UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA

FACULTAD DE INGENIERÍA

ESTUDIO Y EXPERIMENTACIÓN EN UN ROBOT PARALELO STEWART (UPS): REPRODUCCIÓN DE TRAYECTORIAS Y CONTROL EN EL ESPACIO DE JUNTAS.

TESIS

QUE PARA OBTENER EL GRADO DE:

MAESTRO EN INGENIERÍA

CAMPO DE CONOCIMIENTO-MECATRÓNICA

ING. NOÉ ALFREDO MARTÍNEZ SÁNCHEZ

DIRECTOR DE TESIS:
DR. VÍCTOR JAVIER GONZÁLEZ VILLELA

CODIRECTOR DE TESIS:
M.I. SOUZA JIMÉNEZ JOSÉ ANTONIO

CIUDAD UNIVERSITARIA, NOVIEMBRE 2011

JURADO ASIGNADO:

Presidente:	Dr. Marcelo López Parra.
Secretario:	Dr. Rojas Salgado Ángel Alfonso.
Vocal:	Dr. González Villela Víctor Javier.
1er. Suplente:	Dr. Rocha Cózatl Edmundo Gabriel.
2do. Suplente:	M.I. Martínez Zamudio Patricio.

Lugar donde se realizo la tesis:

México D.F. Ciudad Universitaria.

TUTOR DE TESIS

Dr. Víctor Javier González Villela

_____

Firma

Agradecimientos:

A mi familia por todo su apoyo brindado.

A mi director de tesis el Dr. Víctor Javier González Villela, por sus consejos y asesorías.

A José Antonio Souza Jiménez, por sus consejos y apoyo brindado en la realización de esta tesis.

Al Dr. Rojas Salgado Ángel Alfonso.

Agradezco los apoyos brindados por la DGAPA, UNAM, a través de los proyectos PAPIIT IN108308 y PAPIIT IN114711, con títulos: “Investigación y desarrollo en robótica móvil, robótica paralela y sistemas mecatrónicos” y "Optimación de parámetros para trayectorias de robots con consumo mínimo de energía" respectivamente.

A CONACYT por su apoyo al programa de Maestría.

Índice.

Resumen.

Índice de figuras

Capítulo 1

Capítulo 2

Capítulo 3

Capítulo 4

Capítulo 5

Capítulo 6

Índice de tablas

Apéndices

Resumen

En este trabajo se hacen experimentaciones sobre el robot paralelo Stewart, concretamente se generan trayectorias para ser reproducidas por el robot. El robot consiste en una plataforma fija y una plataforma móvil unidas por 6 articulaciones, con juntas esféricas para la base y juntas universales, que unen los actuadores prismáticos con la plataforma móvil.

Se presenta la cinemática inversa; basándose en el manejo de matrices de transformación homogénea se determina la posición, velocidad y aceleración de cada una de las juntas de las cadenas para una trayectoria definida. También, se presenta el modelado e identificación de los motores de CD y el diseño del controlador de posición. Así mismo, se expone la interfaz diseñada en Matlab, la tarjeta principal de control y la instrumentación del robot.

Finalmente se presentan una serie de experimentos hechos con el fin de mostrar que el mecanismo es capaz de reproducir una trayectoria dada. Para la validación del modelo cinemático y las trayectorias realizadas por el robot se hace uso de un acelerómetro y el software de visión reacTIVision.

Capítulo 1

Generalidades

1.1 Introducción.

Un robot paralelo es aquel cuya estructura mecánica, está formada por un mecanismo de cadena cerrada, en el que el efector final se une a la base por al menos dos cadenas cinemáticas independientes. Los robots paralelos simplifican estas cadenas de forma que cada una de ellas dispone, en general de un único actuador.

Estas estructuras cerradas poseen diversas ventajas, tales como que pueden alcanzar velocidades y aceleraciones mucho más grandes que en el caso de los robots serie, manejan cargas superiores, poseen alta rigidez dado que el peso es repartido entre las diferentes cadenas cinemáticas, y con ellos se logra una mayor precisión. Igualmente presentan ciertas desventajas frente a los robots serie tradicionales como son el hecho de poseer una cinemática más compleja, un espacio de trabajo más pequeño y más difícil de definir, un manejo de las singularidades muy particular para cada estructura, y una mayor exigencia en los algoritmos de control dada la alta dinámica que proporcionan estas estructuras (Merlet, 1994; Merlet, 1997).

Las figuras 1.1a y 1.1b muestran robots de estructura serie y paralela respectivamente.

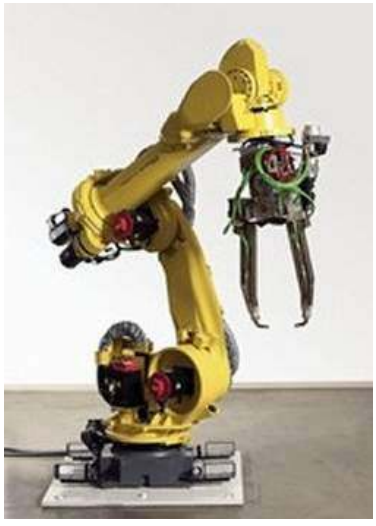


Figura 1.1a Robot serial (Fanuc).

Figura 1.1b Robot paralelo (Hydra-Power system).

1.2 Aportación.

Respecto a robótica paralela en la UNAM se han desarrollado trabajos como “Análisis cinemático de un Manipulador Paralelo Hibrido Tipo Delta [27]” un tipo de robot hibrido que está formado de dos robots paralelos, “Análisis Cinemático y Balanceo de una Plataforma Paralela de 5 GDL[21]”, “Análisis Estático de una Plataforma de seis grados de libertad[24]”, "Análisis Cinemático y Dinámico de un Robot Delta de 3 Grados de Libertad[22], Análisis Cinemático y Dinámico de un Robot Paralelo tipo Diamante[23], Diseño Mecánico de un Robot Paralelo Delta de tres Grados de Libertad[25], Modelación y Simulación Cinemática de un Robot Delta Planar Tipo RR[26], Análisis Cinemático y Dinámico de un Robot Paralelo Espacial RUS [29], en ningún trabajo de los mencionados anteriormente a excepción de [27] se hacen experimentaciones con un prototipo real . Se pretende que este trabajo sirva como un punto de convergencia entre las líneas de investigación y permita desarrollar trabajos futuros relacionados con el control de robots paralelos dado que en la actualidad no hay muchos trabajos relacionados al control de este tipo de robots.

Una importante aportación de esta tesis es el banco de pruebas que consiste en la instrumentación del robot paralelo Stewart y el desarrollo de su tarjeta de control, que será utilizado como plataforma experimental para futuros trabajos con este robot en el laboratorio de instrumentación y control.

1.3 Planteamiento del problema.

1.3.1 Hipótesis.

Dada una trayectoria en base a la cinemática inversa del robot Stewart el sistema mecánico es capaz de reproducir esta trayectoria con un controlador en el espacio de juntas. Esta hipótesis será comprobada experimentalmente.

1.4 Objetivos.

1.4.1 Objetivo general.

Reproducción de trayectorias en el robot paralelo.

1.4.2 Objetivos particulares.

1.-Obtención de las ecuaciones de la cinemática inversa del robot Stewart.
2.-Modelado y diseño del controlador de los motores del robot.
3.-Instrumentación y realización de la tarjeta controladora de motores
4.-Interfaz gráfica en Matlab.
5.-Comparación de los resultados obtenidos en la simulación con los resultados experimentales.

1.5 Justificación.

Las aplicaciones de los robots paralelos se encuentran en una etapa de crecimiento y desarrollo siendo estos robots todavía un tema abierto de investigación.

La principal justificación de este trabajo de investigación, es contar con una plataforma experimental con su hardware de control para poder evaluar diferentes esquemas de control, relacionados con el seguimiento de trayectorias y otras aplicaciones.

Los robots manipuladores paralelos que actualmente se construyen a nivel comercial y que presentan diferentes tipos de aplicaciones son [4]:

El robot Delta desarrollado por R. Clavel utilizado en industrias de manipulación de alimentos. Figura (1.2a).

El robot Tricept patentado por K.N. Neuman, utilizado en maquinas herramientas de media precisión e incursionando en el campo de la medicina. Figura (1.2b).

La plataforma Gough Stewart, la cual es el manipulador paralelo mas construido y es utilizada para orientación de antenas, telescopios, péneles solares, aislamiento y producción de vibraciones, posicionamiento de microscopios y pacientes, simuladores de vuelo de avión, vehículos elevadores, ensamble de piezas y como máquina herramienta de precisión media.

Figura 1.2a Robot Flexpicker. Figura 1.2b Robot Tricept. Figura 1.2c Robot Stewart.

Algunos laboratorios ubicados principalmente en Canadá, USA, Alemania, Italia, Francia, Korea del Sur, Japón, Suiza, están realizando investigaciones con los robots paralelos. Investigaciones tales como: Calibración en robots de cinemática paralela del Laboratorio Robotics and Mechanism Laboratory Canada. Robot paralelo de alta precisión y aislamiento de vibraciones Parallel Robotics Group USA, Universidad de Wyoming. Operación a altas velocidades y aceleraciones con robots paralelos en el campo de la automatización y producción laboratorio Collaborative Research Center 562 TU Braunschweig Alemania. Robots paralelos como centro de maquinado, Parallel Robotics Group, Universidad de Swiss Federal Institute of Technology Lausanne (EPFL), Suiza. Robot

Stewart para el mantenimiento de válvulas submarinas, PMAR lab, Universidad de Jenova Italia. Robot paralelo para aplicaciones de endoscopia, INRIA, COPRIN Project, Francia. Robot paralelo redundante como simulador de vuelo, Seoul National University, Korea del Sur. Aplicaciones medicas con robots paralelos, Robotics Laboratory, Nanyang Technological University, Singapore, Se puede observar que estos tipos de robots tienen muchas aplicaciones en diversos campos del conocimiento.

1.6 Metodología.

La metodología a seguir es la siguiente: se estudia la cinemática generando el modelo cinemático inverso y se desarrolla la solución numéricamente, se generan varias trayectorias para verificarlas en el prototipo, se hace la identificación del modelo dinámico de los actuadores de robot para posteriormente hacer un control de posición sobre ellos, se desarrolla el hardware compuesto por la tarjeta de control de motor, generación de una interfaz gráfica en Matlab para el envió de posiciones a la tarjeta de control, y análisis de resultados.

1.7 Características generales de los robots paralelos.

Las aplicaciones con robots paralelos han ido creciendo, son utilizados en determinadas aplicaciones donde ofrecen ventajas que los robots series no por su configuración mecánica.

A continuación se enumeran una serie de ventajas y desventajas de los robots paralelos, frente a los robots seriales.

En forma general los robots paralelos presentan las siguientes ventajas:

Los accionamientos de potencia conectan directamente la base del robot al efector final debido a esto, los accionamientos de potencia sirven de elementos estructurales y actúan de manera simultánea, lo que les da la capacidad de manipular cargas muy superiores a su propio peso. Por tanto, el elevado ratio carga/peso de estos mecanismos proporciona una alta eficiencia energética.

Las estructuras paralelas son mecanismos que ofrecen alta rigidez y muy bajo peso, esto hace que presenten unas características en cuanto a precisión claramente superiores a las de los robots serie.

Presentan elevadas velocidades de operación, en comparación con cualquier otro tipo de estructura robótica.

De las desventajas podemos mencionar de forma general:

La cinemática de los mecanismos paralelos es más complicada. En ocasiones esto obliga a recurrir a sensores redundantes para poder establecer un lazo de control.

El espacio de trabajo suele ser pequeño comparativamente hablando. Además no es tan sencillo su cálculo, pues la posición y la orientación están fuertemente acopladas.

El problema de las configuraciones singulares es más complejo, y debe resolverse específicamente para cada topología.

No existe, como ocurre en los robots serie, un modelo dinámico general para los mismos. Esto dificulta el desarrollo de algoritmos de control de carácter general (5).

1.8 Configuración cinemática y grados de libertad.

La figura 1.3 muestra la estructura cinemática que se analizara, la cual consiste en un arreglo en paralelo de seis cadenas cinemáticas idénticas que se distribuyen de acuerdo a la configuración cubica o 3-3 de la plataforma Stewart, el robot consta de 6 juntas esféricas para la base y de 6 juntas universales para la unión del efector final.

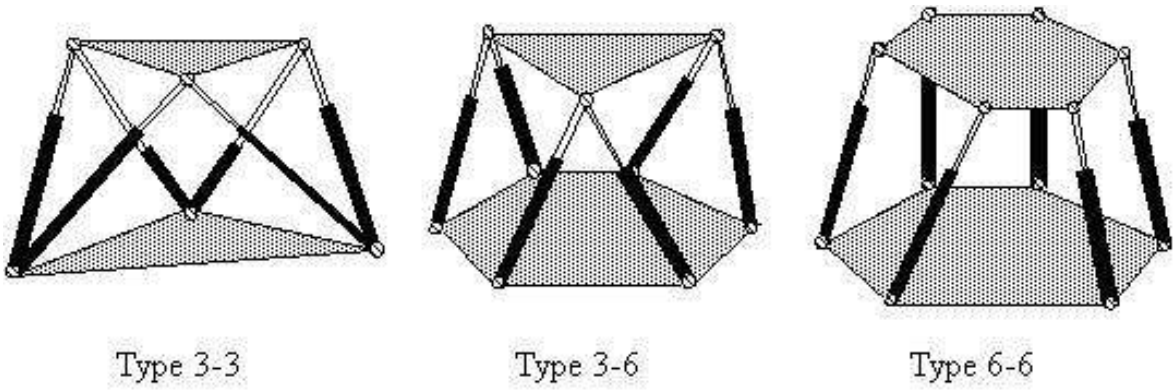


Figura 1.3 Configuraciones robot Stewart.

Figura 1.4 Prototipo robot Stewart (UPS) configuración cubica.

Grados de libertad

En general, el número de grados de libertad de un mecanismo de lazo cerrado no es obvio y puede calcularse mediante la fórmula de Grubler,

$$F = 6(l - n - 1) + \sum_{i=1}^{n} f_i,$$

en donde F es el número total de grados de libertad en el mecanismo, l es el número de vínculos (incluyendo la base), n es el número total de articulaciones y fi es el número de grados de libertad asociados con la i-esima articulación.

Utilizando la formula de Grubler se verifica que la plataforma Stewart tiene 6 grados de libertad

El número de articulaciones es de 18 (6 universales, 6 esféricas y 6 prismáticas en los actuadores). El número de vínculos es de 14 (2 piezas para cada actuador, el efector final y la base). La suma de todos los grados de libertad de las articulaciones es de 36. Utilizando la formula de Grubler, podemos verificar que el número total de grados de libertad es de seis:

F=6(14-18-1)+36=6.

Capítulo 2

Análisis Cinemático

2.1 Introducción.

El análisis del sistema inicia con el estudio de la cinemática inversa; basándose en el manejo de matrices de transformación homogénea se determina la posición de cada una de las juntas de las cadenas para una trayectoria definida.

El análisis cinemático se enfoca al estudio de la geometría de los eslabones y su movimiento sin considerar las causas que lo originan, en este capítulo se desarrolla el análisis de posición, velocidad y aceleración de la plataforma Stewart posteriormente se realizan algunas trayectorias para que sean reproducidas por el robot.

La matriz de transformación homogénea es una matriz 4x4 que transforma un vector de posición expresado en coordenadas homogéneas desde un sistema hasta otro sistema de coordenadas. En la ecuación (2.1) se presenta una matriz de transformación homogénea, se puede considerar que consiste en cuatro submatrices:

$$T = \begin{bmatrix} R_{3x3} & P_{3x3} \\ - & - \\ f_{3x3} & 1x1 \end{bmatrix} = \begin{bmatrix} matriz\_de\_rotacion & vector\_de\_posición\_transformación \\ - & - \\ transformación\_de\_perspectiva & escalado \end{bmatrix} \quad (2.1)$$

La submatriz de 3x3 superior izquierda representa la matriz de rotación; la submatriz superior derecha 3x1 representa el vector de posición del origen del sistema de condenadas rotado con respecto al sistema de referencia; la submatriz superior derecha de 1x1 representa el escalado. Las matrices de translación y rotación básicas en los ejes “X”, “Y”, “Z” respectivamente son:

$$Tz1 = \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.2)$$

$$Tz2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.3)$$

$$Tz3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & z \end{bmatrix} \qquad (2.4)$$

$$Tz4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\theta x & -S\theta x & 0 \\ 0 & S\theta y & C\theta x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.5)$$

$$Tz5 = \begin{bmatrix} C\theta y & 0 & S\theta y & 0 \\ 0 & 1 & 0 & 0 \\ S\theta y & 0 & C\theta y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.6)$$

$$Tz6 = \begin{bmatrix} C\theta z & -S\theta z & 0 & 0 \\ S\theta z & C\theta z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.7)$$

2.1.1 Marcos de referencia.

En la figura 2.1 se muestra un esquema de la plataforma Stewart configuración 3-3 cubica con la enumeración de sus actuadores. El análisis cinemático es realizado usando matrices homogéneas y resuelto de forma iterativa.

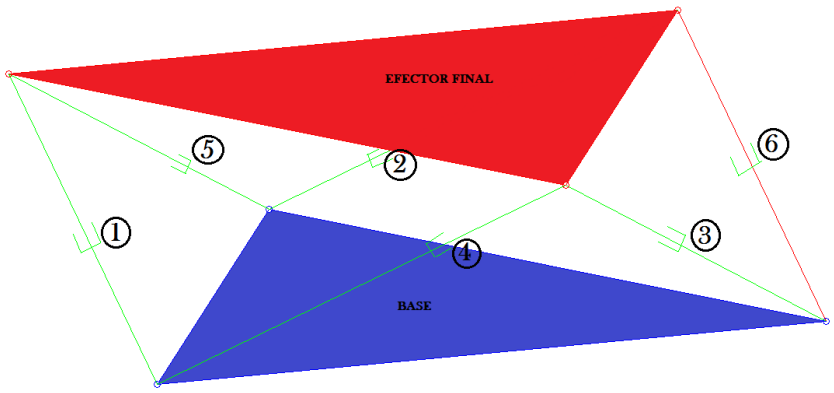


Figura 2.1 Plataforma Stewart, ubicación de actuadores.

2.2 Análisis de la posición.

Se mostrara el análisis de una sola cadena cinemática considerando que para el resto el análisis será esencialmente el mismo, i tomara valores de 1-6 dependiendo de la cadena cinemática a analizar.

En la figura 2.2 se muestra a detalle las bases, que van del sistema i0, j0, k0, al sistema 2. De acuerdo al sistema i0, j0, k0, se orienta el eje x en dirección a uno de los vértices del triángulo, generando el sistema 1, posteriormente dicho sistema se traslada generando el sistema 2. La secuencia de transformaciones es:

T02i=Tz6(**δ1i**).Tz1(**x21i**)

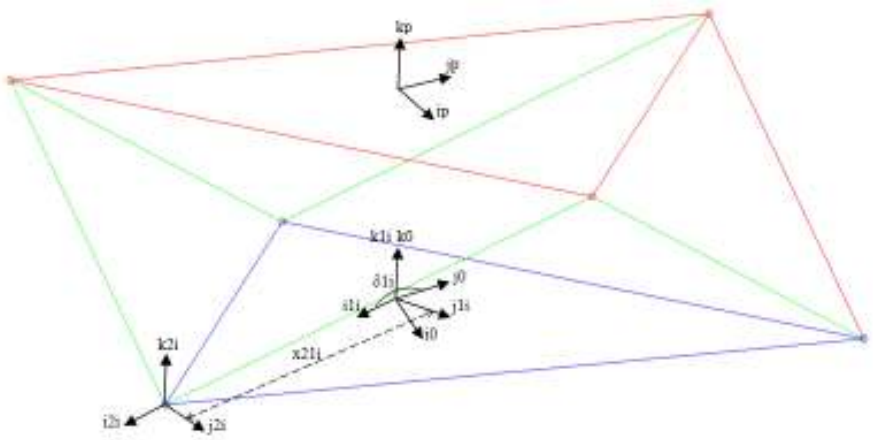


Figura 2.2 Marco de referencia de 0 a 2.

Para alinear el eje z con la dirección del actuador como se muestra en la figura 2.3 se realizan tres rotaciones dando lugar a los sistemas 3, 4, 5, una vez alineado se genera el sistema 6 con una rotación que representa un ángulo variable de la junta esférica y el sistema 7 se genera con el desplazamiento z76i que corresponde con la distancia constante de la base a la junta esférica. Las transformaciones quedan como sigue:

T27i=Tz6(**δ32i**).Tz4(**δ43i**).Tz6(**δ54i**).Tz6(**Θ65i**).Tz3(**z76i**)

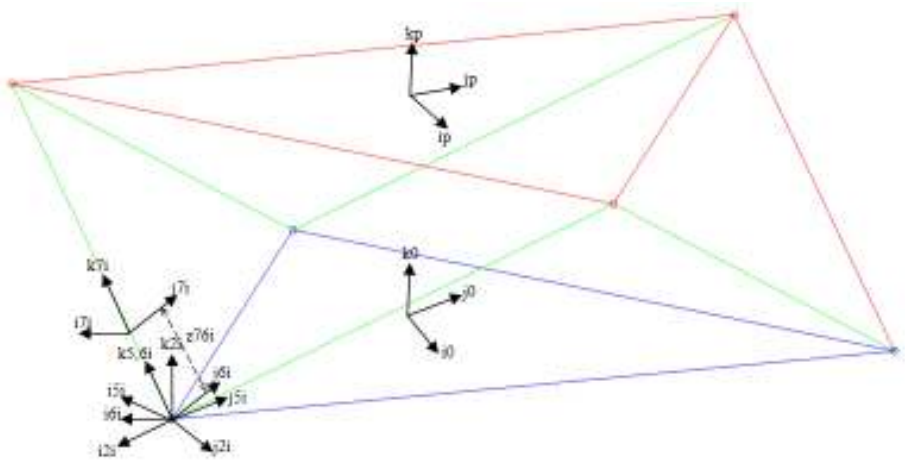


Figura 2.3 Sistema de referencia de 2 a 7.

En la figura 2.4 se muestra el sistema 8,9 generado por 2 rotaciones que representan los ángulos variables de la junta esférica, posteriormente se genera el sistema 10 con la traslación z109i que representa la longitud del actuador. De modo que la secuencia de transformaciones es:

$$T710i = Tz5(\theta\mathbf{87i}).Tz4(\theta\mathbf{98i}).Tz3(\mathbf{z109i})$$

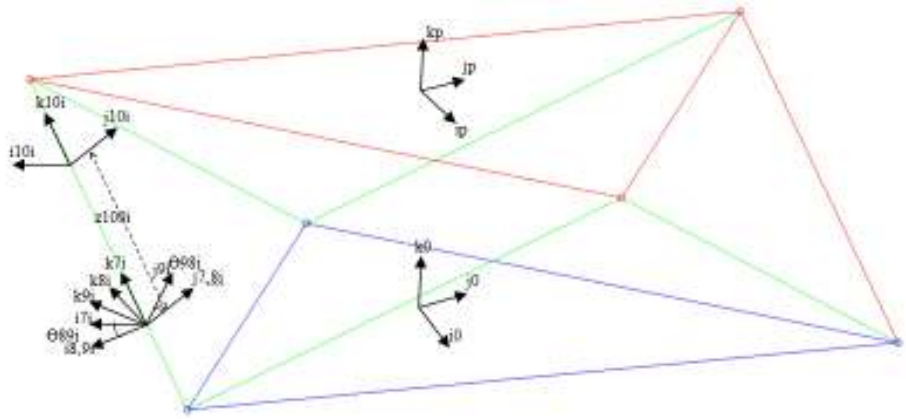


Figura 2.4 Sistema de referencia de 7 a 10.

En la figura 2.5 se muestra los sistemas 11 y 12 generados con las 2 rotaciones de la junta universal posteriormente se genera el sistema 13 con la distancia que separa la junta con la plataforma superior. La secuencia de transformaciones es:

$$T1013i = Tz4(\theta\mathbf{1110i}).Tz5(\theta\mathbf{1211i}).Tz3(\mathbf{z109i})$$

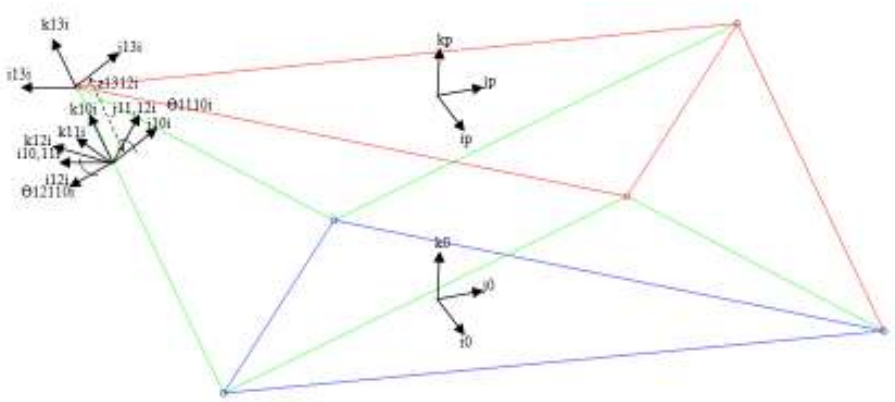


Figura 2.5 Sistema de referencia de 10 a 13.

La base 14 mostrada en la figura 2.6 se forma alineando la base ip, jp, kp, con el vértice del triángulo superior la base 15 se genera con una traslación constante hacia el vértice del triángulo, se generan 2 giros para llegar a la base 18 donde la base ya se encuentra alineada con el actuador, finalmente para cerrar la cadena cinemática se genera el sistema que va de ip, jp, kp, a partir del sistema i0, j0, k0, dando tres translaciones y tres rotaciones.

$$TP18i = Tz6(\mathbf{\delta 14Pi}).Tz1(\mathbf{x1514i}).Tz6(\mathbf{\delta 1615i}).Tz4(\mathbf{\delta 1716i}).Tz6(\mathbf{\delta 1817i})$$

$$T0Pi = Tz1(\mathbf{xp}).Tz2(\mathbf{yp}).Tz3(\mathbf{zp}).Tz5(\boldsymbol{\psi}).Tz4(\theta).Tz6(\phi)$$

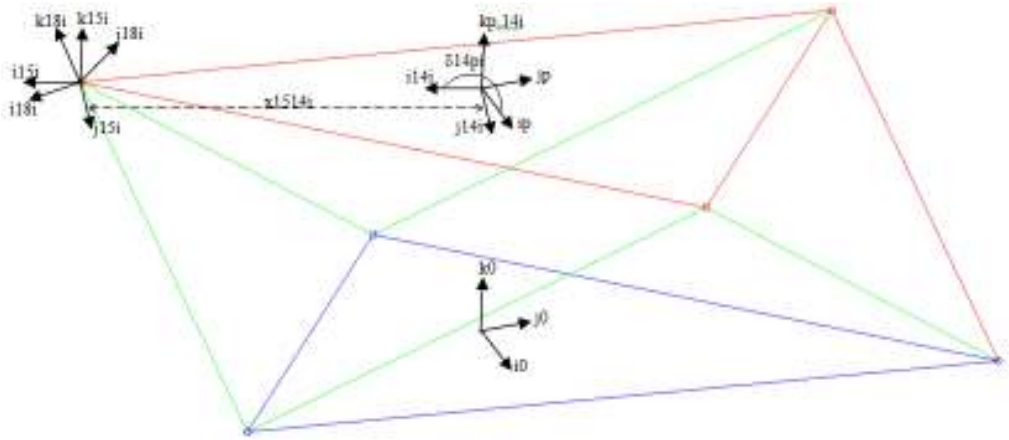


Figura 2.6 Sistema de referencia de P a 18.

Agrupando las matrices:

$$T02i = Tz6(\mathbf{\delta 1i}).Tz1(\mathbf{x21i})$$

$$T27i = Tz6(\mathbf{\delta 32i}).Tz4(\mathbf{\delta 43i}).Tz6(\mathbf{\delta 54i}).Tz6(\theta\mathbf{65i}).Tz3(\mathbf{z76i})$$

$$T710i = Tz5(\theta \mathbf{87i}).Tz4(\theta \mathbf{98i}).Tz3(\mathbf{z109i})$$

$$T1013i = Tz4(\theta \mathbf{1110i}).Tz5(\theta \mathbf{1211i}).Tz3(\mathbf{z109i})$$

$$TP18i = Tz6(\boldsymbol{\delta} \mathbf{14Pi}).Tz1(\mathbf{x1514i}).Tz6(\boldsymbol{\delta} \mathbf{1615i}).Tz4(\boldsymbol{\delta} \mathbf{1716i}).Tz6(\boldsymbol{\delta} \mathbf{1817i})$$

$$T0P = Tz1(\mathbf{xp}).Tz2(\mathbf{yp}).Tz3(\mathbf{zp}).Tz5(\boldsymbol{\psi}).Tz4(\theta).Tz6(\phi)$$

La ecuación de posición matricial es:

T02i.T27i.T710i.T1013i=T0P.TP18i (2.8)

De esta forma nuestro problema se reduce a lo siguiente:

Análisis cinemática inverso

Dada la posición y orientación de la plataforma móvil, ip, jp, kp, ψ, θ, ϕ, hallar los valores $\theta 65i, \theta 87i, \theta 98i, \theta 1110i, \theta 1211i$ y z109i que definen el movimiento de las juntas que conforman la plataforma.

Las posiciones angulares correspondientes a cada cadena con respecto a la plataforma deprenden de la orientación y de la trayectoria de la misma, los valores para el cálculo numérico se muestran en la siguiente tabla:

	Ch1	Ch2	Ch3	Ch4	Ch5	Ch6
δ1i	300º	180º	60º	-60º	180º	60º
x21i(cm)	15.11	15.11	15.11	15.11	15.11	15.11
δ32i	-30º	-30º	-30º	210º	210º	210º
δ43i	55º	55º	55º	55º	55º	55º
δ54i	45º	45º	45º	45º	45º	45º
z76i(cm)	4	4	4	4	4	4
z1312i(cm)	4.5	4.5	4.5	4.5	4.5	4.5
δ14Pi	240º	120º	0º	0º	240º	120º
x1514i(cm)	15.11	15.11	15.11	15.11	15.11	15.11
δ1615i	30º	30º	30º	30º	30º	30º
δ1716i	55º	55º	55º	55º	55º	55º
δ1817i	45º	45º	45º	45º	45º	45º

Tabla 2.1 Parámetros de cada cadena cinemática.

2.3 Análisis de velocidad.

El análisis de velocidad parte del supuesto de que se conoce por completo la posición y orientación de cada componente del sistema por qué son el resultado del análisis de posición, es importante por qué en el caso de continuar con el estudio hasta la dinámica nos permitirá, por ejemplo, calcular la

potencia requerida en el sistema para producir y controlar el movimiento propuesto, en esta tesis no se trata el análisis dinámico.

Análisis cinemática inverso

Dada la velocidad lineal del centroide de la plataforma móvil $\dot{\boldsymbol{xp}}, \dot{\boldsymbol{yp}}, \dot{\boldsymbol{zp}}$ y su velocidad angular $\dot{\boldsymbol{\theta}}, \dot{\boldsymbol{\varphi}}, \dot{\boldsymbol{\psi}}$ hallar la velocidad angular de cada una de las juntas, a saber $\dot{\boldsymbol{\theta}}\mathbf{65_1}, \dot{\boldsymbol{\theta}}\mathbf{87_1}, \dot{\boldsymbol{\theta}}\mathbf{98_1}, \dot{\boldsymbol{\theta}}\mathbf{110_1}, \dot{\boldsymbol{\theta}}\mathbf{211_1}, \dot{\mathbf{z}}\mathbf{109_1}$ en cada una de las 6 cadenas.

De forma general podemos expresar que la derivada de la matriz de transformación homogénea es,

$\dot{T}zj(\boldsymbol{q}) = Tzj(\boldsymbol{q}).Dzj(\dot{\boldsymbol{q}})$ para j=1-6 (2.9)

donde $Dzj(\dot{\boldsymbol{q}})$ es el operador diferencial de la matriz homogenea

Derivando con respecto al tiempo la ecuación (2.8)

$$\dot{T}02i.T27i.T710i.T1013i + T02i.\dot{T}27i.T710i.T1013i + T02i.T27i.\dot{T}710i.T1013i + T02i.T27i.T710i.\dot{T}1013i = \dot{T}0P.TP18i + T0P.\dot{T}P18i \quad (2.10)$$

Donde

$$\dot{T}02i = \dot{T}P18i = 0$$

Las transformaciones anteriores tienen distancias y ángulos constantes. Reescribiendo la ecuación. (2.10):

$$T02i.\dot{T}27i.T710i.T1013i + T02i.T27i.\dot{T}710i.T1013i + T02i.T27i.T710i.\dot{T}1013i = \dot{T}0P.TP18i$$

Donde utilizando la ecuación (2.9)

$$\begin{aligned}\dot{T}27i = {} & \dot{T}z6(\boldsymbol{\delta 32i}).Tz4(\boldsymbol{\delta 43i}).Tz6(\boldsymbol{\delta 54i}).Tz6(\boldsymbol{\theta 65i}).Tz3(\boldsymbol{z 76i}) \\ & + Tz6(\boldsymbol{\delta 32i}).\dot{T}z4(\boldsymbol{\delta 43i}).Tz6(\boldsymbol{\delta 54i}).Tz6(\boldsymbol{\theta 65i}).Tz3(\boldsymbol{z 76i}) \\ & + Tz6(\boldsymbol{\delta 32i}).Tz4(\boldsymbol{\delta 43i}).\dot{T}z6(\boldsymbol{\delta 54i}).Tz6(\boldsymbol{\theta 65i}).Tz3(\boldsymbol{z 76i}) \\ & + Tz6(\boldsymbol{\delta 32i}).Tz4(\boldsymbol{\delta 43i}).Tz6(\boldsymbol{\delta 54i}).\dot{T}z6(\boldsymbol{\theta 65i}).Tz3(\boldsymbol{z 76i}) \\ & + Tz6(\boldsymbol{\delta 32i}).Tz4(\boldsymbol{\delta 43i}).Tz6(\boldsymbol{\delta 54i}).Tz6(\boldsymbol{\theta 65i}).\dot{T}z3(\boldsymbol{z 76i})\end{aligned}$$

Eliminando los términos que se hacen cero a causa de las constantes y reescribiendo la ecuación:

$\dot{T}27i = Tz6(\boldsymbol{\delta 32i}).Tz4(\boldsymbol{\delta 43i}).Tz6(\boldsymbol{\delta 54i}).Tz6(\boldsymbol{\theta 65i}).Dz6(\dot{\boldsymbol{\theta}}\boldsymbol{65i}).Tz3(\boldsymbol{z 76i})$

$$\begin{aligned}\dot{T}710i = {} & \dot{T}z5(\boldsymbol{\theta 87i}).Tz4(\boldsymbol{\theta 98i}).Tz3(\boldsymbol{z 109i}) + Tz5(\boldsymbol{\theta 87i}).\dot{T}z4(\boldsymbol{\theta 98i}).Tz3(\boldsymbol{z 109i}) \\ & + Tz5(\boldsymbol{\theta 87i}).Tz4(\boldsymbol{\theta 98i}).\dot{T}z3(\boldsymbol{z 109i})\end{aligned}$$

$$\dot{T}710i = Tz5(\boldsymbol{\theta 87i}).Dz5\left(\dot{\boldsymbol{\theta}}\boldsymbol{87i}\right)Tz4(\boldsymbol{\theta 98i}).Tz3(\boldsymbol{z109i})$$
$$+ Tz5(\boldsymbol{\theta 87i}).\dot{T}z4(\boldsymbol{\theta 98i}).Dz4\left(\dot{\boldsymbol{\theta}}\boldsymbol{98i}\right).Tz3(\boldsymbol{z109i})$$
$$+ Tz5(\boldsymbol{\theta 87i}).Tz4(\boldsymbol{\theta 98i}).\dot{T}z3(\boldsymbol{z109i}).Dz3(\dot{\boldsymbol{z}}\boldsymbol{109i})$$

$$\dot{T}1013i = \dot{T}z4(\boldsymbol{\theta 1110i}).Tz5(\boldsymbol{\theta 1211i}).Tz3(\boldsymbol{z1312i})$$
$$+ Tz4(\boldsymbol{\theta 1110i}).\dot{T}z5(\boldsymbol{\theta 1211i}).Tz3(\boldsymbol{z1312i})$$
$$+ Tz4(\boldsymbol{\theta 1110i}).Tz5(\boldsymbol{\theta 1211i}).\dot{T}z3(\boldsymbol{z1312i})$$

$$\dot{T}1013i = Tz4(\boldsymbol{\theta 1110i}).Dz4\left(\dot{\boldsymbol{\theta}}\boldsymbol{1110i}\right).Tz5(\boldsymbol{\theta 1211i}).Tz3(\boldsymbol{z1312i})$$
$$+ Tz4(\boldsymbol{\theta 1110i}).Tz5(\boldsymbol{\theta 1211i}).Dz5\left(\dot{\boldsymbol{\theta}}\boldsymbol{1211i}\right).Tz3(\boldsymbol{z1312i})$$

$$\dot{T}0P = \dot{T}z1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\theta).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).\dot{T}z2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\theta).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).\dot{T}z3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\theta).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).\dot{T}z5(\boldsymbol{\psi}).Tz4(\theta).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).\dot{T}z4(\theta).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\theta).\dot{T}z6(\phi)$$

$$T0P = \dot{T}z1(\boldsymbol{xp}).Dz1(\dot{\boldsymbol{x}}\boldsymbol{p}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\theta).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Dz2(\boldsymbol{yp}).Dz2(\dot{\boldsymbol{y}}\boldsymbol{p}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\theta).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Dz3(\dot{\boldsymbol{z}}\boldsymbol{p}).Tz5(\boldsymbol{\psi}).Tz4(\theta).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Dz5\left(\dot{\boldsymbol{\psi}}\right).Tz4(\theta).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\theta).Dz4\left(\dot{\theta}\right).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\theta).Tz6(\phi).Dz6(\dot{\phi})$$

2.4 Análisis de la aceleración.

Dada la aceleración lineal del centroide de la plataforma móvil $\ddot{\boldsymbol{x}}\boldsymbol{p}, \ddot{\boldsymbol{y}}\boldsymbol{p}, \ddot{\boldsymbol{z}}\boldsymbol{p}$ y su aceleración angular $\ddot{\boldsymbol{\theta}}, \ddot{\boldsymbol{\varphi}}, \ddot{\boldsymbol{\psi}}$ hallar la velocidad angular de cada una de las juntas, a saber $\boldsymbol{\theta\ddot{6}5_1, \theta\ddot{8}7_1, \theta\ddot{9}8_1, \theta 1\ddot{1}10_1, \theta 1\ddot{2}11_1, i}$ en cada una de las 6 cadenas.

La matriz de aceleración queda generalizando

$$\ddot{T}zj(\boldsymbol{q}) = Tzj(\boldsymbol{q})(Dzj(\ddot{\boldsymbol{q}}) + D^2zj(\dot{\boldsymbol{q}})) \quad (2.11)$$

Derivando con respecto al tiempo la ecuación (2.10)

$$T02i.\ddot{T}27i.T710i.T1013i + T02i.T27i.\ddot{T}710i.T1013i + T02i.T27i.T710i.\ddot{T}1013i$$
$$+ 3\left(T02i.\dot{T}27i.\dot{T}710i.\dot{T}1013i\right) = \ddot{T}0P.\dot{T}P18i + \dot{T}0P.\dot{T}P18i$$

Donde utilizando la ecuación (2.11)

$$\ddot{T}27i = Tz6(\boldsymbol{\delta 32i}).Tz4(\boldsymbol{\delta 43i}).Tz6(\boldsymbol{\delta 54i}).\ddot{T}z6(\boldsymbol{\theta 65i}).Tz3(\boldsymbol{z76i})$$
$$+ Tz6(\boldsymbol{\delta 32i}).Tz4(\boldsymbol{\delta 43i}).Tz6(\boldsymbol{\delta 54i}).\dot{T}z6(\boldsymbol{\theta 65i}).Tz3(\boldsymbol{z76i})$$

$$\ddot{T}27i = Tz6(\boldsymbol{\delta 32 i}).Tz4(\boldsymbol{\delta 43 i}).Tz4(\boldsymbol{\delta 54 i}).Tz6(\boldsymbol{\theta 65 i})\left(Dz6\left(\ddot{\boldsymbol{\theta}}\boldsymbol{65 i}\right)\right.$$
$$\left. + D^2z6\left(\dot{\boldsymbol{\theta}}\boldsymbol{65 i}\right)\right).Tz3(\boldsymbol{z 76 i})$$
$$+ Tz6(\boldsymbol{\delta 32}).Tz4(\boldsymbol{\delta 43 i}).Tz6(\boldsymbol{\delta 54 i}).\dot{T}z6(\boldsymbol{\theta 65 i}).Tz3(\boldsymbol{z 76 i})$$

$$\ddot{T}710i = \ddot{T}z5(\boldsymbol{\theta 87 i}).Tz4(\boldsymbol{\theta 98 i}).Tz3(\boldsymbol{z 109 i}) + Tz5(\boldsymbol{\theta 87 i}).\ddot{T}z4(\boldsymbol{\theta 98 i}).Tz3(\boldsymbol{z 109 i})$$
$$+ Tz5(\boldsymbol{\theta 87 i}).Tz4(\boldsymbol{\theta 98 i}).\ddot{T}(\boldsymbol{z 109 i}) + 3(\dot{T}z5(\boldsymbol{\theta 87 i}).\dot{T}z4(\boldsymbol{\theta 98 i}).\dot{T}z3(\boldsymbol{z 109 i}))$$

$$\ddot{T}710i = Tz5(\boldsymbol{\theta 87 i})\left(Dz5\left(\ddot{\boldsymbol{\theta}}\boldsymbol{87 i}\right) + D^2z5\left(\dot{\boldsymbol{\theta}}\boldsymbol{87 i}\right)\right).Tz4(\boldsymbol{\theta 98 i}).Tz3(\boldsymbol{z 109 i})$$
$$+ Tz5(\boldsymbol{\theta 87 i}).Tz4(\boldsymbol{\theta 98 i})\left(Dz4\left(\ddot{\boldsymbol{\theta}}\boldsymbol{98 i}\right) + D^2z4\left(\dot{\boldsymbol{\theta}}\boldsymbol{98 i}\right)\right).Tz3(\boldsymbol{z 109 i})$$
$$+ Tz5(\boldsymbol{\theta 87 i}).Tz4(\boldsymbol{\theta 98 i}).Tz3(\boldsymbol{z 109 i})\left(Dz3(\ddot{\boldsymbol{z}}\boldsymbol{109 i}) + Dz3(\dot{\boldsymbol{z}}\boldsymbol{109 i})\right)$$
$$+ 3(\dot{T}z5(\boldsymbol{\theta 87 i}).\dot{T}z4(\boldsymbol{\theta 98 i}).\dot{T}z3(\boldsymbol{z 109 i}))$$

$$\ddot{T}1013i = \ddot{T}z4(\boldsymbol{\theta 1110 i}).Tz5(\boldsymbol{\theta 1211 i}).Tz3(\boldsymbol{z 1312 i})$$
$$+ Tz4(\boldsymbol{\theta 1110 i}).\ddot{T}z5(\boldsymbol{\theta 1211 i}).Tz3(\boldsymbol{z 1312 i})$$
$$+ 2(\dot{T}z4(\boldsymbol{\theta 1110 i}).\dot{T}z5(\boldsymbol{\theta 1211 i}).Tz3(\boldsymbol{z 1312 i}))$$

$$\ddot{T}1013i = Tz4(\boldsymbol{\theta 1110 i})\left(Dz4\left(\ddot{\boldsymbol{\theta}}\boldsymbol{1110 i}\right) + D^2z4\left(\dot{\boldsymbol{\theta}}\boldsymbol{1110 i}\right)\right).Tz5(\boldsymbol{\theta 1211 i}).Tz3(\boldsymbol{z 1312 i})$$
$$+ Tz4(\boldsymbol{\theta 1110 i}).Tz5(\boldsymbol{\theta 1211 i})\left(Dz5\left(\ddot{\boldsymbol{\theta}}\boldsymbol{1211 i}\right)\right.$$
$$\left. + D^2z5(\boldsymbol{\theta 1211 i})\right).Tz3(\boldsymbol{z 1312 i})$$
$$+ 2\left(\dot{T}z4(\boldsymbol{\theta 1110 i}).\dot{T}z5(\boldsymbol{\theta 1211 i})\right).\dot{T}z3(\boldsymbol{z 1312 i})$$

$$\ddot{T}0P = \ddot{T}z1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\boldsymbol{\theta}).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).\ddot{T}z2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\boldsymbol{\theta}).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).\ddot{T}z3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\boldsymbol{\theta}).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).\ddot{T}z5(\boldsymbol{\psi}).Tz4(\boldsymbol{\theta}).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).\ddot{T}z4(\boldsymbol{\theta}).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\boldsymbol{\theta}).\ddot{T}z6(\phi)$$
$$+ 6(\dot{T}z1(\boldsymbol{xp}).\dot{T}z2(\boldsymbol{yp}).\dot{T}z3(\boldsymbol{zp}).\dot{T}z5(\boldsymbol{\psi}).\dot{T}z4(\boldsymbol{\theta}).\dot{T}z6(\phi))$$

$$\ddot{T}0P = Tz1(\boldsymbol{xp}).\left(Dz1(\ddot{\boldsymbol{x}}\boldsymbol{p}) + D^2z1(\dot{\boldsymbol{x}}\boldsymbol{p})\right).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\boldsymbol{\theta}).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).\left(Dz2(\ddot{\boldsymbol{y}}\boldsymbol{p})\right.$$
$$\left. + D^2z2(\dot{\boldsymbol{y}}\boldsymbol{p})\right).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\boldsymbol{\theta}).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).\left(Dz3(\ddot{\boldsymbol{z}}\boldsymbol{p}) + D^2z3(\dot{\boldsymbol{z}}\boldsymbol{p})\right).Tz5(\boldsymbol{\psi}).Tz4(\boldsymbol{\theta}).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).\left(Dz5\left(\ddot{\boldsymbol{\psi}}\right) + D^2z5\left(\dot{\boldsymbol{\psi}}\right)\right).Tz4(\boldsymbol{\theta}).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\boldsymbol{\theta}).\left(Dz5\left(\ddot{\boldsymbol{\theta}}\right) + D^2z5\left(\dot{\boldsymbol{\theta}}\right)\right).Tz6(\phi)$$
$$+ Tz1(\boldsymbol{xp}).Tz2(\boldsymbol{yp}).Tz3(\boldsymbol{zp}).Tz5(\boldsymbol{\psi}).Tz4(\boldsymbol{\theta}).Tz6(\phi).\left(Dz6\left(\ddot{\phi}\right) + D^2z6\left(\dot{\phi}\right)\right)$$
$$+ 6(\dot{T}z1(\boldsymbol{xp}).\dot{T}z2(\boldsymbol{yp}).\dot{T}z3(\boldsymbol{zp}).\dot{T}z5(\boldsymbol{\psi}).\dot{T}z4(\boldsymbol{\theta}).\dot{T}z6(\phi))$$

2.5 Trayectoria propuesta.

El movimiento de un cuerpo en el espacio consiste de dos partes: Una trayectoria lineal o curva en el espacio que sigue un punto del cuerpo (en el centro de gravedad o en el efector final del manipulador) y la orientación angular del cuerpo. Ambas partes deben satisfacer condiciones de posición, velocidad y aceleración tanto lineal como angular y ser realizadas en un tiempo definido previamente. A continuación se desarrolla la trayectoria lineal y la trayectoria angular que se propone para la plataforma en función del tiempo.

Trayectoria Lineal

Se define la curva en el espacio como una recta para el movimiento a seguir por un punto del cuerpo.

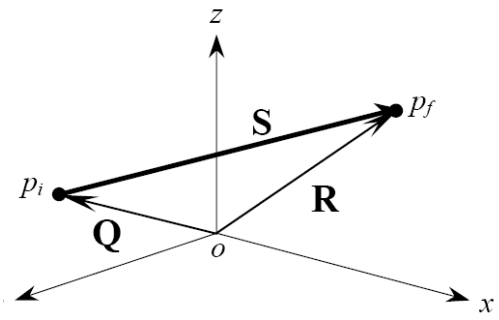


Figura 2.7 Recta en el espacio.

La ecuación vectorial de posición se define a partir de la figura 2.7:

R=Q+S

=Q+su (2.12)

donde s es la magnitud del vector S y u es el vector unitario que define la orientación de S. Para definir R en función del tiempo, se requiere que la magnitud s cambie con respecto al mismo, es decir:

R(t)=Q+s(t)u (2.13)

las ecuaciones vectoriales de velocidad y aceleración se definen como la primera y segunda derivada respecto al tiempo de la ecuación 2.13:

$$V(t) = \dot{s}(t)u$$

$$A(t) = \ddot{s}(t)u \qquad (2.14)$$

ya que Q y u no varían respecto el tiempo, porque están definidos por puntos fijos en el espacio. La magnitud s(t) debe satisfacer condiciones iniciales y finales de posición, velocidad y aceleración, es decir debe satisfacer 6 condiciones, según se muestra en la figura 2.8.

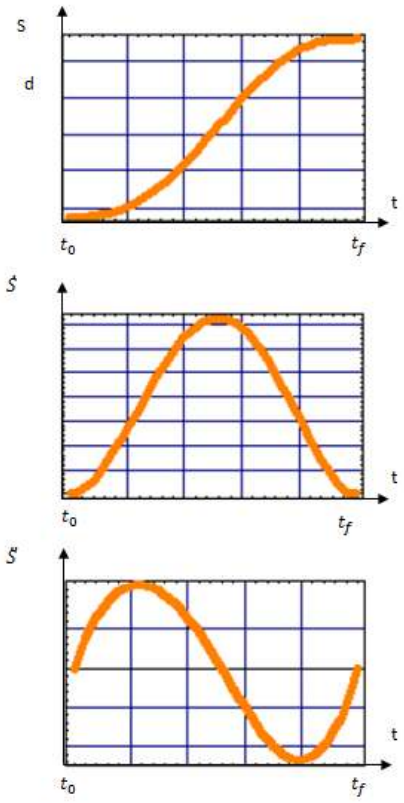


Figura 2.8 Condiciones iniciales.

La primera gráfica indica el cambio de magnitud del vector S, que irá variando de O en un tiempo inicial t_0, a $d = \|S\|$para un tiempo final t_f, los valores de tiempo, t_0 y t_f los definimos de manera arbitraria.

La segunda gráfica es la rapidez con la que la magnitud del vector S cambia respecto al tiempo. Es decir, es la rapidez con que se realiza el traslado del punto p_i a p_f, para un tiempo inicial t_0 y para un tiempo final t_f.

La tercera gráfica es el cambio de la rapidez (aceleración) con que la magnitud del vector S cambia con respecto al tiempo, para un tiempo inicial t_0 y para un tiempo final t_f.

Para satisfacer las 6 condiciones, se empleara un polinomio de quinto grado, ya que este cuenta con 6 coeficientes a determinar. De esta manera, se tiene:

$$s(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5$$

$$\dot{s}(t) = a_1 + 2a_2 t + 3a_3 t^2 + 4a_4 t^3 + 5a_5 t^4 \qquad (2.15)$$

$$\ddot{s}(t) = 2a_2 + 6a_3 t + 12a_4 t^2 + 20a_5 t^3$$

Debido a que existen condiciones iniciales y finales de velocidad y aceleración, se obtienen las derivadas respecto al tiempo del polinomio s(t). A partir de la figura 2.10, para $t = t_0 = 0$ se tienen 3 condiciones iniciales:

$$s(t_0) = s(0) = 0$$
$$\dot{s}(t_0) = \dot{s}(0) = 0 \qquad (2.16)$$
$$\ddot{s}(t_0) = \ddot{s}(0) = 0$$

al sustituir en las ecu.(2.15) se obtiene:

$$s(0) = 0 = a_0 + a_1(0) + a_2(0)^2 + a_3(0)^3 + a_4(0)^4 + a_5(0)^5$$
$$\dot{s}(0) = 0 = a_1 + 2a_2(0) + 3a_3(0)^2 + 4a_4(0)^3 + 5a_5(0)^4$$
$$\ddot{s}(0) = 0 = 2a_2 + 6a_3(0) + 12a_4(0)^2 + 20a_5(0)^3$$

Simplificando:

$$0 = a_0$$
$$0 = a_1$$
$$0 = 2a_2$$

Finalmente, los 3 primeros coeficientes son:

$$a_0 = 0$$
$$a_1 = 0 \qquad (2.17)$$
$$a_2 = 0$$

A partir de la figura 2.10 y repitiendo el proceso para t= t_f se tienen las 3 condiciones finales:

$$s(t_f) = d = \|p_f - p_i\|$$
$$\dot{s}(t_f) = 0 \qquad (2.18)$$
$$\ddot{s}(t_f) = 0$$

donde $p_i = (x_i, y_i, z_i)$, son las coordenadas de los puntos inicial y final de la treyectoria.

La magnitud de la diferencia entre ellos, representa la distancias d que necesitamos recorrer en la línea recta. Al sustituir las ecs. (2.17) y (2.18) en la ec. (2.15) se obtiene:

$$\|p_f - p_i\| = a_3 t_f^3 + a_4 t_f^4 + a_5 t_f^5$$
$$0 = 3a_3 t_f^2 + 4a_4 t_f^3 + 5a_5 t_f^4$$

$$0 = 6a_3 t_f + 12a_4 t_f^2 + 20a_5 t_f^3$$

el sistema de ecuaciones se expresa de la siguiente manera:

$$\begin{pmatrix} t_f^3 & t_f^4 & t_f^5 \\ 3t_f^2 & 4t_f^3 & 5t_f^4 \\ 6t & 12t_f^2 & 20t_f^3 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_2 \end{pmatrix} = \begin{pmatrix} |p_f - p_i| \\ 0 \\ 0 \end{pmatrix} \quad (2.19)$$

al resolver el sistema de la ecuación (2.19)se obtienen los 3 últimos coeficientes:

$$a_3 = 10\frac{\|p_f - p_i\|}{t_f^3}$$

$$a_4 = -15\frac{\|p_f - p_i\|}{t_f^4} \quad (2.20)$$

$$a = \frac{\|p_f - p_i\|}{t_f^5}$$

Sustituyendo las ecs.(2.17) y (2.20) en la ec. (2.15):

$$s(t_f) = 10\frac{\|p_f - p_i\|}{t_f^3}t^3 - 15\frac{\|p_f - p_i\|}{t_f^4}t^4 + 6\frac{\|p_f - p_i\|}{t_f^5}t^5$$

$$\dot{s}(t_f) = 30\frac{\|p_f - p_i\|}{t_f^3}t^2 - 30\frac{\|p_f - p_i\|}{t_f^4}t^3 + 30\frac{\|p_f - p_i\|}{t_f^5}t^4 \quad (2.21)$$

$$\ddot{s}(t_f) = 60\frac{\|p_f - p_i\|}{t_f^3}t - 180\frac{\|p_f - p_i\|}{t_f^4}t^2 + 120\frac{\|p_f - p_i\|}{t_f^5}t^3$$

Finalmente se obtienen las ecuaciones que representan el cambio de la magnitud de la posición, velocidad y aceleración en función del tiempo:

$$s(t_f) = \|p_f - p_i\|\left[10\frac{\|p_f - p_i\|}{t_f^3}t^3 - 15\frac{\|p_f - p_i\|}{t_f^4}t^4 + 6\frac{\|p_f - p_i\|}{t_f^5}t^5\right]$$

$$\dot{s}(t_f) = \|p_f - p_i\|\left[30\frac{\|p_f - p_i\|}{t_f^3}t^2 - 30\frac{\|p_f - p_i\|}{t_f^4}t^3 + 30\frac{\|p_f - p_i\|}{t_f^5}t^4\right] \quad (2.22)$$

$$\ddot{s}(t_f) = \|p_f - p_i\|\left[60\frac{\|p_f - p_i\|}{t_f^3}t - 180\frac{\|p_f - p_i\|}{t_f^4}t^2 + 120\frac{\|p_f - p_i\|}{t_f^5}t^3\right]$$

donde t=tiempo para realizar un movimiento y t_f=tiempo final para terminar el movimiento. Reescribiendo las ecs. (2.13) y (2.14) en función de los puntos de recta:

$$R(t) = Q + s(t)u = (p_i - 0) + s(t)\frac{(p_f - p_i)}{\|p_f - p_i\|}$$

$$V(t) = s(t)u = s(t)\frac{(p_f - p_i)}{\|p_f - p_i\|} \qquad (2.23)$$

$$A(t) = s(t)u = \frac{(p_f - p_i)}{\|p_f - p_i\|}$$

Sustituyendo las ecs. (2.22) en las ecs. (2.23), se obtiene finalmente la ecuación de posición de la velocidad y de la aceleración que debe de seguir la plataforma móvil:

$$R(t) = p_i + \left[10\left(\frac{t}{t_f}\right)^3 - 15\left(\frac{t}{t_f}\right)^4 + 60\left(\frac{t}{t_f}\right)^5\right](p_f - p_i)$$

$$V(t) = \left[30\frac{t^2}{t_f^3} - 60\frac{t^3}{t_f^4} + 30\frac{t^4}{t_f^5}\right](p_f - p_i) \qquad (2.24)$$

$$A(t) = \left[60\frac{t}{t_f^3} - 180\frac{t^2}{t_f^4} + 120\frac{t^4}{t_f^5}\right](p_f - p_i)$$

Orientación angular

Para la orientación se sigue un procedimiento similar, en el entendido de que para este caso, solo se desea pasar de valores iniciales a finales, para la posición, velocidad y aceleración angular de la plataforma móvil, ya que no se requiere cumplir con una trayectoria particular en el espacio. Esto conducirá a las siguientes ecuaciones:

$$\beta(t) = \beta_i + \left[10\left(\frac{t}{t_f}\right)^3 - 15\left(\frac{t}{t_f}\right)^4 + 6\left(\frac{t}{t_f}\right)^5\right](\beta_f - \beta_i)$$

$$\dot{\beta}(t) = \left[30\frac{t^2}{t_f^3} - 60\frac{t^3}{t_f^4} + 30\frac{t^4}{t_f^5}\right](\beta_f - \beta_i) \qquad (2.25)$$

$$\ddot{\beta}(t) = \left[60\frac{t}{t_f^3} - 180\frac{t^2}{t_f^4} + 120\frac{t^4}{t_f^5}\right](\beta_f - \beta_i)$$

Donde el vector $\beta = (\psi, \Theta, \varphi)$. De la misma manera $\beta_i = (\psi_i, \Theta_i, \varphi_i)$ y $\beta_f = (\psi_f, \Theta_f, \varphi_f)$, que se refieren a los valores iniciales y finales.

2.6 Simulación de la trayectoria.

Para el sistema bajo estudio se propone que la trayectoria del centro de la plataforma móvil (x_p, y_p, z_p), tenga como coordenadas inicial y final, las siguientes:

$p_f = (2,2,12)cm$ y $p_i = (0,0,10)cm$

Siendo la orientación.

$\beta_f = (0,0,25)$ y $\beta_i = (0,0,0)$

Para el análisis tomaremos, el siguiente comportamiento de desplazamiento, velocidad y aceleración para en centro de la plataforma móvil en un tiempo de 5 segundos.

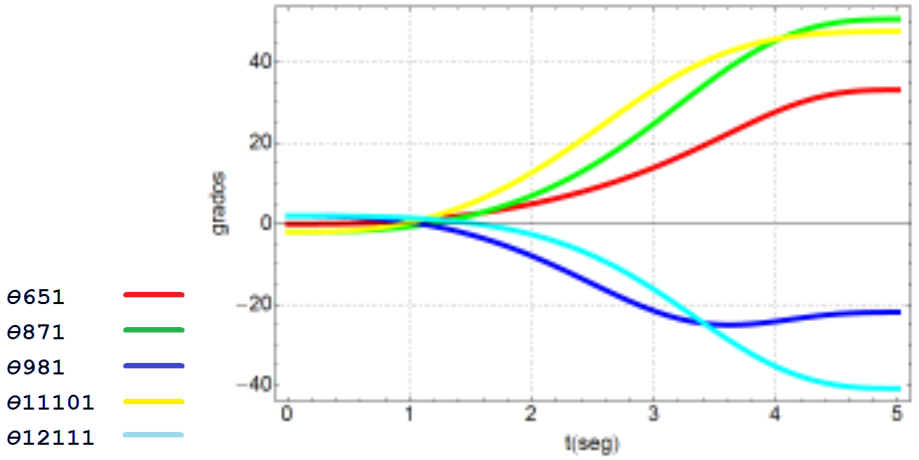


Figura 2.9 Ángulos de la cadena cinemática 1.

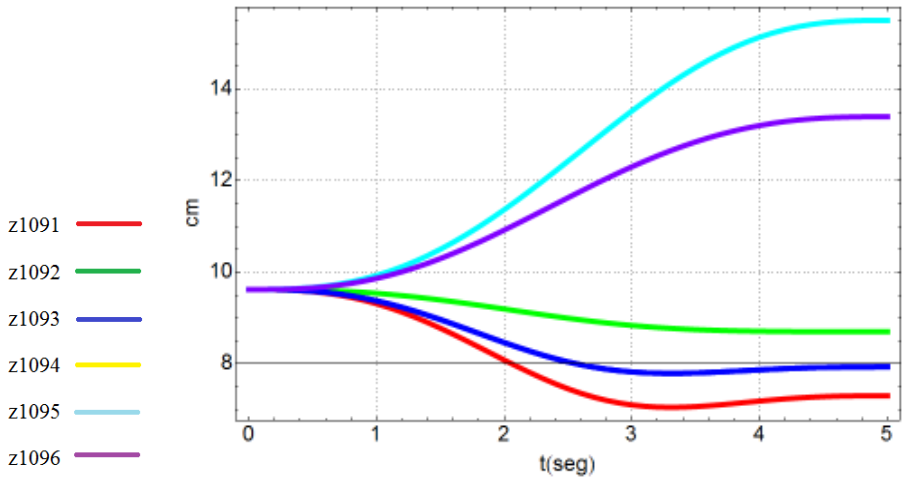


Figura 2.10 Posiciones de los actuadores.

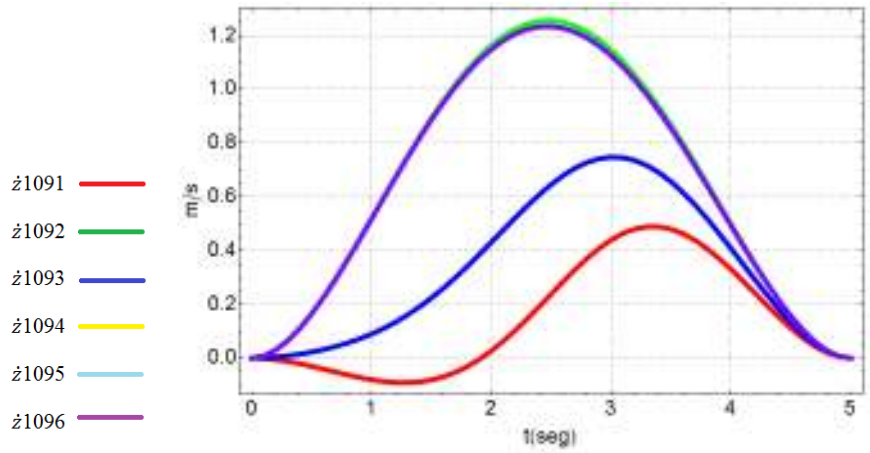


Figura 2.11 Velocidad de los actuadores.

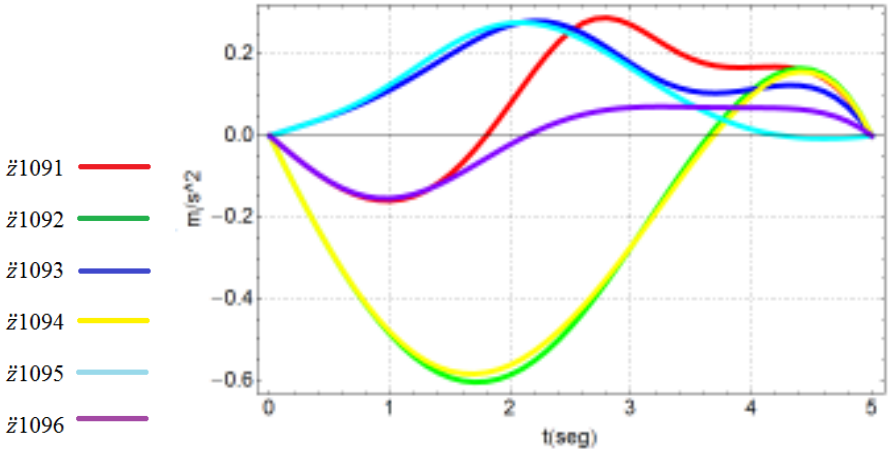
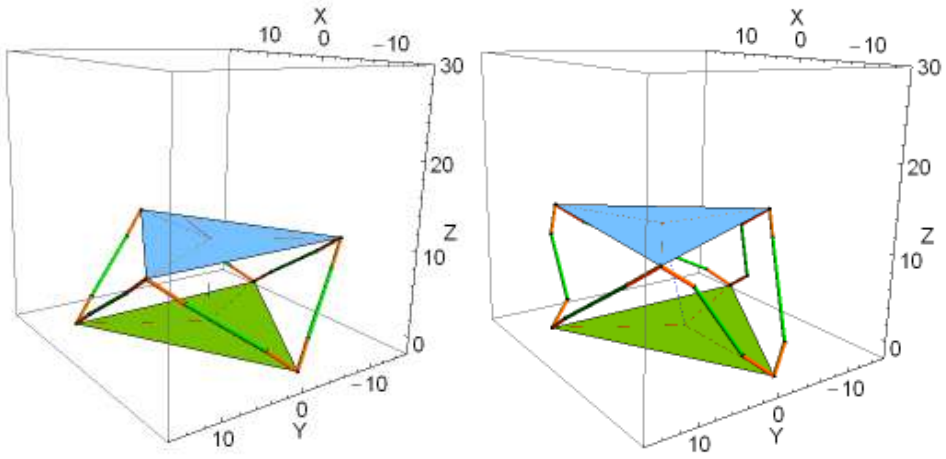


Figura 2.12 Aceleración de los actuadores.



a) b)

Figura 2.13 Posición inicial (a) y final (b) de la plataforma.

Capítulo 3

Modelado e identificación

3.1 Identificación de los motores de corriente directa.

En el prototipo del robot paralelo Stewart se usaron motores de corriente directa con reductor mecánico. Estos motores forman parte una de las articulaciones del robot, la flecha de salida del motoreductor es conectado a un tornillo sinfín, el conjunto en total es el encargado de convertir el movimiento rotacional del motor en movimiento lineal requerido por la junta prismática de la junta del robot.

El robot paralelo utilizara 6 motores de corriente directa de imán permanente del tipo escobilla figura 3.1. Los motores tienen integrado un reductor de velocidad de engranes el cual eleva el par de salida.

Figura 3.1 Motoreductor EGM30.

3.2 Caracterización del motor.

¿Qué es caracterizar?

Realizar algunos experimentos con la Planta (objeto a controlar) con el fin de conocer los valores de los parámetros que intervienen en su modelo matemático.

En este caso el modelo a considerar es el motor de corriente directa y el objetivo de determinar el valor de los parámetros que intervienen en su modelo matemático es realizar simulaciones numéricas, ya sea para realizar algunos análisis en ese modelo, para diseño de un sistema de control de posición o de velocidad o para incorporarlo en un sistema más grande.

Se desea realizar el control de posición del motor de CD, para esto es necesario obtener la función de transferencia y proponer un controlador que cumpla con un comportamiento dinámico deseado.

Se tiene el esquema de control mostrado en la figura 3.2, para realizar el controlador se deben de determinar los parámetros de la planta, estos parámetros se determinan al caracterizar un motor, existen varios métodos para caracterizar un motor, unos usan principalmente pruebas físicas y la respuesta a una señal conocida para poder observar su comportamiento, mientras que otras usan solo la respuesta a una señal conocida para identificar la planta.

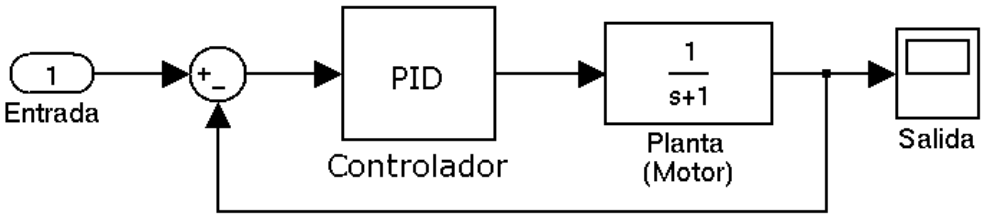


Figura 3.2 Sistema de control de la planta (Propuesto).

A continuación se presentan dos métodos para caracterizar el motor, uno de ellos por pruebas experimentales y el otro por medio del software Matlab, se incluye una comparación de los dos modelos obtenidos.

3.3 Determinación de los parámetros de la planta por medio de pruebas experimentales.

Una forma de expresar el modelo matemático de un motor de CD es por medio de su función de transferencia:

$$T(s) = \frac{\omega(s)}{Vf(s)} = \frac{\frac{Ka}{RJ}}{s + \frac{KaKb}{RJ}} = \frac{a}{s+b}$$

donde:

R = Resistencia de armadura (Ω)

Ka= Constante que relaciona el par y la corriente (Nm/A)

Kb = Constante que relaciona FCEM y la velocidad angular (V s/rad)

J = Carga o momento de inercia (kgm^2)

La caracterización consiste en tres pruebas.

- Prueba a rotor bloqueado
- Prueba a robot libre
- Prueba de la respuesta al escalón

3.3.1 Prueba a rotor bloqueado.

Para obtener la resistencia de armadura $R = Vf/I$ se realiza la prueba a rotor bloqueado y la constante $Ka = \frac{Tm}{I}$. En la tabla 1 se presentan los valores registrados del voltaje de la fuente (Vf) y corriente (I), en la tabla 2 corriente (I) contra el par Tm. En la figura 3.3 y 3.4 se muestra la linealización y las ecuaciones que nos ayudan a obtener los parámetros R y Ka.

V[v]	I[mA]
1	0.2
1.5	180
2	280
2.5	330
3	420
3.5	490
4	540
4.5	600
5	690
5.5	740
6	780
6.5	830
7	1000
7.5	1020
8	1070
8.5	1110
9	1160
9.5	1450
10	1270
11	1580

Tabla 3.1 Prueba a rotor bloqueado V[v] vs I[A].

I[A]	T[Nm]
0.15	0.0057
0.3	0.0114
0.5	0.019
0.69	0.02622
1.04	0.03952
1	0.038
1.15	0.0437
1.3	0.0494
1.4	0.0532

Tabla 3.2 Prueba a rotor bloqueado I[A] vs T[Nm].

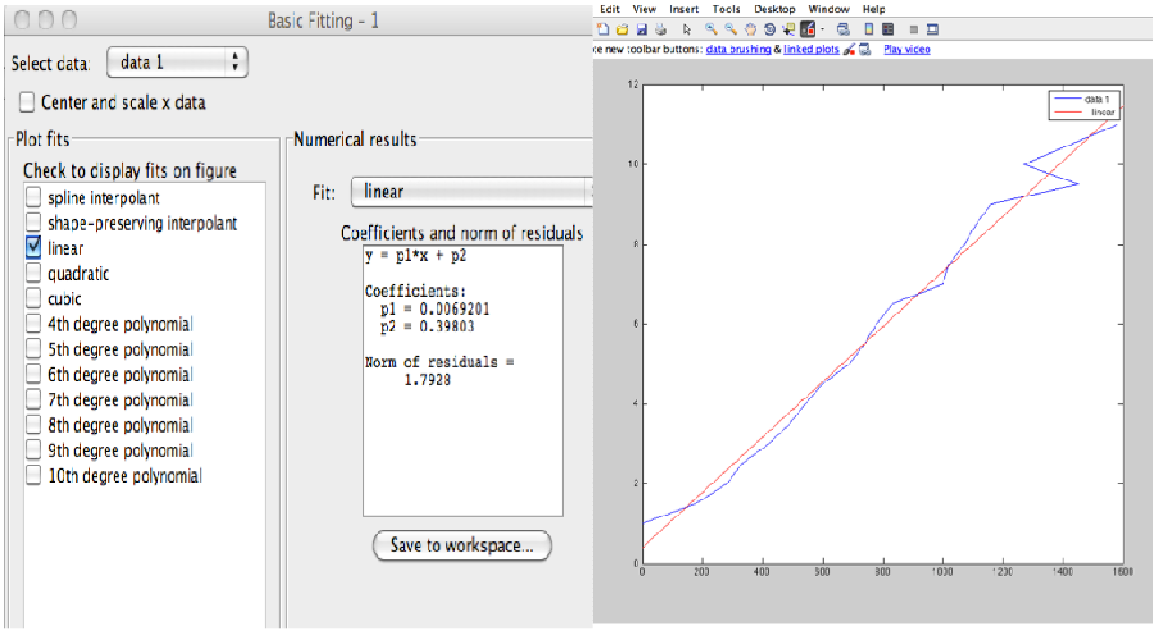


Figura 3.3 Corriente contra voltaje.

De la linealización de los valores de I vs Vf se obtiene la siguiente ecuación de la recta $Vf = 0.0069201i + 0.39803$. Se observa que la pendiente de la ecuación es:

$$.m = \frac{V}{I} = R = 0.0069201$$

Como la corriente está en miliamperes se multiplica este valor por 1000 dando $R = 6.9201\Omega$

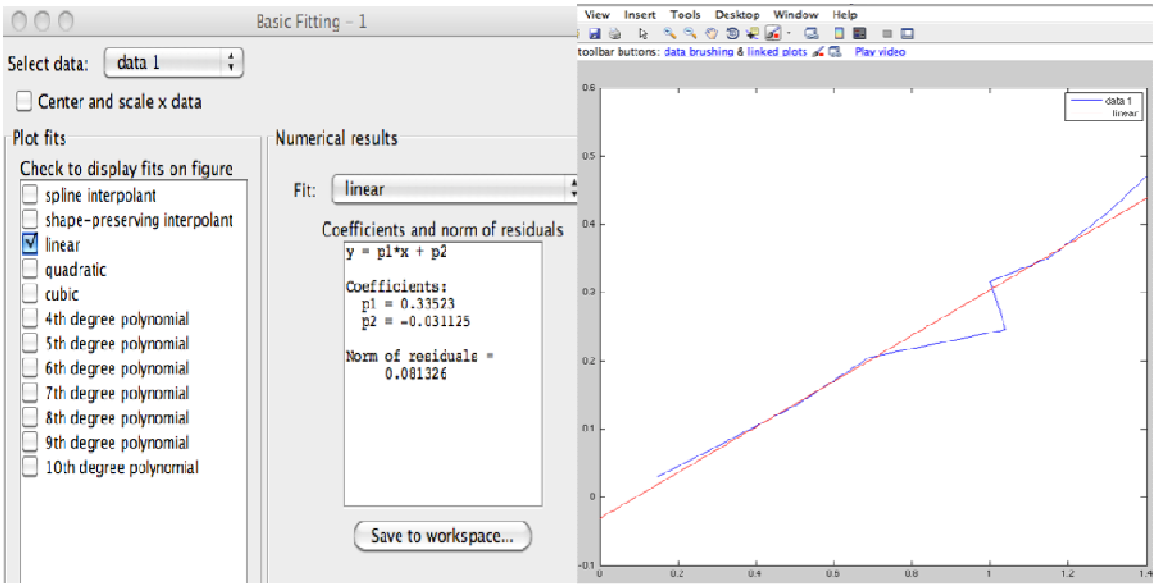


Figura 3.4 Corriente contra par.

De la linealización de los valores de I vs Tm se obtiene la siguiente ecuación de la recta $Tm = 0.33523i - 0.031125$ .Se observa que la pendiente de la ecuación es $m = \frac{Tm}{I} = Ka = 0.33523$.

3.3.2 Prueba a rotor libre.

Posteriormente se realiza la prueba de respuesta a rotor libre la cual consiste en conectar dos motores basados en la configuración de la figura 3.5.

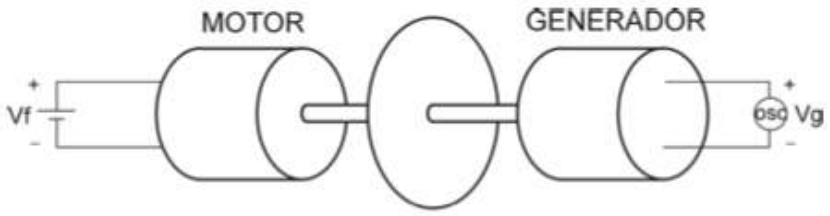


Figura 3.5 Configuración para la prueba de rotor libre y respuesta al escalón.

Como se observa en la figura anterior se necesita acoplar los motores para medir la variación del voltaje generado del motor (generador) cuando aplicamos una señal a la entrada del motor (figura 3.6).

Figura 3.6 Acoplamiento entre los motores caracterizados.

De esta prueba se obtiene la tabla 3.3 y la linealización de los datos (figura 3.7).

V[v]	Tiempo[ms]	w(rad/s)	Vg[v]
2	20	1.745329252	1
2.5	12	2.908882087	1.46
3	10	3.490658504	1.97
3.5	8	4.36332313	2.29
4	6	5.817764173	2.93
4.5	5.5	6.346651825	3.42
5	5	6.981317008	4.05
5.5	4	8.72664626	4.5

6	3.8	9.185943432	4.95
6.5	3.4	10.26664266	5.56
7	2.7	12.92836483	6.09
7.5	2.6	13.42560963	6.59
8	2.4	14.54441043	7.11
8.5	2.2	15.86662956	7.61
9	2	17.45329252	8.18
9.5	2	17.45329252	8.62
10	1.9	18.37188686	9.21
10.5	1.9	18.37188686	9.64
11	1.7	20.53328532	10.14
11.5	1.6	21.81661565	10.64
12	1.5	23.27105669	11.14

Tabla 3.3 Prueba al rotor libre.

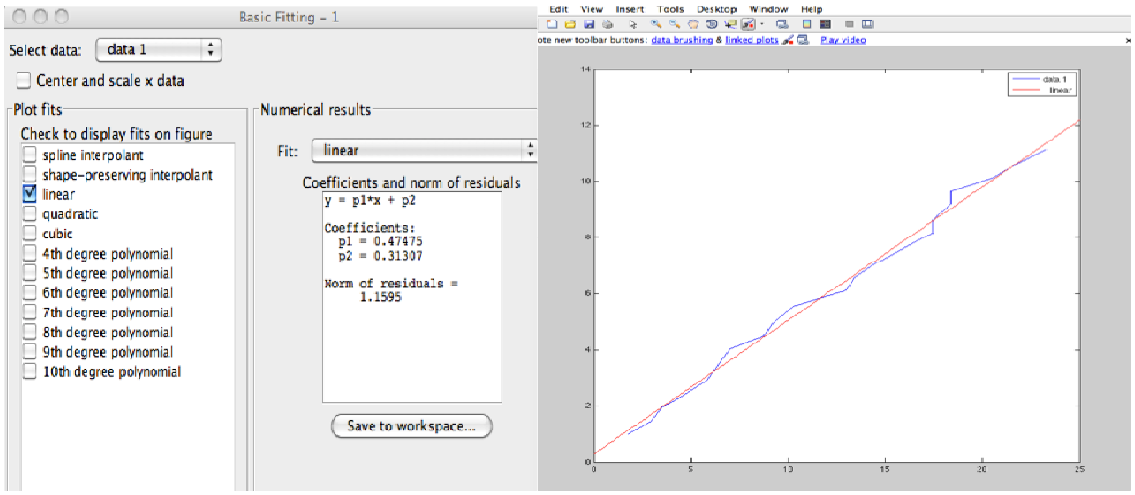


Figura 3.7 Velocidad contra voltaje generado.

De la linealización de los valores de w vs Vg se obtiene la siguiente ecuación de la recta $Vg = 0.4747\omega + 0.31307$ por lo tanto $m = Kb = \frac{Vg}{\omega} = 0.47475$.

3.3.3 Prueba respuesta al escalón.

Para llevar a cabo la prueba al escalón se utilizó la tarjeta de adquisición (DAQ) de LabView NI-USB 6009, usando los motores acoplados como se muestra en la figura 3.8, se usa un motor como "motor" y el otro como generador por el cual se mide el voltaje generado.

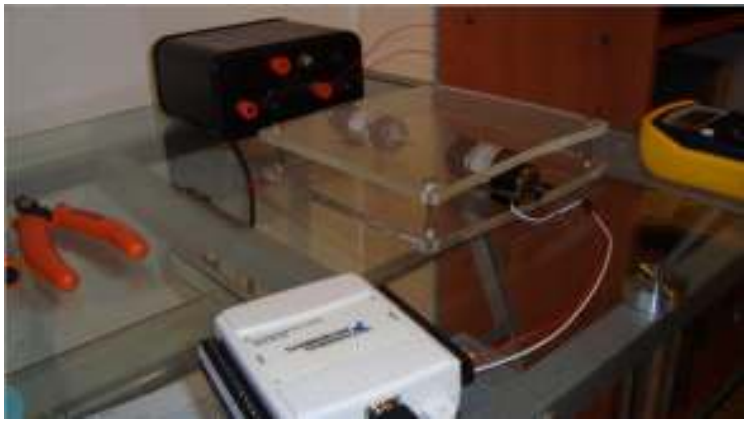
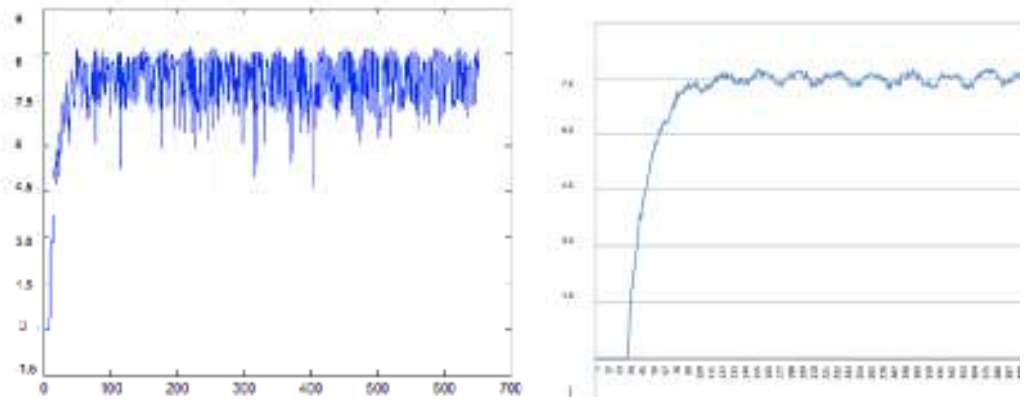


Figura 3.8 Adquisición de datos por medio de una DAQ National Instruments.

Al motor le aplicamos un escalón de 10V y medimos la salida por el motor que funciona como generador con ayuda de la DAQ (en las primeras muestras obtuvimos mucho ruido (Figura 3.9a), por lo que se decide introducir un capacitor de $10\mu f$ para "suavizar o filtrar" los datos ya que la adquisición por parte de la tarjeta es de 10 milisegundos, (Figura 3.9b)



a) b)

Figura 3.9 Señal con ruido (a) debida a la respuesta al escalón, señal filtrada (b).

Aun filtrando los datos adquiridos se observa que existe una variación de la respuesta por lo que tanto se procede a obtener la media de las muestras que se consideran en estado estacionario, este valor resulta de 7.7 V. Por lo que el 63.2% es 4.866, se procede a buscar en los datos el número de muestra que coincida con este valor (si es que alguno coincide, en caso contrario se busca que valor sobre pasa al 4.4886) esto resulta en la muestra 82 y como cada una de las muestras es tomada en intervalos de 10 ms, τ se calcula como la multiplicación del numero de muestras y el intervalo de adquisición.

$$\tau = (82 muestras) * (10 ms) = 0.82 s$$

3.3.4 Obtención del modelo en función de transferencia.

Lo siguiente es calcular b y como el inverso de τ es b, tenemos:

$b = \frac{1}{\tau} = \frac{1}{0.82} = 1.219512$ y como $b = \frac{KaKb}{RJ}$, despejando J queda $J = \frac{KaKb}{Rb} = 0.02332$, a se obtiene; $a = \frac{Ka}{RJ} = \frac{0.33523}{6.9201(0.0232)} = 2.568745$.

Por lo tanto la función de transferencia del motor queda:

$$T(s) = \frac{a}{s+b} = \frac{2.568745}{s+1.219512} \quad (3.1)$$

Los resultados de los datos obtenidos del motor se encuentran en la siguiente tabla

Parámetro	Valor
R (Ω)	6.9201
Ka (N m/A)	0.33523
Kb (V s/rad)	0.47475
J (kgm^2)	0.02332

Tabla 3.4 Caracterización del motor por medio de la herramienta *system identification* de Matlab

3.4 Caracterización del motor por medio de la herramienta system identification de Matlab.

En muchas ocasiones no se puede obtener teóricamente (mediante un modelo matemático) la función de transferencia de una planta, debido a diversos factores. En estos casos podemos hacer una identificación del sistema a partir de mediciones hechas a la entrada y a la salida de la planta. Para este efecto, el programa Matlab tiene una herramienta llamada SYSTEM IDENTIFICATION "Ident".

Como una forma alternativa para la obtención de los parámetros del sistema, se utilizó el acoplamiento de la figura 3.8 y por medio del toolbox de Matlab ident se relacionan las señales del voltaje del motor y el voltaje generado para obtener un modelo aproximado del motor.

Adquisición de las señales de entrada y salida

La frecuencia de muestreo de la señal es de 10000 Hz y se definió un tiempo de adquisición de 6 segundos, suficientes para poder observar el comportamiento del sistema.

El voltaje de entrada al sistema fue de 7 volts, debido a que los datos presentaban ruido se le adiciono un filtro para poder ver las señales originales. Figura 3.10.

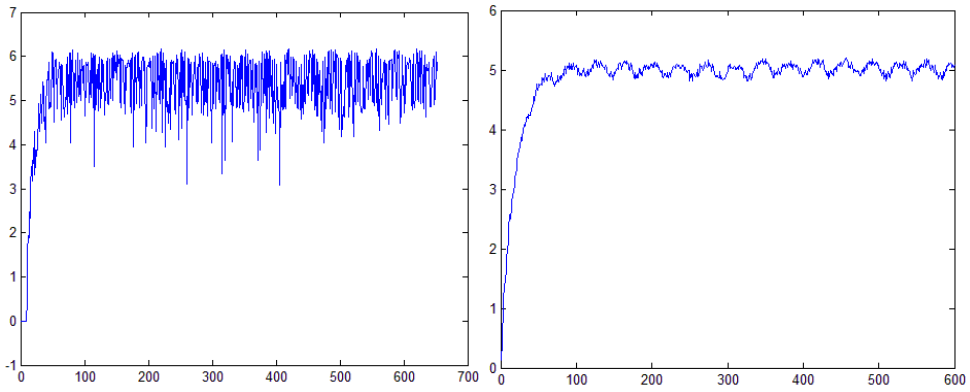


Figura 3.10 Señal con ruido (a), señal filtrada (b).

A continuación se muestra el programa el LabView con el que se realizó la adquisición a una entrada escalón (Figura 3.12)

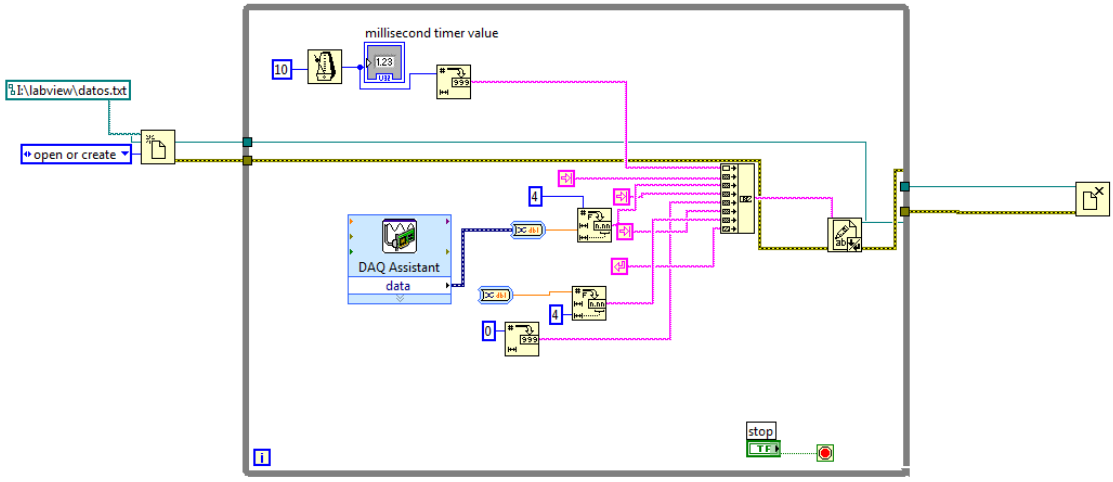


Figura 3.11 Programación de LabView.

En la parte izquierda crea un archivo donde se guardará la adquisición de datos, dentro de la estructura while se adquieren los datos cada 10 milisegundos por medio del bloque DAQ assistant y finalmente guarda todas las capturas en el archivo antes creado.

3.4.1 Obtención del modelo.

Importamos las señales de entrada y salida en la opción *import data*, ponemos nombres en los *edits, input* y *output*, así como el tiempo de muestreo que se empleó para la adquisición.

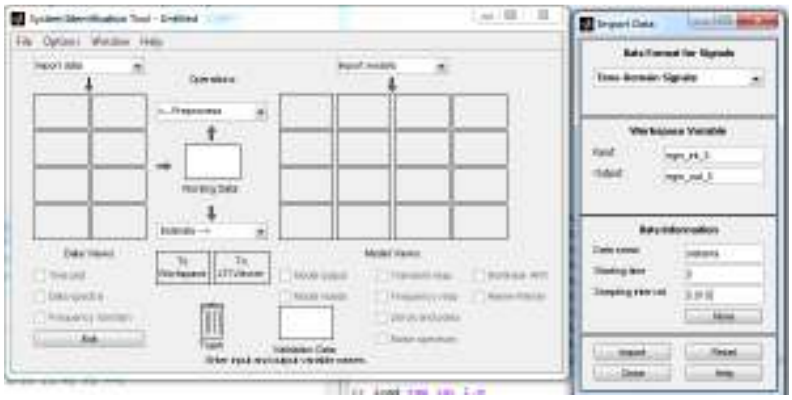


Figura 3.12 Ident: Importación de datos

En la opción *estimate* seleccionamos *process model* el cual hallará un modelo aproximado de la planta.

En nuestro caso tomamos un sistema con 1 polo real Tp1 y una ganancia K, este sería un sistema de primer orden. Luego de definir el tipo de sistema, seleccionamos la opción *estimate* que hallará el modelo aproximado.

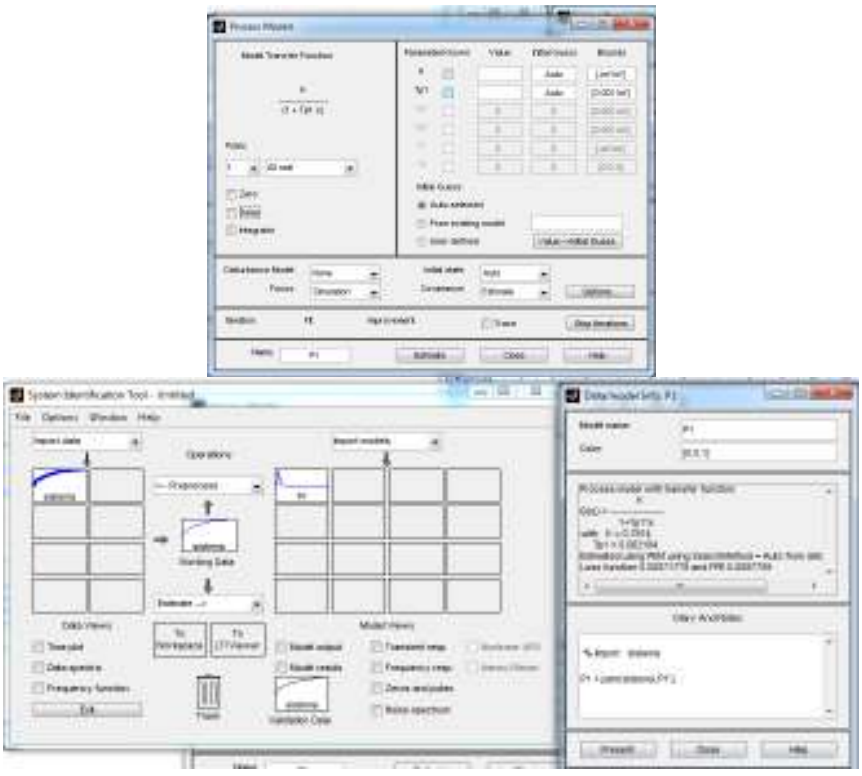


Figura 3.13 Ident: Obtención de la función de transferencia.

En base a una serie de pruebas se obtiene el siguiente modelo

$$G(s) = \frac{K}{1 + Tp1 * s} \qquad (3.2)$$

Con K=0.7814 y Tp1=0.062184.

A continuación se muestra una de las gráficas obtenidas de una de las pruebas

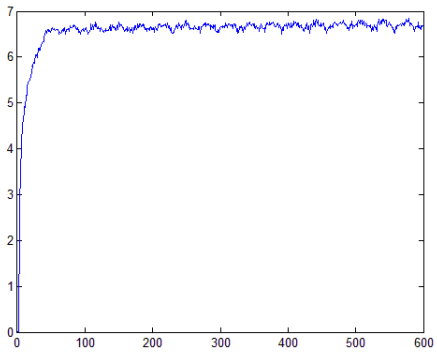


Figura 3.14 Repuesta del sistema obtenido con el Ident.

3.5 Conclusiones de los modelos obtenidos por los dos métodos.

De los modelos obtenidos (experimental y con la herramienta Ident de Matlab) se puede observar que hay una gran similitud de los modelos obtenidos 3.1 y 3.2, la obtención del modelo atreves de las pruebas físicas con la planta resulta un poco tardado y la obtención del modelo matemático con Matlab resulta más rápido y fácil de hacer, se realizo de las dos formas con el fin de validar el modelo.

Capítulo 4

Diseño del control de posición

4.1 Definiciones.

Señales determinísticas

Las señales determinísticas se pueden representar mediante expresiones matemáticas explícitas del tiempo. Son también señales determinísticas aquellas que no poseen una ecuación que las describa pero que están representadas mediante gráficos. El punto a resaltar es que el valor exacto de una señal determinística se puede predecir o calcular por adelantado.

Señales aleatorias

Es aquella en la cual existe un mayor o menor grado de incertidumbre en cuanto a un valor instantáneo futuro. Aunque el valor exacto en un instante dado no se conoce, muchas de las señales aleatorias que se encuentran en los sistemas tienen ciertas características en su comportamiento que permiten describirlas en términos estadísticos o probabilísticos.

Si una señal es determinística que caso tendría medirla si podemos saber para todo t el valor de la señal, por esto se dice que las señales aleatorias proporcionan verdaderamente información.

Señales en tiempo continuo

Es aquella señal que se define sobre un intervalo de tiempo continuo, una señal analógica es una señal en tiempo continuo ya que se puede adoptar un intervalo continuo de valores.

Señales en tiempo discreto

Está definida solo en valores discretos de tiempo (esto es, aquellos en los que la variable independiente t esta cuantificada) En una señal de tiempo discreto, si la amplitud puede adoptar valores en un intervalo continuo, entonces la señal se denomina señal de datos muestreados, esta se puede generar muestreando una señal analógica en valores discretos de tiempo, señal de pulsos modulada en amplitud [16].

Una señal digital es una señal en tiempo discreto con amplitud cuantificada, dicha señal se puede cuantificar mediante una secuencia de números.

Estas señales se pueden describir mediante ecuaciones en diferencias.

Sistemas de control en tiempo discreto

Son aquellos sistemas en los cuales una o más de las variables pueden cambiar solo en valores discretos de tiempo. Estos instantes se denotan mediando kT o t_k $(k=0,1,2,...)$ pueden especificar los tiempos en los que se lleva a cabo alguna medición de tiempo físico o los tiempos en los que se extraen los datos de la memoria de una computadora digital. El intervalo de tiempo entre muestras se supone debe ser lo suficientemente corto de modo que el dato para el tiempo entre estos se pueda aproximar mediante una interpolación sencilla. Se pueden describir mediante ecuaciones en diferencias después de la apropiada discretización de las señales en tiempo continuo.

Proceso de muestreo

Este proceso se emplea siempre que un sistema de control involucra un controlador digital, puesto que son necesarias la operación de muestreo y cuantificación para ingresar datos a ese controlador, al igual de que se da un tiempo de muestreo cuando las mediciones necesarias para control se obtienen en forma intermitente o cuando el controlador se comparte entre varias plantas y de esta manera la señal se convierte en una de datos muestreados [16].

En general a la operación que transforma las señales en tiempo continuo en datos en tiempo discreto se denomina muestreo o discretización, mientras que a la operación inversa que transforma datos en tiempo discreto en una señal en tiempo continuo se le conoce como retención de datos, en la mayoría de los casos esto se realiza manteniendo constante la señal entre los instantes de muestreo sucesivos, a este circuito se le conoce como muestreo y retención (S/H, del inglés Sample and Hold)

Proceso de cuantificación

En este proceso la amplitud analógica muestreada se reemplaza por una amplitud digital (representada mediante un numero binario) luego la señal digital se procesa por medio de una computadora, donde la salida de la computadora es una señal muestreada que alimenta a un circuito de retención, esta salida es una señal en tiempo continuo que alimenta al actuador.

4.2 Conversión de modelos continuos a discretos.

Cuando se va a implementar un controlador por medio de un microprocesador, o un sistema digital, es necesario discretizar el controlador considerando el tiempo de respuesta de todos los dispositivos empleados, se debe tomar en cuenta que todo el sistema responde con la rapidez del elemento más lento, a continuación se describen algunos métodos para discretizar una función.

4.2.1 Método de adelanto.

Este método considera a f(t) constante e igual a f(t+T) en el siguiente intervalo de tiempo, esta consideración se puede expresar como:

$y(t+T)=y(t)+f(t+T)T$ donde T es el periodo de muestreo

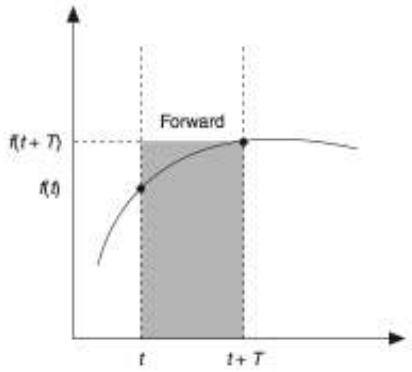


Figura 4.1. Discretización por método de adelanto.

En la figura 4.1 se observa el rectángulo formado por f(t) y f(t+T) aproximado por el compensador de adelanto, T (periodo de muestreo) es el ancho del rectángulo, entre más pequeño sea el valor, o más rápida la toma de la siguiente muestra menor es el error de aproximación. Es importante mencionar que la selección de T depende de la frecuencia y el error permitido en cada una de las aplicaciones.

4.2.2 Método de atraso.

Este método considera a f(t+T) constante e igual a f(t), esta consideración se puede expresar como:

$$y(t+T) = y(t) + f(t)T$$

La siguiente figura muestra el rectángulo formado por el método de atraso

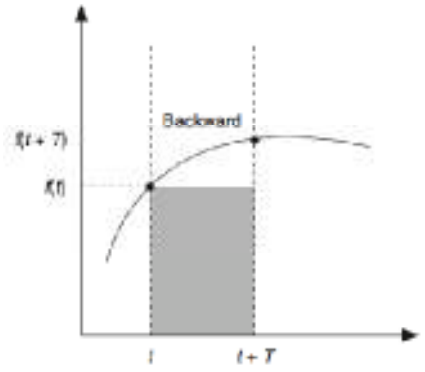


Figura 4.2. Discretización por el método de atraso.

En este método, al igual que en el anterior se observa que entre más pequeño sea el valor de T menor será el error obtenido.

4.2.3 Método trapezoidal.

El método trapezoidal también es llamado Tustin'method. Es un balance de entre el método de adelanto y atraso ya que obtiene un promedio de los rectángulos definidos por estos, el método

trapezoidal usa este valor para aproximar el área bajo la curva. En la siguiente ecuación se ejemplifica es:

$$y(t+T)=y(t)+\frac{[f(t)+f(t+T)]}{2}T$$

El último término de esta ecuación es el área de un trapezoidal de ancho T y base f(t) y f(t+T), figura 4.3.

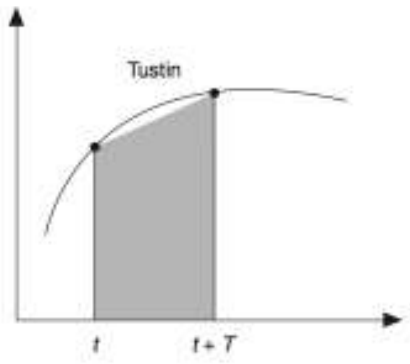


Figura 4.3. Método trapezoidal.

Para implementar un control digital es necesario discretizar la planta y seleccionar un tiempo de muestreo y cuantificación dentro de las capacidades del hardware y software utilizado.

Para convertir un modelo continuo a uno discreto se utiliza la siguiente tabla:

Tabla 4.1. Mapeo para convertir de continuo a discreto.

Método de aproximación	Continuo a discreto
Adelanto	$S \longrightarrow \frac{z-1}{T}$
Atraso	$S \longrightarrow \frac{z-1}{zT}$
Trapezoidal	$S \longrightarrow \frac{2(z-1)}{T(z+1)}$

La tabla 4.1 resume los métodos de aproximación utilizados para mapear una función continua (Trasformada de Laplace) y convertirla en discreta (Transformada Z).

En las ecuaciones anteriores el parámetro T representa el tiempo de muestreo.

4.3 Control digital PID con un microcontrolador.

4.3.1 Objetivo.

Diseñar e implementar un sistema de control digital de posición y velocidad para el motor de corriente directa EGM30, utilizando un microcontrolador PIC 16F877A trabajando a una frecuencia de 20Mhz.

4.3.1 Descripción.

Uno de los objetivos es lograr posicionar el motor EGM30 en una posición dada para esto es necesario que el error de estado estacionario del motor sea cero. Otro requerimiento es que el motor alcance rápidamente su posición final, se desea que el tiempo de establecimiento sea mínimo y tenga un sobrepaso considerable o nulo.

Para la implementación del controlador, dispondremos de un microcontrolador PIC que incorpora todas las funciones y periféricos para realizar estas tareas.

La retroalimentación se hará por medio de un encoder incremental acoplado al eje del motor, este enviara pulsos constantemente al microcontrolador indicándole su posición en todo momento.

Ante la señal del encoder proporcionada, el microcontrolador tomará una decisión en base a la posición actual y su posición deseada enviando una señal pwm de corrección a una etapa de potencia para hacer que el motor gire en el sentido indicado.

4.4.1 Planteamiento del problema.

Se desea diseñar e implementar un controlador discreto para un motor de corriente directa este control estará implementado en un microcontrolador de 8 bits.

4.4.2 Presentación de la planta.

A continuación se presenta la función de transferencia que se obtuvo mediante pruebas experimentales, como la función de transferencia que se obtuvo es de primer orden y está dada con respecto a la velocidad $T(s) = \frac{\omega(s)}{V(s)} = \frac{2.568745}{(s+1.219512)}$, para encontrar una función de transferencia que represente la posición con respecto a V(s) es necesario integrar la función, obteniéndose $T(s) = \frac{\theta(s)}{V(s)} = \frac{2.568745}{s(s+1.219512)}$ esta última función de transferencia es la que se considera para el control de posición del sistema.

4.5 Discretización de la planta de posición.

4.5.1 Método de adelanto, atraso, trapezoidal y simulaciones en Matlab.

Se trabajo en Simulink con la discretización de la planta de posición con el fin de obtener el tiempo de muestreo que más se aproximara a la respuesta de la planta real y que esta no fuera inestable. Se realizó el método de aproximación por adelanto, de atraso y trapezoidal. Después de incorporar diferentes tiempos de muestreo en los tres métodos se observo una mejor convergencia en T=0.05s con una regla de atraso, se muestran a continuación las pruebas que permiten corroborarlo.

Para esta tarea fue necesario encontrar la función de transferencia en lazo cerrado.

$$T(s) = \frac{GH(s)}{1 + GH(s)} = \frac{\frac{2.57}{s^2 + 1.22}}{1 + \frac{2.57}{s^2 + 1.22}} = \frac{2.57}{s^2 + 1.22s + 2.57}$$

Para la regla de adelanto se sustituye $s = \frac{z-1}{T}$

$$\text{HDAd(z)} = \frac{2.57T^2}{z^2 - 2z + 1.22Tz + 2.57T^2 - 1.22T + 1}$$

Con T=0.05

$$\text{HDAd(z)}_{\text{T=0.05}} = \frac{0.006425}{z^2 - 1.93902z + 0.945446}$$

El diagrama de bloques queda de la siguiente manera

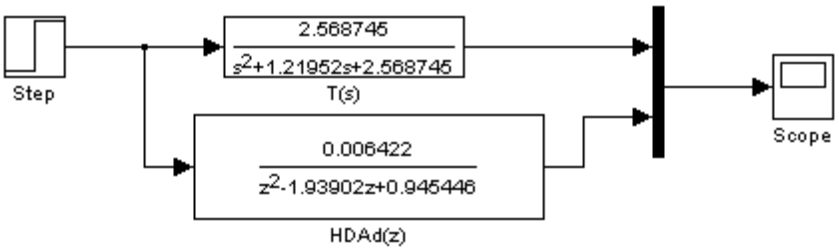


Figura 4.4 Diagrama de bloques de la Regla de adelanto para T=0.05.

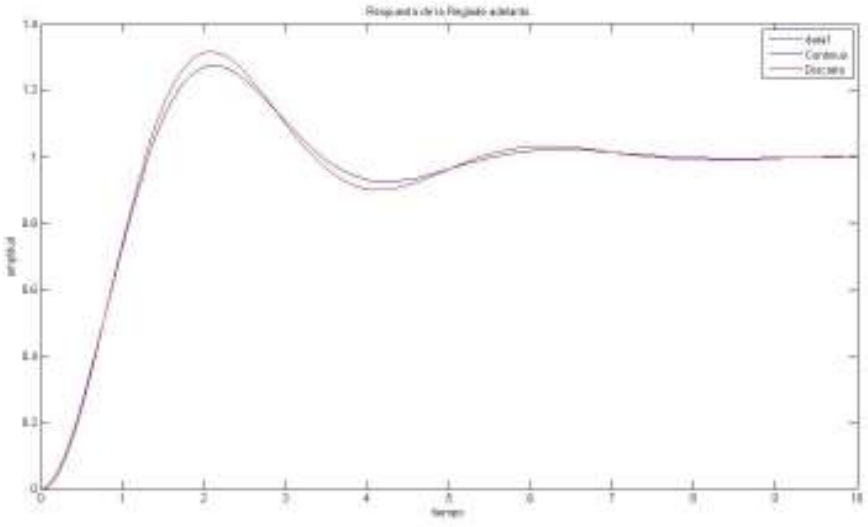


Figura 4.5 Respuesta de la regla de adelanto en planta de posición.

Para la regla de atraso $s = \frac{z-1}{Tz}$

$$HDAt(z) = \frac{2.57T^2z^2}{(1 + 1.22T + 2.57T^2)z^2 + (-2 - 1.22)z + 1}$$

Con T=0.05

$$\text{HDAt(z)} = \frac{0.006433z^2}{1.0674z^2 + 2.06098z + 1}$$

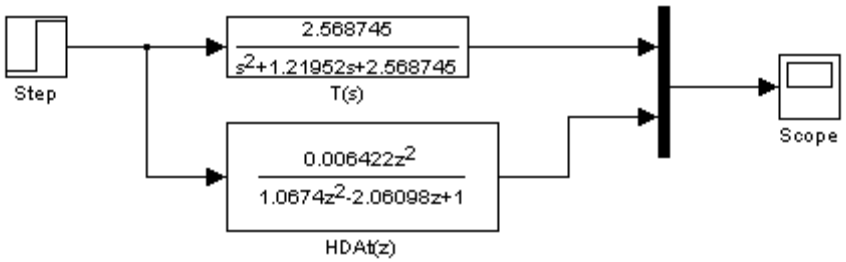


Figura 4.6 Diagrama de bloques de la regla de atraso.

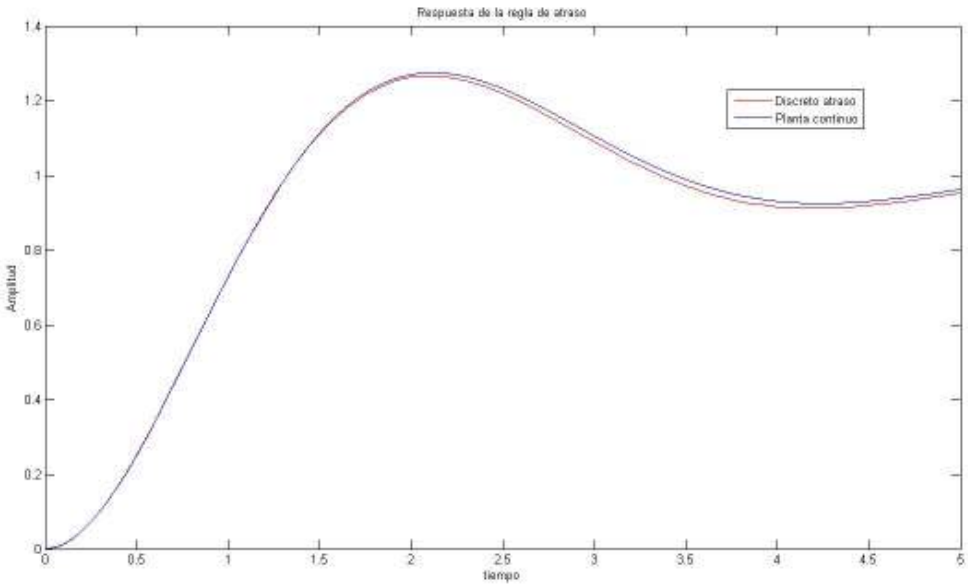


Figura 4.7 Regla de atraso en planta de posición.

Para la regla trapeziodal $\text{s} = \frac{z}{T}\frac{z-1}{z+1}$

$$HDTr(z) = \frac{2.57T^2z^2 + 2.57T^2}{(1 + 1.22T + 2.57T^2)z^2 + (-2 - 1.22)z + 1}$$

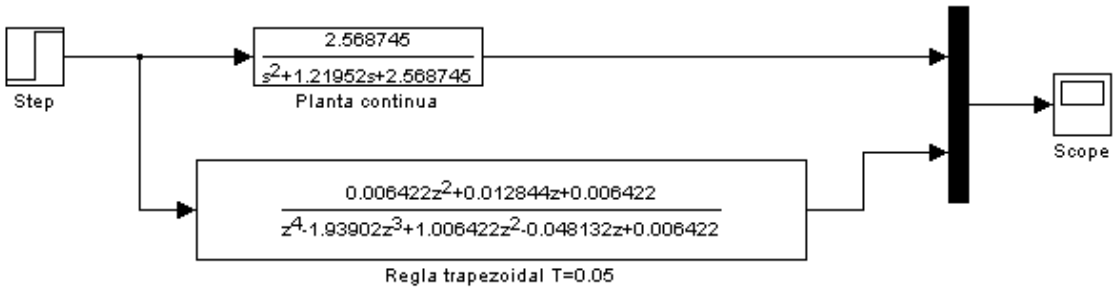


Figura 4.8 Diagrama de bloques para la regla trapezoidal.

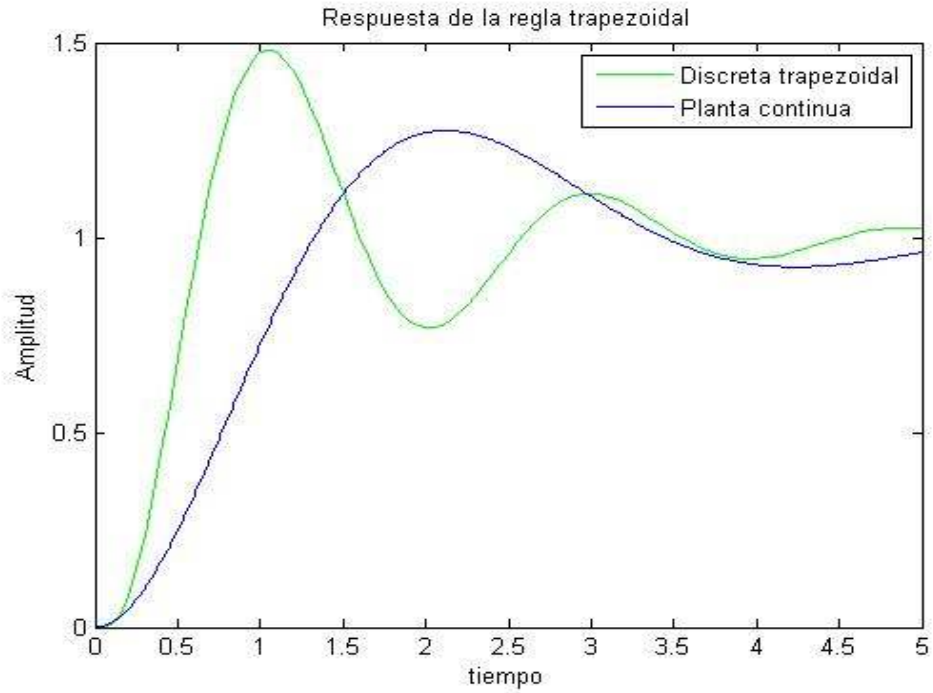


Figura 4.9 Respuesta de la regla trapezoidal en planta de posición.

De los ejercicios anteriores se puede concluir que la respuesta que mejor se apega al comportamiento de la planta en continuo es la regla de atraso. A continuación se realizaron diferentes experimentos en los cuales con la regla de atraso se manejan diferentes tiempos de muestreo con el fin de ver el tiempo que nos ofrece una mejor aproximación.

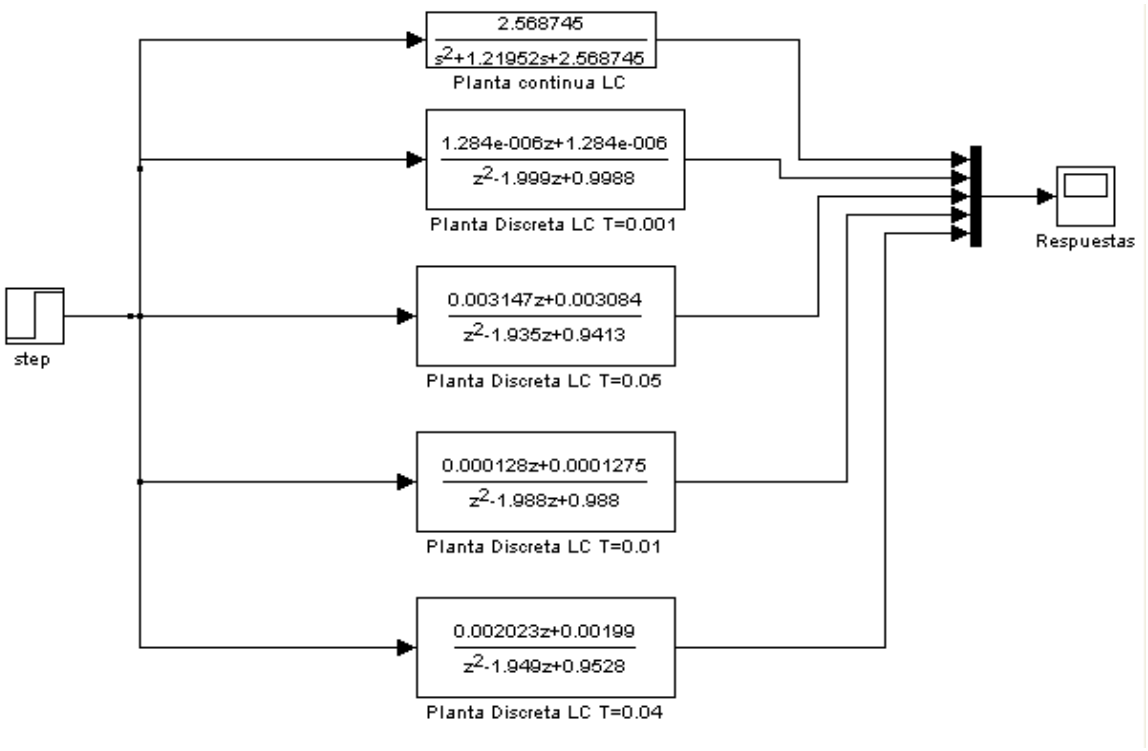


Figura 4.10 Diagrama de bloques para la regla de atraso y diferentes tiempos de muestreo.

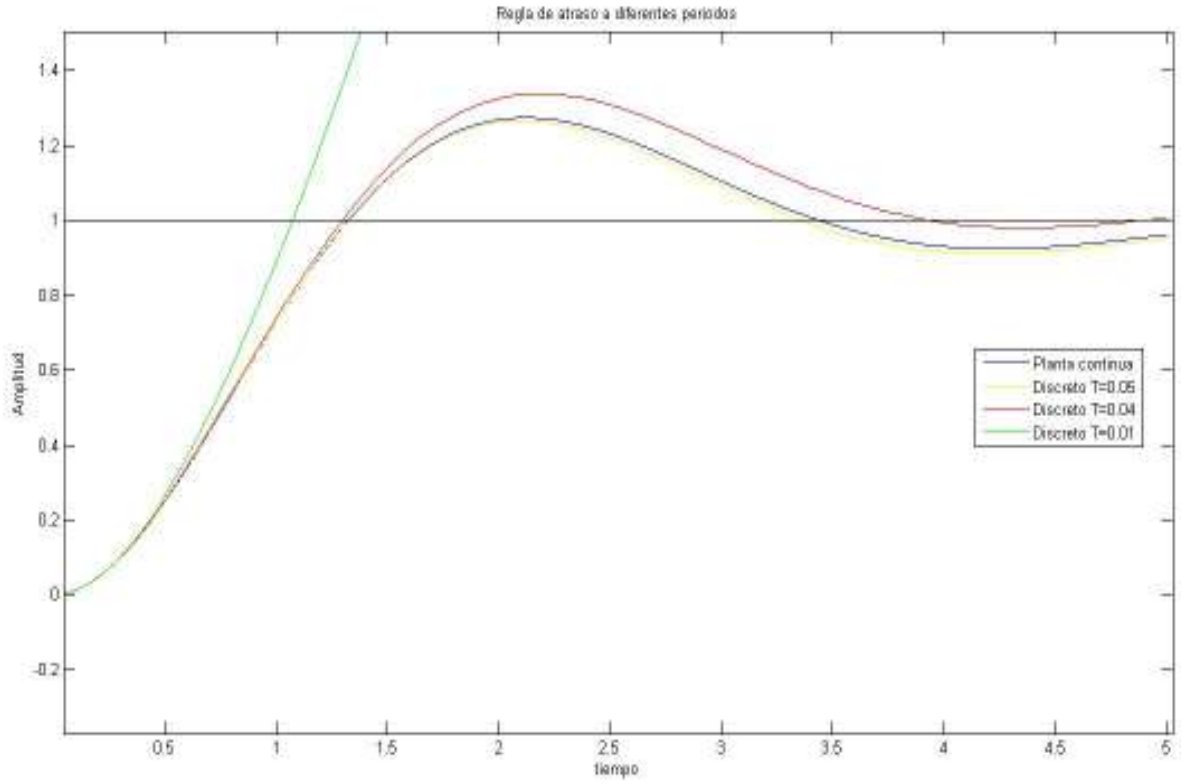


Figura 4.11 Regla de atraso para diferentes periodos de tiempo.

Las pruebas anteriores permitieron confirmar que el período de muestreo T al cual debe estar sometido la planta es T=0.05 ya que con este tiempo la simulación es la que mejor se apega a la planta continua.

4.6 Sistema de control propuesto y justificación.

Se opta por usar al microcontrolador PIC como un dispositivo de control, debido a que es fácil programar el algoritmo del PID además de la flexibilidad de configuración del microcontrolador. La ventaja de usar este tipo de sistemas electrónicos es la rapidez de su respuesta ya que son sistemas dedicados al control de un solo proceso. Una desventaja es que se debe tener un buen nivel de programación y tener buen conocimiento de su arquitectura para implementar el algoritmo de control.

4.7 Controlador PID.

La acción de control PID en controladores analógicos está dada por:

$$u(t) = K\left[e(t) + \frac{1}{Ti}\int_0^t e(t)\,dt + Td\frac{de(t)}{dt}\right]$$

Donde e(t) es la entrada al controlador (señal de error), u(t) es la salida del controlador, K es la ganancia proporcional, Ti es el tiempo integral (o tiempo de reajuste) y Td es el tiempo derivativo (o tiempo de adelanto).

En el dominio S, el controlador PID puede ser escrito como:

$$U(S) = K\left[1 - \frac{1}{Ti.S} + Td.S\right]E(S)$$

La forma discretizada del controlador PID puede ser derivado encontrando la transformada Z, se aplica la regla de atraso para discretizar:

Regla de atraso $S = \frac{(1-z^{-1})}{T}\ S = \frac{(1-\frac{1}{z})}{T}$

$$U(Z) = K\left[1 - \frac{T}{Ti1 - z^{-1}} + Td\frac{(1 - z^{-1})}{T}\right]E(z)$$

O

$$\frac{U(Z)}{E(Z)} = kp + \frac{ki}{1 - z^{-1}} + kd(1 - z^{-1})$$

Donde

$$kp = k,\ \ kd = \frac{kTd}{T},\ \ ki = \frac{kTd}{T}$$

Diseño paralelo del controlador PID

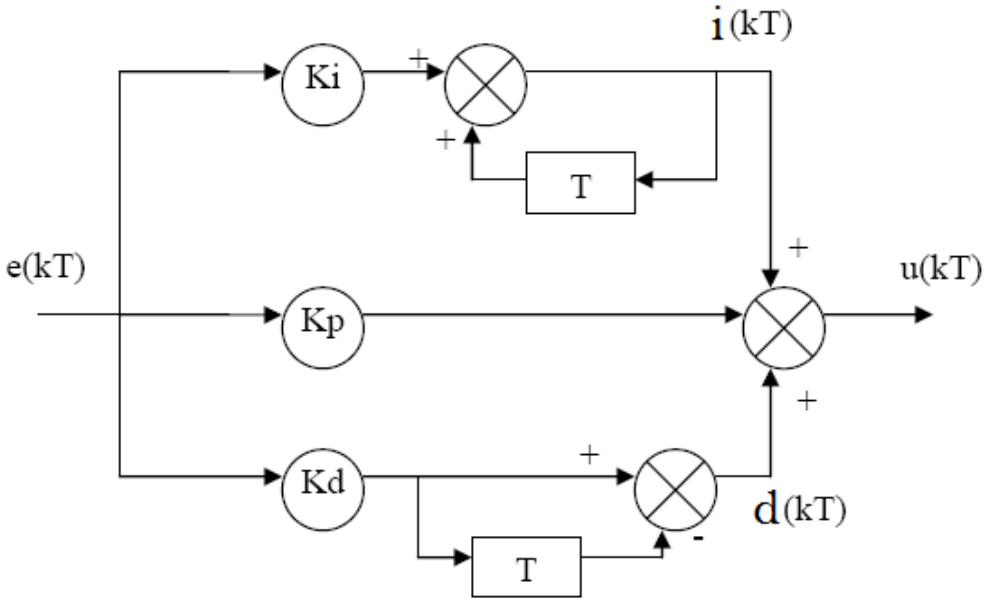


Figura 4.12 Diagrama de un controlador PID.

4.7.1 Algoritmo implementado en el microcontrolador.

El código implementado en el microcontrolador es:

```
  error=set_point-pos_real;        // obtiene el error
   P_term=kp*error;                 // termino proporcional
   I_term=ki*error;                 // termino integral
I_term=I_term+e_previ;
D_term=kd*error-kd*e_prevd;  //termino derivativo

   u=u+(P_term+I_Term+D_Term);   //suma  de los efectos P,I,D
   e_prevd=error;                          // guarda variables
    e_previ=I_term;
```

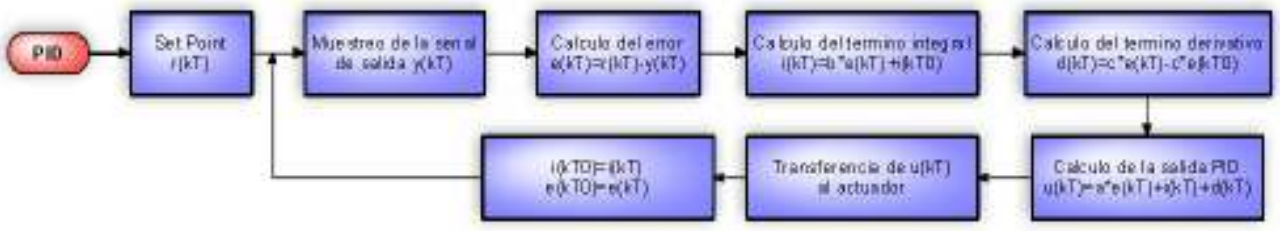


Figura 4.13 Algoritmo de programación del PID digital.

Las ganancias de los controladores se obtuvieron mediante simulaciones en el Simulink de Matlab.

4.7.2 Simulaciones en Simulink.

Se presentan a continuación los resultados de las simulaciones.

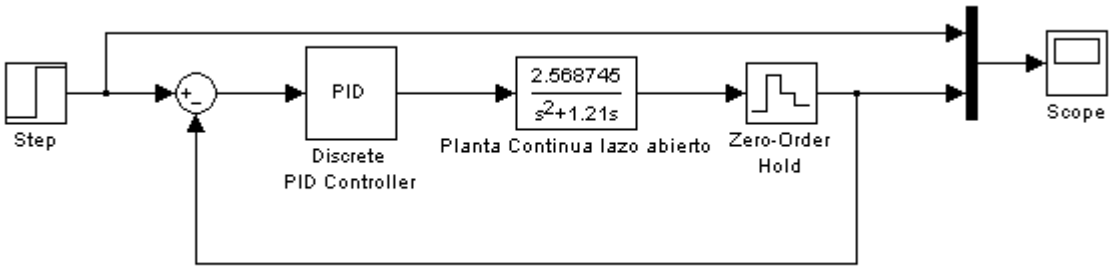


Figura 4.14 Diagrama de bloques de simulación del PID.

De varias pruebas que se realizaron con la planta se reportan a continuación dos pruebas con las cuales el sistema se comporto sin tanto sobrepaso y menor tiempo de establecimiento.

La respuesta al escalón de la primera simulación es:

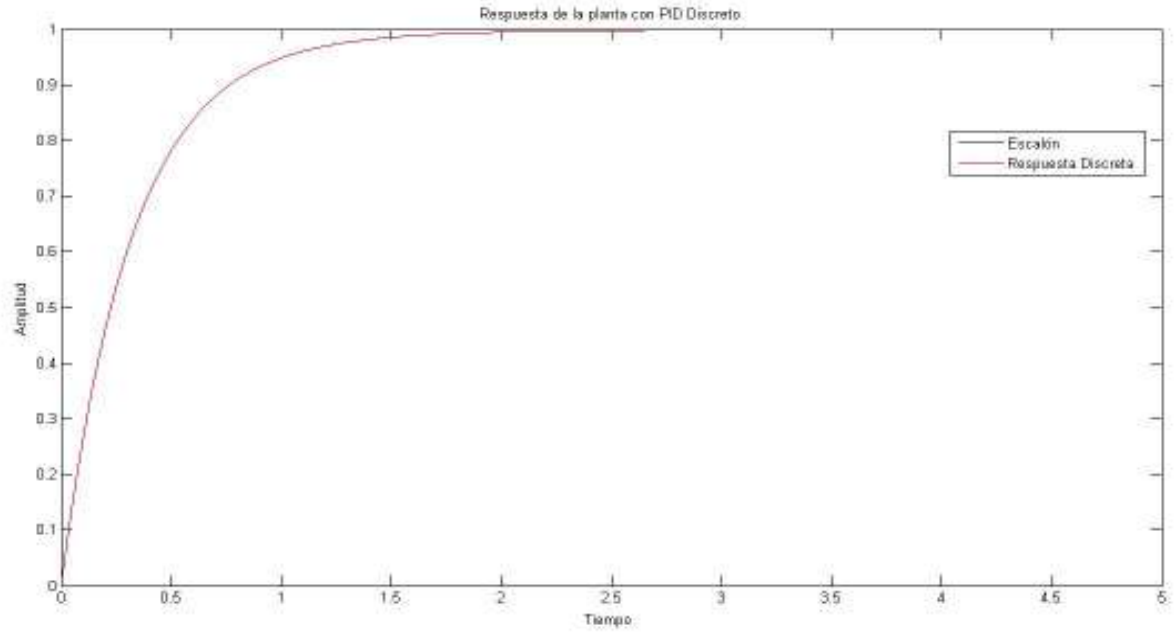


Figura 4.15 Respuesta de la planta con PID Discreto con Kp=1.4, Ki=0 y Kd=1.2.

La respuesta de la segunda simulación es:

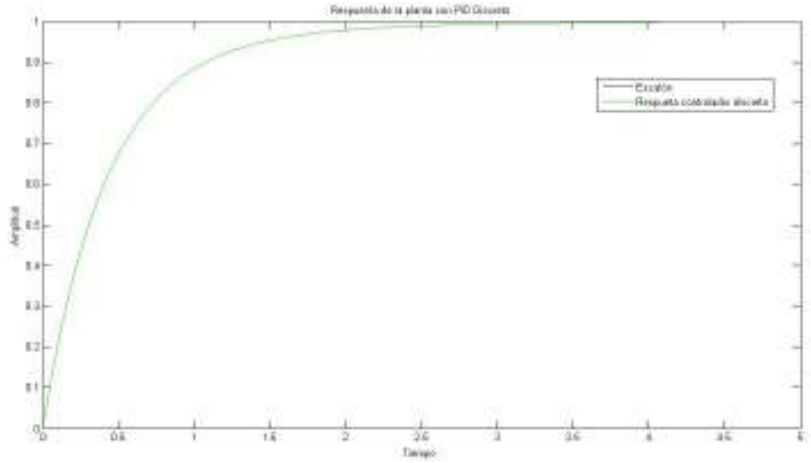


Figura 4.16 Respuesta de la planta con PID Discreto con Kp=1, Ki=0 y Kd=0.9.

Se realizaron simulaciones de una función seno y una función rampa con la finalidad de observar la respuesta de estos controladores ante este tipo de señales.

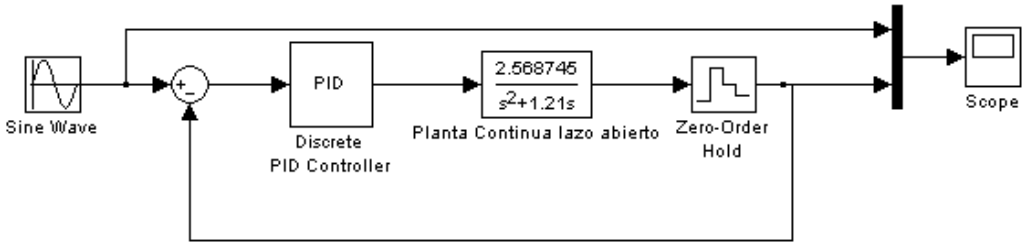


Figura 4.17 Diagrama de bloques del seguimiento de una función seno.

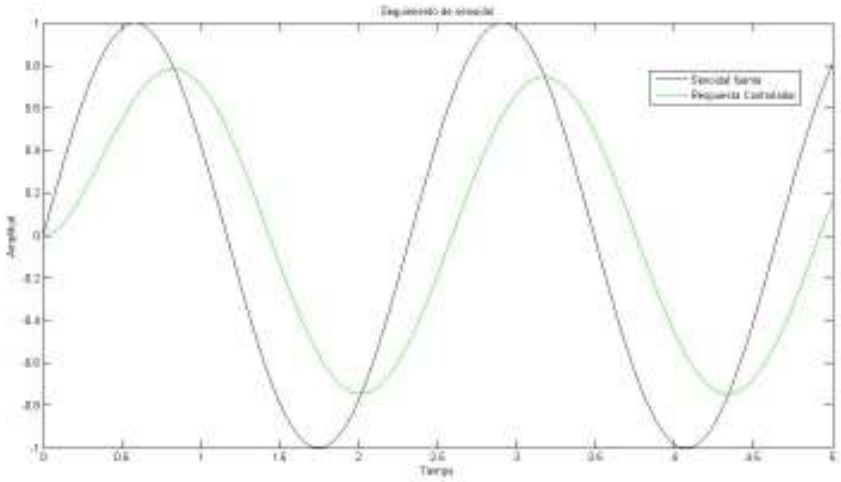


Figura 4.18 Seguimiento de una señal senoidal con Kp=1.4, Ki=0 y Kd=1.2.

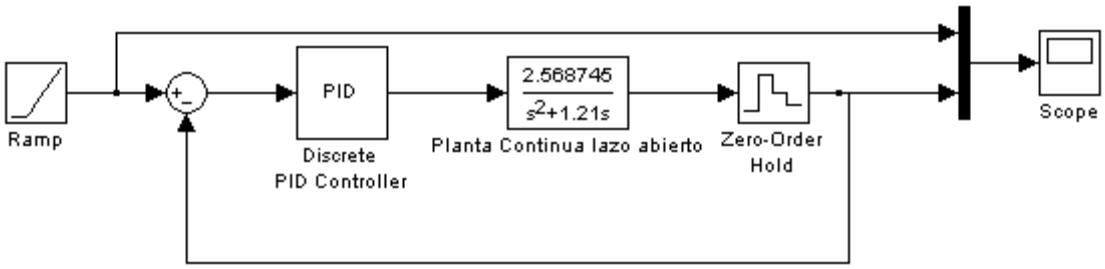


Figura 4.19 Diagrama de bloques del seguimiento de una función rampa.

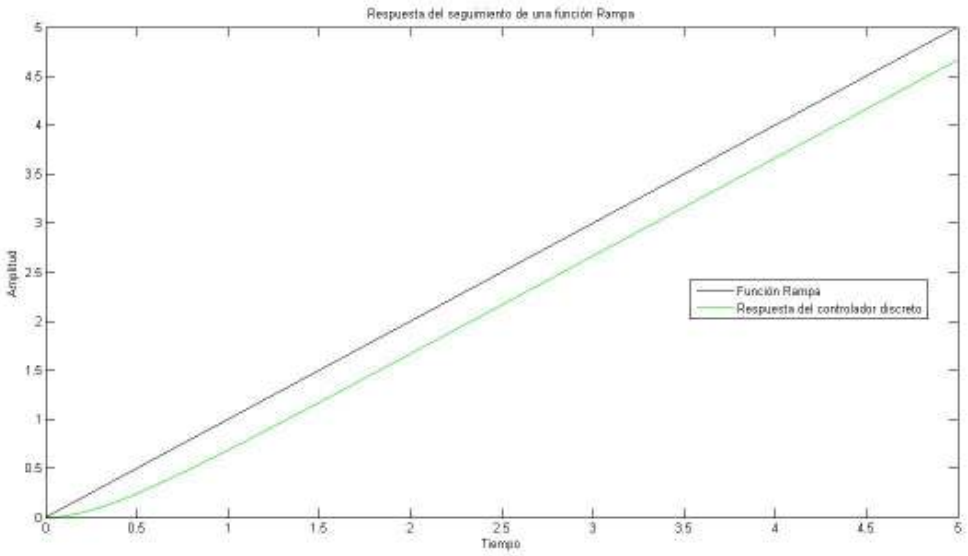


Figura 4.20 Respuesta del sistema a una función Rampa con Kp=1.4, Ki=0 y Kd=1.2.

Con estas simulaciones es posible llevar implementar las ganancias para observar experimentalmente que se cumple con las respuestas.

Controladores para la planta de velocidad

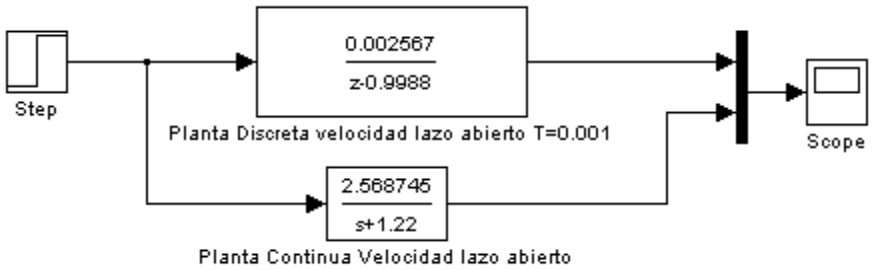


Figura 4.21 Diagrama de la plantas de velocidad a lazo abierto de planta discreta y planta continua.

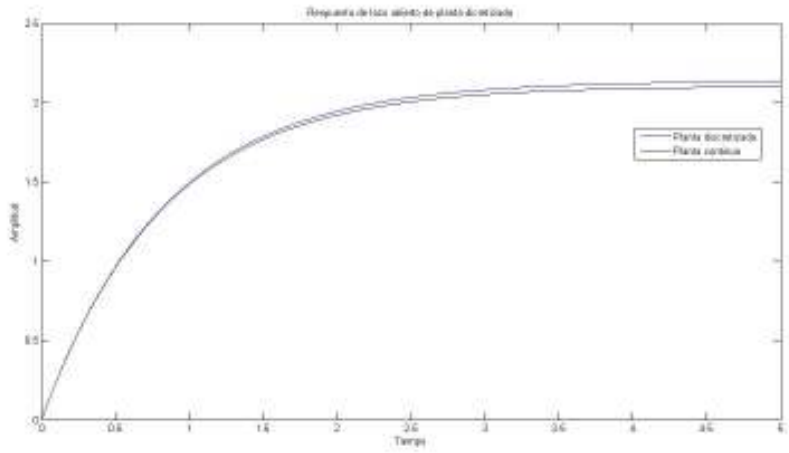


Figura 4.22 Respuesta de lazo abierto en planta discreta y continua de velocidad

La simulación anterior nos permite observar que la planta de velocidad es en su naturaleza estable ya sea discretizada o continua contrario a lo que pasaba con la planta de posición, además de que permite una buena discretización en un tiempo de muestreo de 1ms.

La siguiente figura muestra el diagrama de bloques con el cual fueron calculadas las ganancias hacer un buen control de velocidad.

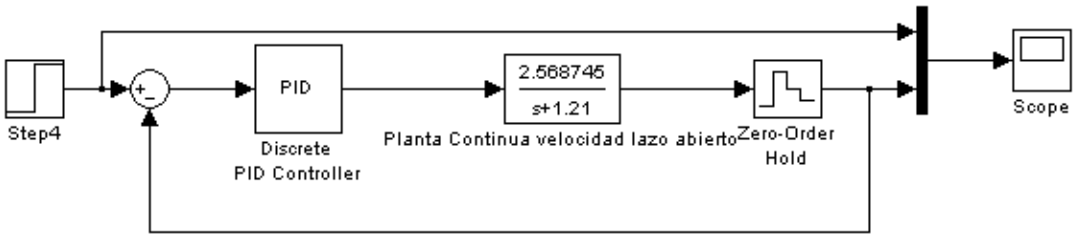


Figura 4.23 Diagrama de bloques para el cálculo de las ganancias en el PID Discreto.

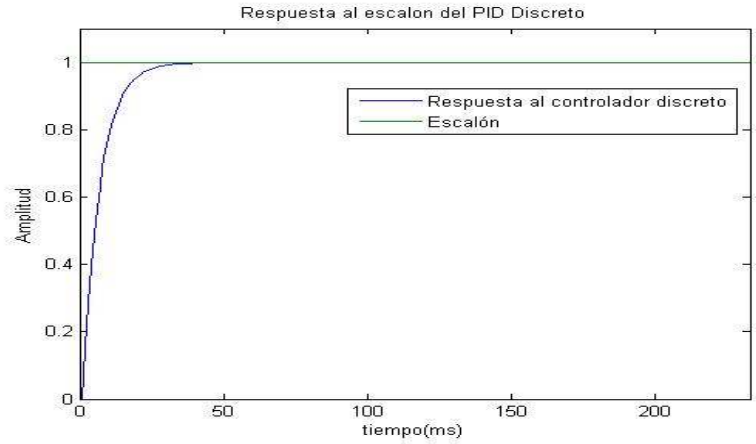


Figura 4.24 Respuesta al escalón PID Discreto con Kp=60, Ki=60 y Kd=0.

4.8 Implementación del controlador.

4.8.1 Periodo de muestreo.

Como periodo de muestreo se toma normalmente 1/10 del tiempo de subida del sistema el lazo cerrado. El tiempo de subida se define como el tiempo necesario para que el sistema pase del 10% al 90% del valor final de la respuesta.

Otro criterio que se puede usar es 1/10 de la constante de tiempo del sistema. La constante de tiempo de un sistema se define como el tiempo necesario para que la respuesta a un escalón unitario alcance el 63% del valor final de la respuesta.

La forma en que se implemento el tiempo de muestreo o tiempo en el cual se calcula el controlador es la siguiente:

Se utilizo el Timer1 del microcontrolador, este es un modulo temporizador contador, es un registro de 16 bits incrementado por el reloj del sistema.

El código que realiza el control se mete a la interrupción que se realizara cierto tiempo (tiempo de muestreo), para esto hay que calcular el valor a cargar en el registro de configuración del timer1 para que cuando pase el tiempo establecido realice la interrupción y haga el cálculo del control.

La ecuación para determinar el valor a cargar en el registro de configuración para generar el tiempo de muestreo es:

$$T = T_{CM}.Prescaler.(65536 - Carga\ TMR1)$$

Donde:

El preescaler es un divisor de frecuencia el que se toma es de 8.

$$T_{CM} = \text{es el ciclo maquina que se puede calcular con la ecuacion}$$

$$T_{CM} = \frac{4}{Fosc}$$

Para nuestro caso es:

$$T_{CM} = \frac{4}{20Mhz} = 200ns$$

Entonces despejando Carga TMR1

$$Carga\ TRM1 = 65536 - \frac{T}{Tcm.Preescaler}$$

Para un tiempo de muestreo de 100ms. El valor a cargar en el Timer1 es de 3036 y para un tiempo de muestreo de 1ms el valor es de 64911.

Las líneas de programación que hacen esto son:

```
#int_TIMER1
void TIMER1_isr(void)
{
(cálculo del control PID)
}
```

setup_timer_1(T1_internal|T1_DIV_BY_8); // configura el timer1 como contador interno con preescaler de 8

set_timer1(53036); para un periodo de muestreo de 20 ms

set_timer1(3060); //inicializa el timer1 para un tiempo de muestreo 100 ms

enable_interrupts(int_TIMER1); //habilita la interrupción del timer1

Este código completo del controlador PID se presentan en el Anexo A.

Los componentes electrónicos, circuitos de control y acondicionamientos de señal se presenta en el anexo E.

4.8.2 Interconexión del controlador y la planta.

Se muestra a continuación un diagrama eléctrico en el programa Proteus, en el cual se muestra como se interconectaron los sistemas de control, de potencia y los encoders.

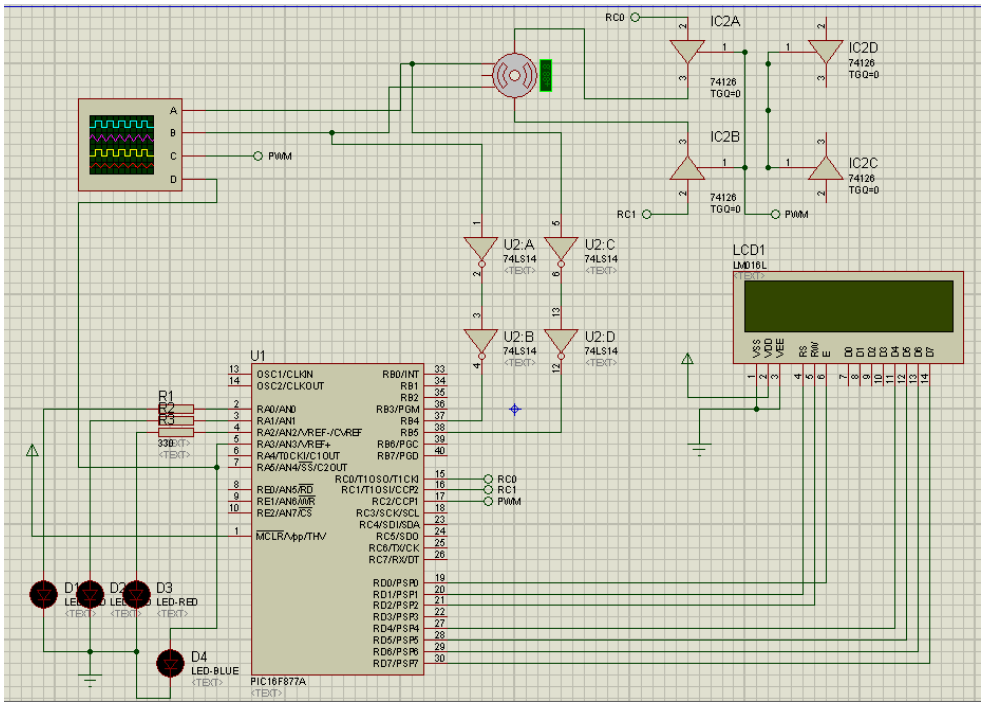


Figura 4.25 Diagrama eléctrico del sistema de control.

4.9 Presentación de resultados.

Se implemento un control PID discreto en el pic, para la posición fue necesario desarrollar el código para la lectura del encoder. El control de la posición como para la velocidad es el mismo y las ganancias se pueden variar dependiendo del desempeño del sistema que se desee. Al trabajar con un procesador exclusivamente dedicado al control del sistema se obtienen grandes velocidades de muestreo, en este caso se uso una velocidad de muestreo con un periodo de1ms pudiéndose obtener más velocidad, esta es una de las ventajas de trabajar con microcontroladores. Se programo el PID discreto en su forma posicional como lo muestra [16], también, existe el esquema de control PID en su forma de velocidad, en la cual se considera la diferencia hacia atrás. Este esquema de control presenta mejores características de respuesta que el esquema en la forma posicional, otra ventaja es que es útil en la supresión de correcciones excesivas en sistemas de control de procesos. Las leyes de control lineales en las formas de control PID, tanto en la forma posicional como en velocidad, son básicas en controles digitales, las leyes se pueden implementar mediante software en una computadora o en un microcontrolador como en este caso, y por lo tanto las restricciones en cuanto a software se eliminan por completo.

Capítulo 5

Desarrollo e Integración de Hardware y Software

5.1 Introducción.

En este capítulo se describen las partes de todo el sistema, compuesto por el robot, motores, sensores y acondicionadores, tarjeta de control e interfaz de control. La construcción del prototipo se realizo gracias al apoyo brindado por la DGAPA, UNAM, a través del proyecto PAPIIT IN108308.

El objetivo del prototipo es permitir el estudio de este tipo de mecanismos, estudiar su cinemática y poder reproducir diversas trayectorias en el espacio con el objetivo de verificar que el robot en estudio es capaz de realizar una tarea en este caso reproducir una trayectoria.

5.2 Descripción del funcionamiento del sistema.

Las ecuaciones cinemáticas y las trayectorias a generar del robot están programadas en el software Mathematica las ecuaciones son resueltas de forma Iterativa con el método numérico Newton Rhapson. La interfaz gráfica que comunica al robot con la computadora fue programada en Matlab. El control de cada uno de los motores se encuentra en los microcontroladores esclavos. En la siguiente figura se muestra el diagrama general de todo el sistema.

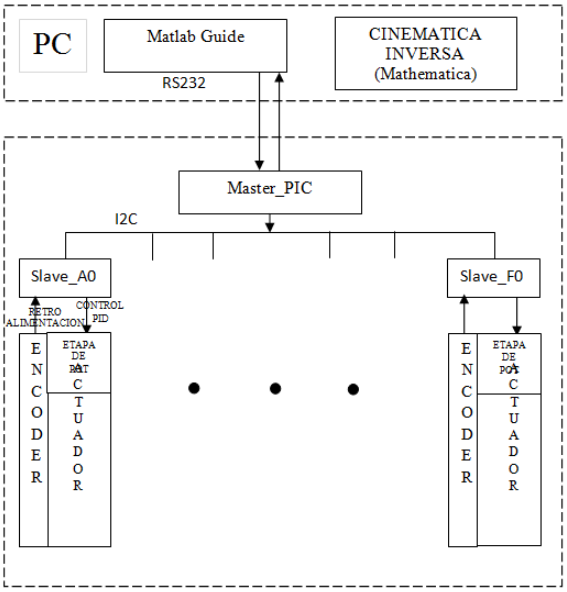


Figura 5.1 Diagrama general del sistema.

A partir de una trayectoria que se define en el espacio de trabajo del robot se generan puntos discretizados en un intervalo de tiempo estos puntos se obtienen con la cinemática inversa, una vez obtenidos se almacenan en Matlab en un archivo .m, estos puntos serán las consignas de posición de cada uno de los motores. La interfaz gráfica programada en Matlab tiene la función de mandar los puntos de cada una de las 6 articulaciones a la tarjeta de control vía puerto serie. Los datos que llegan a la tarjeta de control a través del puerto serie caen en un microcontrolador llamado maestro, este microcontrolador tiene la función de soportar la comunicación entre Matlab y los esclavos, una vez que el maestro ha recibido las consignas de cada motor las envía a cada uno de los 6 diferentes esclavos.

En la siguiente figura se muestra un diagrama de bloques donde se encuentran cada parte del sistema.

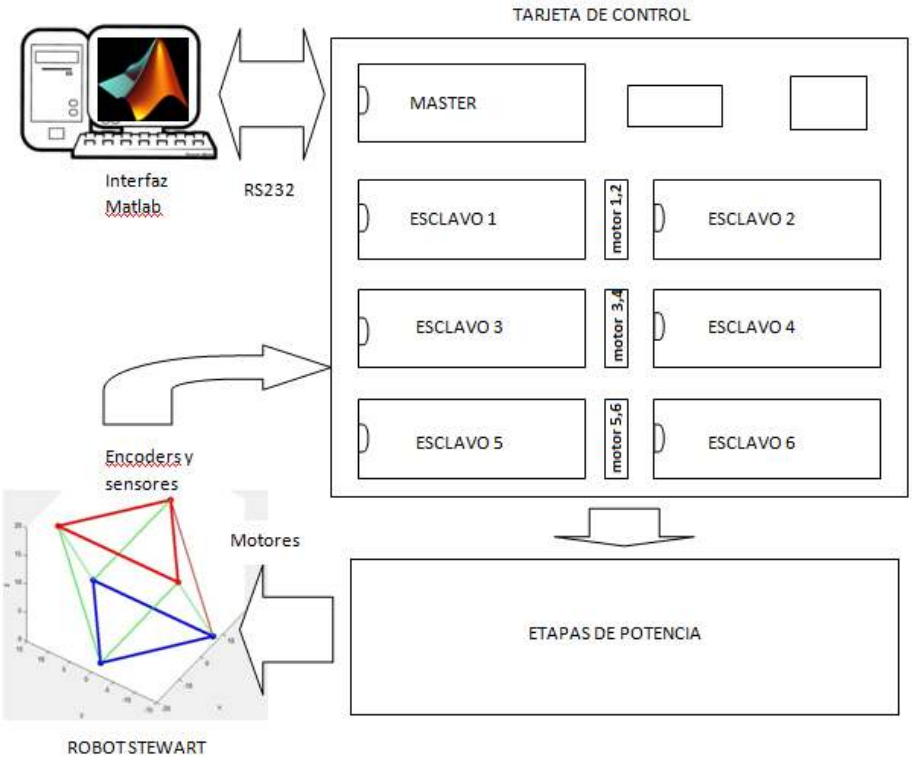


Figura 5.2 Diagrama de bloques del sistema.

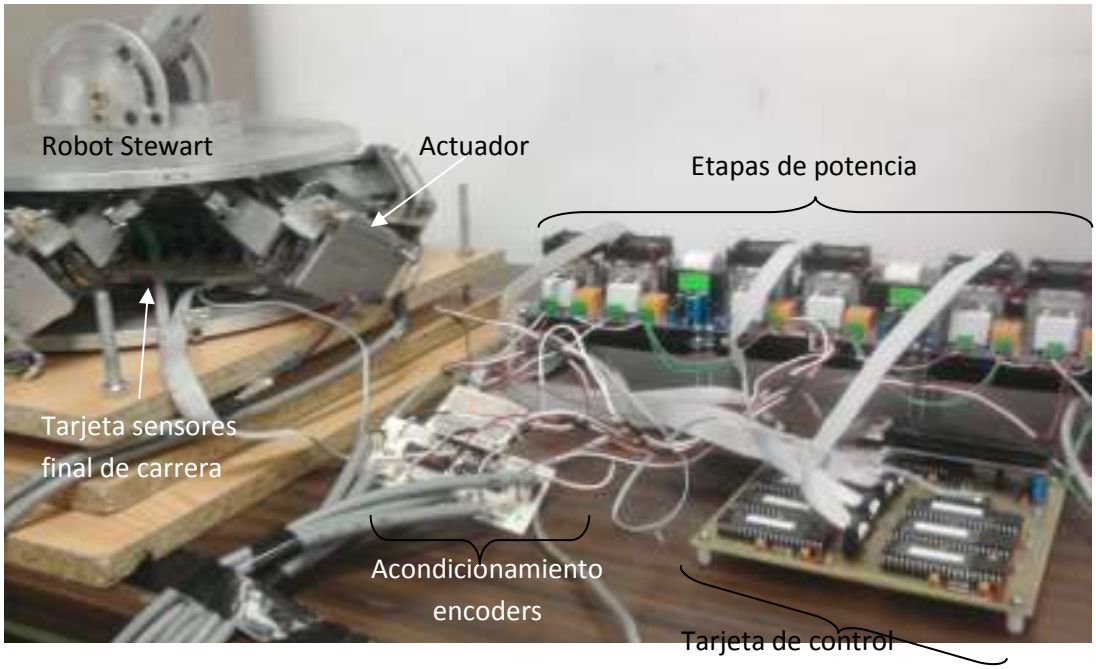


Figura 5.3 Fotografía del sistema completo.

5.3 Estructura física del robot

Como ya se ha mencionado anteriormente el robot Stewart de 6 grados de libertad tiene la configuración conocida como cubica, esa le da al robot mayor capacidad de carga. Cuenta con tres tipos de juntas, esféricas para la base, universales para el plato superior, y prismáticas para los actuadores.

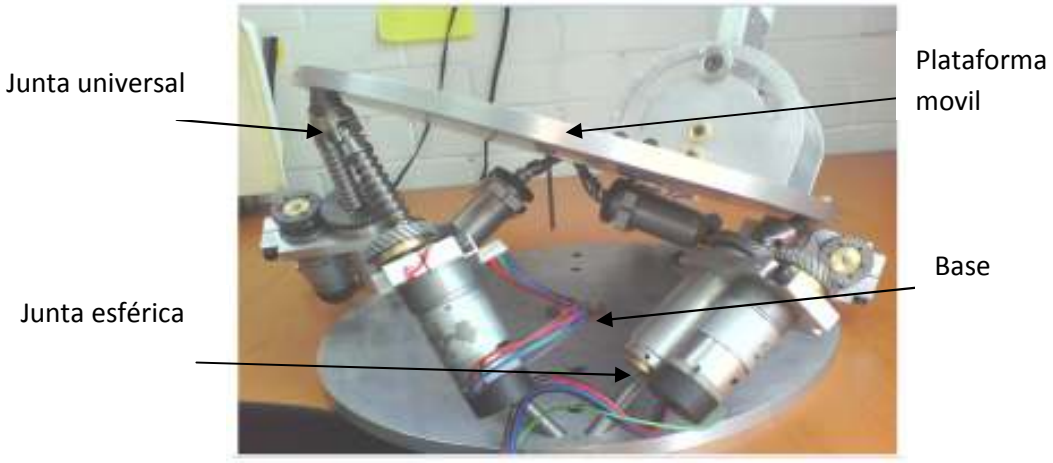


Figura 5.4 Configuración del robot Stewart.

Los actuadores están compuestos por motores de corriente directa, el actuador está compuesto por un tornillo sinfín para convertir el movimiento rotacional en transnacional y así obtener la junta prismática.

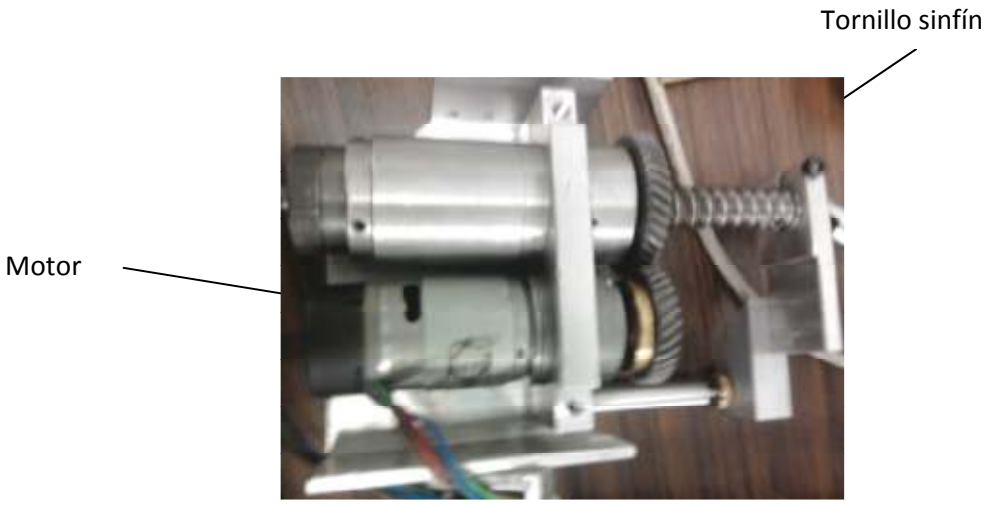


Figura 5.5 Articulación del robot.

5.4 Motores.

El prototipo cuenta con 6 motores de corriente directa con reductor mecánico, estos motores son los encargados de producir el movimiento de la junta prismática.

Los motores utilizados son de corriente directa de imán permanente del tipo escobilla, los motores tienen integrado un reductor de velocidad de engranes el cual eleva el par de salida. Para medir la posición de cada motor se utilizan decodificadores del tipo incremental, los cuales se acoplan directamente a la flecha del motor.

Motor EGM30

El motoreductor contiene una caja de engranes con relación 30:1 y un encoder magnético de efecto hall.

Especificaciones

Voltaje	12v
Torque	1.5kg/cm
Velocidad	170rpm
Corriente	530mA
Velocidad sin carga	150mA
Corriente a motor detenido	2.5ª
Potencia	4.22W
Pulsos del encoder	360

Tabla 5.1 Especificaciones del motor EGM.

Conexiones del encoder.

Color del cable	Conexión
Purpura(1)	Sensor Hall B Vout
Blue(2)	Sensor Hall A Vout
Green(3)	Sensor Hall tierra
Café(4)	Sensor Hall Vcc
Rojo(5)	+Motor
Negro(6)	-Motor

Tabla 5.2 Conexiones encoder.

Dibujo del motor

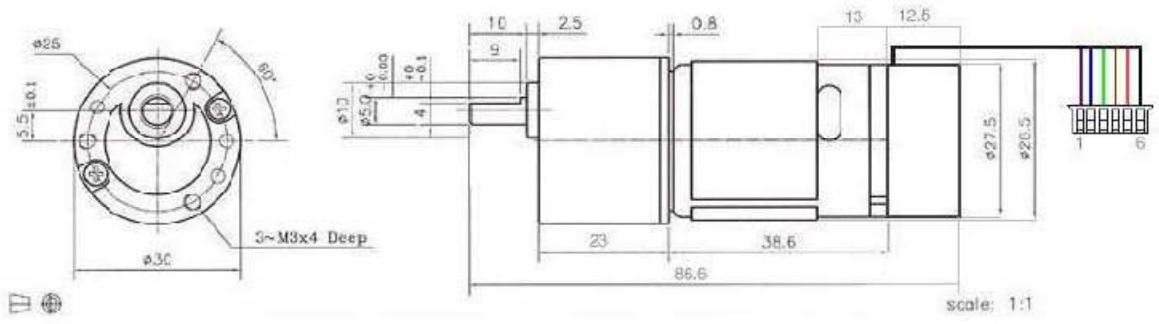


Figura 5.6 Dibujo mecánico del motor.

5.5 Sensores y acondicionamiento.

Se utilizaron switch ópticos H21A1 para los inicios de carrera y los finales de carrera de los actuadores, este tiene la finalidad de formar el home del robot y funciona como protección a los actuadores evitando que estos por alguna razón o fallo lleguen a su límite provocando una sobre corriente en el motor. En la figura 5.7 se muestra el circuito que se realizo y el la figura 5.8 se muestra la colocación de los sensores en los actuadores del robot. En el apéndice F se muestra el diagrama esquemático y el pcb del circuito.

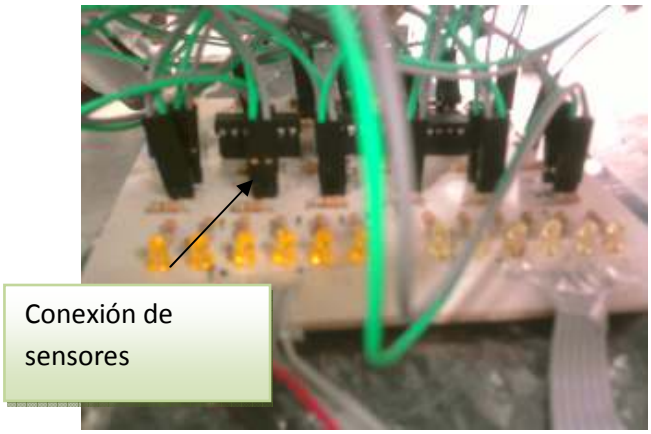


Figura 5.7 Circuito de acondicionamiento sensores. Figura 5.8 Colocación sensores.

Para poder leer los codificadores ópticos de los motores fue necesario hacer una tarjeta de acondicionamiento de las señales cuadradas que nos entregaba el encoder magnético, con el fin de mandar estas señales a la tarjeta de control y que el microcontrolador las procese, este circuito está compuesto por un integrado schmitt trigger el cual cuadra la señal del encoder y la mantiene en el rango de 0 a 5 volts. En el apéndice F se muestra el diagrama esquemático así como su pcb del circuito.

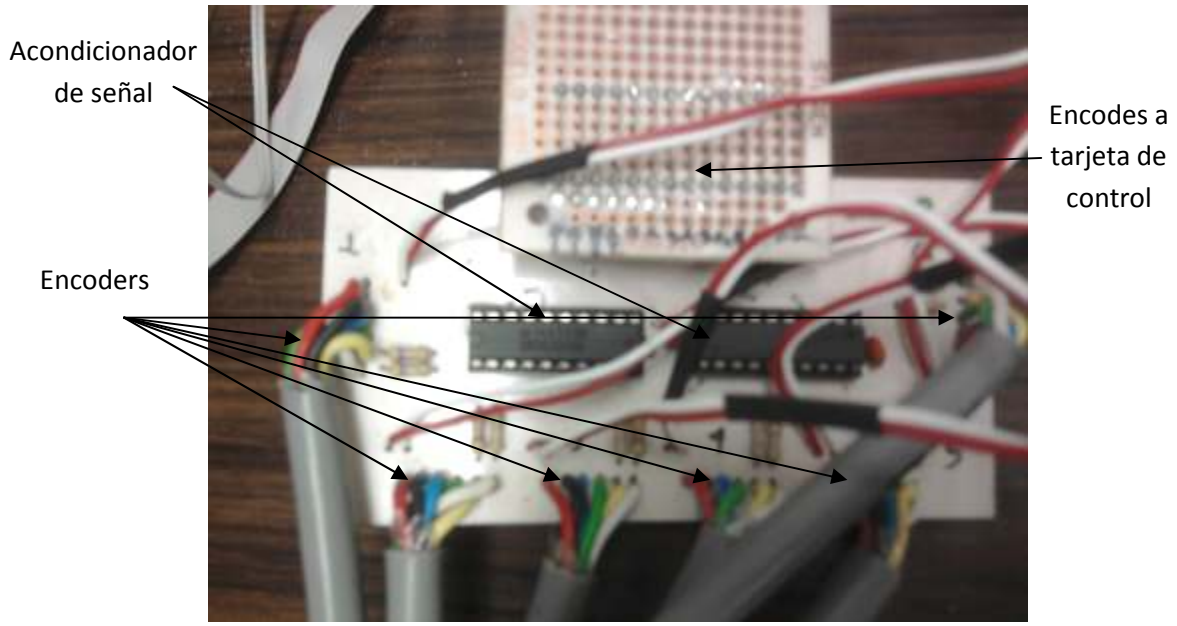


Figura 5.9 Tarjeta de acondicionamiento de encoders.

5.6 Tarjeta de control.

La tarjeta de control que se fabrico, contiene al microcontrolador maestro y también contiene 6 microcontroladores esclavos. La función del microcontrolador maestro es la de establecer la comunicación vía serial (RS232) con la computadora para el envió y recepción de datos, este microcontrolador envía y recibe datos de los 6 microcontroladores esclavos, en el apéndice C se muestra el código fuente. Los microcontroladores esclavos tienen la función de controlar un motor, cada esclavo tiene un control de posición.

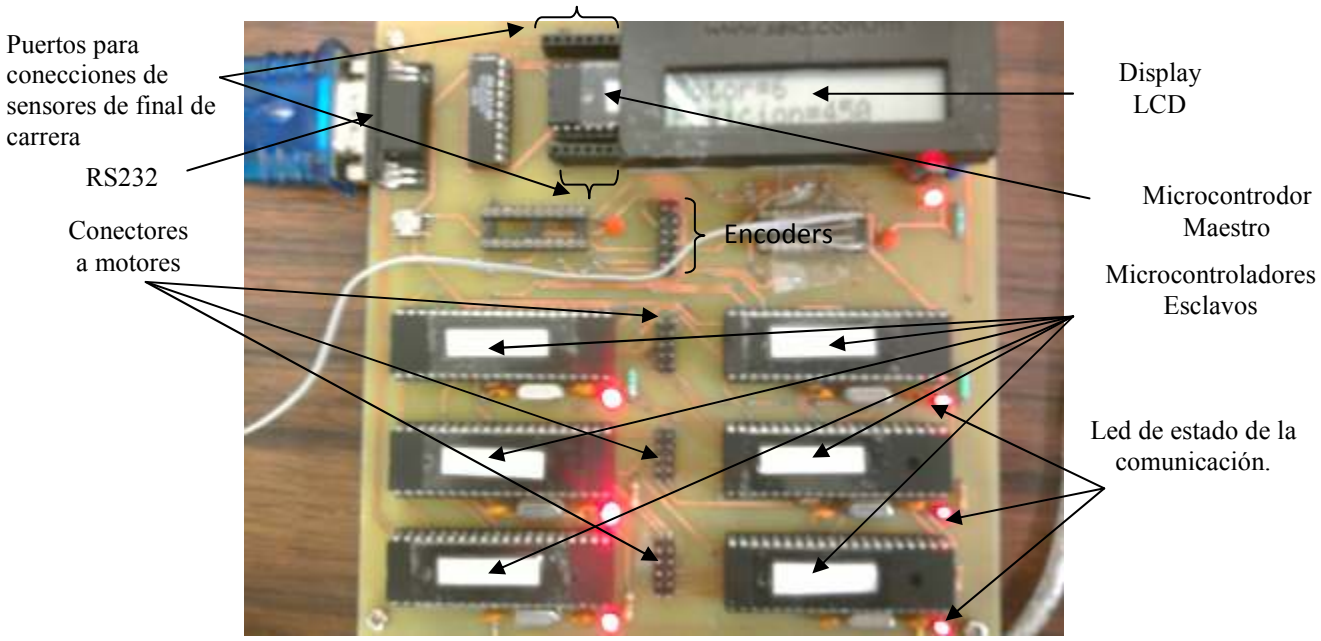


Figura 5.10 Partes de la tarjeta de control.

5.7 Etapas de potencia EMP-3010CD.

Para el control de los motores por medio de la etapa de potencia se deben mandar tres señales, para activar el motoreductor, una señal para el sentido de giro y por ultimo una señal de PWM.

Las etapas de potencia adquiridas tienen las siguientes características:

- Pueden trabajar con un voltaje mínimo de 3[V] hasta un voltaje máximo de 30[V].
- Soportan una corriente máxima de 10[A].
- Cuentan con un sensor de corriente el cual al calentarse los transistores comienzan los ventiladores a funcionar con el fin de reducir el calentamiento en la tarjeta.
- Cada tarjeta puede controlar dos motores
- Cuenta con un botón de reset.
- Para activar a los transistores la señal que envía el microcontrolador primero debe de pasar por un dispositivo electrónico. Para verificar si la información que se le manda es la correcta y activar al motor de una forma segura.
- Cuenta con bornes en los cuales se debe conectar el voltaje para que trabajen los motores.
- Cuenta con su propia fuente de 5[V], para activar los ventiladores y a los demás integrados.
- Debe alimentarse a la toma corriente de 127[V] de CA.
- Cuenta con 10 pines, los cuales están distribuidos de la siguiente manera.

ERROR A	P14
ENABLE A	P15
SENTIDO A	P16
PWM A	P17

ERROR B	P10
ENABLE B	P11
SENTIDO B	P12
PWM B	P13

PUERTO 1

GND	P16	P14	P12	P10
VCC	P17	P15	P13	P11

Figura 5.11 Conexiones de la etapa de potencia.

Para la activación de un motor es necesario que el microcontrolador envié tres señales:

1.-Señal de activación de motor (cero lógico).

2.-Señal de giro del motor (0-Giro en sentido horario, 1-giro en sentido anti horario).

3.-Señal PWM.

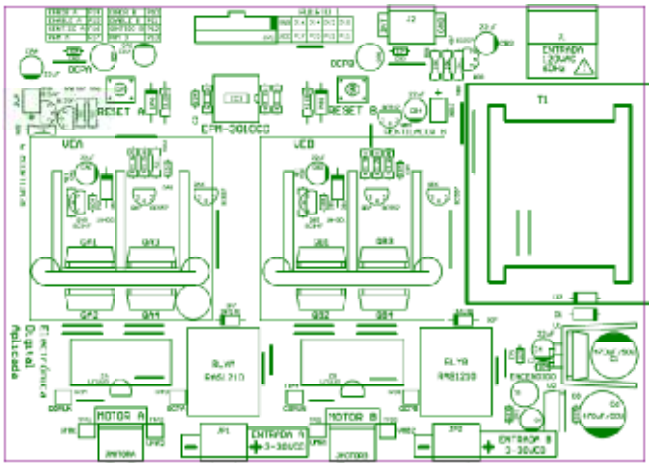


Figura 5.12 Diagrama de etapa de potencia.

5.8 Interfaz gráfica.

La interfaz gráfica fue diseñada en Matlab, esta tiene varias funciones y subprogramas que se detallan a continuación:

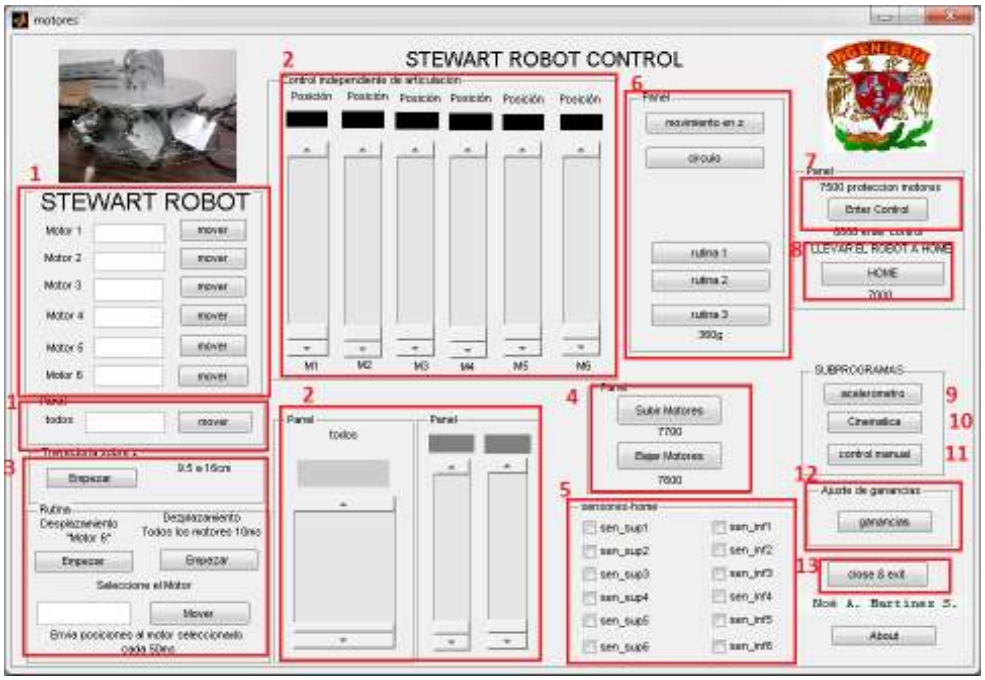


Figura 5.13 Interfaz gráfica de Matlab.

1. Esta parte de la interfaz gráfica tiene la función de controlar un motor de manera independiente, contiene una casilla en la cual se introduce un número dado en pulsos del encoder, para poder mover el motor correspondiente hay que pulsar el botón mover.
2. Esta parte es similar a la anterior con la diferencia de que el control es una barra deslizante con la cual se pueden mover los motores ya sea de manera independiente o todos los motores a la vez.
3. Rutinas. En esta parte se programaron una serie de rutinas con el fin de comprobar el funcionamiento independiente de cada articulación del robot, por ejemplo hay un botón que al pulsarlo genera un movimiento de la plataforma móvil sobre el eje z comenzando en 9.5cm y finalizando en 16cm, también hay un botón con una rutina la cual genera el movimiento de algún motor que se especifique.
4. Esta parte contiene dos botones, un botón para subir de manera manual todos los motores y otro para bajar todos los motores.
5. En este panel se visualizan todas las lecturas de los sensores de inicio de carrera y final de carrera que están colocados en las articulaciones, sen_sup(i) y sen_inf(i) donde i va de 1 a 6 y sen_sup indica el sensor superior de la articulación i y sen_inf indica el sensor inferior de la articulación i.
6. En este panel se programaron una serie de botones los cuales al presionarlos comenzaran a enviar datos con puntos de consigna previamente almacenados y calculados en la cinemática inversa.
7. Habilitación de control, este botón es utilizado después de que el robot llega a home para que pueda comenzar una nueva secuencia de movimientos y también es utilizado para reiniciar el control de posición de los motores enviando un comando que todos los microcontroladores esclavos interpretan como un reset.
8. Home. Cuando se presiona este botón se desactiva la protección de finales de carrera y se manda un dato a los microcontroladores esclavos deshabilitando el control principal y se entra a la rutina de home que llevara al robot a una posición inicial en este caso se selecciono el home cuando todos las articulaciones están contraídas.
9. Subprograma que abre una ventana donde se muestra la medida de posición de la plataforma móvil. Esta medida se hace por medio de un microcontrolador PIC18f4550 y del acelerómetro de Freescale MMA7341L, se obtiene una medida del convertidor analógico y se procesa para obtener el voltaje, el ángulo en radianes y el ángulo en grados de cada eje.

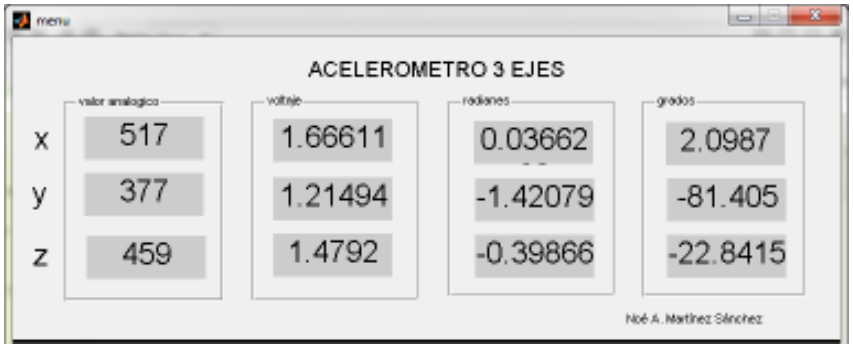


Figura 5.14 Subprograma lecturas de acelerómetro.

10. Subprograma que muestra una interfaz gráfica donde se resuelve la cinemática inversa del robot, en este programa se deben de escribir los parámetros físicos de la plataforma como radio de la base y el radio de la plataforma móvil, el punto a calcular en la cinemática inversa, pulsando el botón calcular dibuja el robot y muestra la longitud que debe de cumplir cada actuador para llegar a esa postura. También el subprograma tiene unos controles deslizantes para poder interactuar con el programa e ir viendo que postura adquiere el robot.

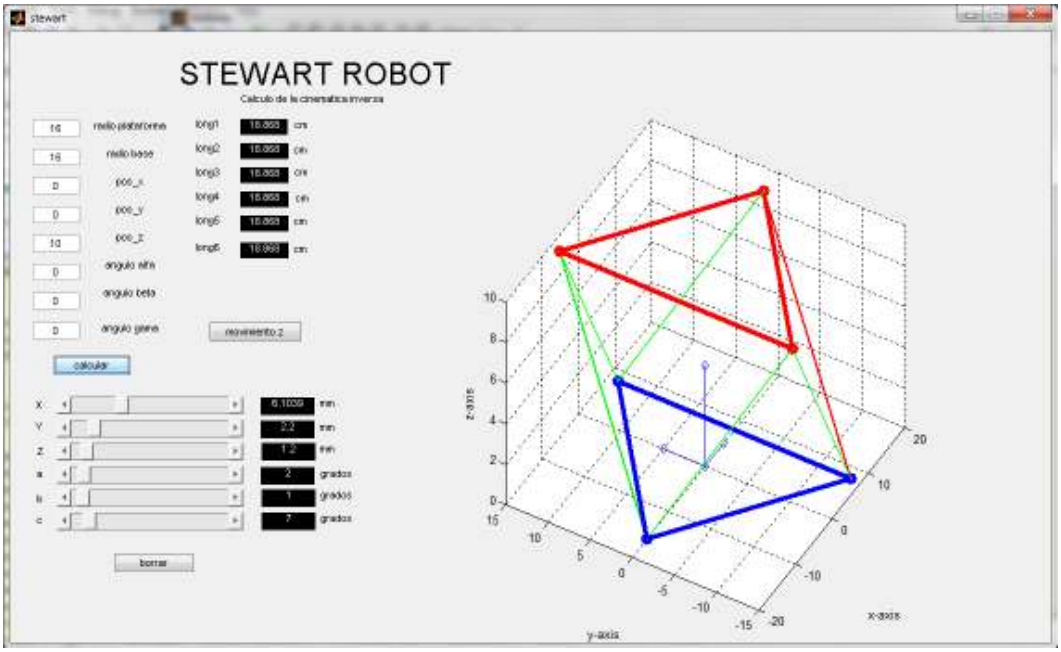


Figura 5.15 Subprograma cinemática inversa.

11. Control Manual. Contiene un subprograma el cual hace un control manual de cada articulación, esto con el fin de evitar estar moviendo cada actuador con la mano.

Figura 5.16 Subprograma control manual de motores.

12. Ajuste de ganancias. Este subprograma sirve para ajustar las ganancias del controlador PID de cada motor.

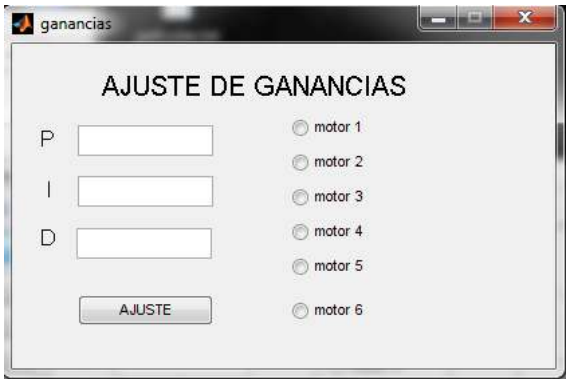


Figura 5.17 Subprograma ajuste de ganancias.

13. Finalizar la comunicación y salir del programa.

Nota: Todos los códigos de programación de la interfaz gráfica se encuentran en el apéndice C.

Capítulo 6

Simulación, Pruebas y resultados

Con el fin de validar el modelo cinemático inverso del robot, se programaron algunas trayectorias que demostraron que el mecanismo es capaz de reproducirlas. Las mediciones de las trayectorias en el plano fueron hechas con un sistema de visión llamado reacTIVision y los ángulos de inclinación de la plataforma superior con un acelerómetro.

La plataforma móvil del robot paralelo seguirá las siguientes trayectorias: línea recta, círculo, ocho, parábola.

Los aspectos a evaluar en la reproducción de trayectorias en el robot Stewart son la trayectoria trazada por el robot contra:

a) Trayectoria de referencia.
b) Error entre la trayectoria de referencia.
c) Error en x de la trayectoria de referencia.
d) Error en y de la trayectoria de referencia.
e) Error del ángulo de la trayectoria de referencia.

Es importante mencionar que las trayectorias se hacen en un tiempo lento debido a que no fue posible mandar los puntos de consigna a los microcontroladores esclavos a gran velocidad.

A continuación se muestran las simulaciones de las trayectorias y los resultados experimentales.

6.1 Línea recta.

6.1.1 Simulación línea recta.

En esta parte se muestra la simulación de la recta obtenida en Mathematica, se muestran los desplazamientos de los actuadores y el punto inicial y final del robot.

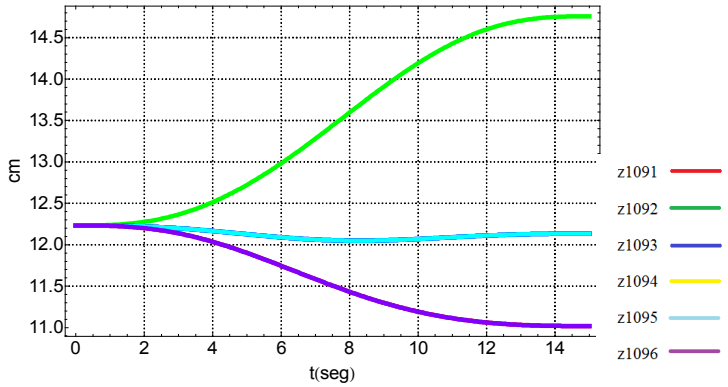
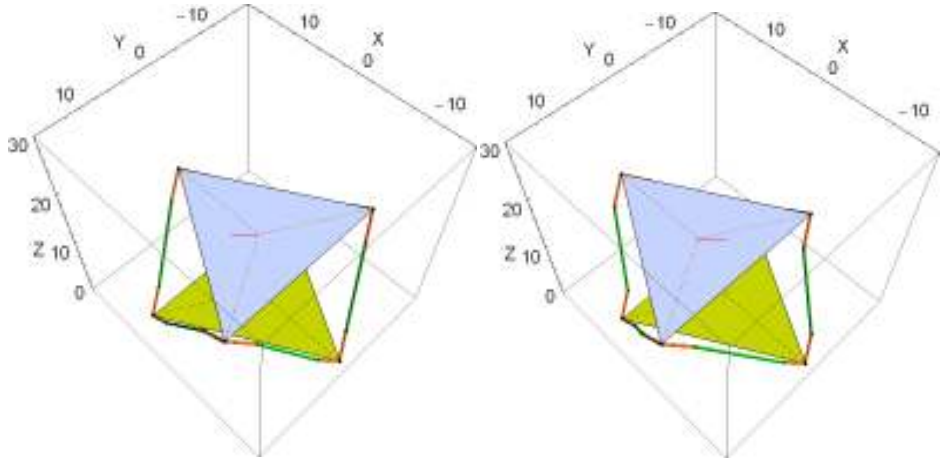


Figura 6.1 Desplazamientos de los actuadores para la línea recta

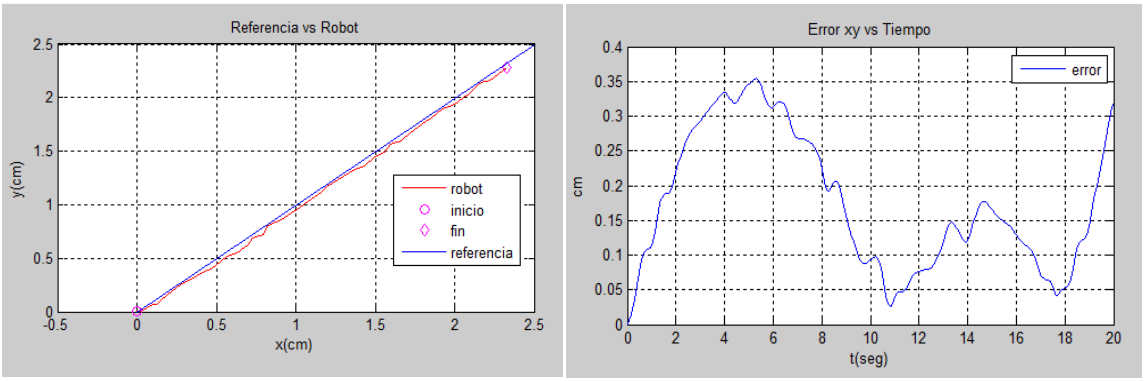


a) b)

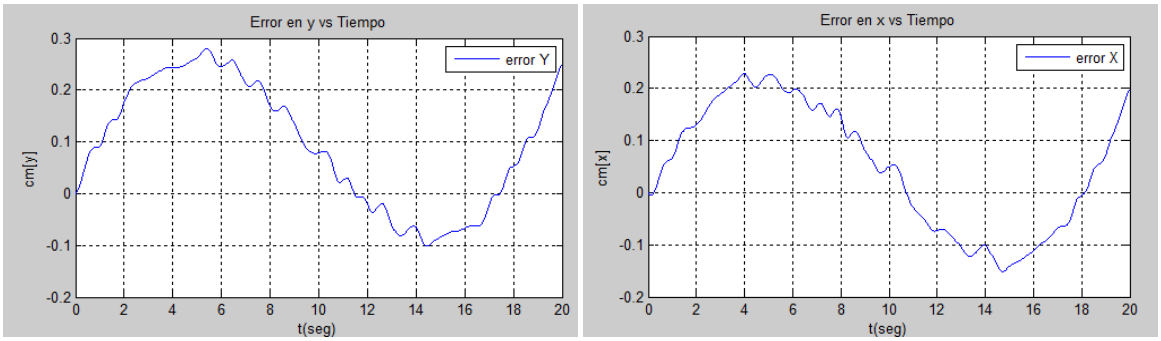
Figura 6.2 a) Posición inicial, b) Posición final.

Punto inicial pi={0,0,14} y punto final pf={2.5,2.5,14}, distancia a recorrer=3.5355, distancia recorrida=3.3868, con z=14cm.

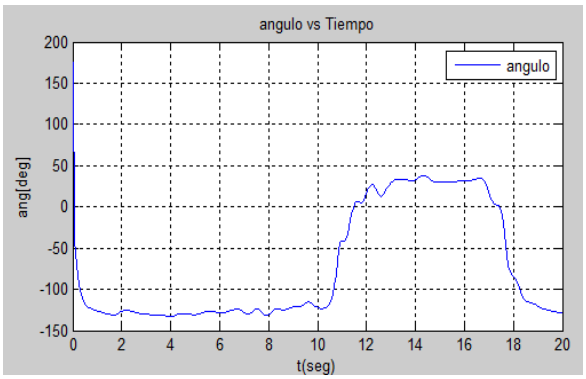
6.1.2 Resultados experimentales línea recta.



a) **b)**



c) **d)**



e)

Figura 6.2 a) Gráfica referencia vs Robot, b) Error xy vs tiempo, c) Error y vs tiempo, d) Error x vs tiempo, e) Ángulo vs tiempo.

6.2 Circulo.

La ecuación paramétrica que define el circulo es: x=cos(t)*2.5 y y=sin(t)*2.5, con z=14cm.

6.2.1 Simulación circulo.

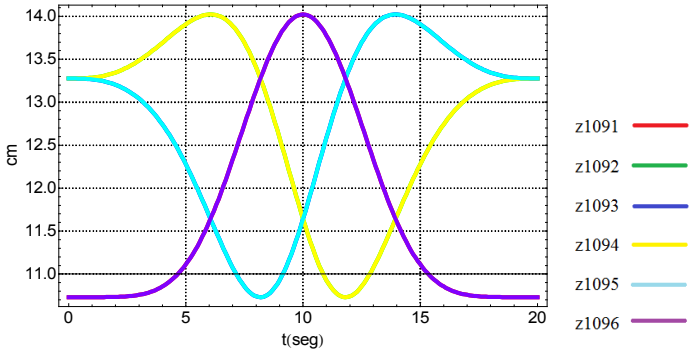


Figura 6.3 Desplazamientos de los actuadores para el círculo.

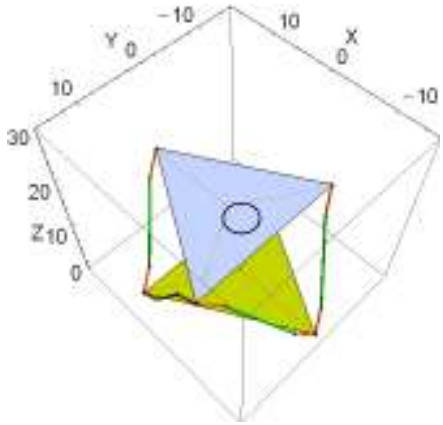
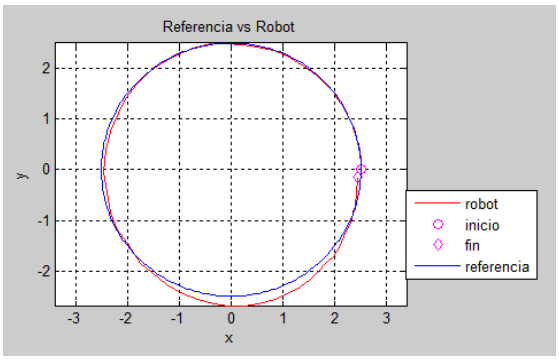
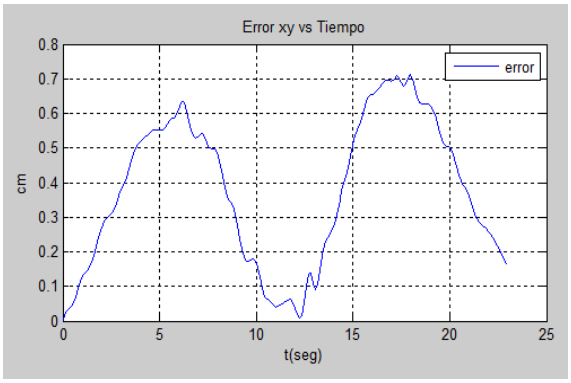


Figura 6.4 Simulación del círculo.

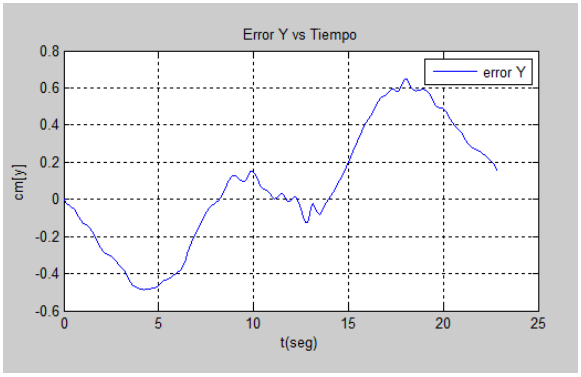
6.2.2 Resultados experimentales circulo.



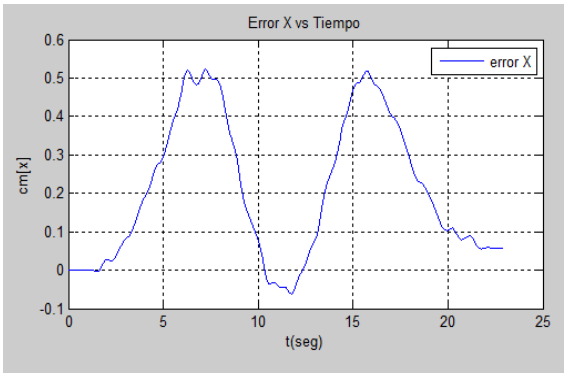
a)



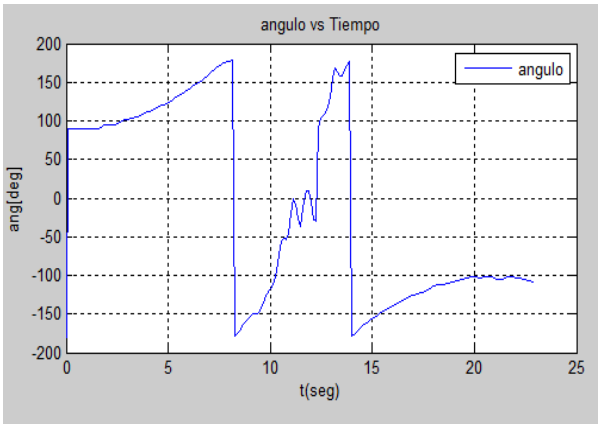
b)



c)



d)



e)

Figura 6.5 a) Gráfica referencia vs Robot, b) Error xy vs tiempo, c) Error y vs tiempo, d) Error x vs tiempo, e) Ángulo vs tiempo.

6.3 Ocho.

La ecuación paramétrica que define al ocho es: y=sin(t)*2.5 y x=sin(t*2)*2.5, con z=14cm.

6.3.1 Simulación ocho.

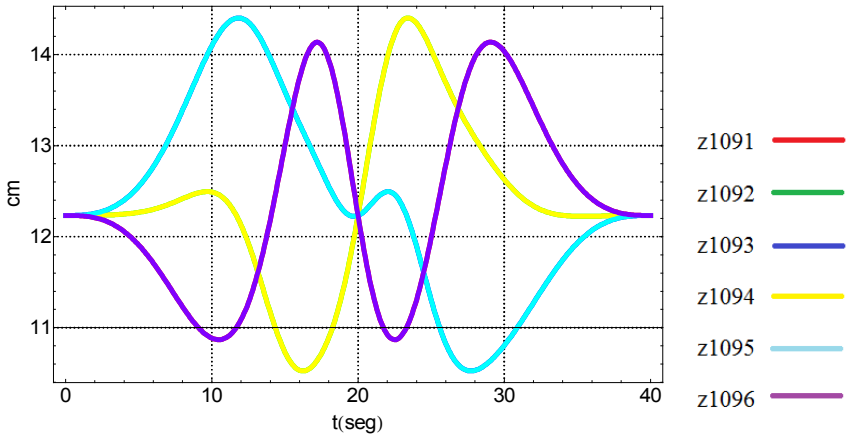


Figura 6.6 Desplazamiento de los actuadores para el ocho.

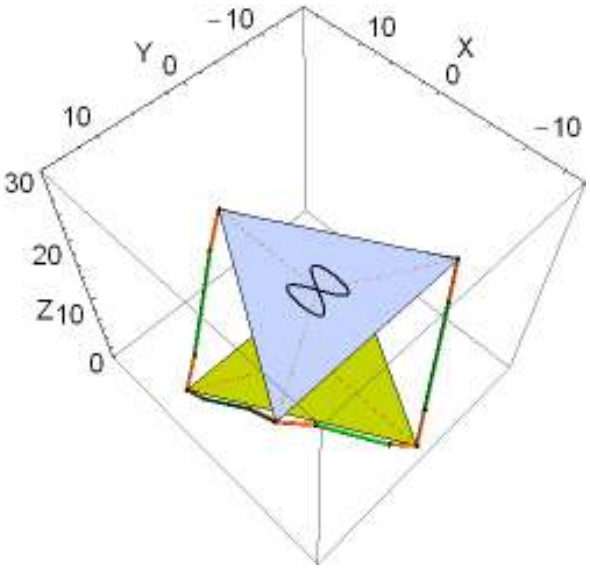
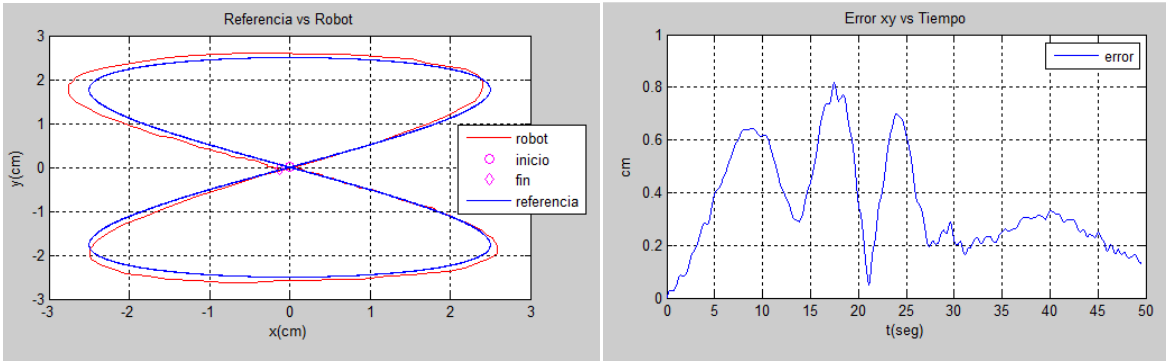
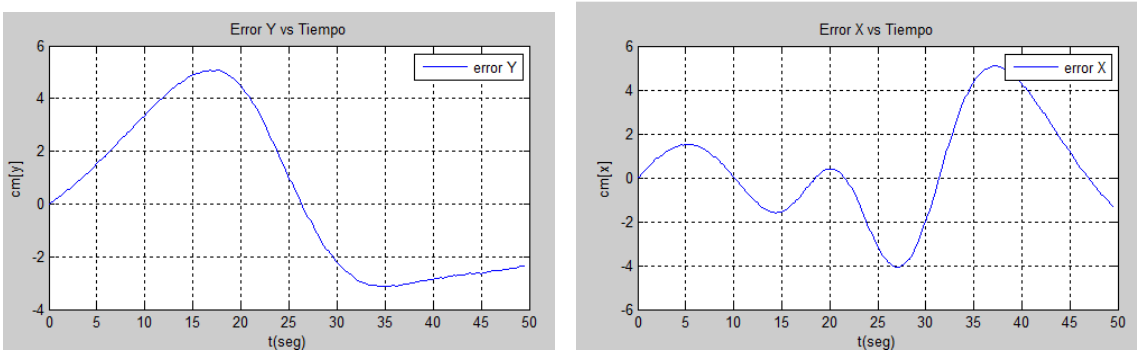


Figura 6.7 Simulación del ocho.

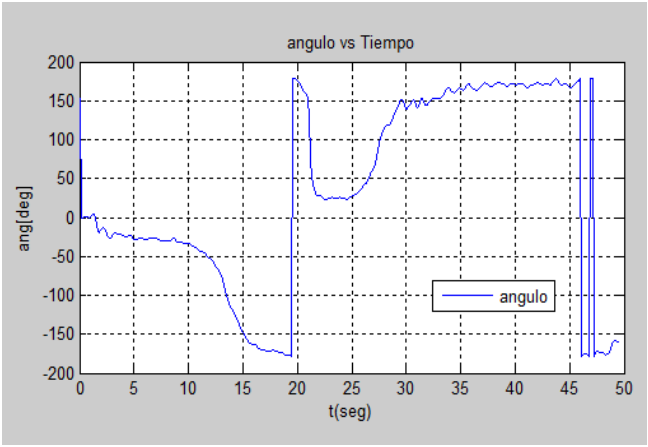
6.3.2 Resultados experimentales ocho.



a) **b)**



c) **d)**



e)

Figura 6.8 a) Gráfica referencia vs Robot, b) Error xy vs tiempo, c) Error y vs tiempo, d) Error x vs tiempo, e) Ángulo vs tiempo.

6.4 Parábola.

La ecuación paramétrica que define a la parábola es: x=y^2, con z=14.

6.4.1 Simulación Parábola.

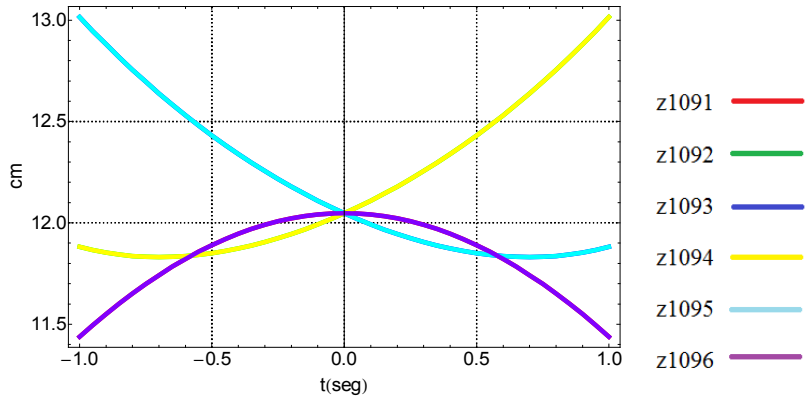


Figura 6.9 Desplazamiento de los actuadores para la parábola.

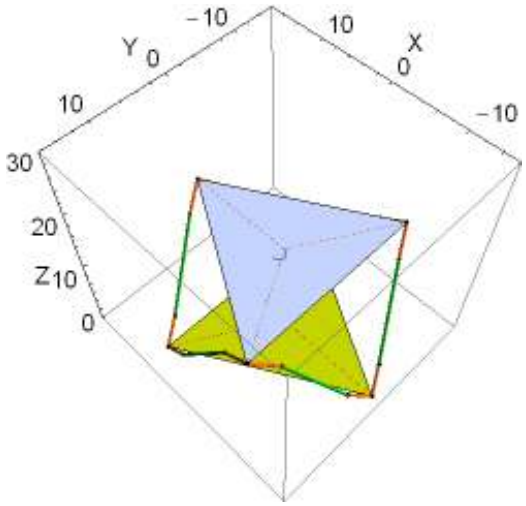
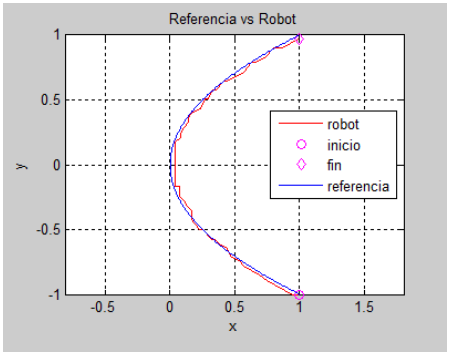
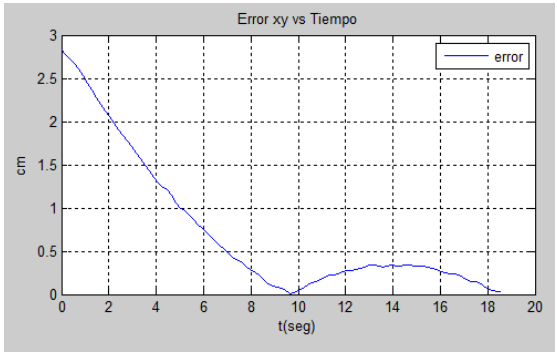


Figura 6.10 Simulación de la parábola.

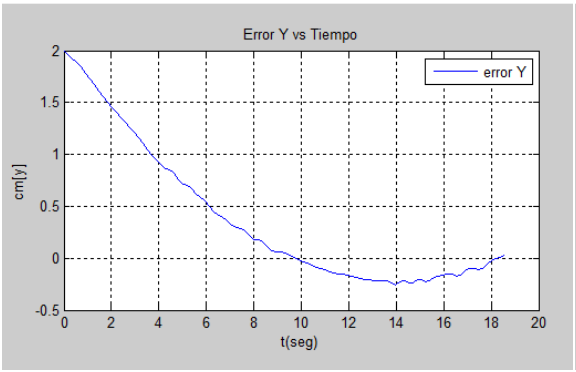
6.4.2 Resultados experimentales parábola.



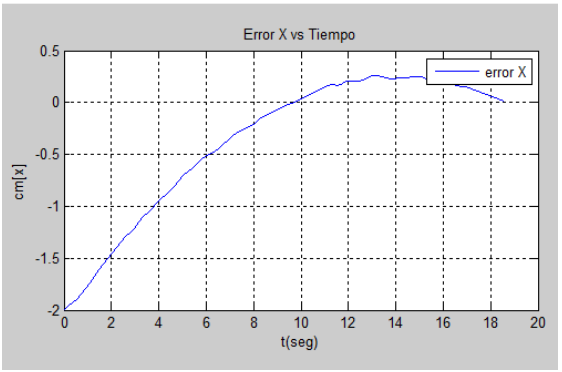
a)



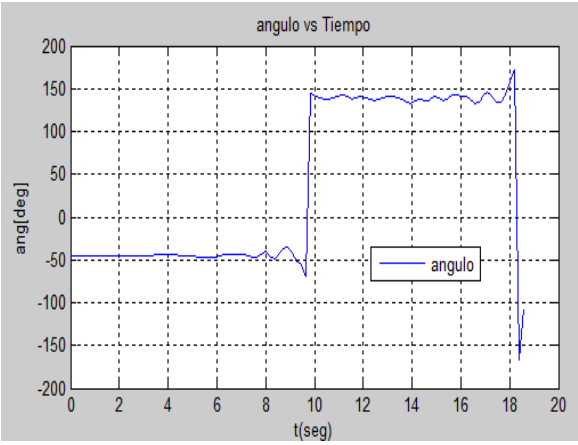
b)



c)



d)



e)

Figura 6.11 a) Gráfica referencia vs Robot, b) Error xy vs tiempo, c) Error y vs tiempo, d) Error x vs tiempo, e) Ángulo vs tiempo.

6.5 Inclinaciones de la plataforma móvil.

En esta prueba se lleva al robot a un ángulo de 15 grados sobre x y y z, se parte de la posición de home (todas las articulaciones contraídas) en la siguiente figura se muestran los resultados donde se puede apreciar que se aproximan los ángulos a su consigna dada.

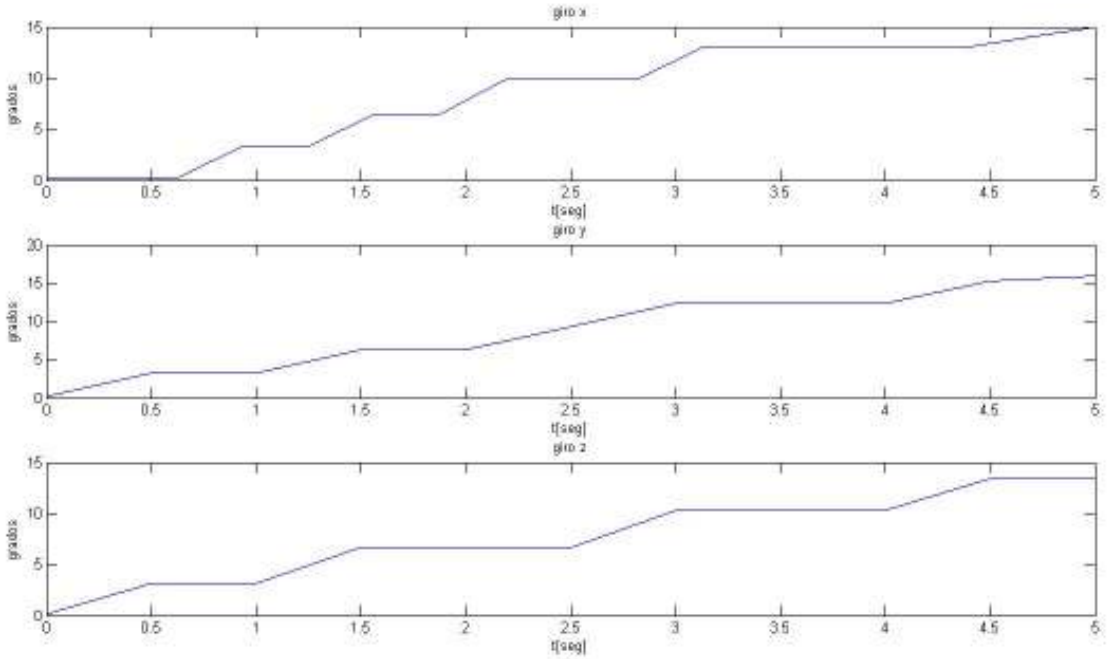


Figura 6.12 Inclinaciones en x,y,z.

6.6 Prueba Estabilidad.

Esta prueba se realizo en base al artículo presentado por José Antonio Souza [55] que también es tema de tesis de doctorado, en este trabajo se muestra la utilización de la plataforma Stewart como sistema de estabilización de una Plataforma móvil elevadora de Personal. En el artículo se evalúa la estabilidad que provee el robot Stewart a todo el sistema robótico.

El sistema propuesto está compuesto de un manipulador paralelo espacial (Plataforma Stewart 6-UPS), un brazo manipulador serial de 3 gdl y un robot móvil 3gdl conectados en serie. La estructura cinemática de dicho robot pretende realizar la tarea de una plataforma elevadora móvil de personas; la particularidad que posee, es la capacidad de cambiar su morfología.

Una plataforma elevadora móvil (PEM) es una maquina móvil destinada a desplazar personas hasta una posición de trabajo. Está constituida como mínimo por una plataforma de trabajo con órganos de servicio, una estructura extensible y un chasis. Existen plataformas sobre camión articuladas y telescópicas, autopropulsadas de tijera, autopropulsadas articuladas o telescópicas, y plataformas especiales remolcables entre otras.

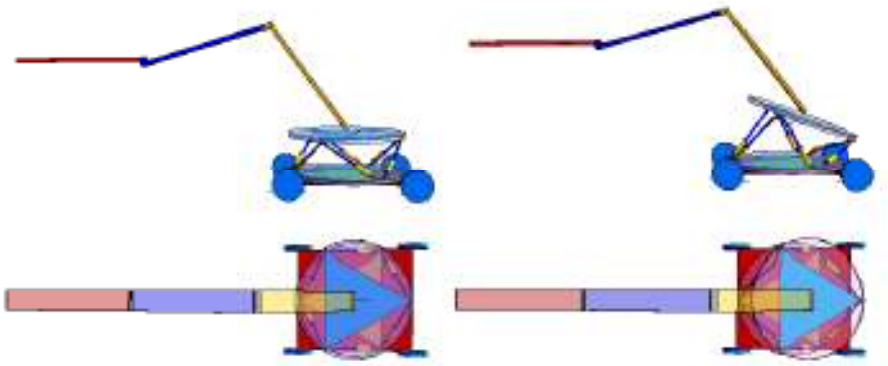
Dentro de los factores de riesgo que involucran el uso de la plataforma elevadora móvil se encuentran las caídas a distinto nivel que pueden ser debidas a: basculamiento de todo el sistema al estar situado sobre una superficie inclinada, el mal estado, la falta de estabilizadores, así como el vuelco del equipo debido a:

-Trabajos con el chasis situado sobre una superficie inclinada.
-Hundimiento o reblandecimiento de toda o parte de la superficie de apoyo del chasis.
-No utilizar estabilizadores, hacerlo de forma incorrecta, apoyarlos total o parcialmente sobre superficies poco resistentes.
-Sobrecarga de las plataformas de trabajo respecto a su resistencia máxima permitida.

Las plataformas móviles cuentan con estabilizadores que son todos aquellos dispositivos o sistemas concebidos para asegurar su estabilidad, dichos estabilizadores pueden ser: gatos, bloqueo de suspensión, ejes extensibles, etc. Sin embargo, actualmente no existe ninguna plataforma que tenga un mecanismo automático que le proporcione estabilidad.

La prueba que se hace es descrita a continuación:

Suponiendo que el órgano terminal de todo el sistema robótico se levanta para realizar una tarea en un punto arbitrario, entonces, el centro de gravedad de todo el sistema cambia y tiende a ser inestable o a una volcadura. Para reducir el riesgo de volcadura debe intervenir la plataforma Stewart modificando la longitud de sus actuadores para de esa manera mover el centro de gravedad del sistema a una zona segura de acuerdo al criterio de [55] y reducir el riesgo de volcadura. La plataforma Stewart se levantara como muestra la figura 6.13.



a) b)

Figura 6.13 Vista lateral y vista superior el brazo está orientado a 180 grados. a) Plataforma cerrada, b) Plataforma inclinada.

La prueba consiste en fijar el brazo en una posición arbitraria y girarlo 360 grados con respecto a su base, se escogen 12 posiciones espaciadas cada 30 grados, para cada una se grabo la posición de la plataforma Stewart que hace al sistema más estable.

Para saber la ubicación del brazo (ángulo), se utilizo el sistema de visión recTIVision, se colocó una cámara por encima del robot y una marca en la base donde gira el brazo figura 6.14, la marca es procesada por reacTIVision y se obtiene el ángulo girado. Para el control de la posición de la plataforma Stewart se utilizo Simulink que por medio de reacTIVision adquiere el ángulo del brazo y envía un dato al microcontrolador maestro que significa una postura en el robot Stewart correspondiente a una de las 12 posiciones. El dato enviado por Simulink y recibido por el microcontrolador maestro significa una postura en el robot. El microcontrolador maestro tiene grabadas todos los puntos de la cinemática inversa de cada una de las 12 posturas a adoptar. Cuando el maestro identifica un ángulo correspondiente al brazo, envía a todos los controladores de cada articulación las posiciones que debe de cumplir la plataforma Stewart para estabilizar el sistema.

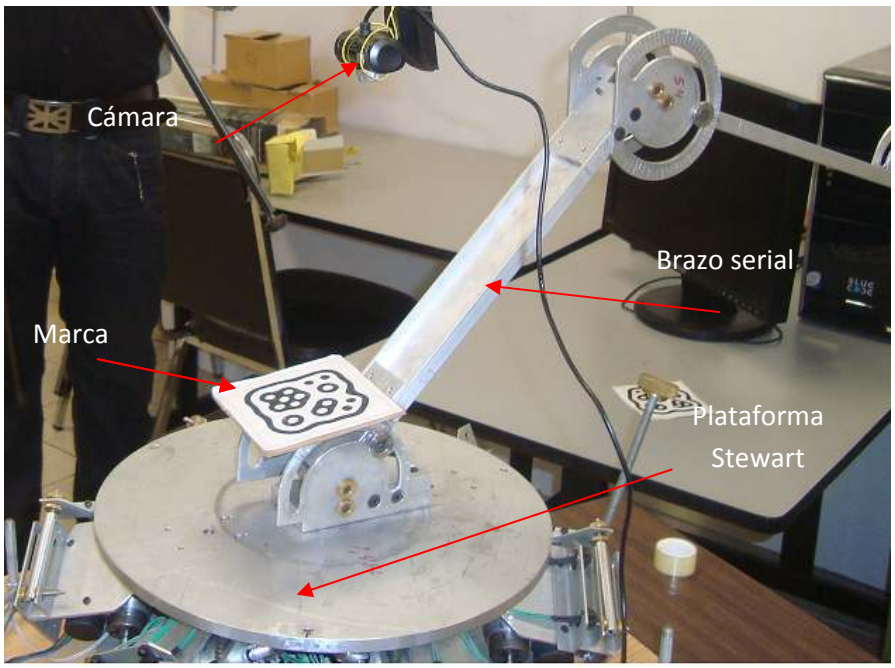


Figura 6.14 Colocación de la cámara y de la marca para la medición del ángulo.

El programa de Simulink se muestra en la siguiente imagen,

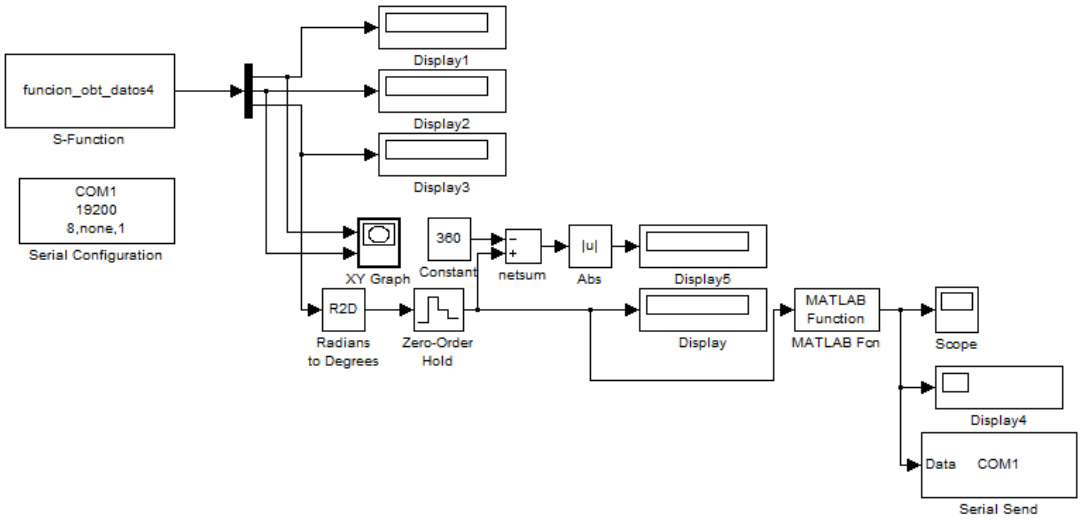


Figura 6.15 Programa Simulink.

Donde:
funcion_obt_datos4: es un script de Matlab que permite utilizar las funciones de Reactivision y obtener el ángulo.
Serial configuration: es la configuración del puerto seria sobre el cual se envían datos a la tarjeta de control.
Matlab fcn: es un scritp que toma la decisión de que dato enviar a la tarjeta dependiendo del ángulo medido.
Serial send: este bloque permite la salida del dato por el puerto serie.

El programa del microcontrolador maestro se encuentra en el apéndice D.

Conclusiones

La creciente demanda de los robots paralelos en sus diferentes áreas de aplicación, ha permitido que en la UNAM, académicos y estudiantes, se interesen en el estudio de estos robots con fin de generar conocimiento y solucionar problemas vigentes en este tipo de manipuladores. Esto se logra a partir de la construcción de prototipos como fue el caso del robot en estudio de esta tesis, el cual hace más fácil la adquisición del conocimiento de su funcionamiento y obtención de valores cuantitativos en base a la experimentación, para a si tener una mejor comprensión de este tipo de sistemas y lograr una buena formación profesional.

Por eso la necesidad de construir prototipos como es el caso del robot paralelo Stewart con la finalidad de enriquecer los conocimientos, buscando con esto el fortalecimiento en áreas de análisis cinemático, dinámico y sobre todo control de estos manipuladores.

En particular en esta tesis se logran reproducir trayectorias generadas por la cinemática inversa del robot, sin embargo no se reproducen fielmente, se tienen errores de seguimiento de un orden pequeño pero considerable esto debido a que el control del robot se hace a lazo abierto o en el espacio de juntas, de la calibración de los instrumentos de medición y a la inexactitud del mecanismo. Para poder asegurar que el robot siga la trayectoria y que el error sea mínimo será necesario hacer el control del robot en el espacio de trabajo considerando el modelo dinámico, hacer uso de la cinemática directa para cerrar el lazo de control o bien hacer el uso de sensores que midan directamente la posición y orientación del efector final.

El mecanismo del robot, en especial los actuadores, tienen algunos defectos mecánicos debido a que es el primer prototipo de este tipo de robot que se realiza. Sin embargo, pese a eso, es suficiente para poder trabajar con el prototipo. Estos defectos mecánicos producen no linealidades y una dinámica cambiante por lo cual no permite la generación de trayectorias de forma exacta y precisa.

Algo que se observo es que la comunicación de la computadora a la tarjeta de control y de la tarjeta de control a la computadora es lenta, esto debido a varios factores tales como: el puerto de comunicación, la computadora, las tareas que realizan el paralelo la computadora.

Se crea todo el hardware necesario para controlar al robot dando entrada para que personas interesadas en el estudio de los robots paralelos tengan la oportunidad de experimentar y realizar sus estudios y experimentos.

Se ha demostrado que pese a la brecha tecnológica que hay en el país es posible hacer estudios sobre temas muy interesantes como lo es la robótica y sobre todo temas que están desarrollándose actualmente en universidades y centros de investigación mundialmente.

Trabajo futuro

En base a las pruebas realizadas en el prototipo, se propone realizar otro prototipo que cubra las deficiencias que se mostraron en este robot, tales como el uso de actuadores lineales eléctricos comerciales, juntas esféricas y universales especiales.

Debido a que este tipo de robots tienen una alta dinámica se hace necesario el uso de controladores diseñados basados en el modelo dinámico una propuesta de este tipo de controladores seria el control robusto o el control adaptativo que hacen uso del modelo dinámico para su diseño.

Promover el uso de plataformas mas robustas de control, como lo son sistemas embebidos, el uso de DSPs y sistemas en tiempo real que sean fácilmente implementados en una estación de trabajo exclusivamente dedicada para eso. Esto permitiría dedicar más tiempo al análisis y a las pruebas que a la construcción de nuestra propia tecnología como tarjetas, etapas de potencia, etc.

Se hace necesario el estudio de las singularidades debido a que este tipo de robots tienen la particularidad de que la orientación y la posición están ligadas provocando singularidades en el espacio de trabajo. Hay muchos problemas abiertos por resolver y optimizar en cuanto a robots paralelos, se ha observado que no hay muchos trabajos en que se hagan pruebas experimentales sobre un robot, es decir, de que no se cuente con un prototipo en el cual poder experimentar y poder demostrar aspectos teóricos.

Apéndice A. Programa de control del PID discreto.

```
/***********************CONTROL DE POSICION*******************************/
/*                  NOE ALFREDO MARTINEZ SANCHEZ                         */
/*                        PID DIGITAL                                    */
/*                       MATERIA: CONTROL APLICADO                       */
/*                       DR. EDMUNDO ROCHA COZATL                        */
/*************************************************************************/
#include <16f877a.h>
#fuses HS,NOWDT,NOPROTECT,PUT,NOBROWNOUT,NOLVP
#use delay(clock=20000000)
#include <lcd.c>
#include <stdlib.h>
#include <math.h>
#use fast_io (a)
#use fast_io (b)
#use fixed_io(c_outputs=pin_c0,pin_c1,pin_c2)

int1 sentido=0;
#byte TRISA=0x85
#byte PORTA=0X05
#byte TRISB=0x86
#byte PORTB=0X06
/**************************************************************/

/**************************************************************/
#define MAX_PWM 1023 /* Maximo PWM (100%) del controlador */
#define MIN_PWM 70 /* Minimo PWM del controlador*/
float error=0;         //error positivo y negativo
signed int16 pos_real=0;        //posicion real
signed int16 set_point=500;    //posicion deseada
float u=0;                    //señal de control(duty pwm)
float kp=10;                //constante proporcional
float ki=50;
float kd=12;
float e_previ=0;        //error previo integral
float e_prevd=0;
float P_term=0;
float I_term=0;
float D_term=0;


/*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*/
#int_TIMER1
void TIMER1_isr(void)
{
error=set_point-pos_real;    // calcula el error
/*****sentido de giro****/
if(set_point==pos_real)
   {
output_high(PIN_C0);
output_high(PIN_C1);
output_high(PIN_A0);
   }
else if(error<0)
   {
output_high(PIN_C0);
output_low(PIN_C1);
output_high(PIN_A1);
   }
else if(error>0)
   {
```

```
output_low(PIN_C0);
output_high(PIN_C1);
output_high(PIN_A2);
   }
/************calculo del pid*******/
error=abs(error);      //valor absoluto del erro
P_term=kp*error;
I_term=ki*error;
I_term=I_term+e_previ;
D_term=kd*error-kd*e_prevd;

u=u+(P_term+I_Term+D_Term);
e_prevd=error;
e_previ=I_term;
u=floor(u);                  // hace un redondeo
u=abs(u);
set_pwm1_duty(u);            //manda salida del controlador
/*%%%%%%%%%%%delimita la salida del controlador%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*/
if(u >= MAX_PWM)
   {
u = MAX_PWM;
set_pwm1_duty(u);
   }
else if(u <= MIN_PWM)
   {
u = MIN_PWM;
set_pwm1_duty(u);       //a Duty de PWM
   }
else
   {
set_pwm1_duty(u);
   }
//  set_timer1(53036);    //   T= 20 MS
//    set_timer1(34286);   //tiempo de muestreo 50 ms
    set_timer1(64911);    //1ms
}

/*%%%%%%%%%%%%%%%%%conteo de encoder%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*/
#int_rb
void detect_rb_change() {
int8 current;
static int8 last=0;
current=PORTB;
/*__________________primer estado_____________________________________________*/
 if ((bit_test(current, 4)==1) && (bit_test(current,5)==0)){   /* 01 */
   if ((bit_test(last,4)) ^ (bit_test(current,5))){       // OR-EX
   pos_real++;                                        //sentido agujas del reloj
sentido=1;
   }
else {
pos_real--;                             //sentido contrario agujas del reloj
sentido=0;
}
 }
/*________________segundo estado______________________________________________*/
else if ((bit_test(current, 4)==0) && (bit_test(current,5)==0)){     /* 00 */
if ((bit_test(last,4)) ^ (bit_test(current,5))){
pos_real++;                                        //sentido agujas del reloj
sentido=1;
}
else {
pos_real--;                             //sentido contrario agujas del reloj
sentido=0;
```

```
}
}
/*_________________tercer_ estado_____________________________________________*/
else if ((bit_test(current, 4)==0) && (bit_test(current,5)==1)){     /* 10 */
if ((bit_test(last,4)) ^ (bit_test(current,5))){
pos_real++;                                          //sentido agujas del reloj
sentido=1;
}
else {
pos_real--;                                 //sentido contrario agujas del reloj
sentido=0;
}
}
/*_________________cuarto_ estado_____________________________________________*/
else if((bit_test(current, 4)==1) && (bit_test(current,5)==1)){     /* 11 */
if ((bit_test(last,4)) ^ (bit_test(current,5))){
pos_real++;                                          //sentido agujas del reloj
sentido=1;
}
else {
pos_real--;                                 //sentido contrario agujas del reloj
sentido=0;
}
 }
last=current;
/*if(sentido==1){
printf(lcd_putc,"\f____=%Ld,%Ld",pos_real,set_point);
printf(lcd_putc,"\n____=%Ld",error);}
else{
printf(lcd_putc,"\f____=%Ld,%Ld",pos_real,set_point);
printf(lcd_putc,"\n____=%Ld",error);}*/
   }

void main() {
set_tris_a(0x00);   //salida
set_tris_b(0x30);
  //setup_adc_ports(AN0);                                    //Canal 0 analógico
  //setup_adc(ADC_CLOCK_INTERNAL);    //Fuente de reloj RC
lcd_init();
port_b_pullups(true);
setup_ccp1(CCP_PWM);
setup_timer_1(T1_internal|T1_DIV_BY_8);
//    set_timer1(53036);    //2° MS
//  set_timer1(34286);   //tiempo de muestreo 50 ms
  set_timer1(64911);    // 1ms
    setup_timer_2(T2_DIV_BY_16,155,1);   //2k
  enable_interrupts(int_TIMER1);
  enable_interrupts(int_RB);    //Habilitación interrupción RB0
  enable_interrupts(global);     //Habilitación general

  do             //Bucle Infinito
{
PORTA=0x00;
printf(lcd_putc,"\n____=%Ld,%Ld",pos_real,set_point);
//printf(lcd_putc,"\f____=%Ld",set_point);
// printf(lcd_putc,"\n____=%Ld",pos_real);
}
while(1);
}
```

Apéndice B. Programa microcontrolador maestro

```
/*********************************************************************/
/*                          MICRO MAESTRO                            */
/*                NOE ALFREDO MARTINEZ SANCHEZ                       */
/*            Recibe las posiciones de los motores matlab            */
/*         imprime numero de motor y posicion                        */
/*         lee los sensores de final e inicio de carrera y los manda a matlab */
/*********************************************************************/

#include <16F877a.h>
#FUSES HS,NOWDT,NOPROTECT,PUT,NOBROWNOUT,NOLVP
#use delay(clock=20MHZ)
#use rs232(baud=19200, xmit=pin_c6, rcv=pin_c7, bits=8, parity=N)
#use i2c(MASTER, SDA=PIN_C4, SCL=PIN_C3, FAST, FORCE_SW)
#include <lcd.c>
#include <stdlib.h>

#use fast_io (a)
#use fast_io (b)

#byte TRISA=0x85
#byte PORTA=0X05
#byte TRISB=0x86
#byte PORTB=0X06

char motor=0;              // motor x
unsigned int8 hi=0;        //posicion alta motor
unsigned int8 lo=0;        //posicion baja motor
unsigned int16 posicion=0;
int8 sensores1=0;
int8 sensores2=0;

unsigned int8 pa=255;   // dato que desactiva los motores cuando estos llegan
unsigned int8 ra=0;     // a sus limites

void lectura()
{
//esta funcion sirve de proteccion si alguno de los motores se pasa del lim-sup o
lim-inf
//manda dato que apaga el motor
sensores1=PORTB;    //Lee los sensores
sensores2=PORTA;    //Lee los sensores
putc(sensores1);    //Envia la lectura a matlab
putc(sensores2);    //Envia la lectura a matlab
}

void sensores()
{
//esta funcion sirve de proteccion si alguno de los motores se pasa del lim-sup o
lim-inf
//manda dato que apaga el motor
sensores1=PORTB;    //Lee los sensores
sensores2=PORTA;    //Lee los sensores
putc(sensores1);    //Envia la lectura a matlab
putc(sensores2);    //Envia la lectura a matlab

if ((bit_test(sensores1, 0)==1)||(bit_test(sensores1, 6)==1))          // PARO
MOTOR1
      {
i2c_start();
       i2c_write(0xA0);       //direccion del esclavo+escritura
       i2c_write(pa);        //enviamos un dato
```

```
i2c_write(ra);
i2c_stop();
      }
else if ((bit_test(sensores1, 1)==1)||(bit_test(sensores1, 7)==1))    // PARO
MOTOR2
      {
i2c_start();
       i2c_write(0xB0);       //direccion del esclavo+escritura
       i2c_write(pa);        //enviamos un dato
i2c_write(ra);
i2c_stop();
      }
else if ((bit_test(sensores1, 2)==1)||(bit_test(sensores2, 0)==1))    // PARO
MOTOR3
      {
i2c_start();
       i2c_write(0xC0);       //direccion del esclavo+escritura
       i2c_write(pa);        //enviamos un dato
i2c_write(ra);
i2c_stop();
      }
else if ((bit_test(sensores1, 3)==1)||(bit_test(sensores2, 1)==1))    // PARO
MOTOR4
      {
i2c_start();
       i2c_write(0xD0);       //direccion del esclavo+escritura
       i2c_write(pa);        //enviamos un dato
i2c_write(ra);
i2c_stop();
      }
else if ((bit_test(sensores1, 4)==1)||(bit_test(sensores2, 2)==1))    // PARO
MOTOR5
      {
i2c_start();
       i2c_write(0xE0);       //direccion del esclavo+escritura
       i2c_write(pa);        //enviamos un dato
i2c_write(ra);
i2c_stop();
      }
else if ((bit_test(sensores1, 5)==1)||(bit_test(sensores2, 3)==1))    // PARO
MOTOR6
      {
i2c_start();
       i2c_write(0xF0);       //direccion del esclavo+escritura
       i2c_write(pa);        //enviamos un dato
i2c_write(ra);
i2c_stop();
      }
}
void ruthome()
{
sensores1=PORTB;    //Lee los sensores
sensores2=PORTA;    //Lee los sensores
/*****************************************************************************/
/*                    conexion de sensores                                   */
/**SENSORES DE INICIO DE CARRERA*******SENSORES DE FIN DE CARRERA*************/
/*   RB0=sen_inf_m1                *      RB6=sen_sup_m1                     */
/*   RB1=sen_inf_m2                *      RA7=sen_sup_m2                     */
/*   RB2=sen_inf_m3                *      RA0=sen_sup_m3                     */
/*   RB3=sen_inf_m4                *      RA1=sen_sup_m4                     */
/*   RB4=sen_ing_m5                *      RA2=sen_sup_m5                     */
/*   RB5=sen_inf_m6                *      RA3=sen_sup_m6                     */
/*****************************************************************************/
```

```
// el esclavo enviara los motores 100 pulsos hacia arriba y despues los mandara
// poco a poco hacia abajo hasta encontrar los sensores de abajo en donde
// reiniciara el contador en cero
if (bit_test(sensores1, 6)==1);

}

void motor1(){      //manda posicion a motor1
i2c_start();
i2c_write(0xA0);
i2c_write(hi);
i2c_write(lo);
i2c_stop();
}
void motor2(){      //manda posicion a motor2
i2c_start();
i2c_write(0xB0);
i2c_write(hi);
i2c_write(lo);
i2c_stop();
}
void motor3(){      //manda posicion a motor3
i2c_start();
i2c_write(0xC0);
i2c_write(hi);
i2c_write(lo);
i2c_stop();
}
void motor4(){      //manda posicion a motor4
i2c_start();
i2c_write(0xD0);
i2c_write(hi);
i2c_write(lo);
i2c_stop();
}
void motor5(){      //manda posicion a motor5
i2c_start();
i2c_write(0xE0);
i2c_write(hi);
i2c_write(lo);
i2c_stop();
}
void motor6(){      //manda posicion a motor6
i2c_start();
i2c_write(0xF0);
i2c_write(hi);
i2c_write(lo);
i2c_stop();
}
void home(){        //matlab manda el numero 7000 que desabilita el control
i2c_start();
i2c_write(0xA0);
i2c_write(hi);
i2c_write(lo);
i2c_start();
i2c_write(0xB0);
i2c_write(hi);
i2c_write(lo);
i2c_start();
i2c_write(0xC0);
i2c_write(hi);
i2c_write(lo);
i2c_start();
```

```
i2c_write(0xD0);
i2c_write(hi);
i2c_write(lo);
i2c_start();
i2c_write(0xE0);
i2c_write(hi);
i2c_write(lo);
i2c_start();
i2c_write(0xF0);
i2c_write(hi);
i2c_write(lo);
i2c_stop();
// el dato enviado, hace que el esclavo entre en una subrutina de movimiento de
// motores para llevar a home al robot
ruthome();    //rutina de sensores
}

#int_rda
void serial_isr()
{
motor=getc();  //numero de motor
lo=getc();
hi=getc();

switch(motor){
case 1:
motor1();
break;
case 2:
motor2();
break;
case 3:
motor3();
break;
case 4:
motor4();
break;
case 5:
motor5();
break;
case 6:
motor6();
break;
case 7:
home();
break;
default:
}
}
void main()
{
   set_tris_a(0xff);   //entrada
   set_tris_b(0xff);   //entrada
lcd_init();
enable_interrupts(int_rda);
enable_interrupts(global);

do{
lectura();             //lectura de los sensores y envio a matlab
posicion=make16(hi,lo);
printf(lcd_putc,"\f____Motor=%u",motor);
printf(lcd_putc,"\n____Posicion=%Lu",posicion);
delay_ms(100);
```

```
    }
while(true);
}
```

Apéndice C. Interfaz gráfica de Matlab.

Movimiento de un solo motor (interfaz gráfica 1)

```
% --- Executes on button press in mov1.
function mov1_Callback(hObject, eventdata, handles)
global s

i=str2double(get(handles.datm1,'String'));
fwrite(s,[1],'uint8')      %posicion
fwrite(s,[i],'int16')     %motor
```

Movimiento de todos los motores con el control deslizante (interfaz gráfica 2)

```
% --- Executes on slider movement.
function slider_todos_Callback(hObject, eventdata, handles)
global s

handles.slider_todos=get(hObject,'Value'); %Carga en handles.slider1 el valor del
slider1
handles.slider_todos=1.*handles.slider_todos;
handles.slider_todos=floor(handles.slider_todos);
set(handles.valor_todos,'String',handles.slider_todos);

fwrite(s,[1],'uint8')      %posicion
fwrite(s,[handles.slider_todos],'uint16')     %motor

fwrite(s,[2],'uint8')      %posicion
fwrite(s,[handles.slider_todos],'uint16')     %motor

fwrite(s,[3],'uint8')      %posicion
fwrite(s,[handles.slider_todos],'uint16')     %motor

fwrite(s,[4],'uint8')      %posicion
fwrite(s,[handles.slider_todos],'uint16')     %motor

fwrite(s,[5],'uint8')      %posicion
fwrite(s,[handles.slider_todos],'uint16')     %motor

fwrite(s,[6],'uint8')      %posicion
fwrite(s,[handles.slider_todos],'uint16')     %motor
```

Subrutinas (interfaz gráfica 3)

```
function pushbutton20_Callback(hObject, eventdata, handles)
global s
load movimiento_z
tic
for i=1:1080
 fwrite(s,[1],'uint8')
 fwrite(s,[movimiento_z(i)'],'uint16')
 fwrite(s,[2],'uint8')
 fwrite(s,[movimiento_z(i)'],'uint16')
 fwrite(s,[3],'uint8')
 fwrite(s,[movimiento_z(i)'],'uint16')
 fwrite(s,[4],'uint8')
 fwrite(s,[movimiento_z(i)'],'uint16')
```

```
 fwrite(s,[5],'uint8')
 fwrite(s,[movimiento_z(i)'],'uint16')
 fwrite(s,[6],'uint8')
 fwrite(s,[movimiento_z(i)'],'uint16')
 pause(0.001)
end
toc

% --- Executes on button press in pushbutton14.
function pushbutton14_Callback(hObject, eventdata, handles)

global s
load rutina

f=str2double(get(handles.num,'String'));

for i=1:1000
fwrite(s,[f],'uint8')      %posicion
fwrite(s,[rutina(i)],'uint16')     %motor
pause(0.04)
end
```

Subir y bajar motores (interfaz gráfica 4)

```
function motor_up_Callback(hObject, eventdata, handles)
global s
           fwrite(s,[1],'uint8')
           fwrite(s,[7700],'uint16')
           fwrite(s,[2],'uint8')
           fwrite(s,[7700],'uint16')
           fwrite(s,[3],'uint8')
           fwrite(s,[7700],'uint16')
           fwrite(s,[4],'uint8')
           fwrite(s,[7700],'uint16')
           fwrite(s,[5],'uint8')
           fwrite(s,[7700],'uint16')
           fwrite(s,[6],'uint8')
           fwrite(s,[7700],'uint16')

function motor_down_Callback(hObject, eventdata, handles)
global s
           fwrite(s,[1],'uint8')
           fwrite(s,[7600],'uint16')
           fwrite(s,[2],'uint8')
           fwrite(s,[7600],'uint16')
           fwrite(s,[3],'uint8')
           fwrite(s,[7600],'uint16')
           fwrite(s,[4],'uint8')
           fwrite(s,[7600],'uint16')
           fwrite(s,[5],'uint8')
           fwrite(s,[7600],'uint16')
           fwrite(s,[6],'uint8')
           fwrite(s,[7600],'uint16')
```

Lectura de sensores (interfaz gráfica 5)

```
%% TIMER EMPIEZA A REALIZAR TAREA ASIGNADA
function tiempo(hObject, eventdata) %(source,eventdata)
global s t portb porta handles1
% hace la lectura de los sensores y lectura de proteccion, si alguno llega
% al limite automaticamente se manda un dato que detiene a ese motor
   portb = fread(s,1,'uint8');
   porta = fread(s,1,'uint8');
```

```
    for i=1:6
        if (bitget(portb, i) == 1)
            set(handles1.(strcat('sen_sup',int2str(i))),'value',1);
            fwrite(s,[i],'uint8')                %motor(i)
            fwrite(s,[7500],'uint16')            %7500 dato de proteccion
        else
            set(handles1.(strcat('sen_sup',int2str(i))),'value',0);
        end
    end

    for j=7:8
        if (bitget(portb, j) == 1)
            set(handles1.(strcat('sen_inf',int2str(j-6))),'value',1);
            fwrite(s,(j-6),'uint8')                %motor(i)
            fwrite(s,[7500],'uint16')            %7500 dato de proteccion
        else
            set(handles1.(strcat('sen_inf',int2str(j-6))),'value',0);
        end
    end

    for k=1:4
       if (bitget(porta, k) == 1)
            set(handles1.(strcat('sen_inf',int2str(k+2))),'value',1);
            fwrite(s,(k+2),'uint8')                %motor(i)
            fwrite(s,[7500],'uint16')            %7500 dato de proteccion
        else
            set(handles1.(strcat('sen_inf',int2str(k+2))),'value',0);
        end
    end
```

Rutinas (interfaz gráfica 6)

Movimiento en z

```
% --- Executes on button press in pushbutton20.
function pushbutton20_Callback(hObject, eventdata, handles)
global s
load movimiento_z
tic
for i=1:1080
 fwrite(s,[1],'uint8')
 fwrite(s,[movimiento_z(i)'],'uint16')
 fwrite(s,[2],'uint8')
 fwrite(s,[movimiento_z(i)'],'uint16')
 fwrite(s,[3],'uint8')
 fwrite(s,[movimiento_z(i)'],'uint16')
 fwrite(s,[4],'uint8')
 fwrite(s,[movimiento_z(i)'],'uint16')
 fwrite(s,[5],'uint8')
 fwrite(s,[movimiento_z(i)'],'uint16')
 fwrite(s,[6],'uint8')
 fwrite(s,[movimiento_z(i)'],'uint16')
 pause(0.001)
end
toc
Circulo
% --- Executes on button press in pushbutton26.
function pushbutton26_Callback(hObject, eventdata, handles)
%trayectoria circulo sobre xy
global s

load long11
```

```
load long22
load long33
load long44
load long55
load long66

tic
for i=1:500
 fwrite(s,[1],'uint8')
 fwrite(s,[long11(i)'],'uint16')
 fwrite(s,[2],'uint8')
 fwrite(s,[long22(i)'],'uint16')
 fwrite(s,[3],'uint8')
 fwrite(s,[long33(i)'],'uint16')
 fwrite(s,[4],'uint8')
 fwrite(s,[long44(i)'],'uint16')
 fwrite(s,[5],'uint8')
 fwrite(s,[long55(i)'],'uint16')
 fwrite(s,[6],'uint8')
 fwrite(s,[long66(i)'],'uint16')
 pause(0.01)
end
toc
```

Inclinación de la plataforma

```
% --- Executes on button press in pushbutton27.
function pushbutton27_Callback(hObject, eventdata, handles)
global s

load z1
load z2
load z3
load z4
load z5
load z6
% nota round(z2(i)-10)
tic
for i=1:51
 fwrite(s,[1],'uint8')
 fwrite(s,[((round(((z1(i)-10))*10))*24)],'uint16')
 fwrite(s,[2],'uint8')
 fwrite(s,[((round(((z2(i)-10))*10))*24)],'uint16')
 fwrite(s,[5],'uint8')
 fwrite(s,[((round(((z3(i)-10))*10))*24)],'uint16')
 fwrite(s,[6],'uint8')
 fwrite(s,[((round(((z4(i)-10))*10))*24)],'uint16')
 fwrite(s,[3],'uint8')
 fwrite(s,[((round(((z5(i)-10))*10))*24)],'uint16')
 fwrite(s,[4],'uint8')
 fwrite(s,[((round(((z6(i)-10))*10))*24)],'uint16')
 pause(0.1)
end
toc
```

Habilitación del control o reset (interfaz gráfica 7) el dato 8000 es interpretado como habilitación de control en los microcontroladores esclavos.

```
function pushbutton18_Callback(hObject, eventdata, handles)
global s
% habilita timer1 del micro e inicializa la posicion real en cero
           fwrite(s,[1],'uint8')
           fwrite(s,[8000],'uint16')
```

```
            fwrite(s,[2],'uint8')
            fwrite(s,[8000],'uint16')
            fwrite(s,[3],'uint8')
            fwrite(s,[8000],'uint16')
            fwrite(s,[4],'uint8')
            fwrite(s,[8000],'uint16')
            fwrite(s,[5],'uint8')
            fwrite(s,[8000],'uint16')
            fwrite(s,[6],'uint8')
            fwrite(s,[8000],'uint16')
```

Home (interfaz gráfica 8)

```
% --- Executes on button press in home.
function home_Callback(hObject, eventdata, handles)
%% Activa la rutina de home y desactiva la proteccion de los motores
global s t h w

fwrite(s,[7],'uint8')          % este dato significa home
fwrite(s,[7000],'uint16')      % este dato es el que se manda a los esclavos
w=0;                           % variable que detiene el timer
stop(t)

%CREAMOS EL OBJETO TIMER Y SE ARRANCA (tiempo de muestreo del puerto serie)
h = timer('TimerFcn',@home, 'ExecutionMode', 'fixedSpacing' ,'BusyMode', 'queue'
,'Period', 0.4);
start (h);
```

Finaliza comunicación (interfaz gráfica 13)

```
% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
global s t h w
fclose(s);
stop(t);   %para timer de lectura sensores
if (w==0)
    stop(h);   %para timer de rutina home
end
delete(s)
clear s
disp('FINALIZA COMUNICACION')
close all
```

Apéndice D. Programa estabilidad.

```
/*****************************************************************************/
/*                          MICRO MAESTRO                                    */
/*                NOE ALFREDO MARTINEZ SANCHEZ                               */
/*    Recibe el angulo que manda simulink que lee reactivision y             */
/*posiciona a los 6 motores en alguna de las 12 posiciones segun sea el caso  */
/*****************************************************************************/

#include <16F877a.h>
#FUSES HS,NOWDT,NOPROTECT,PUT,NOBROWNOUT,NOLVP
#use delay(clock=20MHZ)
#use rs232(baud=19200, xmit=pin_c6, rcv=pin_c7, bits=8, parity=N)
#use i2c(MASTER, SDA=PIN_C4, SCL=PIN_C3, FAST, FORCE_SW)
#include <lcd.c>
#include <stdlib.h>

#use fast_io (a)
```

```
#use fast_io (b)

#byte TRISA=0x85
#byte PORTA=0X05
#byte TRISB=0x86
#byte PORTB=0X06

char motor=0;              // motor x
unsigned int8 hi=0;        //posicion alta motor
unsigned int8 lo=0;        //posicion baja motor
unsigned int16 posicion=0;


/*0 grados*/
int16 a1_0=1176;
int16 a2_0=0;
int16 a3_0=0;
int16 a4_0=1176;
int16 a5_0=1176;
int16 a6_0=1176;
/*30 grados*/
int16 a1_30=0;
int16 a2_30=0;
int16 a3_30=0;
int16 a4_30=1176;
int16 a5_30=1176;
int16 a6_30=1176;
/*60 grados*/
int16 a1_60=0;
int16 a2_60=0;
int16 a3_60=0;
int16 a4_60=1176;
int16 a5_60=1176;
int16 a6_60=0;
/*90 grados*/
int16 a1_90=0;
int16 a2_90=0;
int16 a3_90=1176;
int16 a4_90=1176;
int16 a5_90=1176;
int16 a6_90=0;
/*120 grados*/
int16 a1_120=0;
int16 a2_120=1176;
int16 a3_120=1176;
int16 a4_120=1176;
int16 a5_120=1176;
int16 a6_120=0;
/*150 grados*/
int16 a1_150=0;
int16 a2_150=1176;
int16 a3_150=1176;
int16 a4_150=1176;
int16 a5_150=0;
int16 a6_150=0;
/*180 grados*/
int16 a1_180=0;
int16 a2_180=1176;
int16 a3_180=1176;
int16 a4_180=0;
int16 a5_180=0;
int16 a6_180=0;
/*210 grados*/
```

```
int16 a1_210=1176;
int16 a2_210=1176;
int16 a3_210=1176;
int16 a4_210=0;
int16 a5_210=0;
int16 a6_210=0;
/*240 grados*/
int16 a1_240=1176;
int16 a2_240=1176;
int16 a3_240=1176;
int16 a4_240=0;
int16 a5_240=0;
int16 a6_240=1176;
/*270 grados*/
int16 a1_270=1176;
int16 a2_270=1176;
int16 a3_270=0;
int16 a4_270=0;
int16 a5_270=0;
int16 a6_270=1176;
/*300 grados*/
int16 a1_300=1176;
int16 a2_300=0;
int16 a3_300=0;
int16 a4_300=0;
int16 a5_300=0;
int16 a6_300=1176;
/*330 grados*/
int16 a1_330=1176;
int16 a2_330=0;
int16 a3_330=0;
int16 a4_330=0;
int16 a5_330=1176;
int16 a6_330=1176;

void escribir(int16 a,int16 b,int16 c,int16 d,int16 e,int16 f)
{
       i2c_start();
          i2c_write(0xA0);
          i2c_write(make8(a,1));
          i2c_write(make8(a,0));
       i2c_start();
          i2c_write(0xB0);
          i2c_write(make8(b,1));
          i2c_write(make8(b,0));
       i2c_start();
          i2c_write(0xC0);
          i2c_write(make8(c,1));
          i2c_write(make8(c,0));
       i2c_start();
          i2c_write(0xD0);
          i2c_write(make8(d,1));
          i2c_write(make8(d,0));
       i2c_start();
          i2c_write(0xE0);
          i2c_write(make8(e,1));
          i2c_write(make8(e,0));
       i2c_start();
          i2c_write(0xF0);
          i2c_write(make8(f,1));
          i2c_write(make8(f,0));
       i2c_stop();
}
```

```
#int_rda
void serial_isr()
{
   motor=getc();  //numero de motor
   lo=getc();
   hi=getc();
   posicion=make16(hi,lo);

 switch(posicion){
   case 16368:       // para 0
         escribir(a1_0,a2_0,a3_0,a4_0,a5_0,a6_0);
         break;

   case 16384:        //para 30
      escribir(a1_30,a2_30,a3_30,a4_30,a5_30,a6_30);
         break;

   case 16392:             //para 60
      escribir(a1_60,a2_60,a3_60,a4_60,a5_60,a6_60);
         break;

   case 16400:           //para 90
       escribir(a1_90,a2_90,a3_90,a4_90,a5_90,a6_90);
         break;

   case 16404:        //para 120
       escribir(a1_120,a2_120,a3_120,a4_120,a5_120,a6_120);
         break;

    case 16408:      //para 150
       escribir(a1_150,a2_150,a3_150,a4_150,a5_150,a6_150);
         break;

   case 16412:    //para 180
       escribir(a1_180,a2_180,a3_180,a4_180,a5_180,a6_180);
         break;

   case 16416:     //para 210
       escribir(a1_210,a2_210,a3_210,a4_210,a5_210,a6_210);
         break;

   case 16418:    //para 240
   escribir(a1_240,a2_240,a3_240,a4_240,a5_240,a6_240);
         break;

case 16420:      //para 270
       escribir(a1_270,a2_270,a3_270,a4_270,a5_270,a6_270);
         break;

   case 16422:     //para 300
       escribir(a1_300,a2_300,a3_300,a4_300,a5_300,a6_300);
         break;

    case 16424:  //para 330
      escribir(a1_330,a2_330,a3_330,a4_330,a5_330,a6_330);
         break;
  default:
}
}
void main()
{
   set_tris_a(0xff);   //entrada
   set_tris_b(0xff);   //entrada
```

```
   lcd_init();
   enable_interrupts(int_rda);
   enable_interrupts(global);

do{

   printf(lcd_putc,"\f____Motor=%u",motor);
   printf(lcd_putc,"\n____Posicion=%Lu",posicion);
   delay_ms(100);
    }
    while(true);
}
```

Apéndice E. Tarjetas electrónicas.

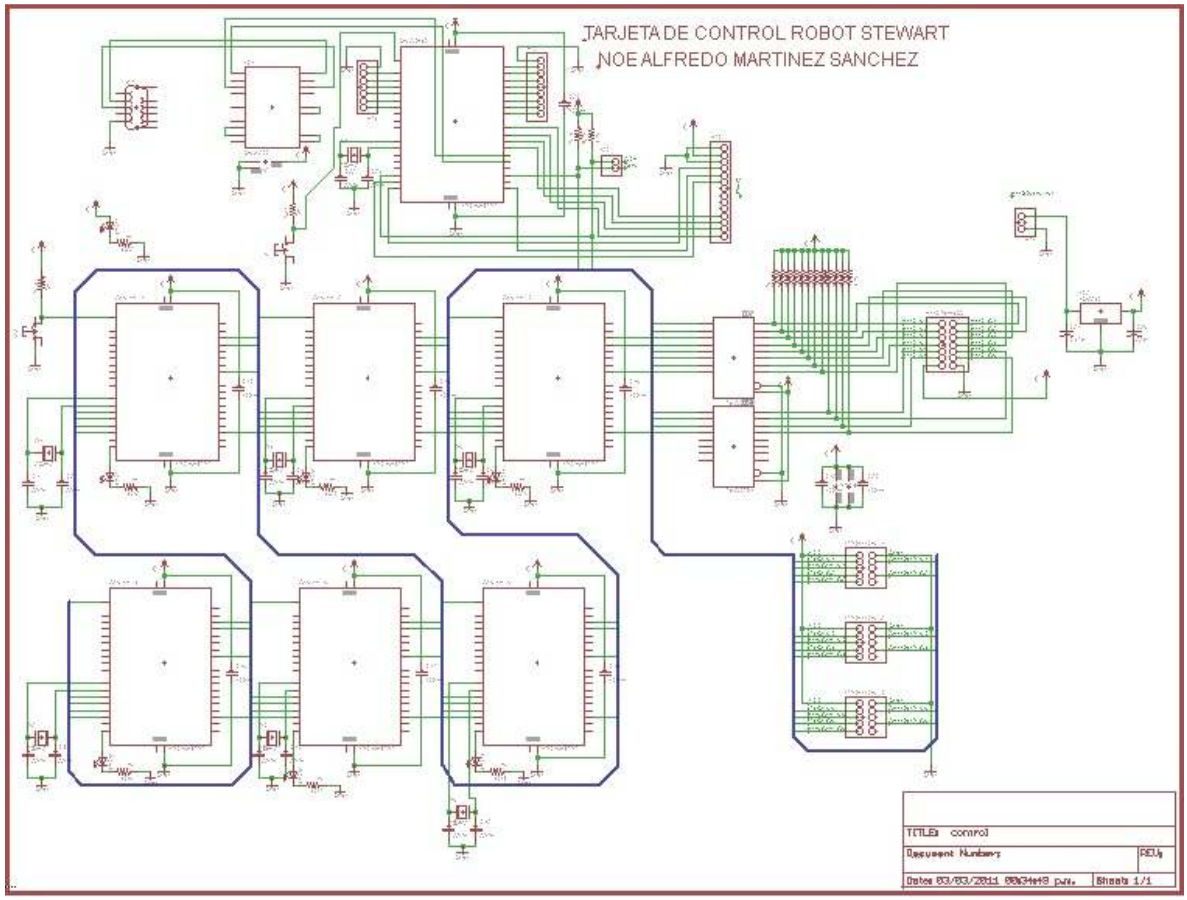


Figura E1. Diagrama esquemático de la tarjeta de control.

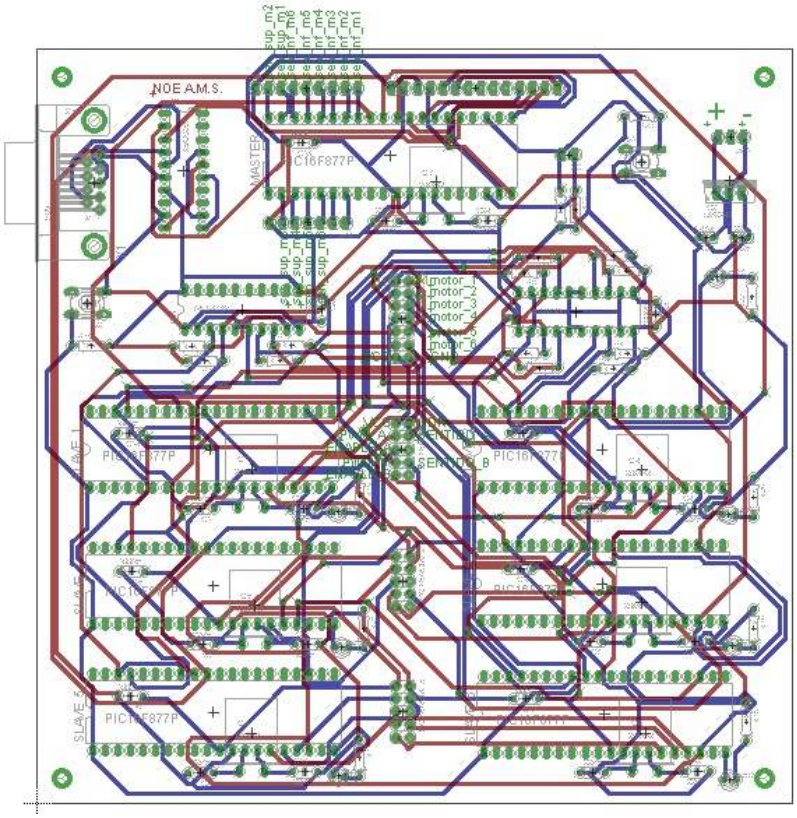


Figura E2. PCB de la tarjeta de control.

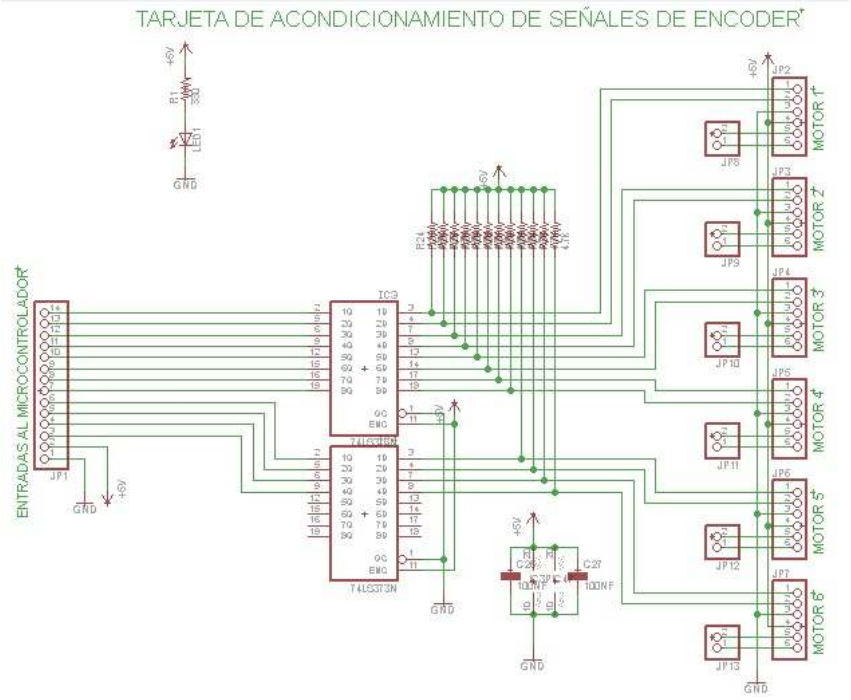


Figura E3. Esquemático de la tarjeta de acondicionamiento de encoders.

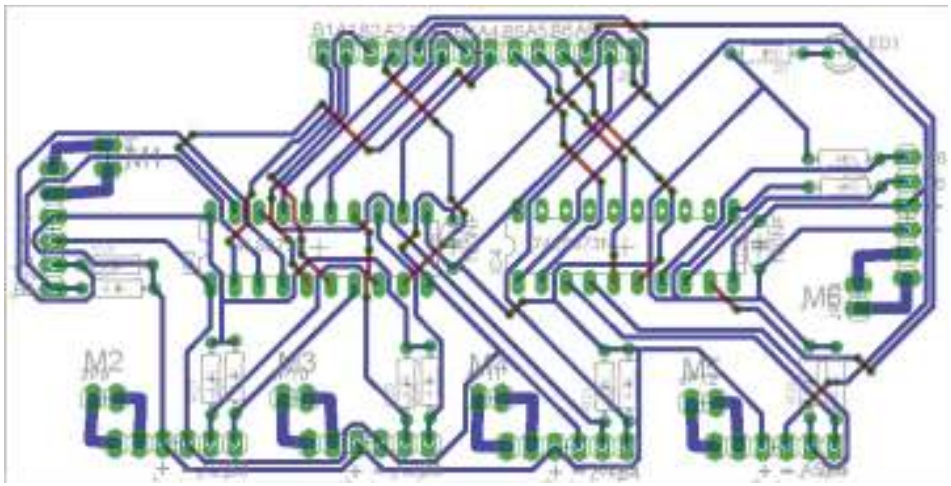


Figura E4. PCB de la tarjeta de acondicionamiento de encoders.

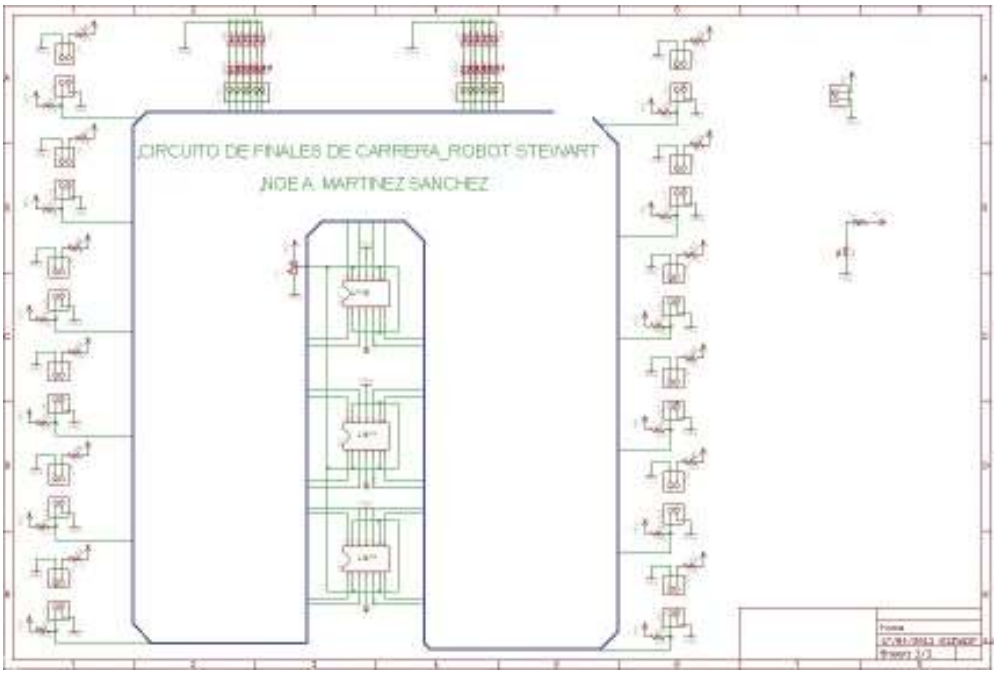


Figura E5. Esquemático de la tarjeta de lectura de sensores de final de carrera.

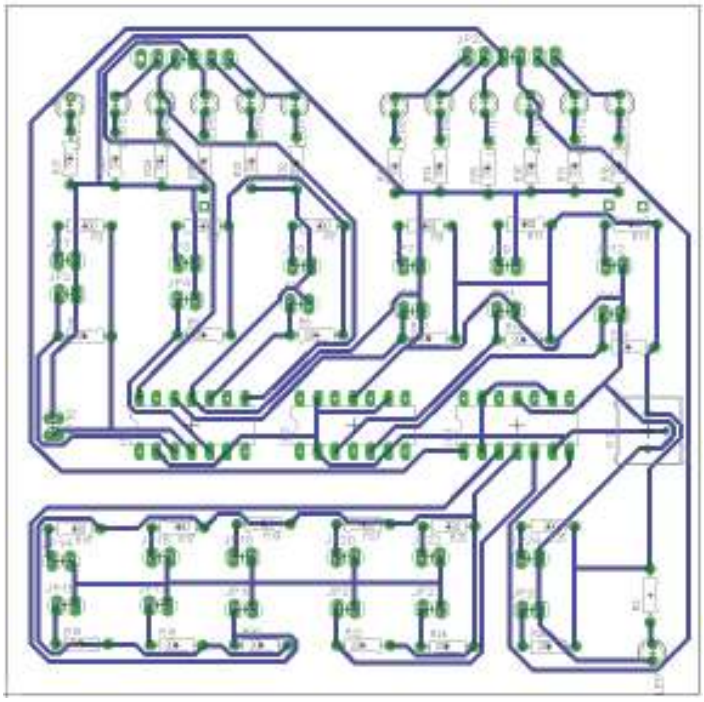


Figura E6. Pcb de la tarjeta de lectura de sensores de final de carrera.

Bibliografía.

[1] Merlet, J.P. (1994). Parallel manipulators: state of the art and perspective, Advanced Robotics, 8 (6), pp. 589-596.

[2] Merlet, J.P. (1997). Les robots parallèles, Paris :Editions Hermès.

[3] Tsai, Lung-Wen. (1999). Robot Analysis: The mechanic of Serial and Parallel Manipulators, John Wiley
& Sons, Inc.

[4] Angeles, J.(2002). Fundamentals of Robotic Mechanical ystems. Theory, Methods and Algorithms, Sppringer.

[5] Craig, John J. (2006). Robótica, Pearson educación.

[6] Slotine ,J.J.E and Li, W. 1991, Applied Nonlinear Control, Prentice-Hall

[7] Hassan K. Khalil. (2002). Nonlinear Systems Third Edition Prentice Hall.

[8] J.P. Merlet.(2000). Parallel Robot. INRIA, Sophia-Antipolis, France, Springer.

[9] FU, K. S. - GONZALEZ, R. C. Robótica: Control, detección, visión e inteligencia (1994), McGraw Hill

[10] Mikell Groover, Mitchell Weiss (1989). ROBOTICA INDUSTRIAL - Tecnología, Programación y Aplicaciones. Editorial Mc Graw Hill.

[11]W. Bolton (2007). Mecatrónica, Sistemas de control Electrónico en la ingeniería mecánica y eléctrica, Alfaomega.

[12] Katsuhiko Ogata (1993). Ingeniería de control moderna, Prentice Hall.

[13] Norman S. Nise (2009). Sistemas de control para ingeniería, Grupo editorial Patria.

[14]Richard L. Burden, J. Douglas Faires (2002). Análisis Numérico, CENAGE Learning.

[15] José Briceño Márquez "Principios de las comunicaciones" Mérida 2005

[16] Katsuhiko Ogata, Sistemas de control en tiempo discreto, Segunda Edición, Pearson Educación 1996

[17] Benjamin C. Kuo, Sistemas de Control Automático, Séptima Edición, Prentice Hall, México 1996.

[18] R. Canales Ruiz, R. Barrera Rivera, Análisis de Sistemas Dinámicos y Control Automático, Editorial Limusa, México, 1976.

[19] Ilian Bonev, El verdadero origen de robots paralelos

[20] García Breijo Eduardo, Compilador C CCS Y simulador Proteus para Microcontroladores PIC, Alfa Omega, Primera edición México 2008.

Tesis

[21] Jorge Enrique González Almanza (2006). Análisis Cinemático y Balanceo de una Plataforma Paralela de 5GDL. Tesis de Maestría Área: Mecánica. UNAM.

[22] Shair Mendoza Flores (2006). Análisis Cinemático y Dinámico de un Robot Delta de 3 Grados de Libertad. Tesis de Maestría Área: Mecánica Aplicada. UNAM.

[23] Guadalupe Dalila García Gálvez (2008). Análisis Cinemático y Dinámico de un Robot Paralelo tipo Diamante. Tesis de Maestría Área: Mecatrónica. UNAM.

[24] Jorge Díaz Velázquez (2007). Análisis Estático de una Plataforma de 6DOF. Tesis de Maestría Área: Mecánica aplicada. UNAM.

[25] Martin Ortega Breña (2005). Diseño Mecánico de un Robot Paralelo Delta de tres Grados de Libertad. Tesis de Maestría Área: Mecánica. UNAM.

[26] Pedro Agustín Ojeda Escoto (2006). Modelación y Simulación Cinemática de un Robot Delta Planar Tipo RR. Tesis de Maestría Área: Diseño Mecánico. UNAM.

[27] Patricio Martínez Zamudio (2009). Análisis Cinemático de un Manipulador Paralelo Hibrido tipo Delta. Tesis de Maestra Área: Mecatrónica. UNAM.

[28] Juan Sánchez Cano (2005). Balanceo Estático de un Mecanismo Espacial de Paralelogramo. Tesis de Maestría Área: Mecánica. UNAM.

[29] Francisco Cuenca Jiménez (2008). Análisis Cinemático y Dinámico de un Robot Paralelo Espacial RUS. Tesis de Doctorado Área. Mecánica Aplicada. UNAM.

[30] Juan Antonio Briones León (2009). Diseño Análisis y Construcción de un Robot Paralelo Transnacional. Tesis de Maestría en Tecnología Avanzada. (IPN)

Artículos

[31] Chin-I Huang and Li-Chen Fu. Adaptative Backstepping Tracking Control of the Stewart Platform. Conference on Decision and Control IEEE 2004.

[32] J.M. Rosario, E. Oliveira, D. Dumur. Conception if stewart-Gough Platform with Reconfigurable Control. International Symposium on Power Electronics, Electrical Drives, Automation and Motion, 2006.

[33] Stephen P. Tseng, S. S. Hu, and J.S Shaw. Control and Motion Cues Generation of Stewart Platform whit a DSP Based Controller. International Confefrence on Mechatronics IEEE 2005.

[34] Se-Han Lee, Jae-Bok Song, Woo-Chun Choi, Daehie Hong. Controller Design for Stewart Platform Using Small Workspace Characteristics. International Conference on Intelligent Robots and Systems IEEE 2001.

[35] Indrawanto, Anindito Santoso. Desing and Control of the Stewart Platform Robot. International Conference on Modeling and Simulation. IEEE 2009.

[36] Shahrak Esmaeil, Bak Thomas, Zamanabadi Roozbeh. Model Predictive Controller Combined whit LQG Controller and Velocity Feedback to Control the Stewart Platform. IEEE 2006.

[37] Chiew, Y. S., Abdul Jalil, M. K. and Hussein, M. Motion Cues Visualization of a Motion Base for Driving Simulator. International Conference on Robotics and Biomimetics, IEEE 2008.

[38] Gao Meng, li Tiemin, Yin Wensheng, Calibration Method and Experiment of Stewart Platform Using Laser Tracker. IEE 2003.

[39] Sung-Hua Chen and Li-Chen Fu, Output Feedback Control with a Nonlinear Observer Based Forward Kinematics Solution of a Stewart Platform. IEEE 2008.

[40] Xiguang Huang, Qizheng Liao, Shimin Wei, Xu Qiang and Shuguang Huang. Forward Kinematics of the 6-6 Stewart Platform with Planar Base and Platform Using Algebraic Elimination. International Conference on Automation and Logistics. IEEE 2007.

[41] MinHee Choi, Weekuk Kim and Byung-Ju Yi. Trayectory Planing in 6-Degrees-of-Freedom Operational Space for the 3-Degree-of-Freedom Mechanism Configured by Constraining the Stewart Platform Structure. International Conference on Control, Automation and Systems 2007.

[42] Sung-Hua Chen and Li-Chen Fu. The Forward Kinematics of the 6-6 Platform Using Extra Sensors.International Conference on Systems, and Cybernetics. IEEE 2006.

[43] G. Satheesh Kumar, T. Nagarajan. Experimental Investigation on Reconfigurable Stewart Platform for Contour Generation. International Conference on Sensing Technology. IEEE 2008.

[44] Sedar Ay, o. Erguven Vatandas, Abdurrahman Hacioglu. The Effect of Radius of Join Location on Workspace Analysis of The 6-6 Stewart Platform Mechanism. IEEE 2009.

[45] Tzung-Cheng Tsai, Yeh-Liang Hsu. Development of a parallel surgical robot with automatic bone drilling carriage for stereotactic neurosurgery. Biomedical Engineering, Aplications, Basis, and Communications, 2007.

[46] S Iqbal, A. I. Bhatti and Q. Ahmed. Determination of realistic Uncertainty Bouns for the Stewart Platform with Payload Dinamics. Conference on Control Applications IEEE 2008.

[47] Jeff M. Wendlandt, S. Shankar Sastry. Desing and Control of a Simplified Stewart Platform for Endoscopy. Conference on Decision and Control, 1994.

[48] J. Gallardo Alvarado, J.M. Rico Martínez, M. Caudillo Ramírez. Análisis Cinemático Directo de un Manipulador Paralelo Esférico Asimétrico. Universidad Nacional Autónoma de México. 2006.

[49] Meneses Xavier, Méndez Mauricio, Cortes Eduardo. Diseño y Control de un Robot Paralelo. Congreso Nacional de Mecatrónica, 2007.

[50] Saravia Darly, López Marlon, Riaza Hector. Revisión del Estado del Arte de Manipuladores Paralelos. Universidad Tecnológica de Pereira 2009.

[51] Andrés Vivas. Robótica Paralela: Aplicaciones Industriales, Modelado y Control. Colombia.

[52] Isidro Zabala, Aplicaciones Actuales de los Robots Paralelos, Congreso Iberoamericano de Ingeniería Mecánica, 2007.

[53] Llian Bonev. El verdadero origen de los robots paralelos. 2003.

[54] Rafael Aracil, Roque J. Saltaren, José M. Sabater, Oscar Reinoso. Robots Paralelos Maquinas con un Pasado para una Robótica del Futuro. Revista Iberoamericana de Automática e Informática Industrial.2006.

[55] J. A. Souza-Jimenez, V. J. Gonzalez-Villela, "Kinematics and tip-over stability analisys for a hybrid serial-parallel mobile manipulator" Proc. of ASME International Mechanical Engineering Congress & Exposition, 2011.

Páginas de Internet

http://www.fundibeq.org/opencms/opencms/PWF/home/index/index.html

http://www.mathworks.com/index.html

http://www.todopic.com.ar/foros/

http://gear.chuckpaiwong.com/deltarobot

http://www.parallemic.org/