciones corresponden a sus equivalentes en Pascal. Ellos permiten presentar mensajes (Write y Writeln) en pantalla y obtener datos desde el exterior (Read).

WRITE y WRITELN

Ambos procedimientos se utilizan para escribir mensajes en pantalla. La diferencia existente, es que WRITE, después de desplegar el mensaje en pantalla, posiciona el cursor al final del mensaje sobre la misma línea, en cambio WRITELN coloca el cursor en la siguiente línea sobre la primera columna. Los diagramas de flujo de ambas instrucciones lo tenemos en las Figs. 6.19 y 6.20.

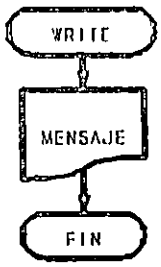


Figura 6.19. Módulo WRITE

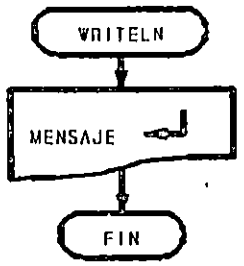


Figura 6.20. Módulo WRITELN

READ

Este procedimiento permite que el usuario introduzca desde el teclado valores a variables del programa.

El único tipo de variables que pueden ser introducidas de esta manera son las variables enteras. El parámetro que se le pasa a este procedimiento es un valor entero, que se utiliza como índice en la tabla de variables para saber la posición que ocupa dicha variable.

El diagrama de flujo de esta instrucción se muestra en la figura 6.21.

Figura 6.21. Módulo READ

6.7.5. PROCEDIMIENTOS GOTO, JMPON, JMPOFF, STOP, PAUSE, DELAY, DO-UNTIL y SUB-END

Este conjunto de instrucciones se utilizan para cambiar el flujo de ejecución del programa.

GOTO, JMPON y JMPOFF

La instrucción GOTO realiza un salto incondicional hacia cualquier otra instrucción. Esta instrucción permite romber el flujo de ejecución secuencial de las instrucciones, y posibilita continuar la ejecución con otra instrucción diferente a la siguiente.

La manera en que GOTO hace su trabajo es manipulando el program_pointer. En el nodo de la instrucción GOTO, se encuentra tanto la dirección del siguiente nodo de la cola, así como la dirección del nodo al que se saltará (llamada en Pascal Direccion_Salto). Antes de entrar a ejecutar el procedimiento GOTO, el traductor cargó en el program_pointer la dirección del siguiente nodo de la cola. En la ejecución del procedimiento GOTO, éste cambia el valor del program_pointer y le asigna el valor del nodo hacia el cual se va a saltar. Recordemos que cada nodo corresponde con una instrucción de programa.

El diagrama de flujo de tal instrucción se muestra en la Fig. 6.22.

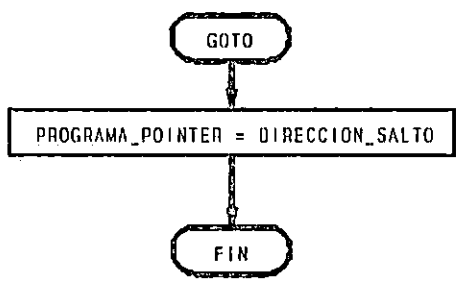


Figura 6.22. Módulo GOTO

Las instrucciones JMPOFF y JMPON también realizan un salto hacia otra instrucción, pero en forma condicional. Para el valor de la condición utilizan cualquiera de los puertos que se utilizan como entrada. En las Fig. 6.23 y 6.24 se muestran sus respectivos diagramas de flujo.

Ambos procedimientos utilizan dos parámetros, el primero es el número de puerto de entrada y el otro, la dirección de salto. La lógica de JMPON es la siguiente: si el puerto de entrada está en ALTO al ejecutar la instrucción, salta a direccion_salto, en caso contrario sigue con la siguiente instrucción en la secuencia del programa. La lógica de JMPOFF es idéntica, pero con lógica negativa (es decir que salta a direccion_salto si el puerto de entrada en BAJO).

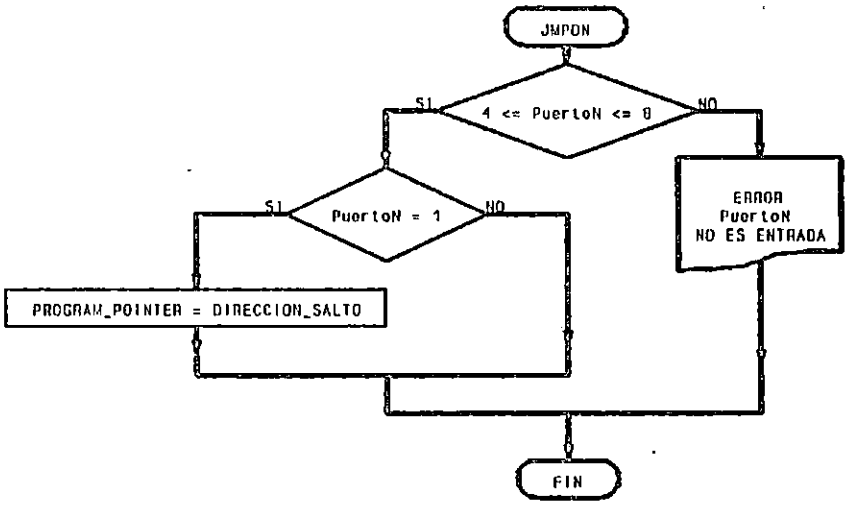


Figura 6.23. Módulo JMPON

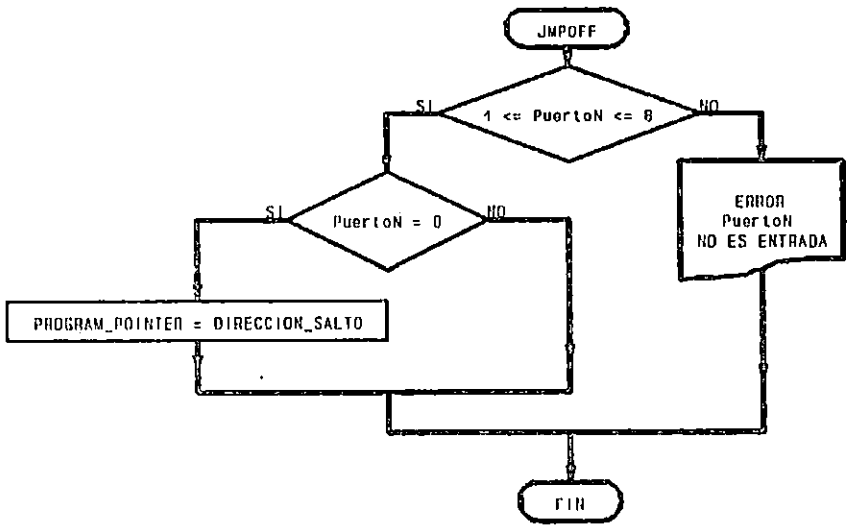


Figura 6.24. Módulo JMPOFF

STOP, PAUSE Y DELAY

Los tres procedimientos detienen el programa en alguna forma. Stop detiene el programa totalmente (regresa al programa monitor), y ajusta el program_pointer a la primera instrucción del programa.

La instrucción PAUSE detiene el programa y regresa al programa monitor. Si el programa se vuelve a ejecutar, la ejecución se podrá continuar desde la instrucción siguiente a PAUSE. Para ello se escoge en la opción Run, del programa monitor, la selección Resume. La opción Start ejecutaría el programa desde

la primera línea, aunque hayamos regresado al programa monitor después de una instrucción PAUSE.

Este procedimiento puede ser utilizado para un proceso de depuración, o para tareas donde se requiere de la intervención humana, en las cuales el tiempo requerido es variable.

DELAY detiene el programa durante Time segundos, luego continua con la siguiente instrucción del programa. Time es un parámetro requerido por la instrucción. Los diagramas de flujo de STOP, PAUSE y DELAY se muestran a continuación.

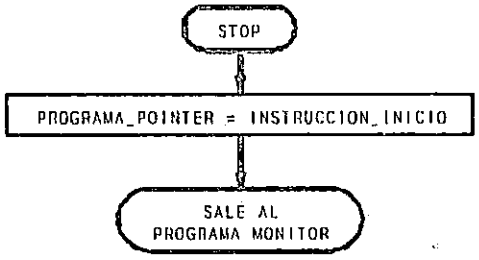


Figura 6.25. Módulo STOP

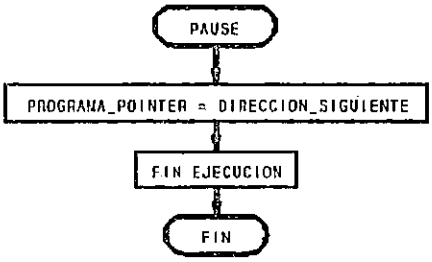


Figura 6.26. Módulo PAUSE

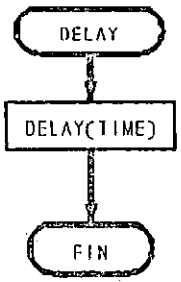


Figura 6.27. Módulo DELAY

DO-UNTIL, SUB-END y CALL

Las instrucciones DO-UNTIL se utilizan para ejecutar un lazo condicional. La forma en que trabaja es la siguiente: cuando el programa llega a la instrucción DO, le asigna un nodo a dicha instrucción; sigue ejecutando todas las instrucciones existentes entre el DO y UNTIL; al llegar a la instrucción UNTIL, el programa evalúa una condición, si la condición es falsa, el programa salta al nodo DO y repite la secuencia de instrucciones entre DO y UNTIL. Si la condición al evaluarse es verdadera, el programa sigue con la siguiente instrucción a UNTIL.

Es evidente que el nodo UNTIL debe contener la dirección de la instrucción siguiente, así como la dirección del nodo DO.

Las instrucciones SUB y END se utilizan para definir subrutinas de programa. Todos los nombres de rutina, así como su dirección de inicio y fin son almacenados en la Tabla de procedimientos. Cuando en el programa principal se llama un procedimiento con la instrucción CALL, se busca en la tabla de procedimientos la dirección del nodo final de la rutina (correspondiente a la instrucción END), y con ella se almacena en el nodo END la dirección del nodo siguiente a CALL (que sería la dirección de retorno de la rutina). Luego, el programa se sigue ejecutando en la primera instrucción de la rutina; al llegar al final de ésta, encuentra el nodo END, el cual contiene la dirección de retorno, y ésta es cargada en el program_pointer, de tal manera que la siguiente instrucción a ejecutar es la siguiente a CALL.

6.7.6. PROCEDIMIENTOS INC y DEC

Estas instrucciones permiten incrementar o decrementar en 1 el valor de un variable entera. Se han pensado para trabajar con lazos DO-UNTIL, en cuya condición de salida utilicen un contador, el cual es incrementado o decrementado con INC y DEC.

Los parámetro de ambos procedimientos es un valor entero, el cual se utiliza como índice en la tabla de variables. Este índice apunta a la variable sobre la cual se va a realizar la operación de incremento o decremento. Los diagramas de flujo de ambas instrucciones se muestran a continuación:

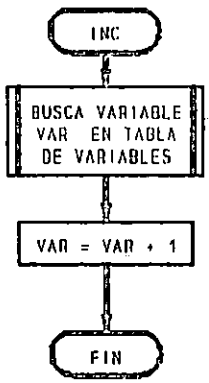


Figura 6.28 Módulo INC

Figura 6.29.Módulo DEC

6.8. MODO EDITOR

En la sección 5.5.3 se mencionó que el sistema operativo del software de control de robot tiene 3 modos: monitor, ejecución y edición. Hasta este momento se ha visto los modos monitor y modo ejecución.

En vista que hoy en día existen una gran variedad de editores de texto de gran potencia y capacidad, el diseñar un software para edición de programas sería un trabajo ya realizado por otro programadores. Por tal razón, se utilizará un editor muy

común en el mercado, que es el SIDEKICK V 1.56.

La característica por que la se escogió este editor, es porque se puede ejecutar manteniéndose residente en memoria, lo cual permite activarlo con solo presionar dos teclas (CTRL+ALT).

Dicha característica permite, que mientras el usuario esté corriendo el programa monitor, pueda accesar el editor proporcionado por el SIDEKICK sin la necesidad de salirse del programa monitor. Detallaremos mejor este procedimiento a continuación.

Antes de ejecutar el programa monitor, se carga el programa SIDEKICK. Cuando el SIDEKICK se ejecuta, el control del computador regresa al sistema operativo DOS, y aparentemente no hace ningún cambio, pero realmente el SIDEKICK se encuentra residente en memoria.

Estando el prompt del DOS, y con el SIDEKICK cargado en memoria, se ejecuta el programa monitor. Si estando en el programa monitor, deseamos editar algún archivo, únicamente invocamos el SIDEKICK presionando CTRL+ALT, con lo cual nos aparece el Main Menu del Sidekick.

Del menú principal seleccionamos la opción Notepad, el cual permite crear y editar archivos de texto. Después de seleccionar Notepad, nos pregunta el nombre de archivo a editar, digitamos el nombre y nos aparece la ventana de edición del SIDEKICK. Si deseamos regresar al programa monitor, únicamente presionamos ESC.

Algunos comandos elementales del SideKick son:

Función F2: guarda en disco el archivo que actualmente editamos.

Función F3: Crea un archivo de disco. En dado caso, ya existe el archivo, entonces lo carga a memoria para su edición.

Función F9: Expande la ventana de edición del SideKick.

Función F10: Comprime la ventana de edición del SideKick.

ESC: Regresa al modo monitor.

CTRL+ALT: Estando el programa monitor, permite accesar el SideKick. Si se presionan ambas teclas estando en SideKick, nos regresa al programa monitor.

El APENDICE D detalla la mayoría de comandos de edición del SideKick, así como describe las acciones que realizan las teclas de función.

CONCLUSIONES DEL CAPITULO VI

En este capítulo se definen una serie de procedimientos que permiten controlar el brazo para aplicaciones de mediana complejidad. Este conjunto de procedimientos, así como un conjunto de reglas sintácticas, definen lo que es el lenguajes de robot desarrollado.

Este conjunto de procedimientos, no son solo aplicables a este robot en particular, sino que puede ser generalizado a cualquier robot controlado por motores eléctricos, independiente de su tamaño o estructura física.

En este punto se está en capacidad de desarrollar programas de aplicación que hagan uso de todas las capacidades que posee el sistema brazo robótico.

Hay que hacer notar, que la lógica de implementación del lenguaje puede ser usada, no solo para desarrollar un lenguaje orientado a robot, sino también para cualquier otra aplicación que se desee, como algún instrumento que requiera ser programado, etc.

La idea principal de los algoritmos que permiten la codificación del lenguaje se tomaron de los algoritmos que utilizan los lenguajes ensambladores para producir un código ejecutable a partir de un programa fuente.

El lenguaje desarrollado tiene la ventaja principal de permitir su ampliación con nuevos procedimientos y funciones, que pueden enriquecer el lenguaje. Esta característica es la característica de expandibilidad, que permite expandir el lenguaje con nuevos procedimientos y funciones, que puedan mantener el lenguaje a nivel del desarrollo tecnológico alcanzado.

REFERENCIAS BIBLIOGRAFICAS

[1] Borland International. Turbo Pascal Tutor. Version 4.0. IBM version. Francia, 1987.

[2] Borland International. Turbo Pascal 4.0. Owner's Handbook. Francia, 1987.

[3] Haskell, Richard E.. IBM PC-8088. Assembly Lanquaje Programming. Rochester: Rehi Books, Inc., 1986.

[4] Grupo del Buch Engineering Co. Inc. Robotics Trainings Systems, Concepts and Aplications.

[5] Peralta, Juan y Portillo, Marcos. Diseño del Sistema Locomotor y Sensor de un Microrobot. Universidad de El Salvador, proyecto. 1992.

[6] Engelber, Joseph F. Robótica Industrial. Edit. McGraw Hill, España, 1a. Edición, 1989.

[7] Andeen, Gerry B. Robot Design Handbook. Edit McGraw Hill, España, 1a. Edición, 1988.

CONCLUSIONES Y RECOMENDACIONES GENERALES

1. La principal conclusión que se obtuvo al desarrollar el trabajo es que la robótica es un campo de la ingeniería aplicada, que integra tres áreas: ingeniería mecánica, eléctrica e informática. Teniendo tal visión, para un desarrollo de la robótica, ya se cuentan con recursos humanos que poseen amplios conocimientos de las áreas que la integran. Por lo tanto, para el desarrollo del área de robótica, el recurso fundamental que hace falta en nuestro medio es el económico, el cual posibilitaría integrar un grupo multidisplinario de investigación y desarrollo del área.

2. El trabajo se desarrollo modularmente, lo cual permite expandir y aplicar nuevas tecnologías en el sistema robótico. Los tres módulos principales que componen el brazo robot son: 1o) el manipulador, 2o) la circuiteria eléctrica y 3o) el software del sistema. A su vez, la parte eléctrica se compone de tres interfases: interfase controlador de los motores, interfase de sensores e interfase de comunicación con la computadora.

3. El módulo que controla los motores de paso puede manejar una amplia variedad de dichos motores, que utilicen corrientes de devanado entre 0.1A y 0.5A. Para manejar motores de mayor potencia (con corriente de devanado mayores a 0.5A), el integrado SAA1027 puede ser utilizado, pero sus salidas deben ser conectadas a transistores de potencia, los cuales proporcionarán la corriente necesaria a los devanados del motor de paso.

4. El software completo fue realizado en Pascal y Ensamblador del 8086. Se puede crear un proyecto cuyo objetivo sea crear una versión de tal software en lenguaje C, el cual posee cualidades ideales para desarrollar aplicaciones de instrumentación.

5. Para el desarrollo del área de robótica en la UES, se hacen las siguientes recomendaciones:

   a. Implementar una materia de robótica en la cual se estudien los aspectos fundamentales del área, y cuyo proyecto de materia sea construir un brazo robot de aplicación didáctica. Es de hacer notar que la UES cuenta con los recursos necesarios para tal proyecto.

b. Impulsar trabajos de graduación, cuyo objetivo principal sea el construir un brazo robótico de aplicación industrial, y que tenga las características necesarias para ser comercializado. Para lograr dicho efecto, se deben crear grupos multidisplinario, que abarque tanto estudiantes de Ingeniería Mecánica como de Ingeniería Eléctrica.

c. Impulsar las relaciones con la empresa privada, en busca de apoyo económico, incentivando a los productores con la mejora que obtendrían en su producción al usar brazos robot.

ANEXOS

ANEXO A

CALCULO DEL PESO DE LA ESTRUCTURA.

En un principio, se puede establecer que el peso de la estructura tiene una relación proporcional con la longitud del brazo. Para ello se establecerán ciertas ecuaciones, que nos darán una estimación de la variación del peso respecto a la estructura.

Estableceremos las siguientes variables:

Wa: peso en Newton del enlace del antebrazo.

Wb: peso en Newton del enlace del brazo.

L: longitud global del brazo.

Se partirá de la Ec. 3.8:

$$W = \frac{0.8 \cdot f}{L} - 0.4\ (Wa/2+3Wb/2) \qquad (3.8)$$

El objeto de este análisis es establecer una relación entre el peso W y la longitud L. Observamos en la Ec. 3.8 que aparte de W y L, se encuentra el peso de Wa y Wb involucrados.

Los pesos Wa y Wb están representando las fuerzas gravitatorias que ejerce la estructura del brazo, tanto enlaces como poleas, piñones, ejes, etc. El enlace son las planchas de aluminio que unen dos articulaciones, y forman el esqueleto del brazo, sobre el cual se apoyan el resto de elementos del brazo.

Se ha asumido que Wa representa el peso ejercido por los elementos que se encuentran en el antebrazo, y Wb es el peso debido al brazo. Si denominamos Wa' y Wb' al peso debido exclusivamente a los enlaces del antebrazo y brazo, podemos establecer una relación entre Wa' y Wb' con la longitud L.

A continuación calcularemos el volumen de los enlaces del antebrazo y brazo, Va y Vb, dando los siguientes valores:

El antebrazo está compuesto por una plancha de aluminio cuyas características son :

Ancho = 2"

Espesor = 2mm de espesor

Longitud = 0.5L

El brazo está compuesto por dos planchas de aluminio cuyas características son:

Ancho = 1 1/4"

Espesor = 2mm de espesor

Longitud = 0.5L

El volumen, en metros vendría dado por la expresión

$$V = 2 \cdot \text{ancho} \cdot \text{espesor} \cdot \text{longitud}.$$

El 2 se usa ya que son dos planchas de aluminio, lo cual el volumen total es el doble del volumen de un enlace.

Evaluando Va y Vb, obtenemos:

$$Va = L / 9842 \ m^3 \qquad (A.1)$$

$$Vb = L / 15748 \ m^3 \qquad (A.2)$$

Donde L está en metros.

La densidad del aluminio es de $\mathcal{E} = 0.1$ lb/pulg^3 (2773.8 Kg/m^3), con lo cual podemos obtener el peso Wa' y Wb' en función de L, de la expresión:

$$W = m \cdot g = (\mathcal{E} \cdot V) \cdot g \quad \text{---->} \quad g \text{ es la gravedad.}$$

Por tanto:

$$Wa' = (2773.8 \cdot L/9842) \cdot 9.8 = 2.76L$$

y

$$Wb' = (2773.8 \cdot L/15748) \cdot 9.8 = 1.73L$$

Se ha encontrado una relación entre Wa' y Wb' con la longitud L. Entre la Wa y Wa', así como Wb y Wb', no existe una relación matemática, ya qué la diferencia entre ambas es el peso del resto de la estructura (engranes, poleas, ejes, tornillos, etc.), que no tienen un peso en relación directa con la longitud L. Para solventar este problema, se hará la siguiente suposición: el peso total de la estructura del antebrazo, Wa, será dos veces el peso sus enlaces, Wa'.

Por lo tanto:

$$Wa = Wa' = 5.52L \qquad (A.3)$$

y

$$Wb = Wb' = 3.46L \qquad (A.4)$$

Sustituyendo las Ecs. A.3 y A.4 en Ec. 3.8, obtenemos:

$$W = \frac{0.8 \cdot f}{L} - 0.4\ (5.52L/2 + 3 \cdot 3.46L/2)$$

Evaluando y reduciendo, obtenemos:

$$W = 0.8 \cdot f/L - 3.18L \qquad (A.5)$$

De las especificaciones, se sabe que $W \geq 5$ lbs (22.27N). Si A.5 se plantea como una inecuación que cumple esta especificación, obtenemos:

$$W = 0.8 \cdot f/L - 3.18L \geq 22.27 \qquad (A.6)$$

Si establecemos una función f(L), en la cual relacionamos el factor de amplificación en función de la longitud L, obtenemos:

$$f \geq 27.84L + 3.96L^2$$

Si graficamos esta función, para $0 \leq L \leq 1$ mt, el resultado lo observamos en la Figura A.1.

Análisis Gráfico.

La parabólica que relaciona a f con L, se ha denominado Fw(L). También se han graficado las relaciones dadas por las Ecs. 3.9.b y 3.13.b, que son las limitantes de la resolución y la velocidad. En la gráfica hay ciertas zonas de vital importancia. La gráfica Fv(L) representa el límite impuesto por la velocidad máxima, de esta forma, al realizar el diseño, no se puede tomar un par (f, L) que éste arriba de Fv(L).

La gráfica Fw(L) representa el límite impuesto por el peso, y en el proceso de diseño del brazo, no se puede tomar un par (f,L) qué esté por debajo de esta gráfica, Fw(L), ya que no cumpliría la especificación del peso (el peso máximo a levantar sería menor de 5 lbs). De esta forma el rango de diseño es el área entre las gráfica Fv(L) y Fw(L).

De la figura A.1, en el rango $0 \leq L \leq 0.5$ mts, la función parabólica Fw(L) tiene una aproximación lineal muy buena a la función. Para ello, observe los pares (0,0), (3,0.1), (6,0.2), (9,0.3), (12,0.4) y (15, 0.5), los cuales describen una línea

recta, que viene descrita por:

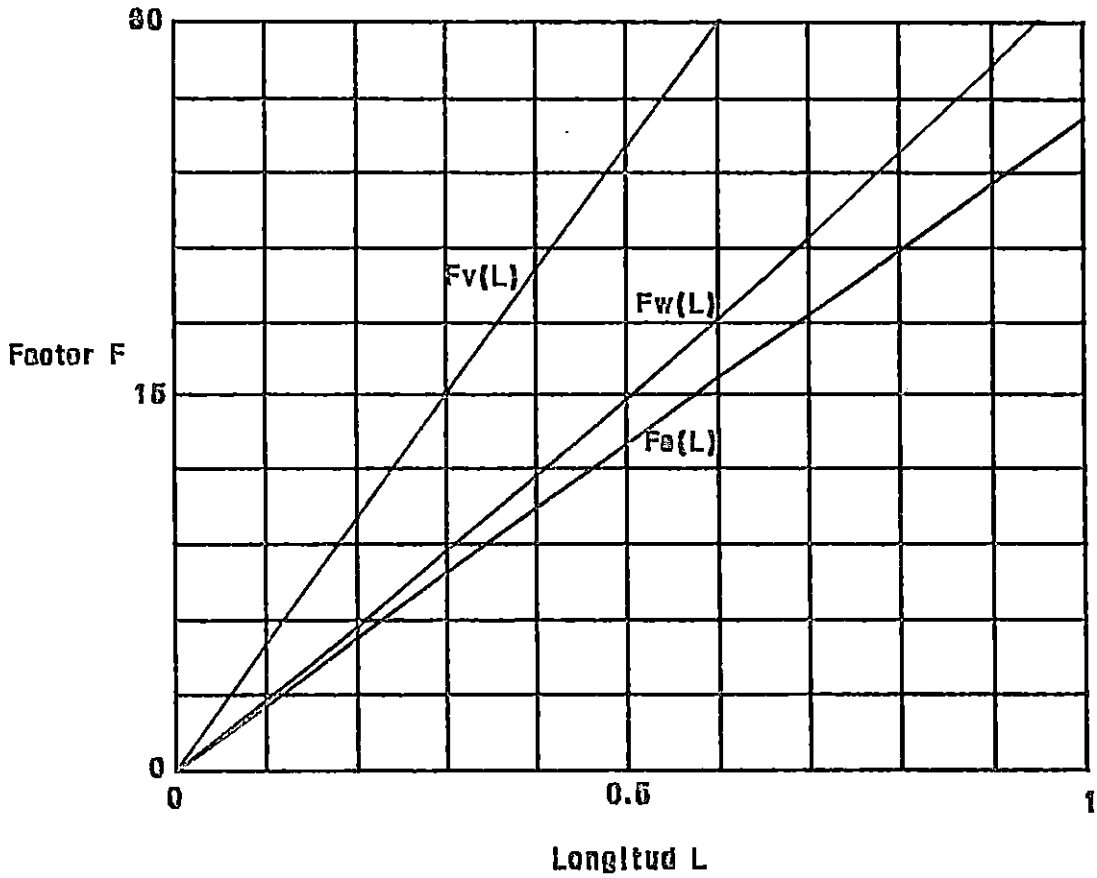


Figura A.1. El factor de amplificación en función de la longitud L del brazo.

$$f = (15/0.5) \cdot L$$

es decir: $$f \geq 30 \cdot L \qquad (A.7)$$

valida para $$0 \leq L \leq 0.5 \text{ mts}$$

En el análisis estático del antebrazo, se asumió Wa= 11b y Wb=11b, lo cual nos dio una limitante para el peso de:

$$f \geq 32.3 \cdot L. \qquad (3.14.b)$$

De lo cual concluimos que la limitante considerada al diseñar ($f \geq 32.3$) satisface el comportamiento de la carga con respecto a la longitud en el rango de $0 \leq L \leq 0.5$ mts.

ANEXO B

TABLA B.1. Especificaciones principales del IC SAA1027 PHILIPPS

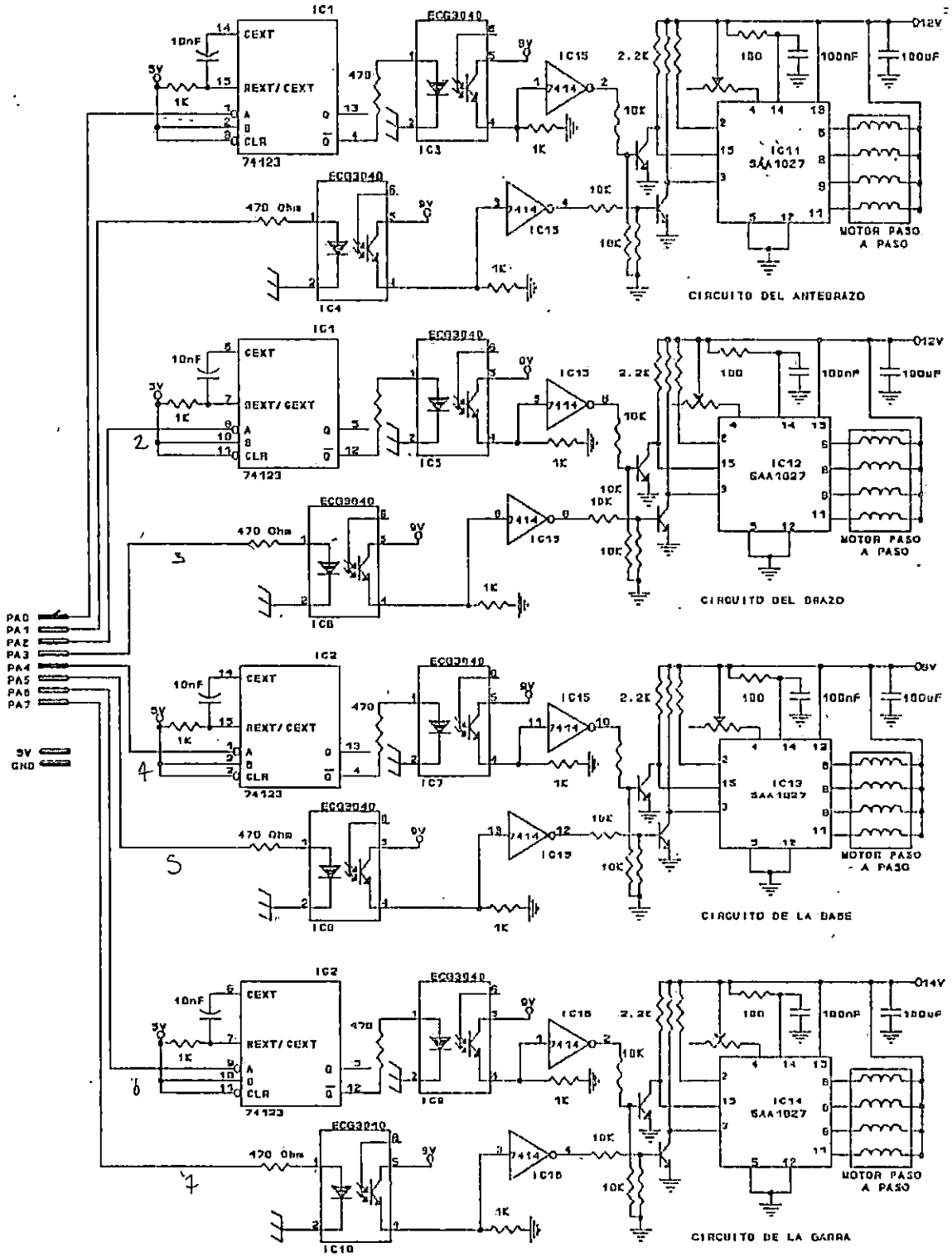
Banda de tensiones de alimentación de 9.5 a 18V.
Corriente sin carga igual a 4.5 mA.
Corriente de entrada I_{OL} = 30µA.
Corriente de salida I_{OL} = 500 mA.

TABLA B.2. Especificaciones generales del IC SAA1027

Parámetro	Símbolo	Min.	Tip.	Máx.	Uni.
Alimentación Vcc1 (pins 13 y 14). Corriente de alimentación, Vcc1 no cargado; Entradas HI; pin 4 abierto.	Icc	2	4.5	6.5	mA
Entradas C, M y R (pines 15, 3 y 2). Tensión de entrada					
HI	V_{IH}	7.5	–	–	V
LO	V_{IL}	–	–	4.5	V
Corriente de entrada					
HI	I_{IH}	–	1	–	µA
LO	I_{IL}	–	30	–	µA
Resistor externo Rx (pin 4). Tensión en Rx con Vcc = 12V ± 5% R4 = 130Ω ± 5%	V_{RX}	3	–	4.5	V
Salidas Q1 a Q4 Tensión de salida LO					
con I_{OL} = 350 mA	V_{OL}	–	500	1000	mV
con I_{OL} = 500 mA	V_{OL}	–	700	–	mV
Corriente de salida					
LO	I_{OL}	–	–	500	mA
HI con VQ1 = 18V	I_{OH}	–	–	50	µA

ANEXO C:

CIRCUITO GLOBAL DE CONTROL DE MOTORES



ANEXO D

COMANDOS BASICO DEL SIDEKICK

CURSOR MOVEMENTS:	COMMAND	KEY
Character left	Ctrl-S	<-
Character right	Ctrl-D	->
Word left	Ctrl-A	Ctrl-<-
Word right	Ctrl-F	Ctrl-->
Line up	Ctrl-E	^
Line down	Ctrl-X	
Scroll up	Ctrl-W	
Scroll down	Ctrl-Z	
Page up	Ctrl-R	PgUp
Page down	Ctrl-C	PgDn

CURSOR MOVEMENTS:	COMMAND	KEY
To left on line	Ctrl-Q-S	Home
To right on line	Ctrl-Q-D	End
To top of page	Ctrl-Q-E	Ctrl-Home
To bottom of page	Ctrl-Q-X	Ctrl-End
To top of file	Ctrl-Q-R	Ctrl-PgUp
To end of file	Ctrl-Q-C	Ctrl-PgDn
To end of file & insert date/time	Ctrl-Q-O	
To beginning of block	Ctrl-Q-B	
To end of block	Ctrl-Q-K	
To last cursor position	Ctrl-Q-P	

INSERT & DELETE	COMMAND	KEY
Insert mode on/off	Ctrl-V	Ins
Insert line	Ctrl-N	
Delete line	Ctrl-Y	
Delete to end of line	Ctrl-Q-Y	
Delete right word	Ctrl-T	
Delete char under cursor	Ctrl-G	Del
Delete left character	Ctrl-H	<--

BLOCK COMMANDS	COMMAND	KEY
Mark block begin	Ctrl-K-B	F7
Mark block end	Ctrl-K-K	F8
Mark single word	Ctrl-K-T	
Hide/display block	Ctrl-K-H	
Copy block	Ctrl-K-C	
Move block	Ctrl-K-V	
Delete block	Ctrl-K-Y	
Read block from disk	Ctrl-K-R	
Write block to disk	Ctrl-K-W	
Sort block	Ctrl-K-S	

MISC. COMMANDS	COMMAND	KEY
Save note file	Ctrl-K-D	F2
Paste block	Ctrl-K-E	
Cut and paste		F4

Paste block lets you paste a block of text from the Notepad to any other program.

Cut and paste lets you cut data from the screen and paste it into the Notepad.

MISC. COMMANDS	COMMAND	KEY
Tab	Ctrl-I	
Repeat last find	Ctrl-L	
Control character prefix	Ctrl-P	
Re-format line	Ctrl-B	
Set right margin	Ctrl-O-R	
Find	Ctrl-Q-F	
Find & replace	Ctrl-Q-A	
Auto tab on/off	Ctrl-Q-I	
Restore line	Ctrl-Q-L	
Read date and time	Ctrl-Q-T	

When off, you may edit text produced by editors using the 8th bit of the characters (like WordStar). When on, you can see the entire 256-character IBM PC character set, including the semigraphics.

The right margin is set to column 65 by default (unless changed during installation). You may change this value with the Ctrl-O-R command.

When you reach the right margin, a line break will automatically be inserted, and the last word is moved to the new line if it does not fit within the margins.

The current right margin is saved permanently when

text. A paragraph of text ends with a blank line. For example, this is the first paragraph on this page.

This is the second paragraph. The first paragraph ends in line 4 because of the blank line following it.

This is still the second paragraph. An indent does not start a new paragraph.

You can disable word wrap by setting the right margin to 250.

FUNCTION KEYS

F1 Help. The help key will display detailed help about Notepad. Depending on which features you are using you will get different help texts.

F2 Save. Saves the contents of your note file on disk. In order to let you switch disks this is never done automatically, and you must therefore remember to save for instance before you shut down the computer. You may also use the WordStar command Ctrl-K-D to save.

F3 New note file. Define the name of a file to be used as note file. The default file name is NOTES (unless changed in Setup).

F4 Cut and paste. The screen as it were before Notepad was opened will appear. You may then mark a block of text on the screen with the normal block markers and paste it into the Notepad with the Ctrl-K-C command.

F9 Expand window. Pressed once, this key causes the arrow keys to move the borders of the Notepad window outwards, expanding its size. When pressed again, F9 returns the arrows to their normal use.

F10 Contract window. Pressed once, this key causes the arrow keys to move the borders of the Notepad window inwards, contracting its size. When pressed again, F10 returns the arrows to their normal use.

Marcado de bloques:
Hold down the Ctrl-key and type K and B to mark the block beginning (upper left corner). Use the ^ and -> keys to move the cursor to the end of the block, the lower right corner. The block is marked on the screen as the cursor moves along.

ANEXO E:
DIAGRAMA CIRCUITAL DE LOS SENSORES