UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

INGENIERÍA INDUSTRIAL
AUTOMÁTICA Y ELECTRÓNICA INDUSTRIAL

PROYECTO FIN DE CARRERA

Caracterización y Control de los Motores del Tobillo del Robot Humanoide RH-2

AUTOR: GUILLERMO GALLARDO FERNÁNDEZ
TUTORA: CONCEPCIÓN ALICIA MONJE MICHARET

Julio 2010

A mi familia.

Índice

Índice de Figuras

Capítulo 1

INTRODUCIÓN Y OBJETIVOS

1.1. INTRODUCCIÓN

Este trabajo forma parte del novedoso proyecto de desarrollo del robot humanoide RH-2 como mejora de RH-1. Proyecto realizado por el equipo de trabajo Robotic Lab de la Universidad Carlos III de Madrid, primer laboratorio de esta naturaleza en España y uno de los pocos a nivel mundial. Una imagen del RH-1 se presenta en la Figura 1.

Figura 1 RH-1

El proyecto de humanoides realizado en la Universidad Carlos III de Madrid consta de dos proyectos ya construidos: RH-0 y RH-1.

El RH-1 es un humanoide que mide 1.50m de altura, pesa 50Kg y cuenta con 21 grados de libertad distribuidos en piernas, brazos y tronco.

El equipo de investigación de la Universidad consiguió que Rh-1 anduviese a 0,7 km/h en un máximo de cuatro pasos. A simple vista parecerá una minucia, sin embargo, muy pocos robot humanoides logran desplazarse con robustez y seguridad. En movimientos más lentos el robot llegó a dar 10 pasos [1] y [2].

La estructura del robot es muy parecida a la de un ser humano tanto en su peso como en su altura. En realidad algunas de sus partes tienen la única función de asemejarse al cuerpo humano, como por ejemplo, los brazos, que no son básicos a la hora de mantener el equilibrio, sino que son más para asemejar a la mímica de las personas, es decir, se persigue la idea de conseguir un robot de apariencia y gestos análogos a los humanos.

El RH-1, al igual que el RH-0, son sistemas mecánicos de 21 Grados De Libertad (GDL), los cuales se distribuyen de la siguiente forma:

- Piernas: dispone cada una de 6 GDL distribuidos entre el tobillo, la rodilla y la cadera. La cadera posee 3 GDL, la rodilla tiene 1 GDL y el tobillo posee los otros 2 GDL.
- Brazos: cada brazo tiene 4 GDL distribuidos entre el hombro, el codo y la muñeca. En el hombro existen 2 GDL, en el codo hay únicamente 1 GDL y el otro sita en la muñeca.
- Tronco y Cabeza: hay un grado de libertad que permite mover los hombros.

1.1.1. RH2

Entre las características básicas del Robot Humanoide RH-2, que le permitan tener un amplio campo de aplicación tanto en el sector industrial como en el de servicios, se encuentran:

- Locomoción bípeda estable.
- Dos brazos con pinza articulada en los extremos para manipular objetos.
- Un cuerpo que albergue el sistema de control.
- Una cabeza con sensores de visión y sonido, para orientarse y desplazarse por el entorno de trabajo y recibir órdenes por voz.
- Autonomía en la toma de decisiones y necesidades de energía.
- Capacidad de realizar tareas en entornos no estructurados, tales como naves industriales, locales comerciales, centros sanitarios y educativos y entornos domésticos, donde debe ser capaz de con su entorno y desarrollar su función.

Resulta interesante desarrollar este tipo de robots con sus características, capaz de interactuar con los seres humanos de forma eficiente y amigable en sus entornos cotidianos, y en los que estén presentes elementos y utensilios propios de la actividad humana. El fin último es la mejora de la calidad de vida y el apoyo a personas con discapacidades entre otros.

Las tareas que podrá realizar son de los siguientes tipos:

- Asistencia a humanos: el humanoide debe ser capaz de interactuar con las personas de forma eficiente en un entorno humano. Se pueden citar como ejemplos la asistencia a personas dependientes, ayudantes personales en oficinas, centros sanitarios y educativos y otros tipos de servicios públicos.
- Manipulación y transporte de mercancías: de forma individual como cooperando con otros robots u operarios humanos y capacidad para mover una gran variedad de objetos.
- Operaciones de mantenimiento, limpieza y tareas similares: especialmente cuando supongan cierto peligro para la integridad física de un operario humano.
- Vigilancia y salvamento: tanto aplicaciones de inspección visual y de seguridad, como tareas de salvamento frente a desastres naturales y situaciones de emergencia.
- Entretenimiento y educación: el robot debe ser capaz de realizar actividades de entretenimiento como juegos y deportes, y servir por ejemplo, como guía de un museo.

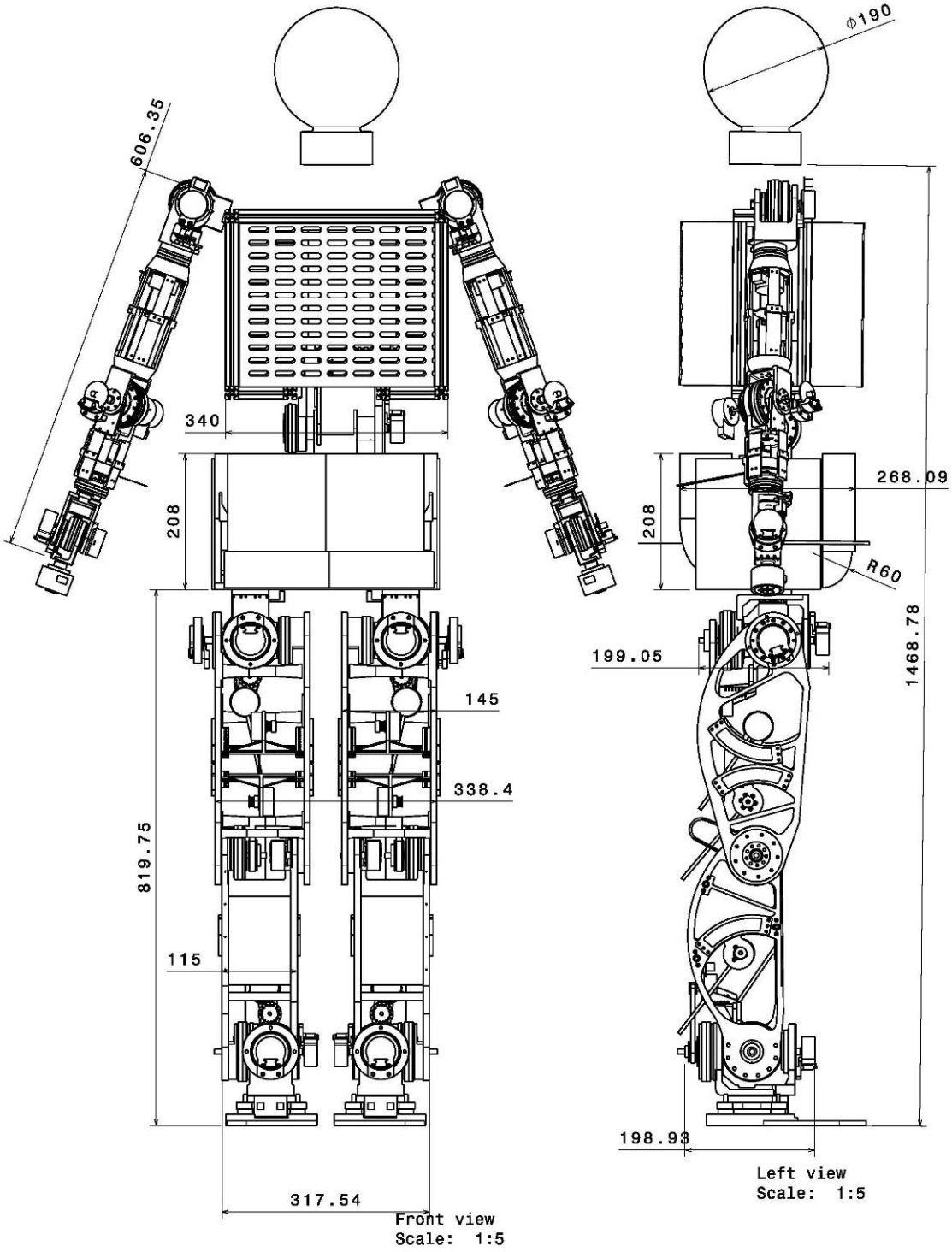


Figura 2 RH2 –Plano

Se estima que su peso será de unos 60 Kg. y llegará a alcanzar una velocidad de 1 Km/h durante la caminata. En la Figura 2 se presenta un plano de RH-2 en el que se incluyen sus medidas. Se observa que la altura pasa a ser aproximadamente 1.65 m, siendo esta más acorde a la de un humano.

El nuevo robot dispone de 24 grados de libertad, 3 más que los modelos anteriores, distribuidos de la siguiente forma:

- Piernas: dispone cada una de 6 GDL distribuidos entre el tobillo, la rodilla y la cadera.
- Brazos: dispone cada uno de 5 GDL, uno más que el RH-1, distribuidos entre el hombro, el codo y la muñeca.
- Tronco: el tronco posee 2 GDL en plano transversal, que le permite el giro en ese plano sin tener que mover las piernas y otro en plano frontal, que le permite regular su inclinación.

El esquema de la Figura 3 resume lo explicado anteriormente:

Extremidad		**GDL**	
Piernas	**Tobillo**	**2**	6
	Rodilla	1	
	Cadera	3	
Brazos	Muñeca	1	5
	Codo	2	
	Hombro	2	
Tronco	Tronco	2	2

Figura 3 GDL del RH2

1.2. OBJETIVOS

El objetivo de éste proyecto es realizar la caracterización y control del tobillo del Robot Humanoide RH-2 (Figura 4 y Figura 5), con el fin de estudiar la robustez mecánica del prototipo y el correcto funcionamiento de los motores utilizados para sus movimientos.

Como ya se ha comentado, el tobillo posee dos grados de libertad, por lo que se usarán dos motores iguales en cada uno de los tobillos del humanoide (ver Figura 6).

A priori, no es necesario realizar estos primeros estudios usando técnicas matemáticas para la caracterización, debido a que no es un prototipo definitivo. Por este motivo se usarán técnicas de "*prueba-error*" para el cálculo de los PID y la función de transferencia que caracteriza el sistema.

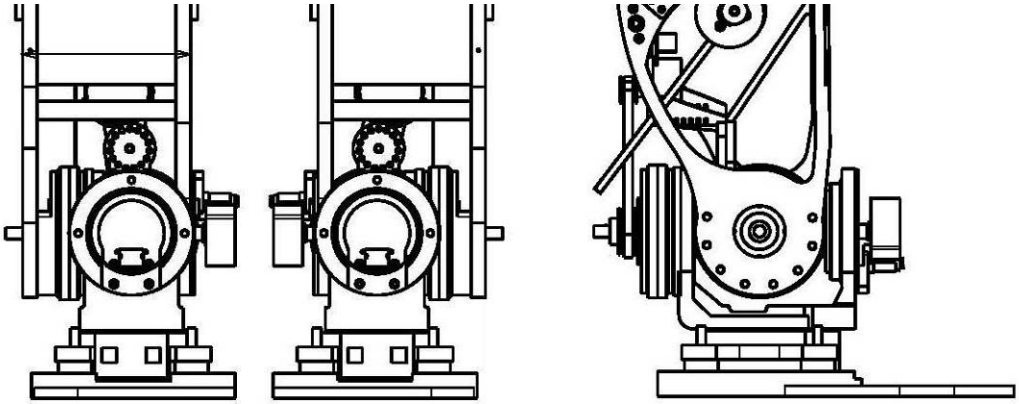


Figura 4 Tobillo del RH-2

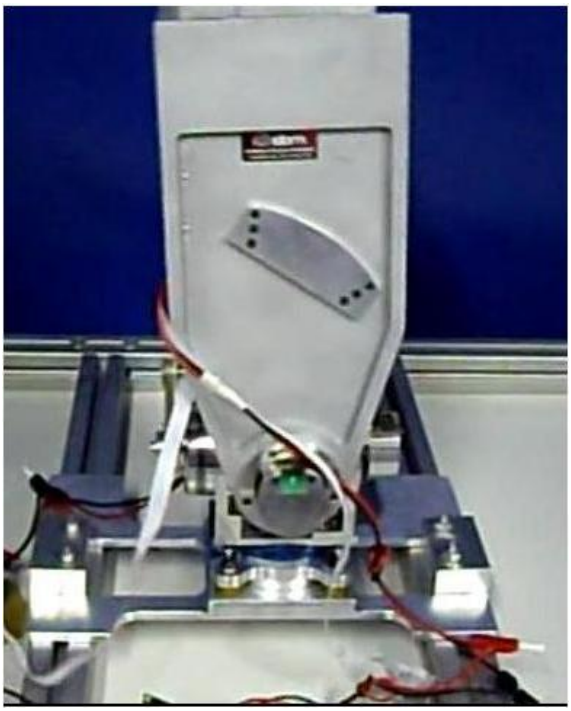


Figura 5 Prototipo del pie del RH2

El primer paso, una vez elegido el motor, es caracterizarlo, es decir, obtener su Función de Transferencia (FT) para así poder modelarlo y controlarlo según los requerimientos técnicos del RH-2. Para ello es necesario el uso de dos programas, Easy Motion Studio (EMS) y Matlab. El primero es un software esencial, ya que es el encargado de controlar el driver del motor y por tanto el que proporciona las señales y los datos necesarios para su caracterización. Una vez obtenidos estos datos se utilizará Matlab, que a través de la herramienta IDENT permite obtener la FT del motor. Con este modelo matemático del sistema se simula su comportamiento y se realizarán toda clase de pruebas ya sea en lazo abierto o lazo cerrado del tobillo, con lo que se podrá analizar la respuesta generada ante todo tipo de entradas.

Una vez analizado el comportamiento del motor se pasa a realizar las pruebas reales para asegurar su correcto funcionamiento. Para ello se conectan los motores con los drivers a las articulaciones del tobillo. Con el EMS se generan las trayectorias de movimiento, teniendo en cuenta las limitaciones de cada eje especificadas en el diseño. En una primera prueba se generan unas trayectorias suaves para verificar que no hay ningún problema en el diseño mecánico. Seguidamente se realiza una prueba llevando a cada motor al límite de su funcionamiento, haciéndolos girar al mismo tiempo durante un largo

periodo de tiempo. Cuando se ha comprobado que los límites de diseño son correctos y que el motor elegido es el adecuado, se pasa a hacer una prueba añadiendo al tobillo una mayor carga mediante una serie de pesos. Se repiten las mismas pruebas y se comprueba su correcto funcionamiento y su robustez mecánica.

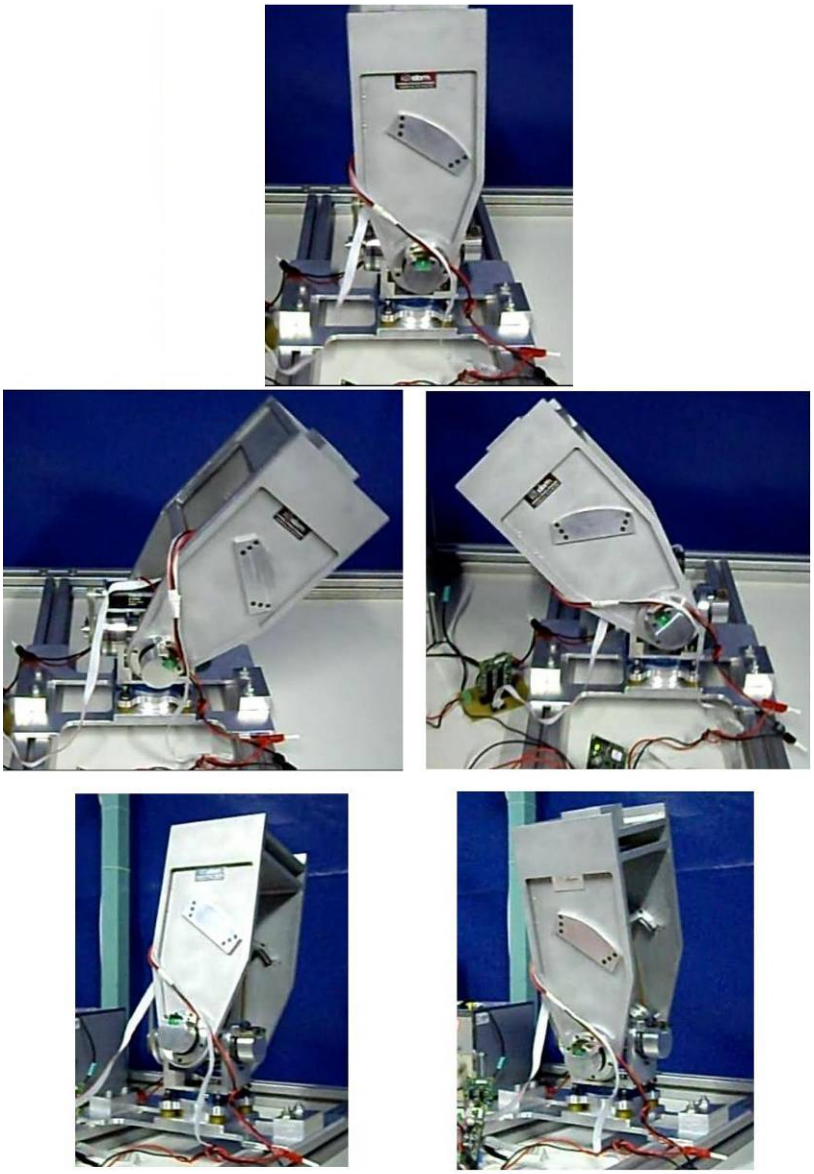


Figura 6 Grados de libertado del tobillo

1.3. ESTRUCTURA DEL DOCUMENTO

La estructura de este documento está dividida en 7 capítulos y los anexos que lo complementan:

- El capítulo 2 incluye una descripción de los elementos necesarios para la realización del proyecto, así como el software que se ha utilizado para llevarlo a cabo.
- La forma en la que se han configurado los lazos de control del motor del tobillo y la metodología seguida para caracterizarlos y obtener su función de transferencia, se describen en los capítulos 3 y 4, respectivamente. Siguiendo con la línea de caracterización y control en el capítulo 5 se desarrolla el modo de diseñar un PID, mediante ajuste manual, en lazo cerrado para garantizar que el sistema responda ante perturbaciones e incertidumbres del modelo.
- Los ensayos realizados tanto en vacío (únicamente con el peso propio del tobillo) como a plena carga simulada, mediante una barra, se desarrollan en el capítulo 6.
- El capítulo 7 versa sobre las conclusiones obtenidas tras la finalización de la fase experimental y futuros trabajos a realizar para mejorar el funcionamiento del RH-2.
- Por último, las bibliografías y las referencias consultadas se encuentran, al igual que los anexos, al final del documento.

Capítulo 2

ELEMENTOS DEL SISTEMA Y SOFTWARE UTILIZADO

Como ya se ha comentado, el objetivo principal de este proyecto es el modelado y control de los motores usados para generar los movimientos del tobillo del humanoide. A continuación se hará una descripción de los elementos hardware y software empleados para ello.

2.1. MOTOR MAXON

El motor usado (ver Figura 7) para generar el movimiento de las articulaciones estudiadas en este proyecto es un motor de marca Maxon. El modelo es RE 35 de 35 milímetros de diámetro, alimentación DC, escobillas de grafito y una potencia de 90 vatios. Su referencia es: 273757. Este motor no lleva incorporado sensores de efecto Hall.

La hoja de características de este motor se encuentra en el Anexo I. En él se pueden observar los datos del motor y las especificaciones necesarias para su correcto funcionamiento. Dichos datos y especificaciones serán introducidos en el EMS para el control de los motores.

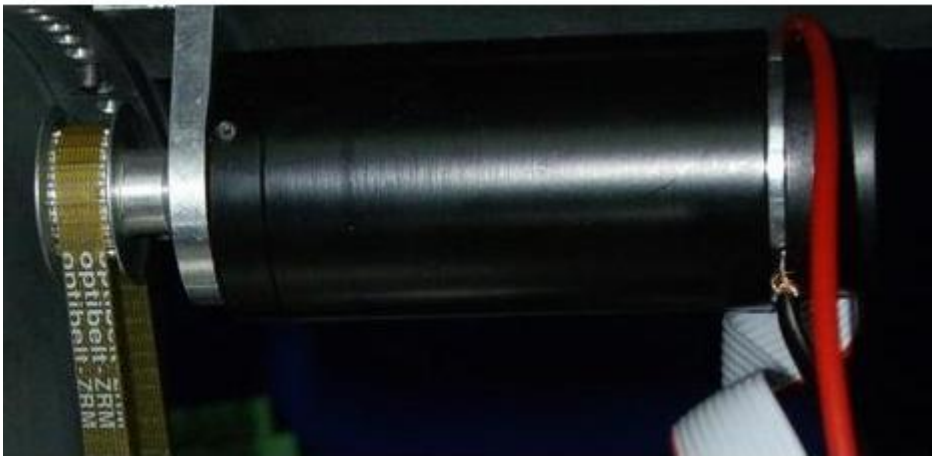


Figura 7 Motor Maxon (REF: 273757)

2.1.1. Grados de Libertad de la articulación

En la Figura 8 se representan de forma esquemática los dos grados de libertad de la estructura que corresponde a:

- Giro en X (plano frontal)
- Giro en Y (plano sagital)

Esta estructura está diseña de tal forma que el máximo giro en X es de unos 20º y de 70º en el caso del giro en Y.

Con el objetivo de comprobar las robustez del prototipo y que el diseño mecánico ha sido adecuado, se generarán movimientos límites y bruscos, así se asegurará su funcionamiento y fiabilidad.

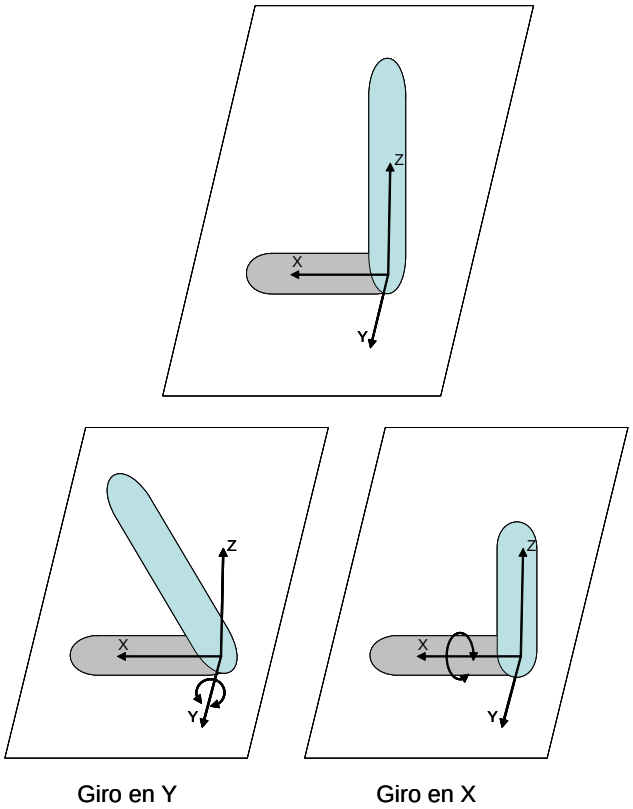


Figura 8 Esquema de los giros de las articulaciones

2.1.2. Características, Pines y Conexión

Para hacer funcionar correctamente al motor, es necesario conectar las señales del encoder y las señales ABC de control. La Figura 7 representa esquemáticamente dicho conexionado. Siendo los cables Rojo y Negro los correspondiente a las señales ABC de control y el cable IDE el correspondiente a las del encoder.

Como ya se ha comentado, este modelo no lleva incorporado sensores de efecto Hall. Al ser un motor DC brush, los únicos pines que es necesario conectar son el 1 y el 2 (ver Figura 11) correspondientes a los cables rojo (A+) y negro (A-) de las señales de control del motor.

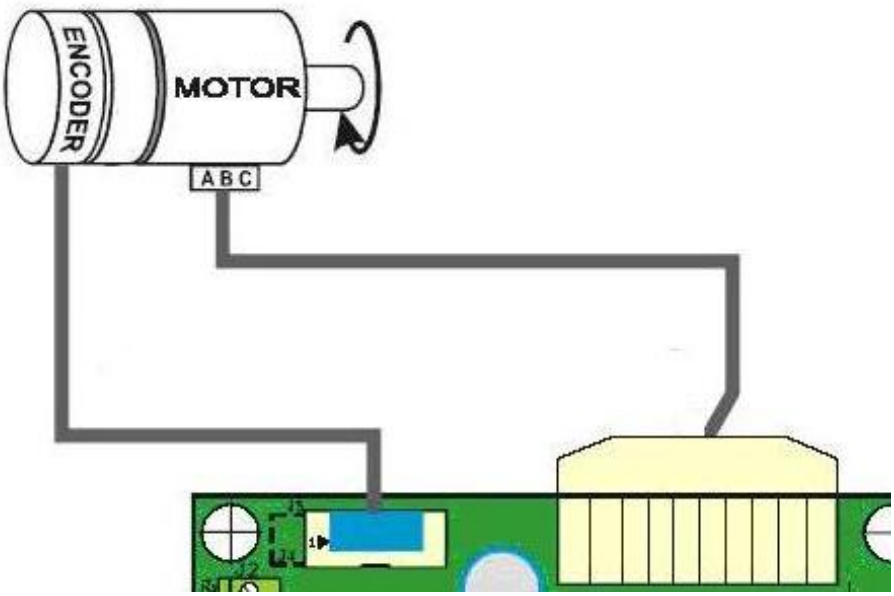


Figura 9 Esquema conexión Motor-Extension Board I

En la Figura 9 y Figura 10 se muestra la conexión física de estos dos pines de las señales de control (cables rojo y negro). También se representa en esta misma figura las conexiones de las señales del encoder, siendo estas las correspondientes a la tabla de la Figura 11.

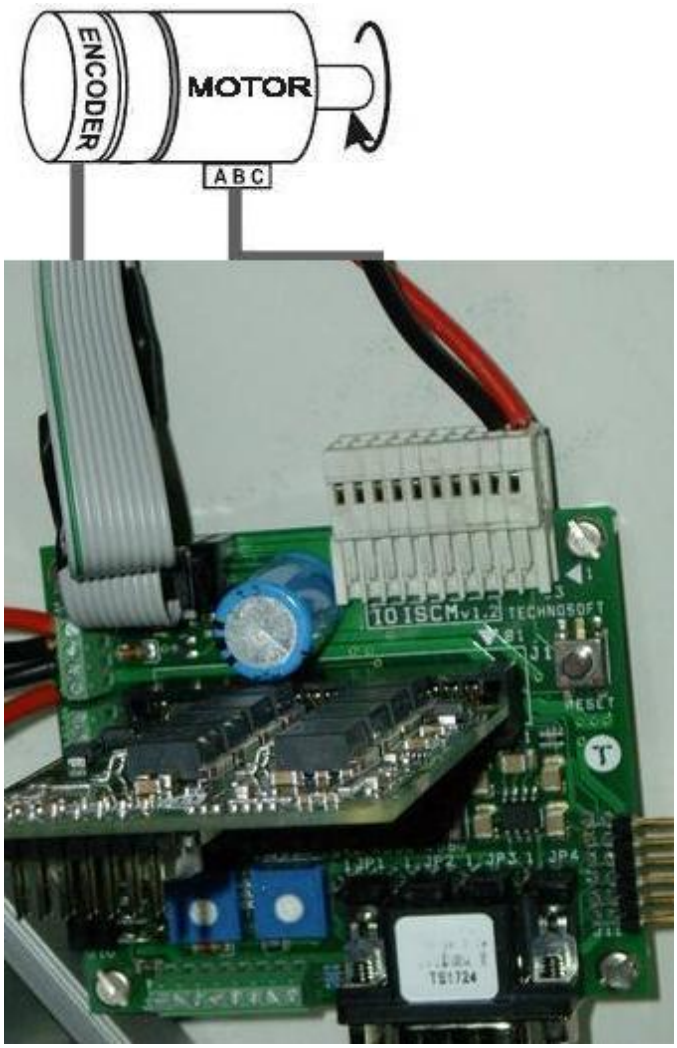


Figura 10 Esquema conexión Motor-Extension Board II

Pin	Pin name	Type	Function
1	A / A+	I	Phase A for brushless motors Motor+ for DC brush motors
2	B / A-	I	Phase B for brushless motors Motor- for DC brush motors
3	C / B+	I	Phase C for brushless motors Motor+ for DC brush motors
4	BRAKE / B-	I	Brake output (for external brake resistor) for brushless motors Motor- for DC brush motors
5	GND	-	Ground
6	+5V	I	Positive terminal of the 5V logic supply
7	H1 / SIN	O	Hall 1 signal for digital Hall sensor SIN signal for Linear Hall sensor
8	H2 / COS	O	Hall 2 signal for digital Hall sensor COS signal for Linear Hall sensor
9	H3	O	Hall 3 signal for digital Hall sensor
10	GND	-	Ground

Figura 11 Señales control del motor

Pin	Pin name	Type	Function
1	GND	-	Ground
2	+5V	I	Positive terminal of the +5V logic supply
3	GND*	-	Ground
4	+5V	I	Positive terminal of the +5V logic supply
5	ENCA-	O	Single-ended encoder A signal Differential encoder negative A input
6	ENCA+	O	Single-ended encoder A signal Differential encoder positive A input
7	ENCB-	O	Single-ended encoder B signal Differential encoder negative B input
8	ENCB+	O	Single-ended encoder B signal Differential encoder positive B input
9	ENCZ-	O	Single-ended encoder Z signal Differential encoder negative Z input
10	ENCZ+	O	Single-ended encoder Z signal Differential encoder positive Z input

Figura 12 Señales encoder del motor

Cabe destacar que en las tablas de la Figura 11 y de la Figura 12 se ha querido mostrar, por comodidad para el lector, el tipo de señal (I -> INPUT, O-> OUTPUT) desde el punto de vista del motor. Es decir, las señales “I” son señales de entrada al motor y las “O” de salida del mismo. En apartados siguientes se muestra esta misma tabla desde el punto de vista de la Extension Board.

2.2. SISTEMA DE REDUCCIÓN

Se han utilizado dos correas de transmisión reductoras iguales en cada uno de los dos motores, estas correas serán las encargadas de transmitir el movimiento del motor al eje de la articulación a través del harmonic drive (ver Figura 13 y Figura 14).

La reducción total del conjunto es de 1/320, dividida en:

- Reducción harmonic drive: 1/160
- Reducción de las correas de transmisión: 1/2

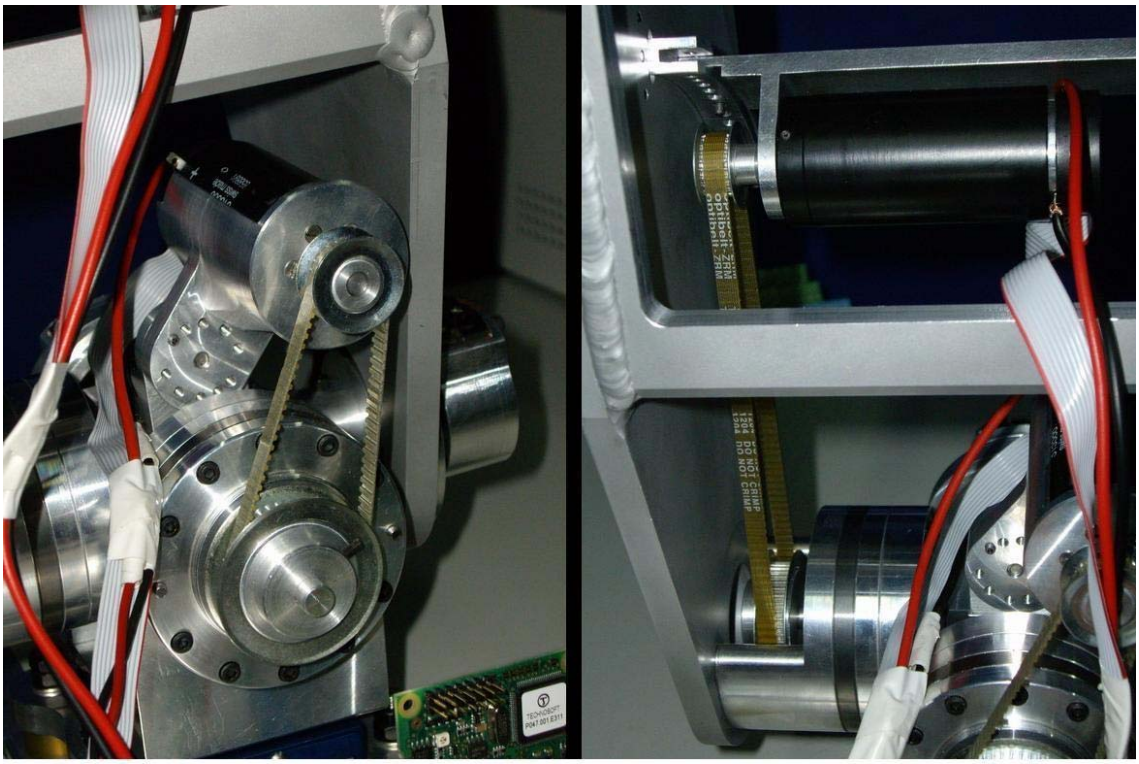


Figura 13 Correas de distribución

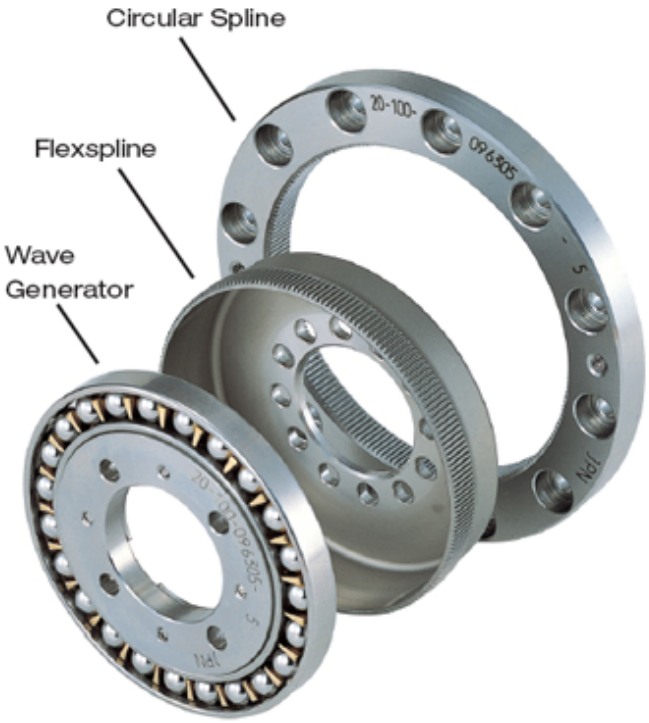


Figura 14 Harmonic drive

2.3. ISCM 8005 INTELLIGENT SERVO DRIVE

La placa Intelligent Servo Drive es el hardware que sirve de interfaz entre el motor y el ordenador de control, es decir, su función es manejar toda la información relativa al movimiento del motor, procesa las señales de entrada y salida del motor actuando según proceda y según haya sido programada. En la Figura 15 se muestra dicho hardware.

Figura 15 ISCM 8005 Intelligent Servo Drive

Este hardware se encarga de recoger y suministrar al motor toda la información necesaria para que funcione tal y como requiera el diseñador. Es decir, una vez realizadas las conexiones oportunas, el Drive excita al motor para que empiece a moverse, recogiendo la información de su posición y velocidad a través de las pines que van conectados al encoder del motor (Ver en Figura 16 y Figura 17 los pines A7-A9 y B7-B9).

2.3.1. Pines y Conexiones

El ISCM 8005 INTELLIGENT SERVO DRIVE, consta de conexiones en sus dos caras, teniendo un total de 34 repartidas de igual forma en cada una de ellas y siendo complementarias las de misma numeración, es decir:

A16 y B16: CAN-Bus

A17 y B17: RS-232

Y así sucesivamente (ver Figura 16 y Figura 17)

En la Figura 16 y Figura 17, se muestran las 34 posibles conexiones que están disponibles. Como puede observarse, las conexiones están distribuidas por grupos, así los pines 2,3, 4 y 5 son los correspondientes a la estación de cada una de las fases del motor y los pines 7,8 y 9 son los correspondientes a las señales del encoder [3].

Pin	Pin name	TML name	Type	Function/Alternate function/ Comments
A1	+Vmot	-	I	Positive terminal of the **motor supply**: 12 to $48V_{DC}$ for ISCM4805v1.2 12 to $80V_{DC}$ for ISCM8005v1.2
A2	A / A+	-	O	• **Phase A** for brushless motors • **Motor+** for DC brush motors
A3	B / A-	-	O	• **Phase B** for brushless motors • **Motor-** for DC brush motors
A4	C / B+	-	O	• **Phase C** for brushless motors • **Motor+** for DC brush motors
A5	BRAKE /B -	-	O	• **Brake output** (for external brake resistor) for brushless motors • **Motor+** for DC brush motors
A6	+5V	-	O	Positive terminal of the 5V logic supply (internally generated)
A7	ENCA+	-	I	• Single-ended encoder A signal • Differential encoder positive A input
A8	ENCB+	-	I	• Single-ended encoder B signal • Differential encoder positive B input
A9	ENCZ+ / CAPI	-	I	• Single-ended encoder Z signal • Differential encoder positive Z input
A10	H1 / SIN	-	I	• Hall 1 signal for digital Hall sensor • SIN signal for linear Hall sensor
A11	IN#38 / PULSE	IN#38 / PULSE	I	• 5V compatible input • Can be used as PULSE input in Pulse & Direction motion mode
A12	IN#2 / LSP	IN#2 / LSP	I	• 5V compatible input • Positive limit switch
A13	ENABLE	ENABLE	I	5V compatible input. Connect to +5V to disable PWM outputs
A14	FDBK	AD2	I	Monopolar 0…+5V or bipolar –10V…+10V analogue input. May be used as analogue position or speed feedback (from a tachometer)
A15	GND	-	-	Ground
A16	CAN_H	-	I/O	Can-Bus positive line (positive during dominant bit)
A17	TX232	-	O	RS-232 Data Transmission

Figura 16 Pines conexión ISCM 8005 Intelligent Servo Drive (cara A)

B1	+Vmot	-	I	Positive terminal of the **motor supply**: 12 to $48V_{DC}$ for ISCM4805v1.2 12 to $80V_{DC}$ for ISCM8005v1.2
B2	A / A+	-	O	• **Phase A** for brushless motors • **Motor+** for DC brush motors
B3	B / A-	-	O	• **Phase B** for brushless motors • **Motor-** for DC brush motors
B4	C / B+	-	O	• **Phase C** for brushless motors • **Motor+** for DC brush motors
B5	BRAKE / B-	-	O	• **Brake output** (for external brake resistor) for brushless motors • **Motor+** for DC brush motors
B6	+Vlog	-	I	Positive terminal of the logic supply: +12 to $+48V_{DC}$
B7	ENCA-	-	I	• Single-ended encoder A signal • Differential encoder negative A input
B8	ENCB-	-	I	• Single-ended encoder B signal • Differential encoder negative B input
B9	ENCZ-	-	I	• Single-ended encoder Z signal • Differential encoder negative Z input
B10	H2 / COS	-	I	• Hall 2 signal for digital Hall sensor • COS signal for linear Hall sensor
B11	H3	-	I	Hall 3 signal for digital Hall sensor
B12	IN#24 / LSN	IN#24 / LSN	I	• 5V compatible input • Negative limit switch
B13	RESET		I	RESET signal – connect to +5V to reset the board
B14	REF	AD5	I	Monopolar 0…+5V or bipolar –10V…+10V analogue input. May be used as analogue position, speed or torque reference.
B15	GND	-	-	Ground
B16	CAN_L	-	I/O	CAN-Bus negative line (negative during dominant bit)
B17	RX232	-	I	RS-232 Data Reception

Figura 17 Pines conexión ISCM 8005 Intelligent Servo Drive (cara B)

2.4. IO ISCM V1.2 EXTENSION BOARD

Como las conexiones directas del Servo Drive son de difícil acceso y poco manejables, es necesario utilizar en un primer momento y para realizar las pruebas, una Extension Board para así facilitar el conexionado y ser más manejable de cara a la persona que lo manipule. Una vez terminado el periodo de pruebas se podría optar por seguir utilizando la Extension Board dentro de la implementación final del prototipo del robot humanoide o, si el espacio lo requiere, utilizar directamente las conexiones de la Intelligent Servo Drive soldando las conexiones.

El IO ISCM v1.2 Extension Board (ver Figura 18) es un módulo auxiliar que permite evaluar rápidamente las capacidades que ofrece el Drive para el control de los

movimientos del motor. Este hardware es compatible con el ISCM8005 v1.2 Intelligent Servo Drive, el cual se conecta a este módulo a través del conector J1.

Gracias a IO ISCM v1.2 Extension Board, se consigue tener un acceso directo a los recursos principales del Drive Board, lo que facilitará sustancialmente el trabajo.

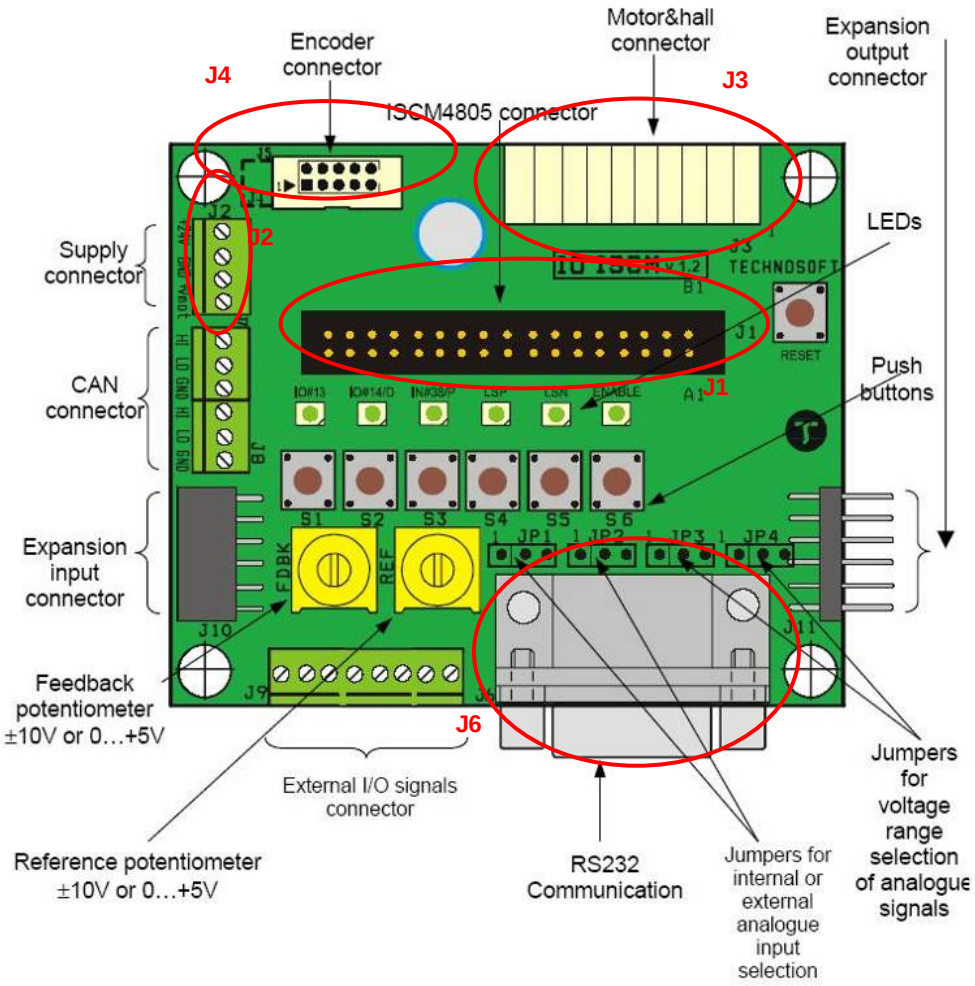


Figura 18 ISMC 8005 v1.2 Extension Board

Para la realización de este proyecto es interesante estudiar con detenimiento cinco de los once conectores de los que dispone esta Extension Board.

(Nota: Si se desea ampliar la información de los otros 7 conectores ver referencia [3])

A continuación se enumeran los once conectores, siendo los destinadas a estudio los conectores J1, J2, J3, J4 y J6:

- J1 – conector de 2x17 pines y 0.1", usado para conectar la IO ISCM v1.2 Extension Board y ISCM4805/8005 v1.2 Intelligent Drives
- J2 – conector para alimentación
- J3 – conexión para el servo motor
- J4 – conector para señales diferenciales y single-ended del encoder
- J5 – conector para señales single-ended del encoder
- J6 – conector RS-232
- J7 y J8 – conectores del CAN bus
- J9 – conector para señales externas I/O
- J10 y J11

Las conexiones del conector J1 se representan en la Figura 16 y en la Figura 17.

La alimentación de la Intelligent Board y del motor se realiza desde el J2 a través de los pines 1 y 4, respectivamente, utilizando una tierra común que se conectará en los pines 2 y 3 de este conector (ver Figura 19). Como puede observarse, la alimentación permitida va desde los 12 hasta los 48 V_{DC} para el Drive y de los 12 a 80 V_{DC} para el motor. Como la tensión máxima que proporcionan las fuentes utilizadas es de unos 33 V_{DC}, será ésta la utilizada para ambas alimentaciones y así obtener la máxima potencia disponible. Como se verá más adelante, con la tensión y corriente que proporciona esta fuente (ver Figura 20) será suficiente para alimentar al conjunto tanto en vacío como en plena carga.

Pin	Pin name	Function
1	+Vlog	Positive terminal of the logic supply: +12 to +48V_{DC}
2	GND	Ground
3	GND	Ground
4	+Vmot	Positive terminal of the motor supply: 12 to 48V_{DC} for ISCM4805v1.2 12 to 80V_{DC} for ISCM8005v1.2

Figura 19 J2 - Conexiones alimentación de la placa

Figura 20 Fuente de Alimentación

A través del conector J3 (ver Figura 21) se enviarán las señales de control desde el Drive hasta el motor. Puesto que estamos ante un motor con escobillas de grafito, serán los pines 1 y 2 los únicos usados en este conector, recibiendo dichas señales desde los pines A2, A3, B2 y B3 del Drive. Estas señales serán salidas de este hardware.

Pin	Pin name	Type	Function
1	A / A+	O	Phase A for brushless motors Motor+ for DC brush motors
2	B / A-	O	Phase B for brushless motors Motor- for DC brush motors
3	C / B+	O	Phase C for brushless motors Motor+ for DC brush motors
4	BRAKE / B-	O	Brake output (for external brake resistor) for brushless motors Motor- for DC brush motors
5	GND	-	Ground
6	+5V	O	Positive terminal of the 5V logic supply
7	H1 / SIN	I	Hall 1 signal for digital Hall sensor SIN signal for Linear Hall sensor
8	H2 / COS	I	Hall 2 signal for digital Hall sensor COS signal for Linear Hall sensor
9	H3	I	Hall 3 signal for digital Hall sensor
10	GND	-	Ground

Figura 21 J3 - Señales control Extension Board

Para poder cerrar el lazo y realimentar el dispositivo se utilizará el J4. Así, se podrá diseñar un PID y se controlará la posición y velocidad del motor en todo momento. El encoder del motor será el encargado de enviar la información de su posición actual, información que será recibida a través de los pines 5, 6, 7, 8, 9 y 10 del J4 (ver Figura 22) de la Extension Board y que será reenviada a los pines A7, A8, A9, B7, B8 Y B9 del Drive. Del conjunto de estas señales y del diseño de un PID se obtendrá la excitación del motor adecuada al diseño.

Pin	Pin name	Type	Function
1	GND	-	Ground
2	+5V	O	Positive terminal of the +5V logic supply
3	GND*	-	Ground
4	+5V	O	Positive terminal of the +5V logic supply
5	ENCA-	I	Single-ended encoder A signal Differential encoder negative A input
6	ENCA+	I	Single-ended encoder A signal Differential encoder positive A input
7	ENCB-	I	Single-ended encoder B signal Differential encoder negative B input
8	ENCB+	I	Single-ended encoder B signal Differential encoder positive B input
9	ENCZ-	I	Single-ended encoder Z signal Differential encoder negative Z input
10	ENCZ+	I	Single-ended encoder Z signal Differential encoder positive Z input

Figura 22 J4 - Señales encoder Extension Board

La función del conector RS-232 (J6) es posibilitar la conexión de la placa al ordenador, representado en la Figura 23. Así se podrá interactuar con el Drive, programándolo y analizando las señales generadas durante el funcionamiento.

Pin	Pin name	Type	Function
1	NC	-	Not connected
2	TX232	O	RS-232 Transmission
3	RX232	I	RS-232 Data Reception
4	NC	-	Not connected
5	GND	-	
6	NC	-	Not connected
7	NC	-	Not connected
8	NC	-	Not connected
9	NC	-	Not connected

Figura 23 J6 - Conexión RS-232

En la Figura 24 se representa esquemáticamente la conexión del Drive a la Extension Board (J1) desde dos vistas, planta y perfil. Se incluye también una foto real de dicha conexión (Figura 25).

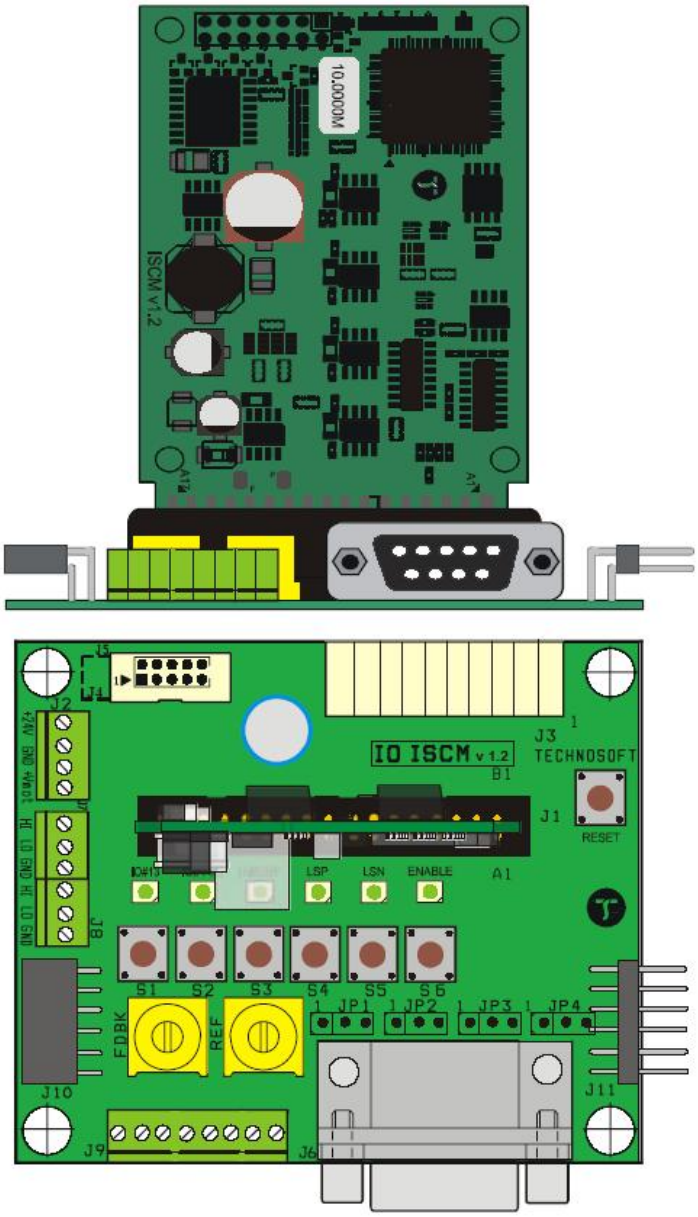


Figura 24 Montaje del Drive en la Extension Board

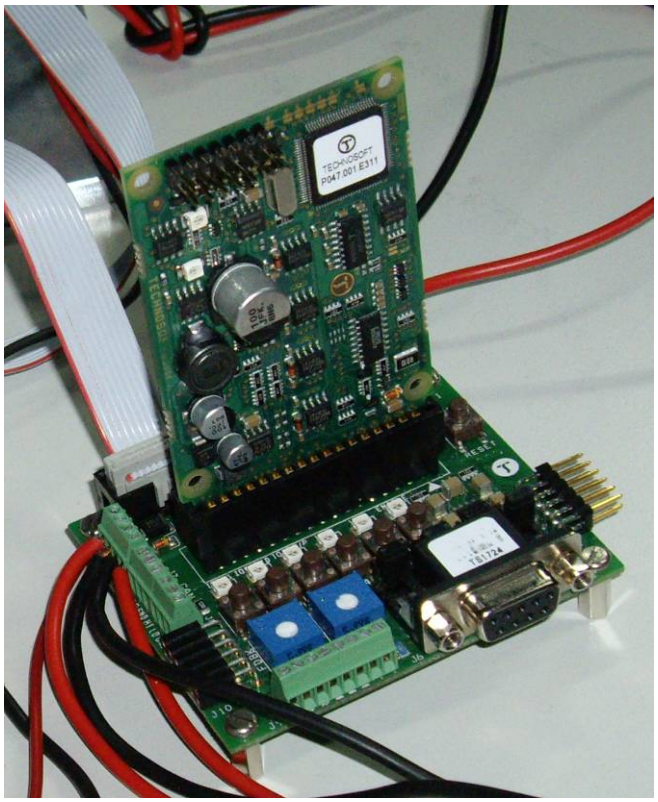


Figura 25 Montaje del Drive en la Extension Board (Foto real)

2.5. EXTENSION BOARD DESARROLLADA EN LA UC3M

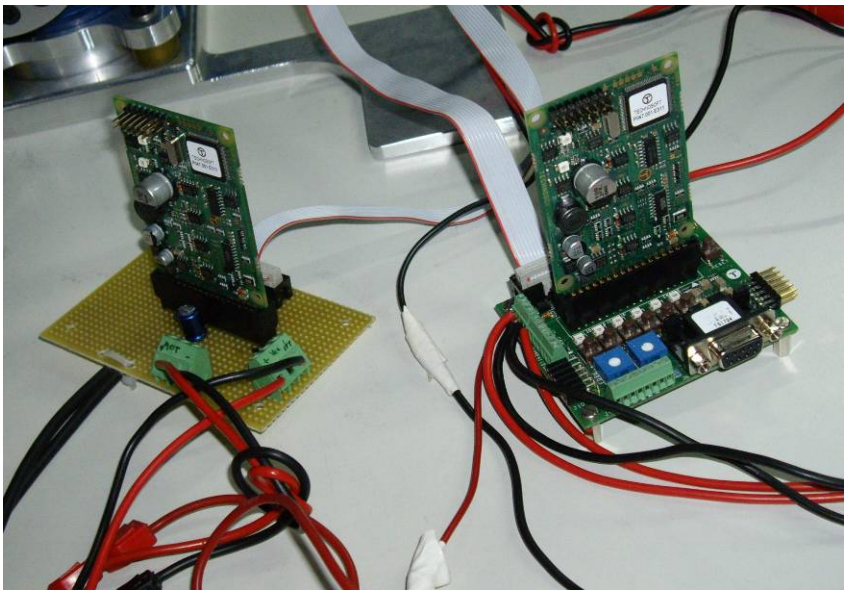


Figura 26 Drives y Extension Board

Es necesario utilizar dos motores para obtener los dos grados de libertad del sistema, por lo que se requieren dos Drives y dos Extension Board (Figura 26). La segunda Extension Board utilizada será una desarrollada en el UC3M, más simple que la anterior pero con la misma funcionalidad y fiabilidad. La única diferencia es que ésta tan solo tiene cinco conectores, frente a los once de la placa comercial, que serán los indicados anteriormente en el apartado 2.4. Esto hace que sea más fácil de manejar ya que está dotada sólo de los conectores que serán utilizados durante este proyecto (ver Figura 27).

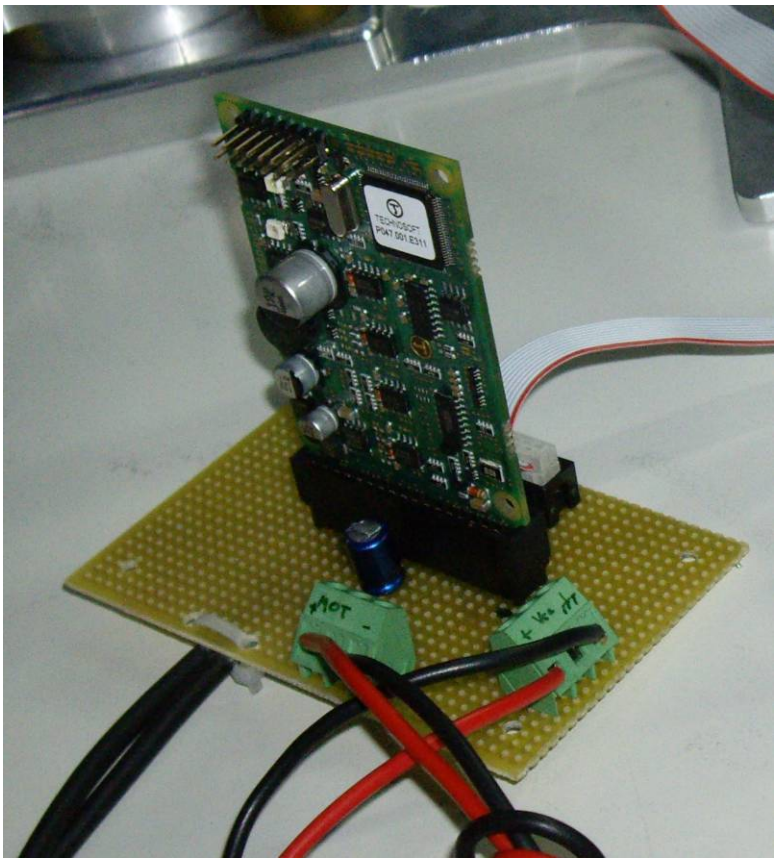


Figura 27 Extension Board desarrollada en la UC3M

2.6. ESQUEMA DE CONEXIONADO COMPLETO

Finalmente el conjunto del sistema está formado por:

Cantidad	**Concepto**	**Modelo**
2	Motor DC escobillas	Maxon ref. 273757
2	Drive	ISCM 8005 INTELLIGENT SERVO DRIVE
2	Extension Board	IO ISCM V1.2 EXTENSION BOARD
1	Ordenador	---
1	Fuente de Alimentación	---
2	Software	Easy Motion Studio y Matlab

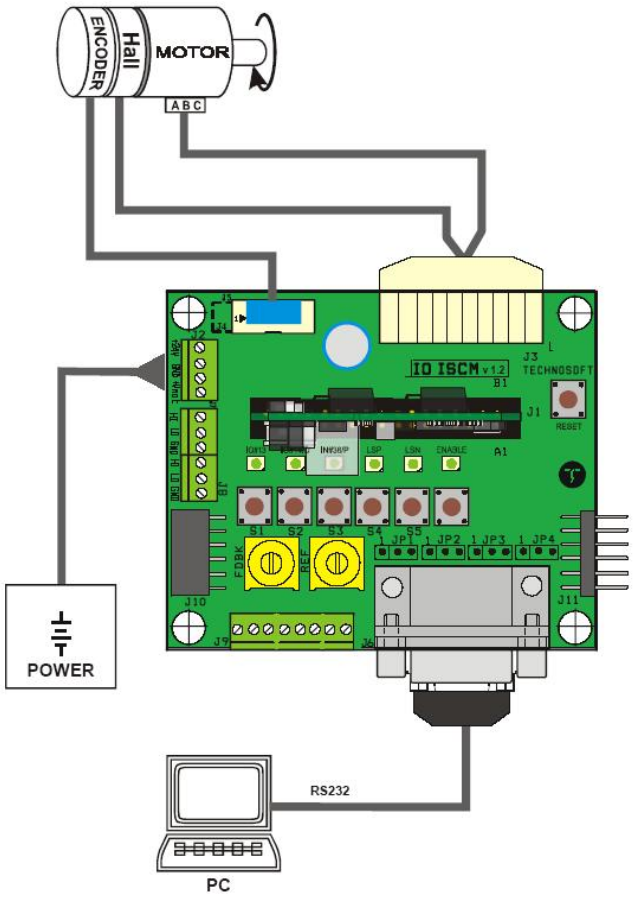


Figura 28 Esquema de conexiones general

En la Figura 28 se muestra esquemáticamente las conexiones del sistema y en la Figura 29 las conexiones en los componentes hardware reales.

Nota: Los software Easy Motion Studio y Matlab se detallan en los apartados 2.7 y 2.8, respectivamente.

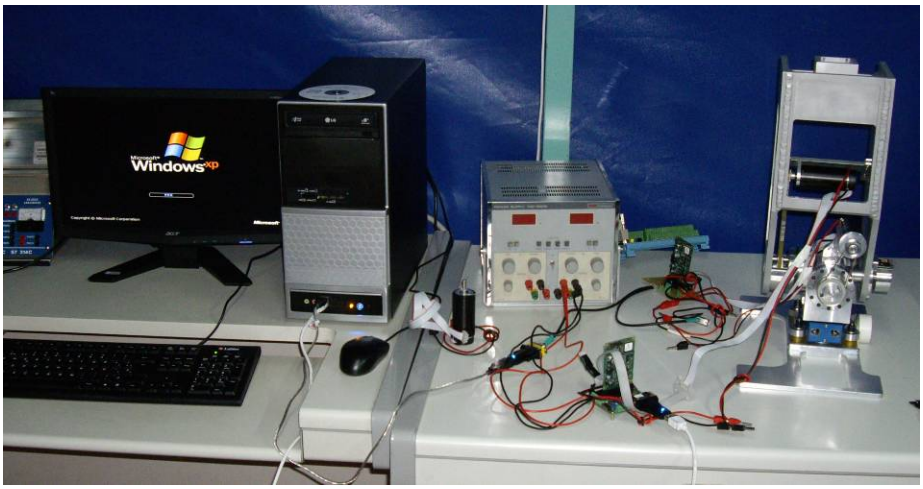


Figura 29 Conexiones hardware del prototipo

2.7. EASY MOTION STUDIO

Easy Motion Studio (EMS) es un software que permite configurar los Drive y motores usados en este proyecto. También es la herramienta que permite programar los movimientos que serán transmitidos al motor a través del Drive, facultando a éste, como ya se ha comentado, la capacidad de administrar las señales necesarias para dicho movimiento del motor. Adicionalmente permite comparar las trayectorias teóricas y reales para así analizar el grado de aceptación de la configuración que se ha realizado.

A continuación se muestra, a modo de esquema, los pasos a seguir para crear un nuevo proyecto con Easy Motion[3].

2.7.1. Creación de un Proyecto

El primer paso es la creación de un Nuevo Proyecto. En éste se incluirán los datos necesarios para el correcto funcionamiento del Drive y el motor. Estos datos permanecerán invariables durante todo el proyecto, ya que los definen físicamente.

Para crear un nuevo proyecto se seguirá la ruta: "*Proyect -> New Proyect".* Una vez hecho esto, se abre una ventana como la que vemos en la Figura 30. Al elegir la opción *New* se pasa a configurar el driver que se utilizará durante el proyecto. En el caso de este proyecto

la opción que hay que elegir es *"PLUG IN DRIVES -> ISCM8005 CANOPEN -> BRUSHED MOTOR -> Incremental Encoder, Tacho".*

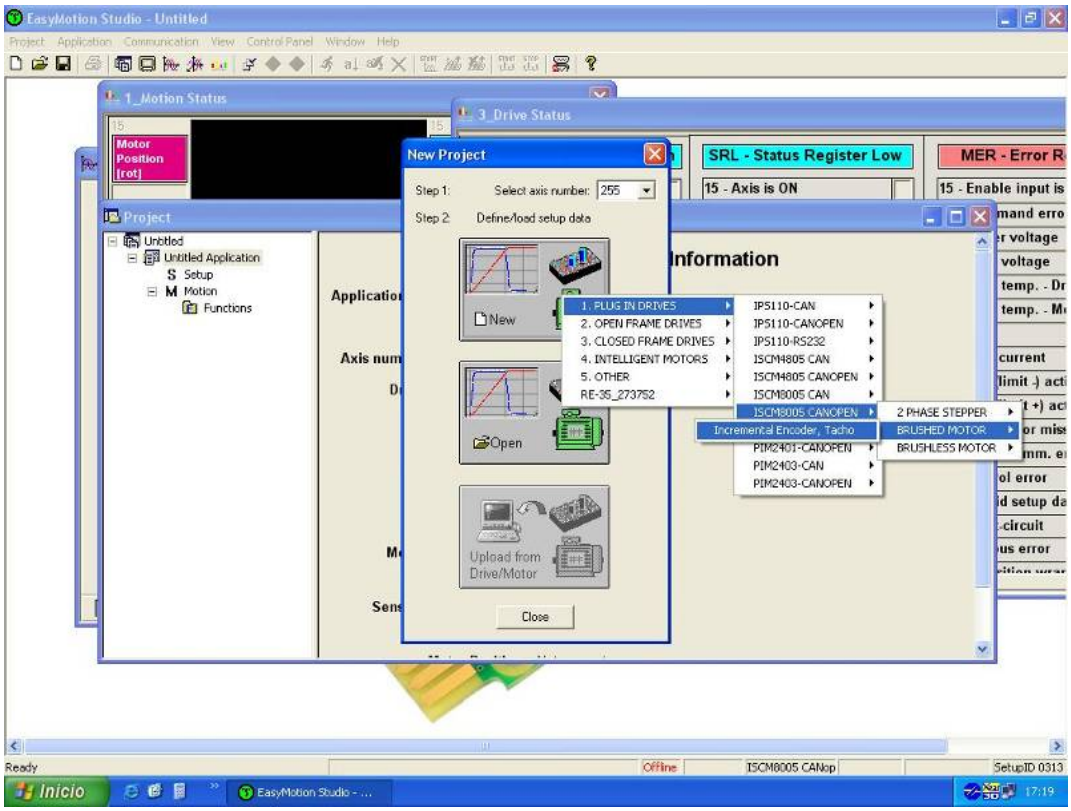


Figura 30 EMS - Crear proyecto I

Una vez elegida esta opción se abrirá la siguiente ventana (Figura 31), que resume las características del proyecto.

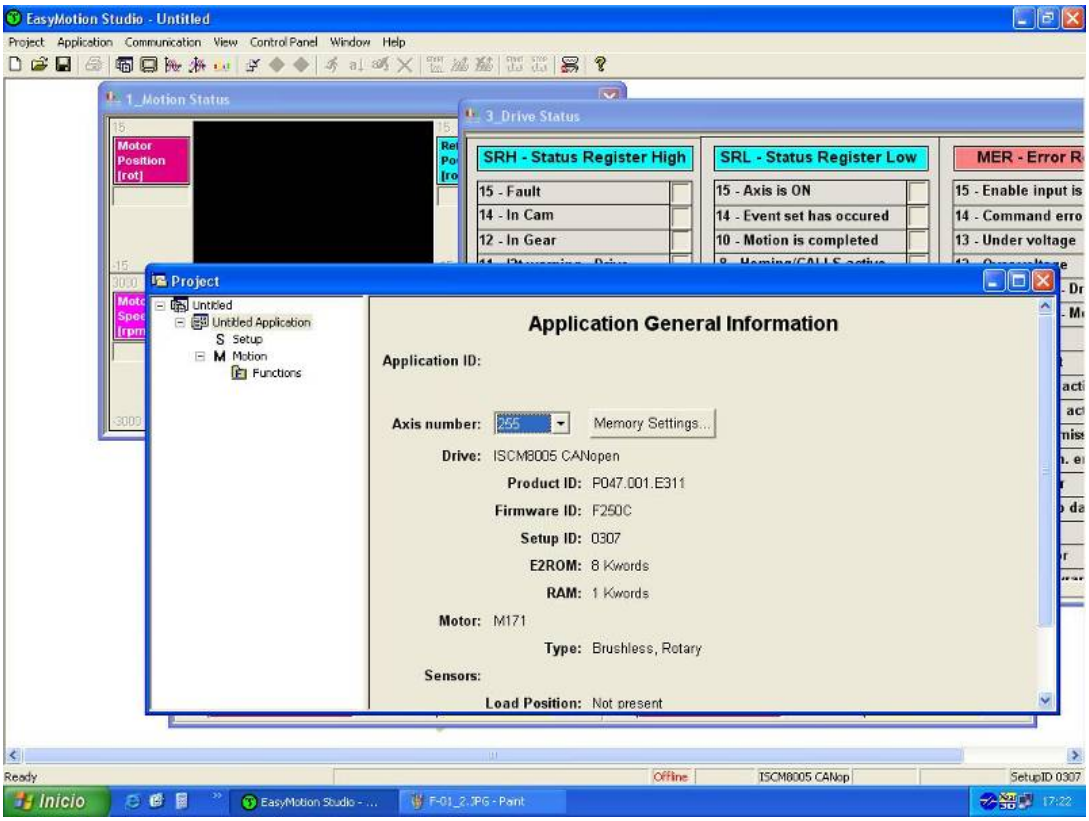


Figura 31 EMS - Crear proyecto II

Tal y como se muestra en la Figura 32, pinchando en "*SETUP -> New*" se podrá configurar el motor e introducir sus datos característicos. Los datos para definir el motor se obtienen de la hoja de características del mismo (ver Anexo I) y han de introducirse y grabarse en la ventana de la Figura 33. Además, se seleccionará la opción *Incremental encoder on motor* y el modo transmisión: *Rotary to rotary*.

En los Capítulos 3 y 4 se incluye una descripción más detallada del software.

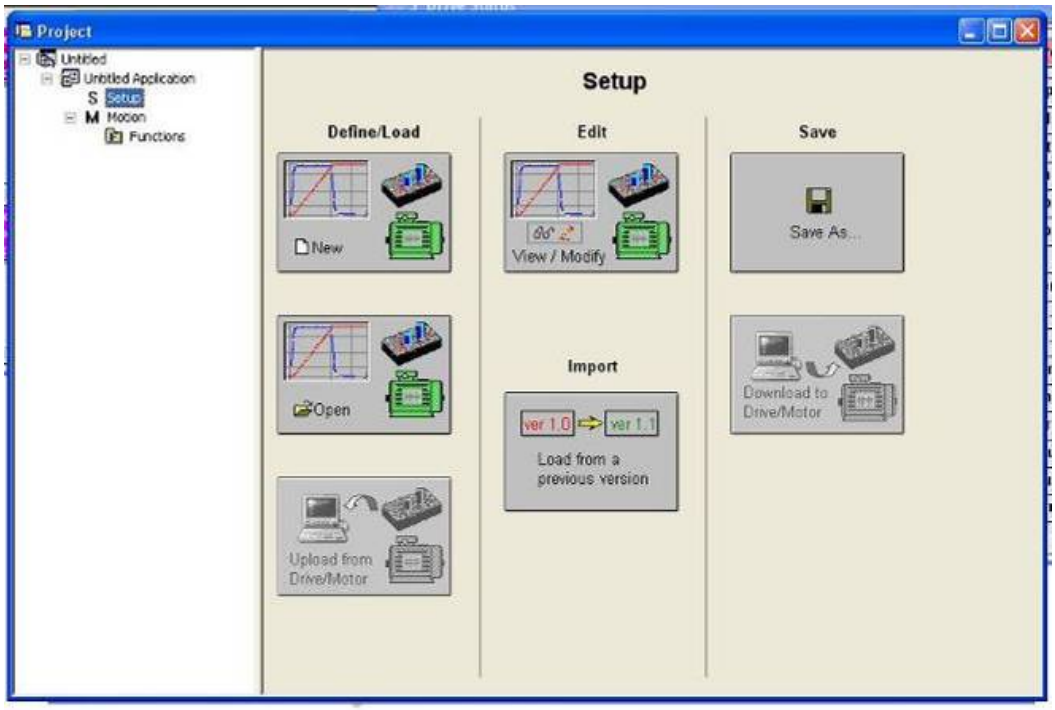


Figura 32 EMS - Setup

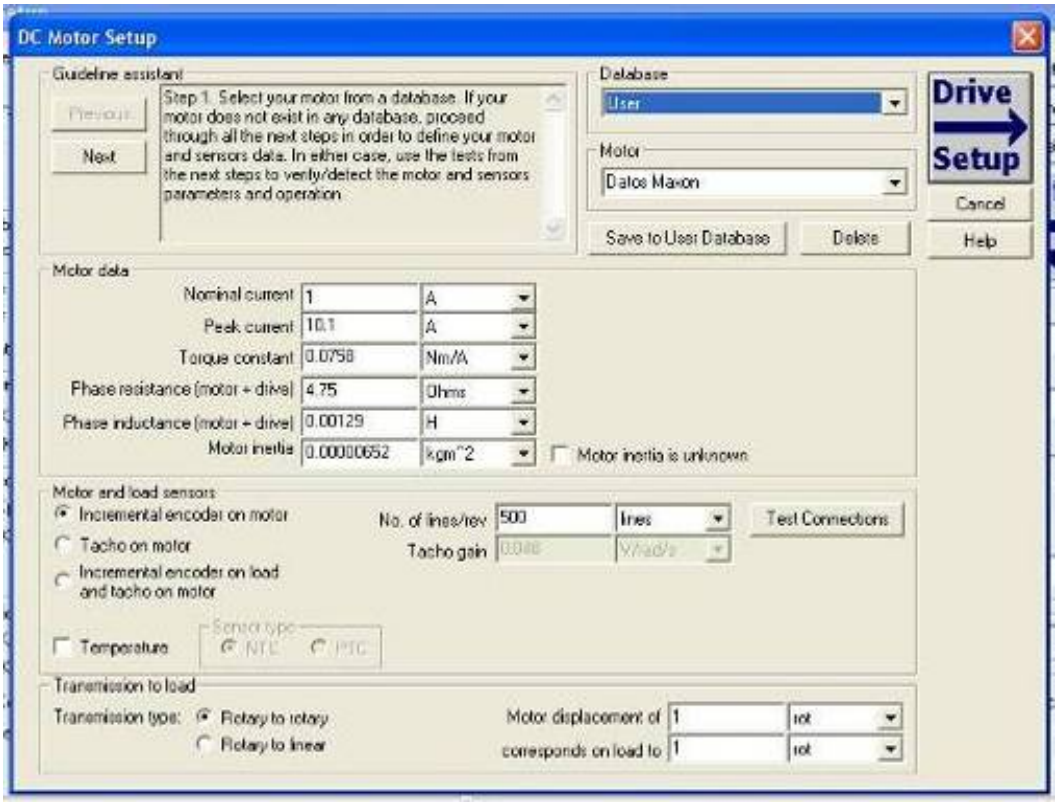


Figura 33 EMS - DC Motor Setup

2.8. MATLAB

MATLAB es un lenguaje de computación técnica de alto nivel y un entorno interactivo para desarrollo de algoritmos, visualización de datos, análisis de datos y cálculo numérico. Con MATLAB, se pueden resolver problemas de cálculo técnico más rápidamente que con lenguajes de programación tradicionales, tales como C, C++ y FORTRAN.

En Matlab se pueden usar una amplia gama de aplicaciones que incluyen procesamiento de señales e imágenes, comunicaciones, diseño de sistemas de control, sistemas de prueba y medición, modelado y análisis financiero y biología computacional. Los conjuntos de herramientas complementarios (colecciones de funciones de MATLAB para propósitos especiales, que están disponibles por separado) amplían el entorno de MATLAB permitiendo resolver problemas especiales en estas áreas de aplicación.

En este proyecto se han utilizado dos toolboxes de Matlab:

- Toolbox de identificación (Ident)
- Toolbox de simulación (Simulink)

A continuación se resumen brevemente:

2.8.1. Ident

Ident es una toolbox de Matlab que permite obtener el modelo matemático, es decir, la función de transferencia de un sistema físico. Para ello es necesario proporcionar a la herramienta dos archivos con los datos de entrada y salida [4].

Usando el comando “ident” aparecerá la ventana de la Figura 34:

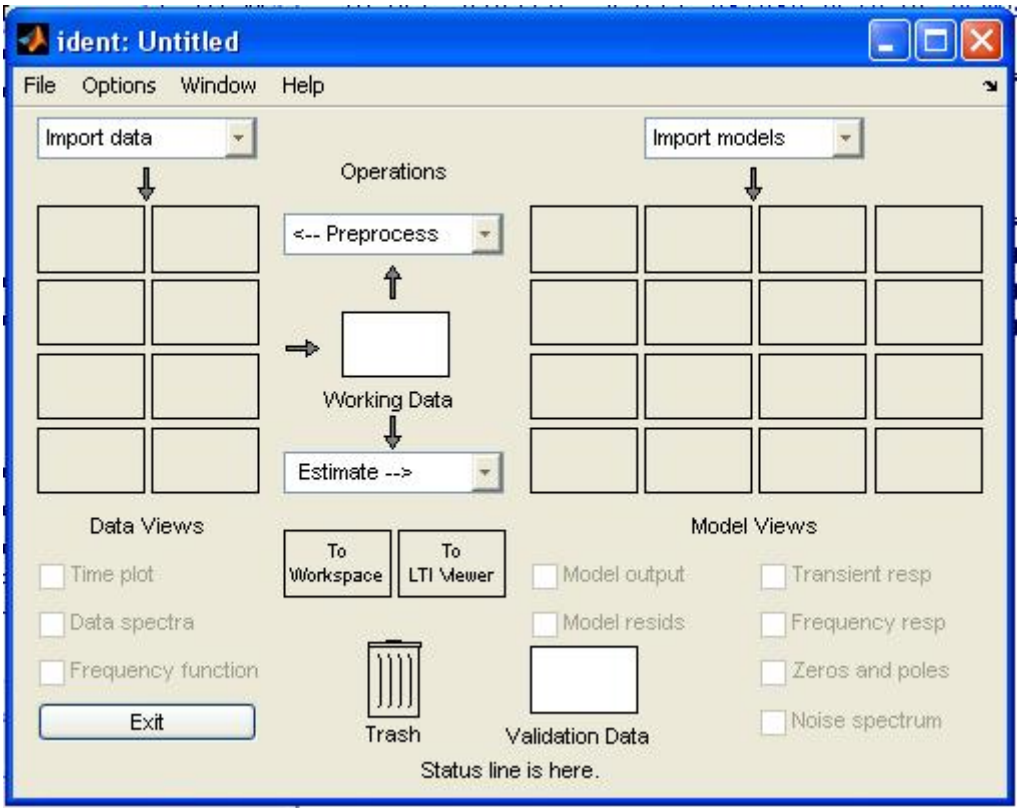


Figura 34 IDENT

Aquí se importarán los datos obtenidos en la simulación. Para importarlos se debe poner *time domain data* donde esta *import data* y aparecerá esta ventana de la Figura 35:

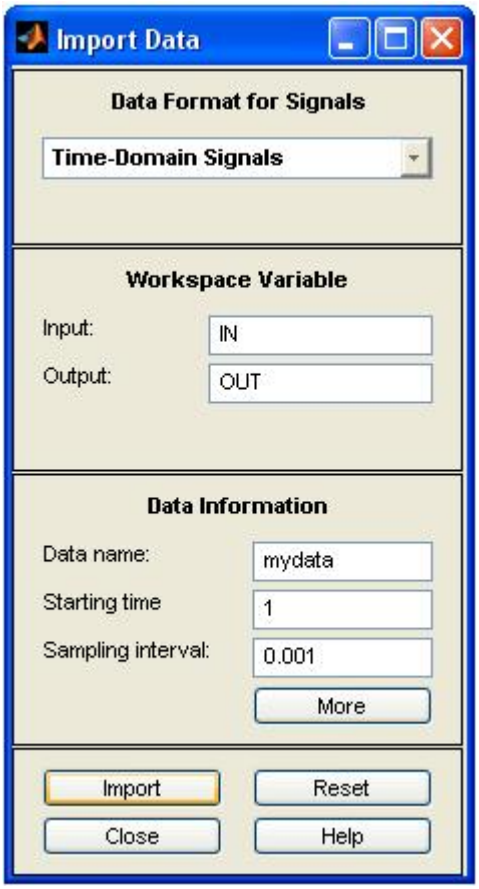


Figura 35 Ident - Importar datos en el dominio del tiempo

- Input: Se escribirá el *"Nombre de la variable", en este caso IN*

- Output: Se escribirá *"Nombre de la variable", OUT*

Previamente hay que generar la entrada (IN), el modo de generarla se representa en la Figura 36.

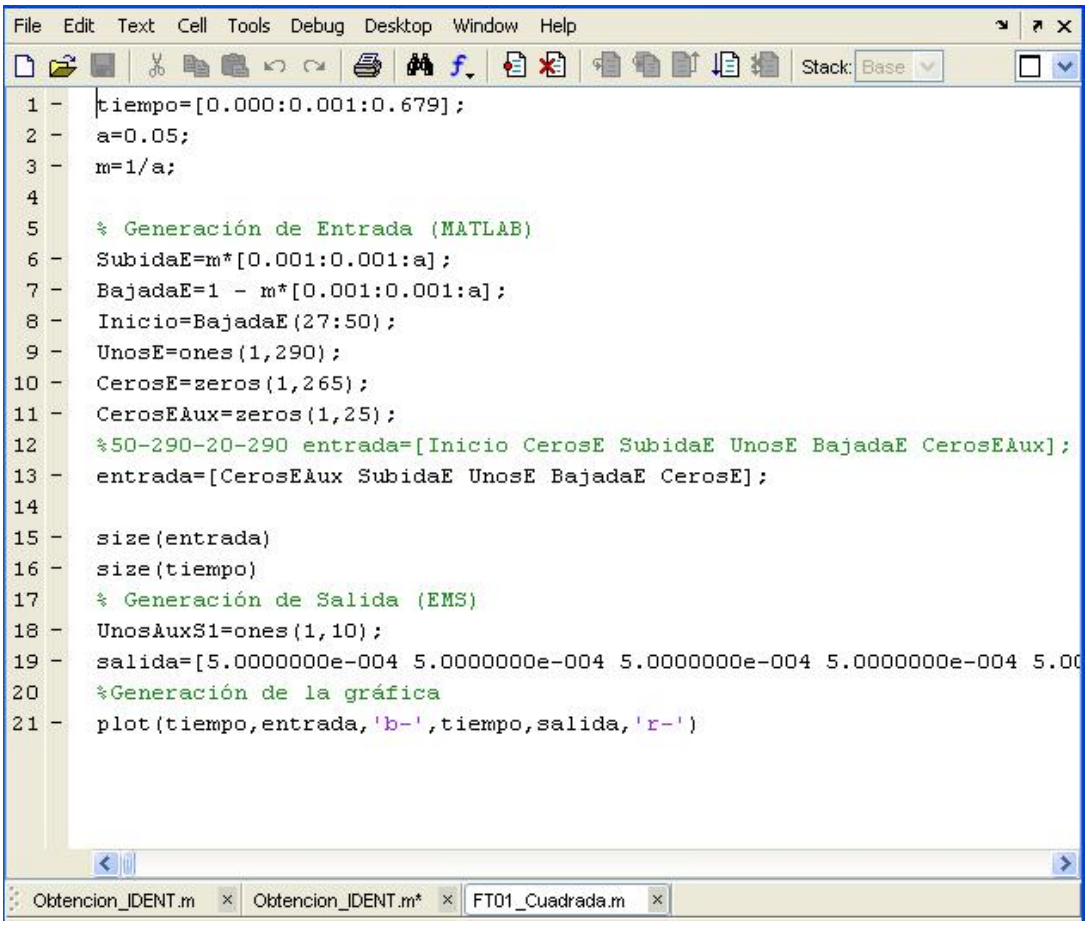


Figura 36 Generar entrada escalón

La variable OUT será la salida generada del EMS, la cual hay que procesar con el software explicado en el apartado 2.9.

Tras importar los datos aparece la ventana de la Figura 37:

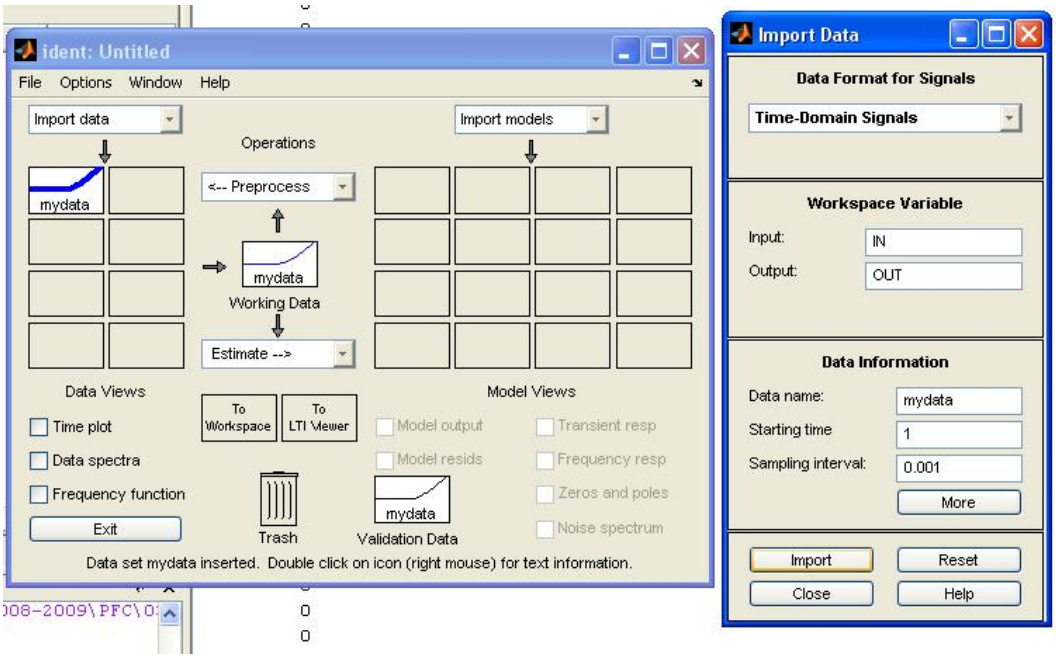


Figura 37 IDENT - Import

Es importante que la FT sea de segundo grado, por lo que se fuerza a que genere un sistema ARX221 (ver Figura 38).

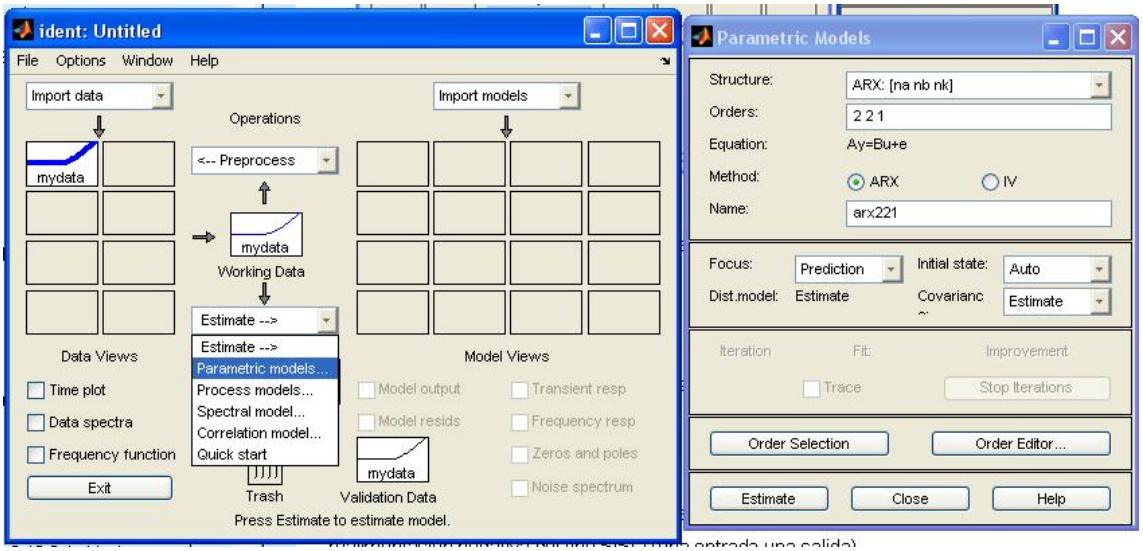


Figura 38 IDENT - ARX221

Haciendo click en el botón "*estimate*" aparece el modelo representado (Figura 39):

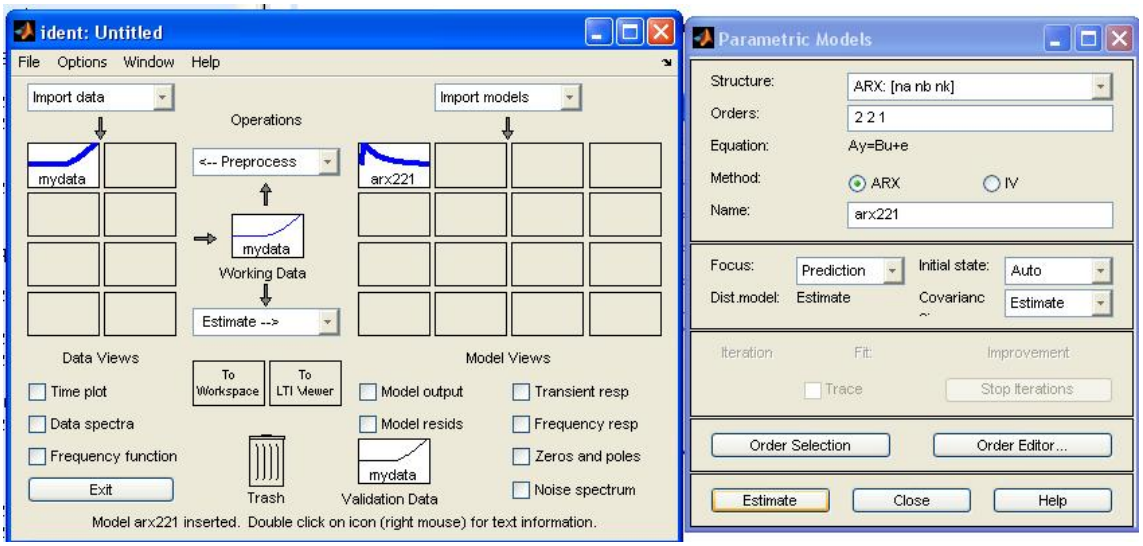


Figura 39 IDENT - Estimate

Se pueden activar las opciones que se deseen para obtener la información necesaria para el estudio del sistema.

Para obtener la función de transferencia del modelo construido hay que arrastrar el modelo generado (ARX221) a 

.

Una vez realizado este paso basta con volver a la consola de Matlab y teclear el nombre del modelo (arx221 en este caso) y se obtiene la FT (Figura 40).

```
>> arx221
Discrete-time IDPOLY model: A(q)y(t) = B(q)u(t) + e(t)
A(q) = 1 - 1.832 q^-1 + 0.8349 q^-2

B(q) = 0.1551 q^-1 - 0.1524 q^-2

Estimated using ARX from data set mydata
Loss function 1.09897e-007 and FPE 1.11198e-007
Sampling interval: 0.001
```

Figura 40 IDENT - FT

Nota1: En el DVD adjunto se incluyen todos los archivos y simulaciones realizadas.

Nota2: En el Anexo III está disponible la hoja de características de ésta función

2.8.2. Simulink

2.8.2.1. INTRODUCCIÓN

Simulink es un entorno de programación visual, que funciona sobre el entorno de programación Matlab.

Es un entorno de programación de más alto nivel de abstracción que el lenguaje interpretado Matlab (archivos con extensión .m). Simulink genera archivos con extensión .mdl (de "model").

Simulink viene a ser una herramienta de simulación de modelos o sistemas, con cierto grado de abstracción de los fenómenos físicos involucrados en los mismos. Se hace hincapié en el análisis de sucesos, a través de la concepción de sistemas (cajas negras que realizan alguna operación).

Se emplea arduamente en Ingeniería Electrónica en temas relacionados con el procesamiento digital de señales (DSP), involucrando temas específicos de ingeniería biomédica, telecomunicaciones, entre otros. También es muy utilizado en Ingeniería de Control y Robótica [5].

2.8.2.2. FUNCIONAMIENTO

El primer paso es pasar la FT de discreto a continuo, para ello se procede como indica la Figura 41 (función d2cm de matlab):

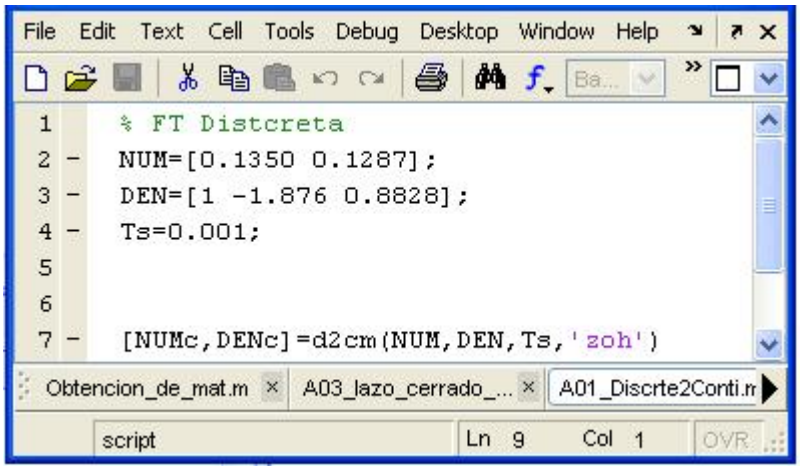


Figura 41 Simulink - d2cm

Las variables NUMc y DENc almacenan los valores del numerador y denominador de la FT en continuo.

Usando el comando “simulink” aparecerá la ventana de bloques de simulink, una vez ingresado todos los necesarios para este proyecto se obtiene la Figura 42:

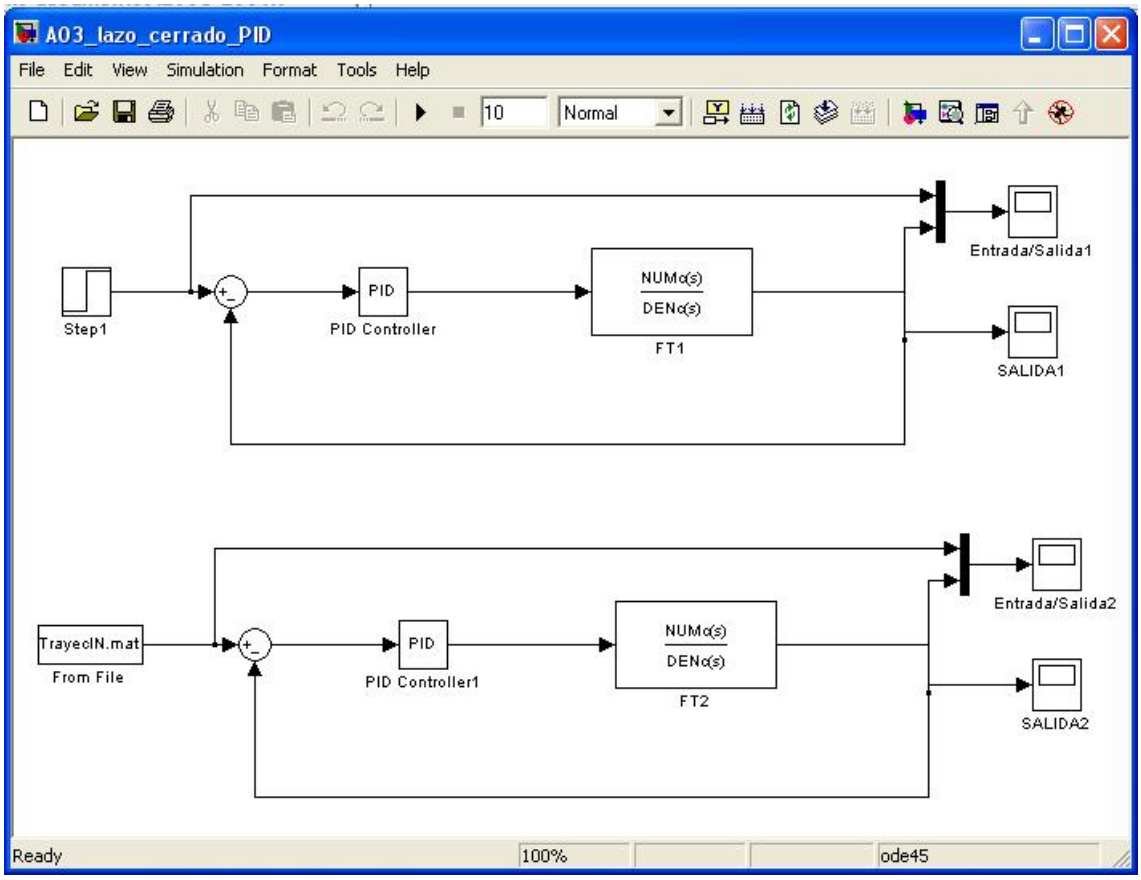


Figura 42 Simulink - Bloques

En el Capítulo 4, se aporta más información sobre estas dos herramientas.

2.9. PROGRAMAS PARA LA TRANSFORMACIÓN Y ADAPTACIÓN DE DATOS

Para procesar los datos que se obtienen en el EMS es necesario adaptarlos a un formato que entienda MATLAB y viceversa. Por éste motivo se han desarrollado dos aplicaciones que resuelven estos problemas.

El código de estos programas, realizados en C, está disponible en el DVD adjunto y en el Anexo IV.

Capítulo 3

CONFIGURACIÓN DE LAZOS DE CONTROL DEL MOTOR EN EASY MOTION STUDIO

3.1. AJUSTE AUTOMÁTICO DEL PID INTERNO

Previamente al ajuste del PID es necesario realizar las conexiones oportunas. En el caso de la Extension Board comercial, los conectores necesarios para las conexiones mencionadas anteriormente son (ver Figura 43, Figura 44 y Figura 45):

- J1: Driver
- J2: Alimentación
- J3: Señales de control del motor
- J5: Encoder
- J6: RS-232

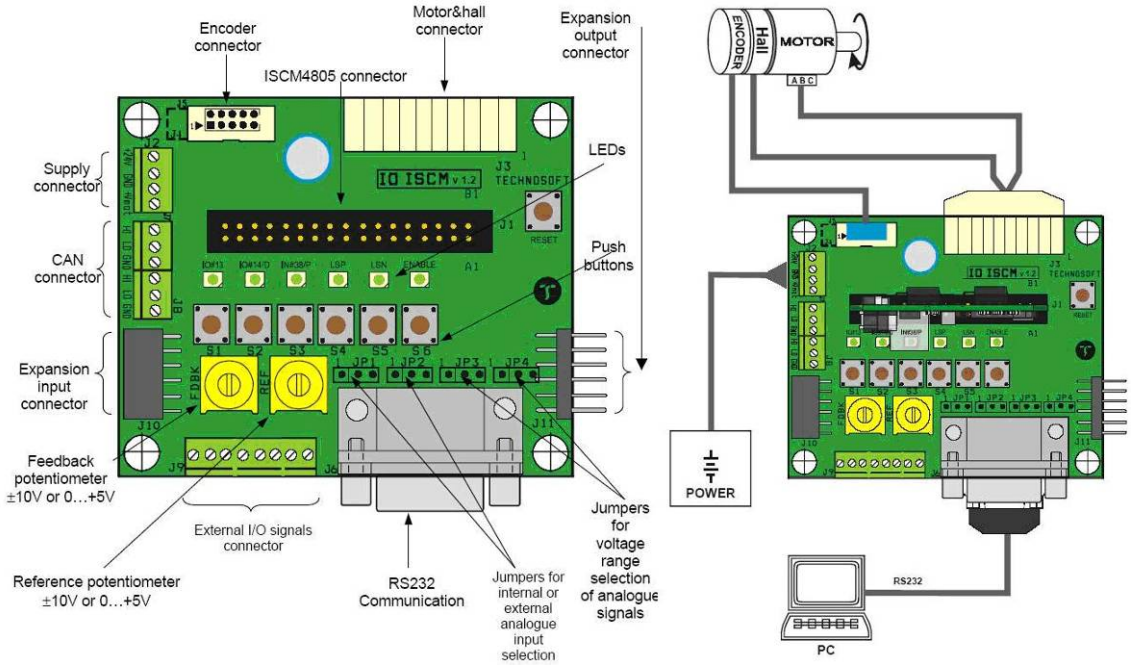


Figura 43 Esquema conexiones Extension Board comercial

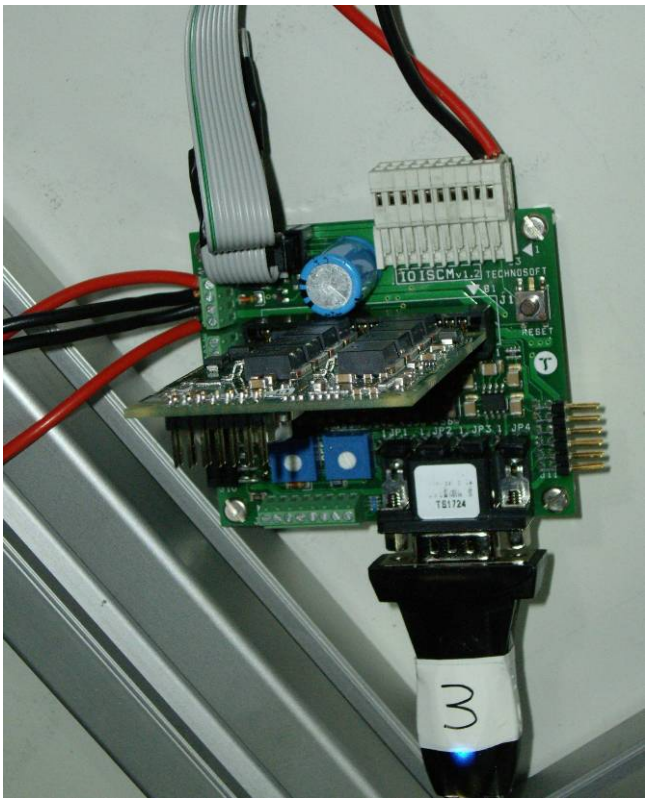


Figura 44 Conexiones Extension Board comercial

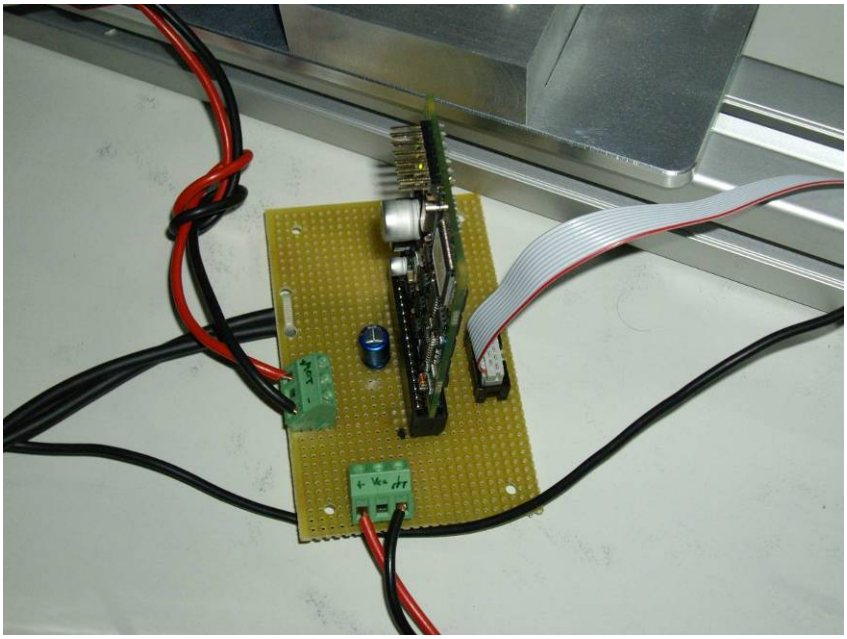


Figura 45 Conexiones Extension Board uc3m

Nota: V_{mot} en Figura 45 es el equivalente al J3 de la placa comercial.

Una vez familiarizado con todos elementos del sistema, el primer paso es ajustar el PID propio del motor, cerrando internamente el lazo y obteniendo su Función de Transferencia (FT).

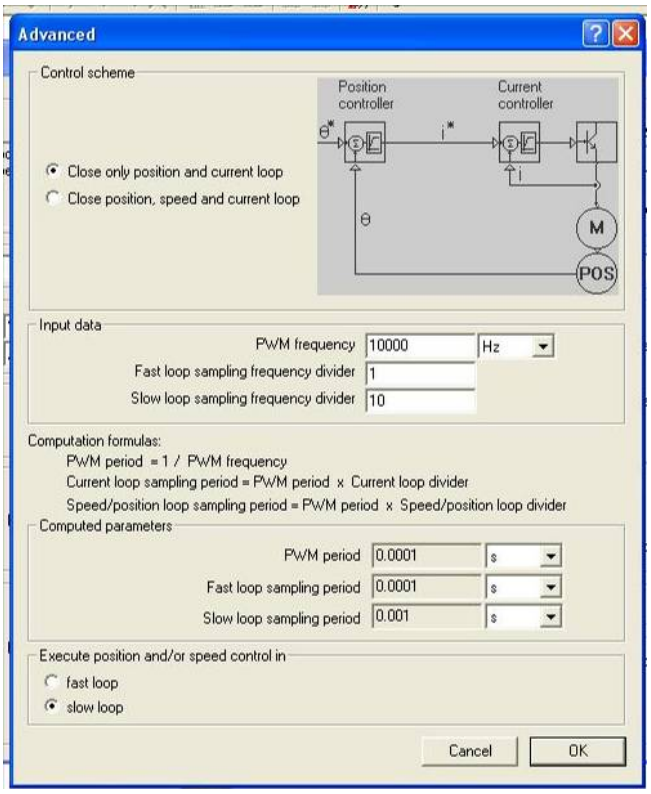


Figura 46 Modo de control posicional

Para ello es necesario utilizar el software EMS que a través de su interface permite modificar manualmente el PID interno del motor, pudiendo además seleccionar el modo de control para el sistema (Figura 46). En el recuadro “Control schume” se observa que hay dos posibles modos de control:

- “Close Only position and current loop”
- “Close position, Speedy and current loop”

En el caso de este proyecto se elegirá la primera opción, es decir, “Close Only position and current loop”, ya que el objetivo principal es obtener una buena precisión de movimientos, más allá que la rapidez de respuesta.

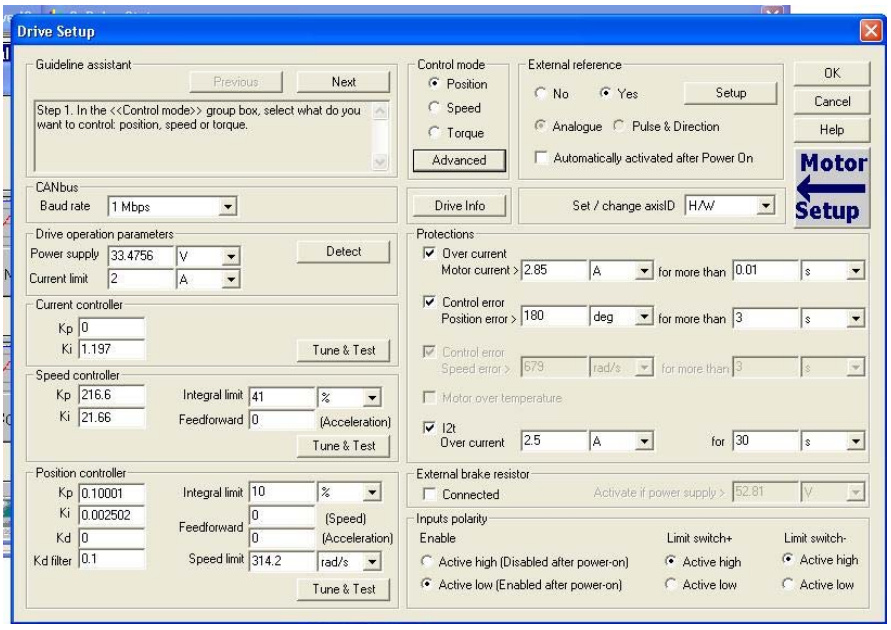


Figura 47 EMS - Drive Setup

En el apartado 2.7.1 se comentó la operativa a seguir para crear un proyecto en EMS. Una vez creado un proyecto nuevo, al pulsar sobre Drive Setup (ver Figura 33) se abre una nueva ventana (Figura 47). En esta se puede elegir el modo de control pulsando sobre Advanced, que como ya se ha comentado será posicional.

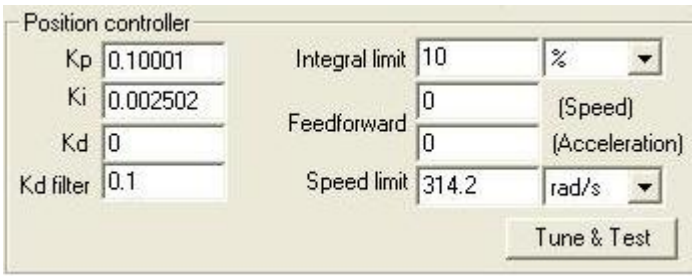


Figura 48 Tune & Test

Tras realizar esta configuración previa, se pasará a configurar los controladores internos del driver. Para realizar las primeras pruebas y obtener los valores K_p, k_i y k_d es recomendable ajustar estos controladores automáticamente, sabiendo a priori que los resultados obtenidos de este modo no serán los mejores, siendo, en la mayoría de los casos, necesario ajustarlos manualmente estudiando los resultados obtenidos. Esto se consigue con la opción *"Tunig Test"* (Figura 48 y Figura 49).

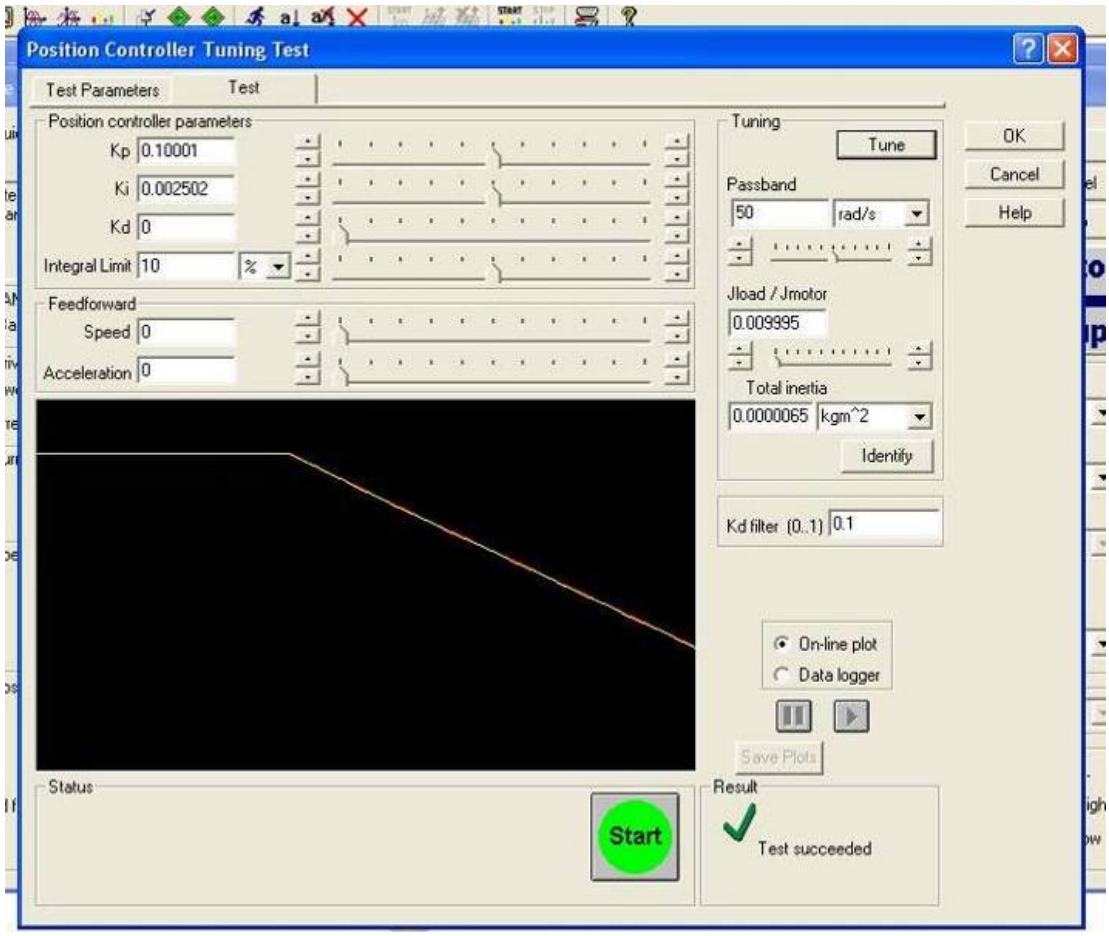


Figura 49 EMS - Position Controller Tunning Test

En la Figura 49 puede observarse como el resultado del test tras realizar el tuning es satisfactorio. Además la grafica muestra que los valores de las constantes son aceptables, ya que la salida de posición se ajusta sin error apreciable a la referencia.

Sin embargo, en el Capítulo 4 al introducir entradas tipo “escalón” se aprecia que la salida no sigue a la señal de referencia. Por este motivo será necesario ajustar manualmente el PID, procedimiento que se indica en el Capítulo 4.

Capítulo 4

CARACTERIZACIÓN DE LOS MOTORES DEL TOBILLO EN MATLAB

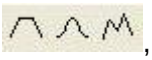
4.1. INTRODUCCIÓN

Llegado a este punto, el siguiente paso es obtener la Función de Transferencia del sistema. Para ello se utilizará la Toolbox Ident de Matlab.

Como se comentó en el apartado 2.8.1, es necesario disponer de un fichero con los datos de una señal de entrada (entrada escalón – "IN") y un segundo fichero con los datos de la respuesta del sistema ("OUT"). A continuación se detalla el método utilizado para obtener estos dos archivos.

4.1.1. Generación de trayectorias – "IN"

La generación de las trayectorias que seguirán los motores se utilizará en un primer momento para calcular la Función de Transferencia del sistema y una vez obtenida ésta, para generar los movimientos de los propios motores.

Con el fin de obtener un modelo matemático del sistema, inicialmente se programará una entrada escalón, para ello se hará clic en uno de estos tres botones  , abriéndose la ventana de la Figura 50.

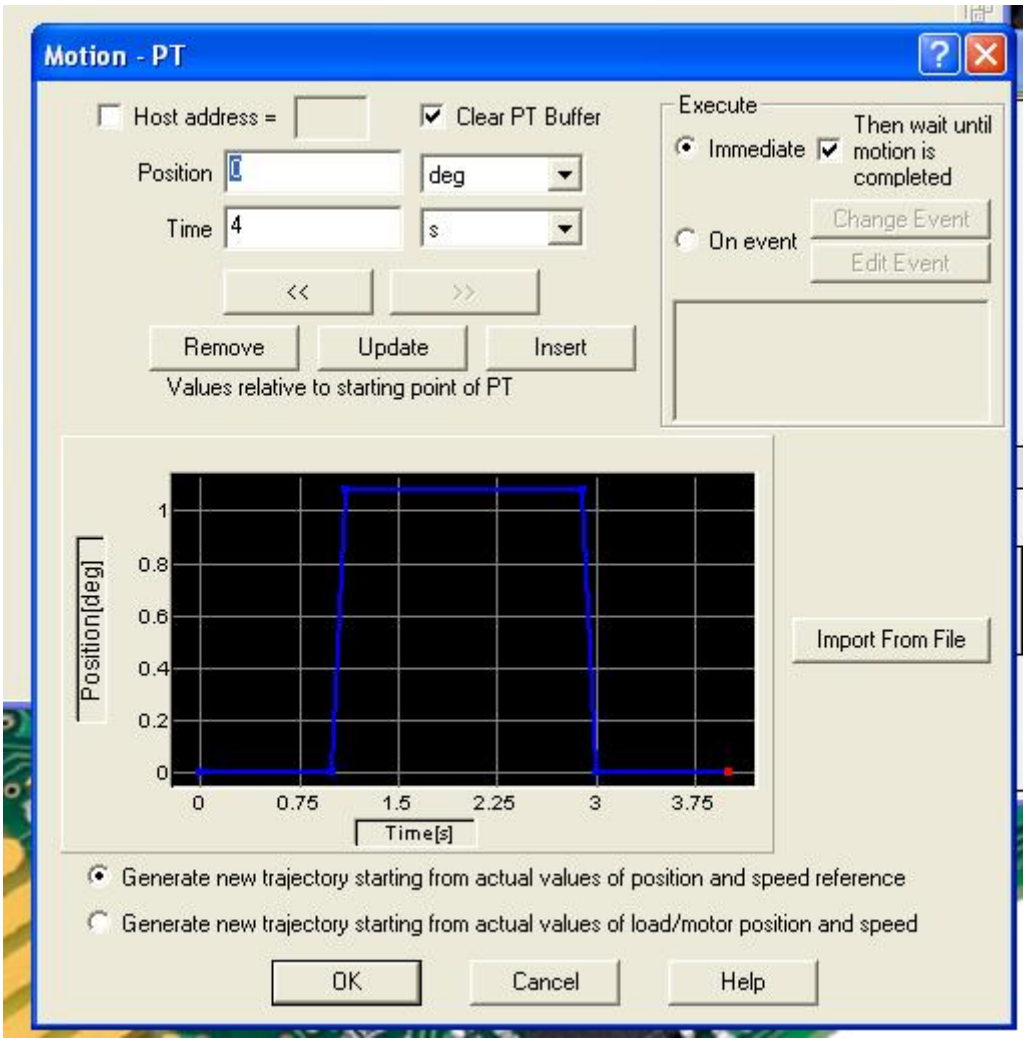


Figura 50 Entrada escalón

Llegado a este punto, es necesario introducir los valores de posición y tiempo correspondientes a la trayectoria escalón que se deseen generar. A continuación se representa un posible ejemplo de los valores a introducir:

Posición (deg)	Tiempo (s)
0	0
0	1
1	1.1
1	2.9
0	3
0	4

Obteniéndose una ventana como la de la Figura 51.

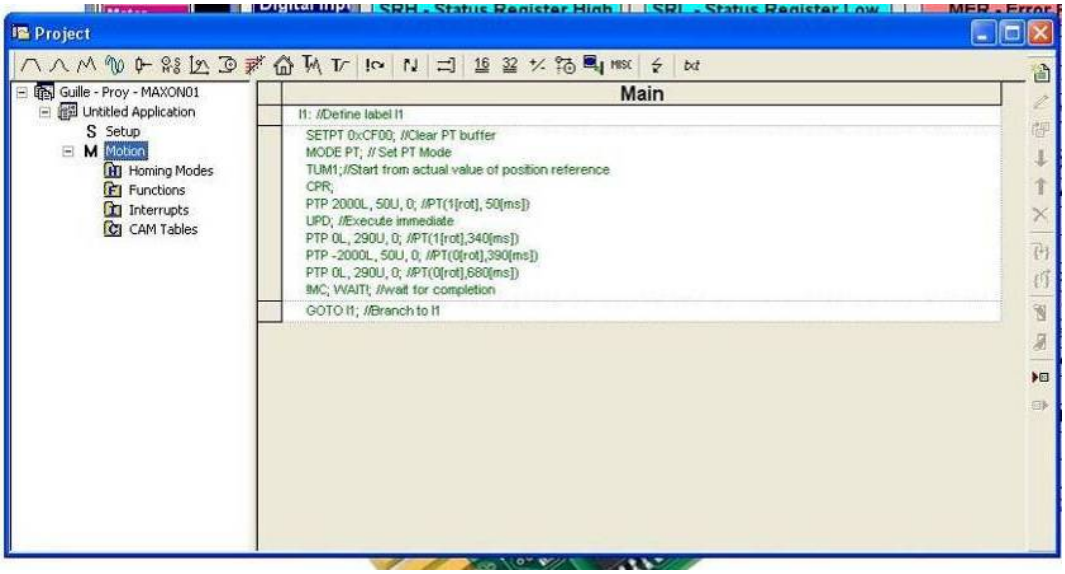
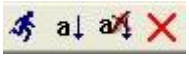


Figura 51 EMS - Programación de Trayectorias

Cuando la trayectoria con entrada escalón ha sido programada, es necesario introducir ésta en la memoria del driver. Para ello, una vez realizadas las conexiones oportunas, se hará click en el botón  .

Los botones  sirven para interactuar con los motores:

- El primero, los pone en funcionamiento, su tecla de acceso directo es F5
- El segundo y el tercero ejecutan o detienen paso a paso la trayectoria programada
- El último detiene el movimiento de los motores

4.1.2. Respuesta del sistema – "OUT"

Una vez cargados los datos en el driver del motor y estando en funcionamiento, se hará click en el botón  para visualizar y estudiar las respuesta del sistema (Figura 52).

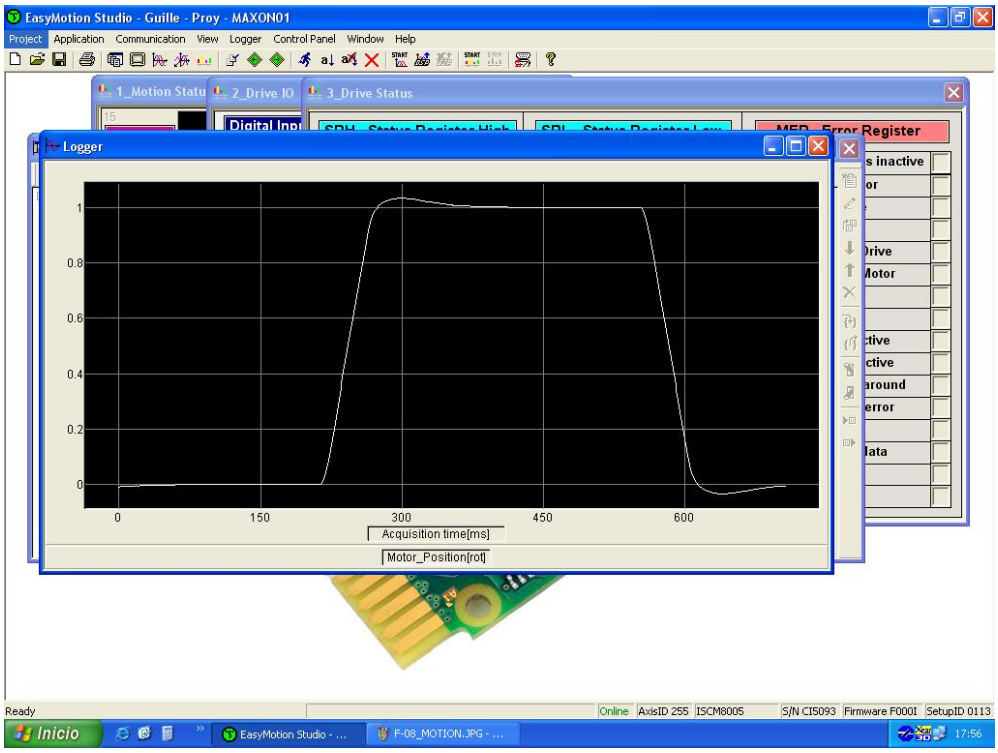


Figura 52 EMS – Logger

Este paso tiene una gran importancia. Como se representa en Figura 53, se puede obtener desde el Logger un fichero .txt con la información contenida en la gráfica y ser ésta estudiada y tratada en Matlab.

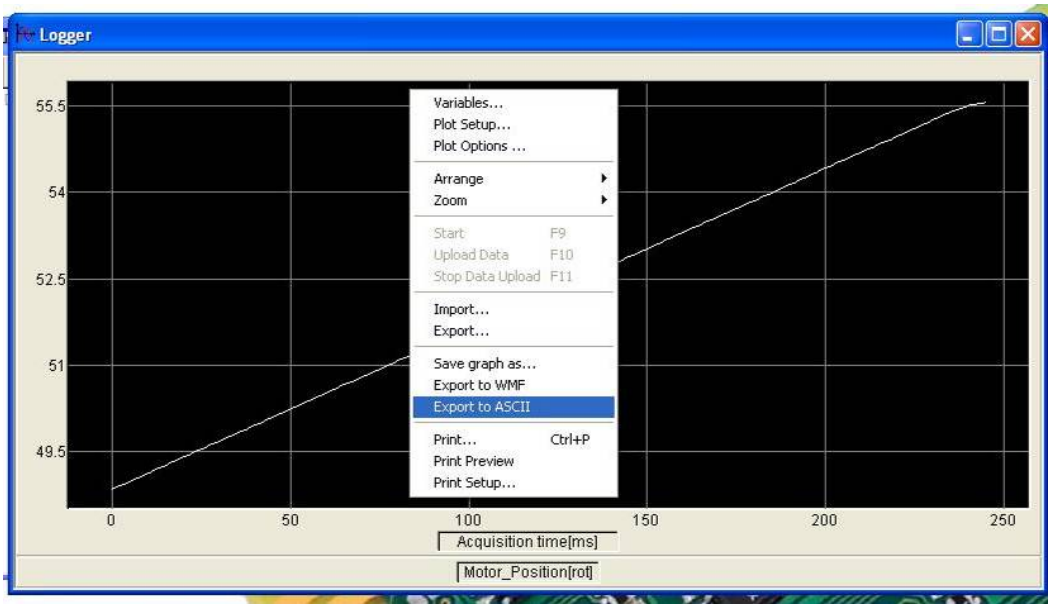


Figura 53 Exportar a .txt la salida desde logger

En Figura 54 se muestran las opciones de configuración que soporta el Logger de EMS. Entre las que destacan:

- Aumentar el buffer para visualizar un mayor rango de datos
- Controlar la posición y velocidad
- Corriente instantánea

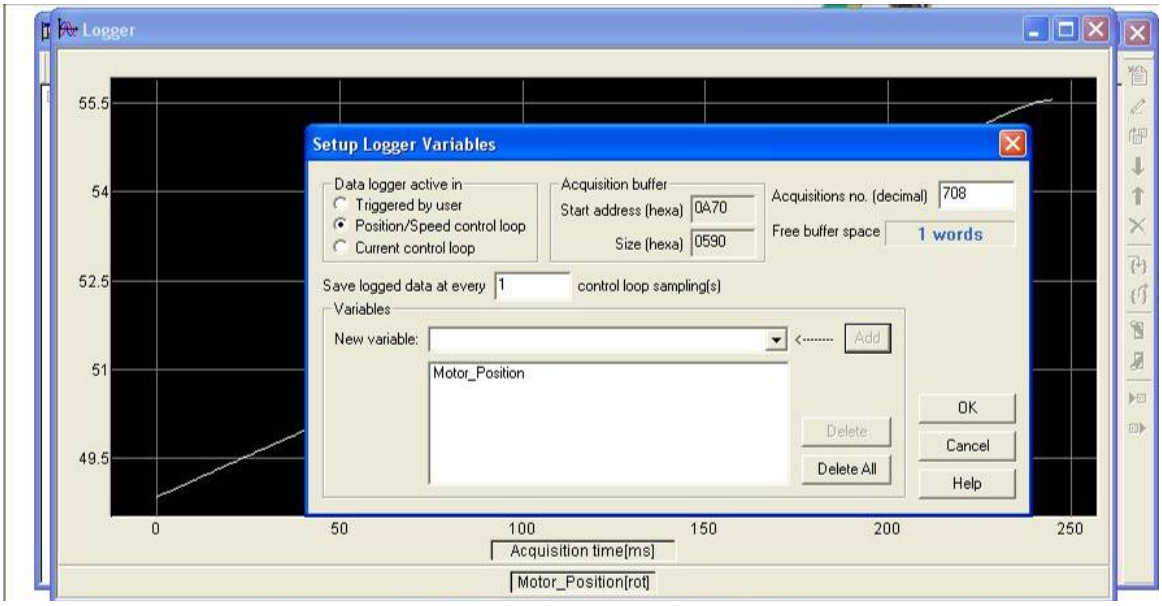


Figura 54 Opciones adicionales de Logger

4.2. FUNCIÓN DE TRANSFERENCIA

Inicialmente y haciendo uso de EMS se ajusta el PID interno que está disponible en el driver del sistema. Tal y como se ha comentó el apartado 2.7.1, el primer ajuste se realizará de manera automática utilizando las herramientas propias del sistema. A priori, éste no será el más apropiado para el sistema pero será valido como primera aproximación y sin duda servirá para hacer un primer estudio de las respuestas del sistema.

Una vez que se tienen los valores del PID aproximado, se comenzará a utilizar MATLAB. Serán las herramientas IDENT y SIMULINK las encargadas de simular el sistema y así ver el funcionamiento teórico esperado.

Por lo tanto, con IDENT se obtiene la función de transferencia (FT) el sistema, haciendo especial hincapié en que la función de transferencia obtenida sea de segundo grado para simplificar lo máximo posible dicho sistema (modelo ARX221).

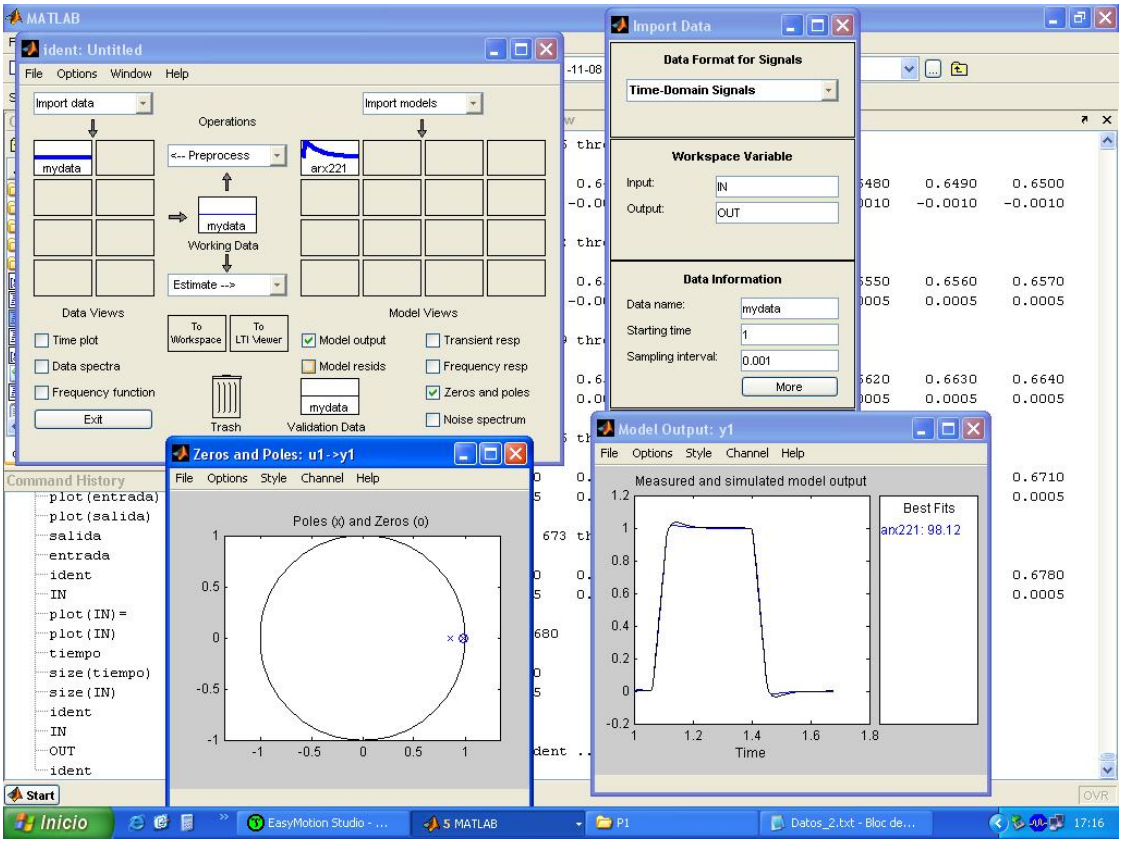


Figura 55 Ident - FT1

Haciendo uso de los datos obtenidos en el autoajuste de los valores del PID interno del motor e introduciendo éstos en la herramienta Ident, se consigue la FT deseada. Utilizando un ARX221 se obtiene la Figura 55. En ésta se observan dos polos y un cero dentro del círculo unidad, por lo que el sistema es estable, consiguiendo con ello una aproximación del 98.12%. El modelo matemático que representa el sistema del tobillo en lazo abierto es (ver Figura 56):

$$FT = \frac{0.1551 \cdot Z - 0.1524}{Z^2 - 1.832 \cdot Z + 0.8349}$$

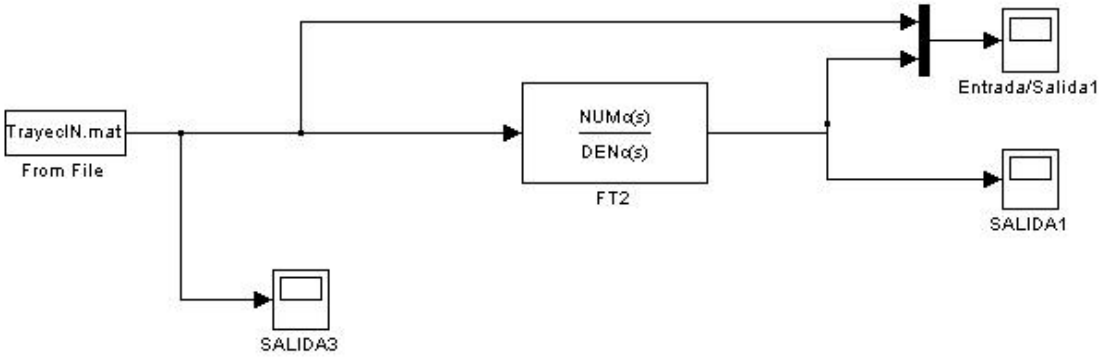


Figura 56 Esquema del sistema en Lazo Abieto

4.2.1. Ajuste PID interno

Una vez obtenida la FT se simula el comportamiento en SIMULINK. Se utilizarán dos tipos de entradas, una inicial que será de tipo escalón y otra que se aproxima a un posible movimiento real del tobillo.

El siguiente paso es analizar la respuesta obtenida, teniendo en cuenta si el sistema es subamortiguado o sobreamortiguado.

Una vez llegado a este punto es posible que la respuesta obtenida no se ajuste tal y como se desea, exponiendo al sistema a comportamientos no precisos, existiendo riesgo de vibraciones y movimientos no controlados. Por lo tanto es necesario reducir estos errores.

Como inicialmente se ha ajustado el PID interno automáticamente, cabe esperar que esta respuesta no sea adecuada, por ello es necesario ajustar el sistema las veces que sea necesario utilizando alguna técnica como las comentadas en el apartado 5.2.

En la Figura 57 se representa la respuesta del sistema simulando el sistema con la función de transferencia ya calculada. Se puede observar que la respuesta no es del todo adecuada, pudiéndose mejorar.

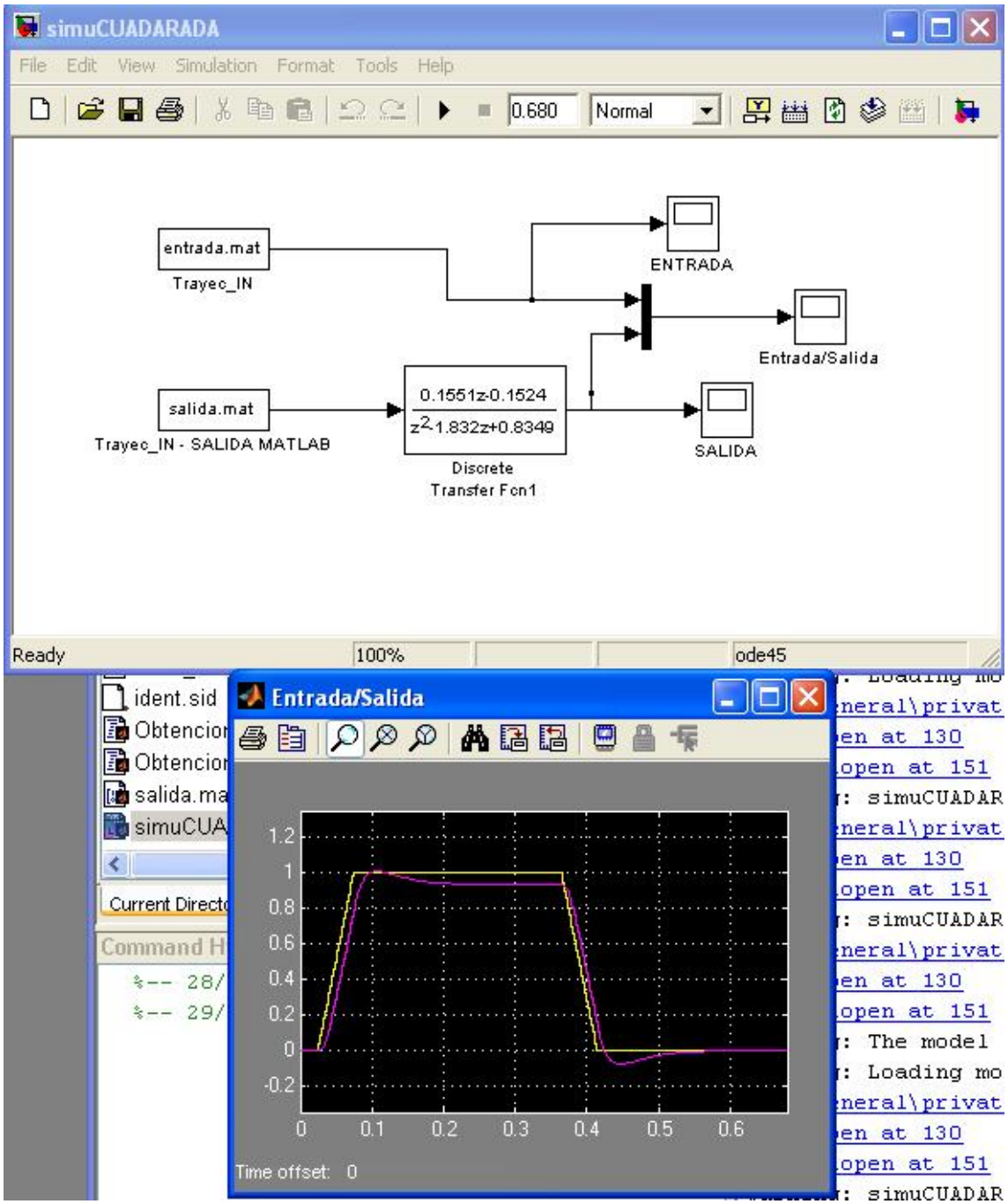


Figura 57 FT1 – Ki=0.00252

Siguiendo los paso de ajuste manual de PID explicado en el apartado 5.2, se obtiene un mejora respecto al sistema anterior (ver Figura 58).

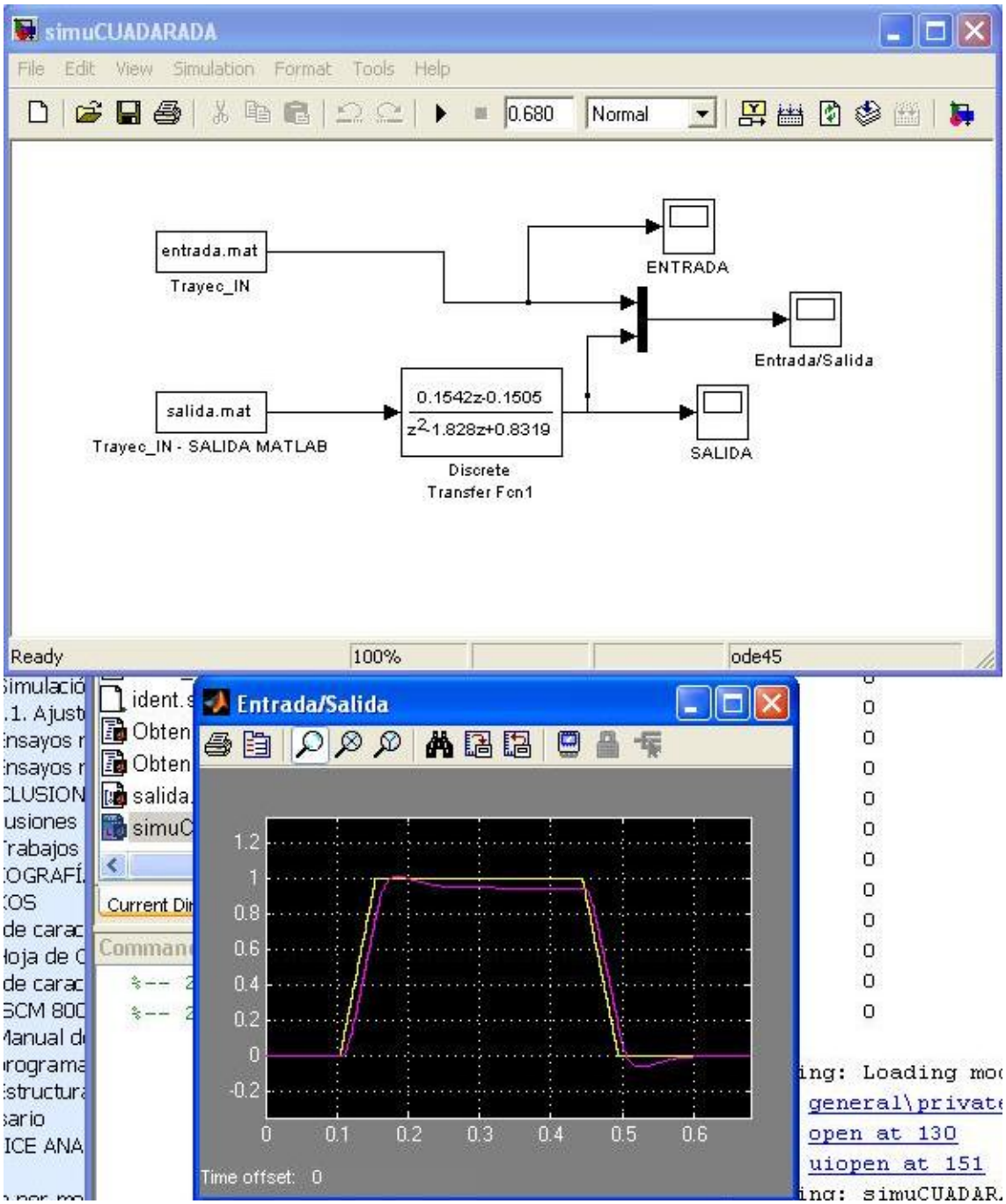


Figura 58 FT2 - Ki=0.00351

Finalmente, en la Figura 59, Figura 60 y Figura 61 se representa la respuesta del sistema utilizada. La respuesta ante entra entrada de datos reales de movimiento (Figura 61) se ajusta muy bien, por lo que tomamos por bueno ésta FT como representación del tobillo.

$$FT = \frac{0.1352 \cdot Z - 0.1287}{Z^2 - 1.876 \cdot Z + 0.8828}$$

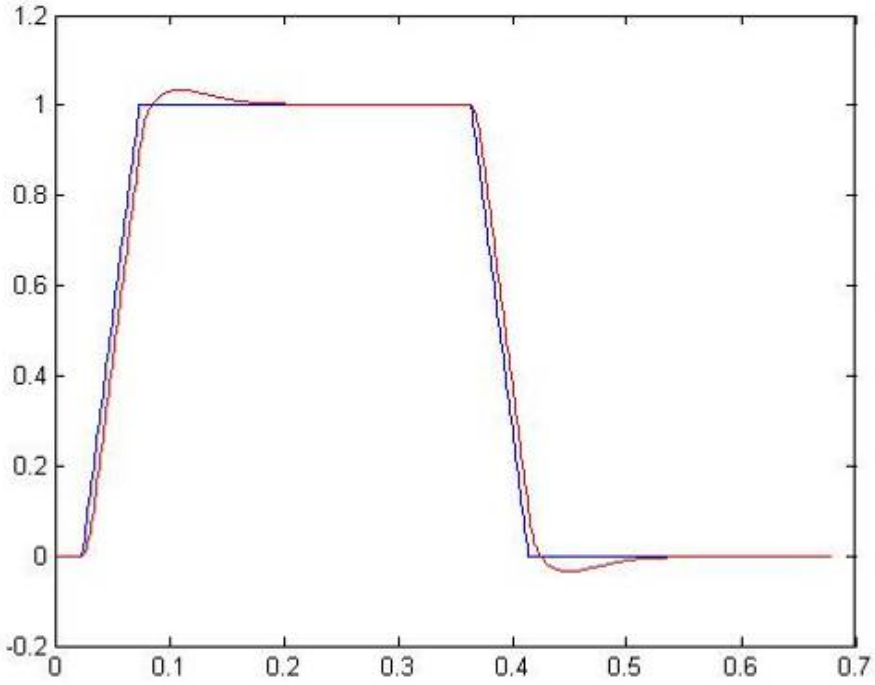


Figura 59 Respuesta final

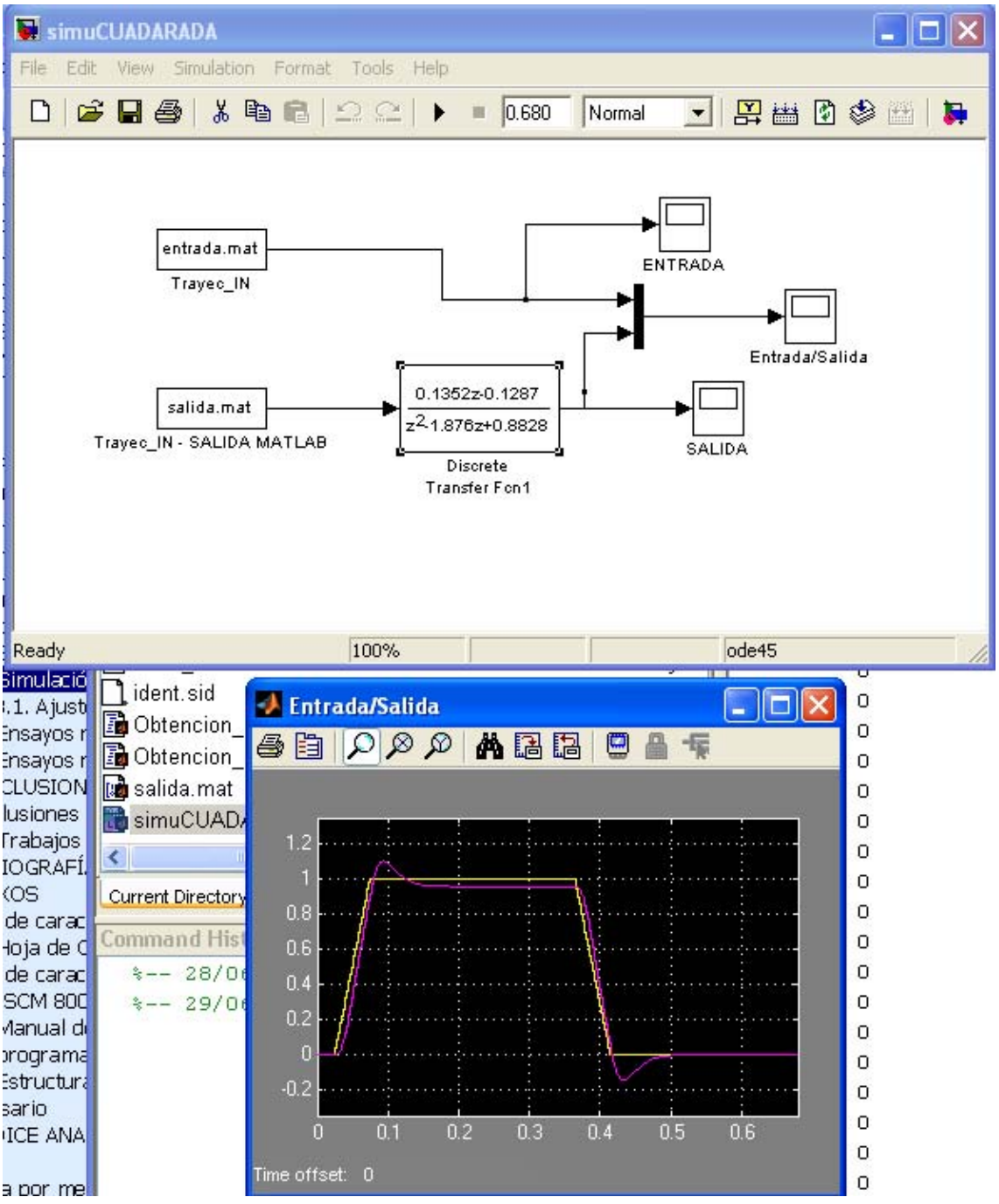


Figura 60 FT3 – Entrada escalón

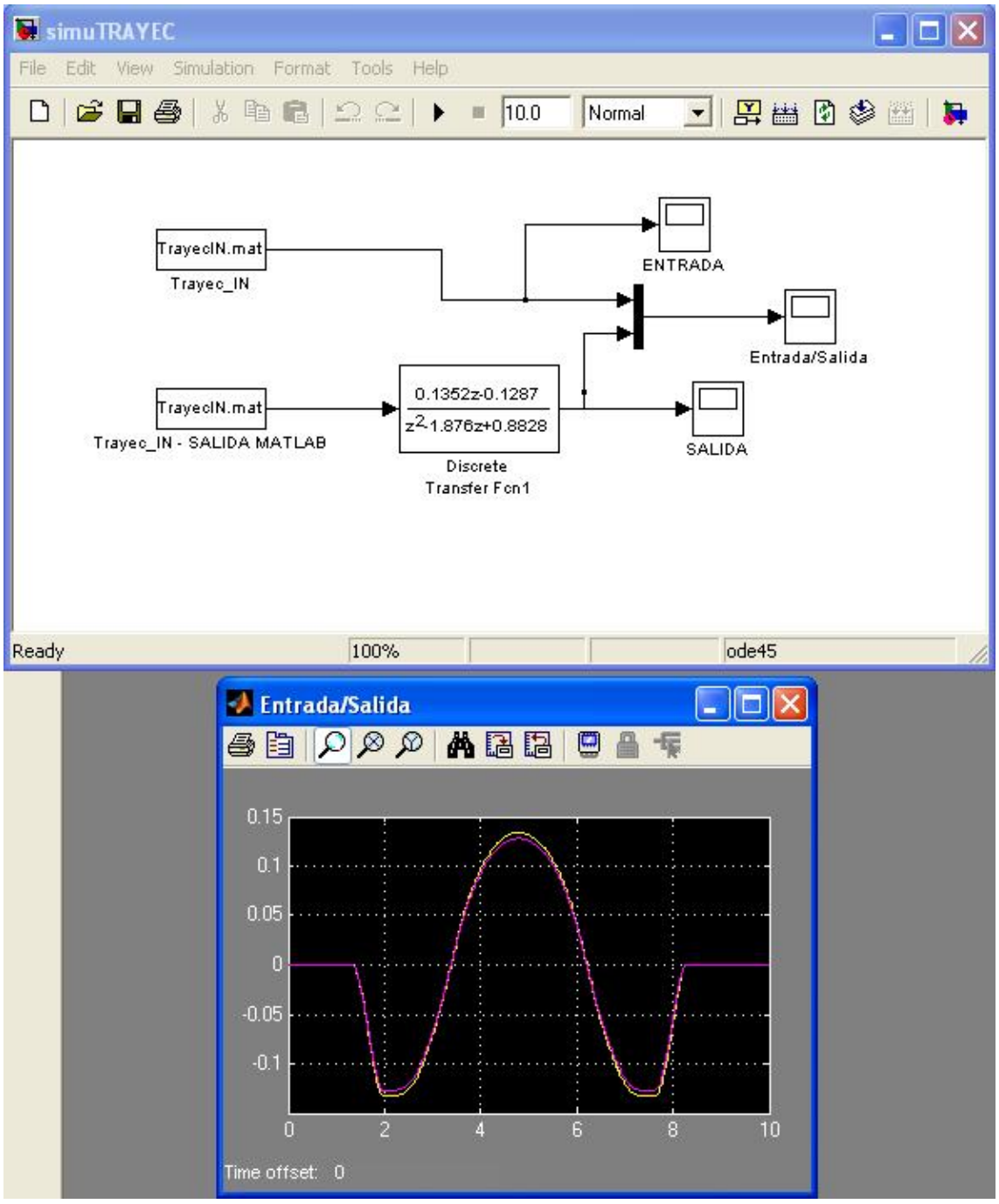


Figura 61 FT3 - Entrada Real

En la Figura 62 se muestran los nuevos valores del PID interno y en la Figura 63 la representación del Logger de este nuevo sistema.

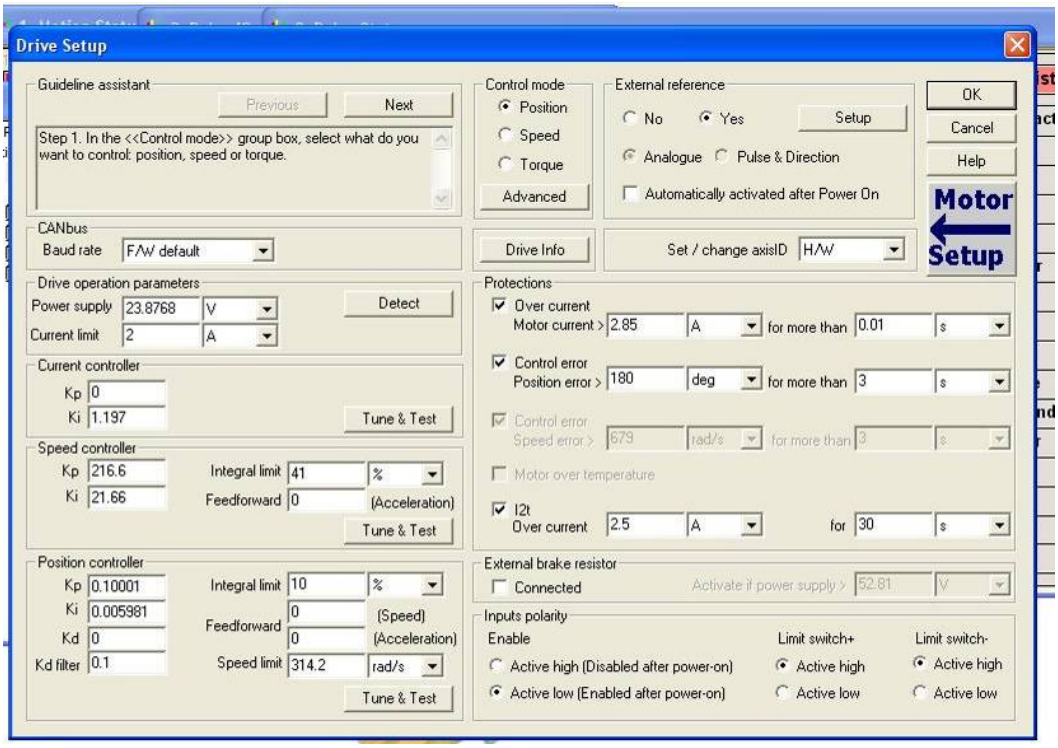


Figura 62 FT3 - Valores PID

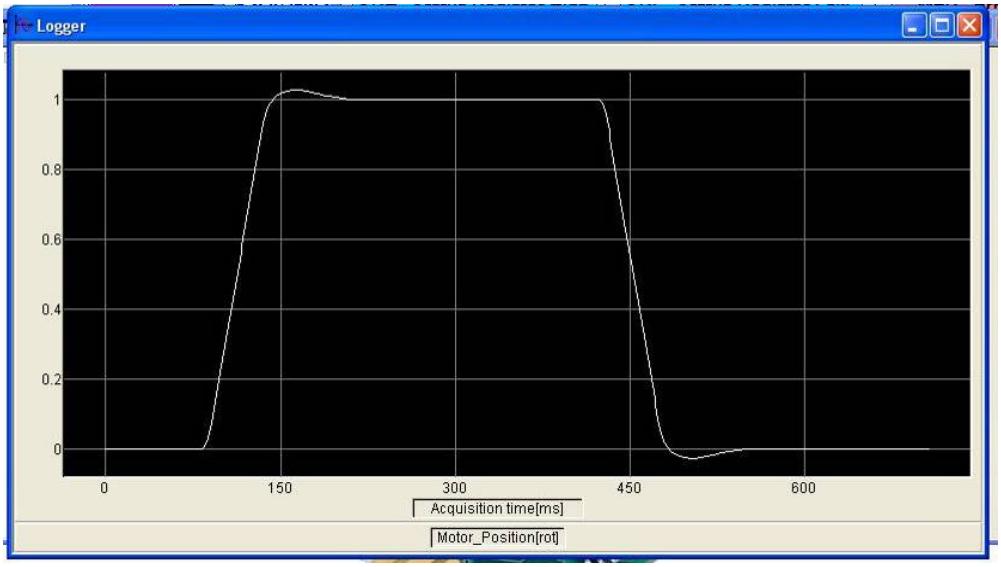


Figura 63 FT3 – Logger

Finalmente, en el flujograma de la Figura 64 se muestra a modo resumen los pasos a seguir para llegar a este punto del proyecto.

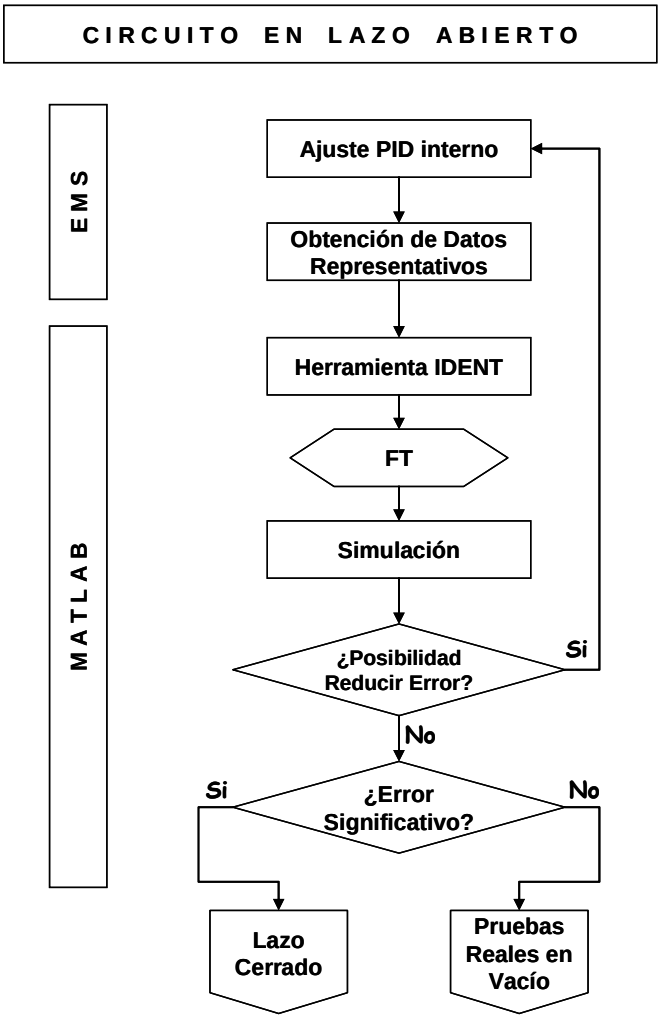


Figura 64 Diagrama de flujo de la caracterización y control simulado del sistema en lazo abierto

Capítulo 5

CONTROL EN LAZO CERRADO DEL TOBILLO

Para poder hacer un seguimiento de la posición del tobillo más preciso y garantizar que el sistema sea robusto ante perturbaciones e incertidumbres del modelo, es necesario cerrar un lazo de control. En nuestro caso el lazo se ha cerrado solamente en simulación, dejando para un proyecto futro la implementación real del lazo cerrado.

5.1. ESQUEMA DEL SISTEMA EN LAZO CERRADO

En la Figura 65 se muestra el esquema del sistema en lazo cerrado. Se ha incluido el PID diseñado y la función de transferencia que simula el comportamiento del motor, así como la entrada de datos y la respuesta del circuito.

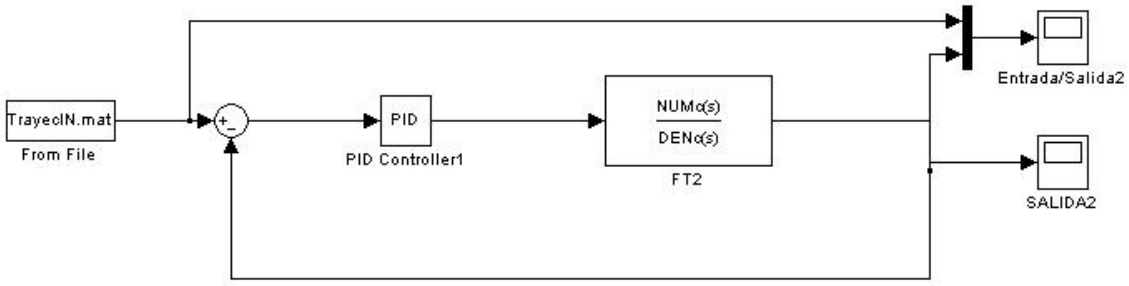


Figura 65 Esquema del circuito en lazo cerrado

5.2. CONTROLADOR PID

Un PID (Proporcional Integral Derivativo) es un mecanismo de control por realimentación que se utiliza en sistemas de control industriales. Un controlador PID corrige el error entre un valor medido y el valor que se quiere obtener calculándolo y luego sacando una acción correctora que puede ajustar al proceso acorde. Como se observa en la Figura 66, el algoritmo de cálculo del control PID lo componen tres parámetros distintos: el proporcional, el integral, y el derivativo. El valor Proporcional determina la reacción del error actual. El Integral genera una corrección proporcional a la integral del error, esto nos asegura que aplicando un esfuerzo de control suficiente, el error de seguimiento se reduce a cero. El Derivativo determina la reacción del tiempo en el que el error se produce. La suma de estas tres acciones es usada para ajustar al proceso vía un elemento de control. Ajustando estos tres valores en el algoritmo de control del PID, el controlador puede

proveer un control diseñado para lo que requiera el proceso a realizar. La respuesta del controlador puede ser descrita en términos de respuesta del control ante un error y el grado de oscilación del sistema. Nótese que el uso del PID para control no garantiza control óptimo del sistema o la estabilidad del mismo. Algunas aplicaciones pueden solo requerir de uno o dos modos de los que provee este sistema de control. Un controlador PID puede ser llamado también PI, PD, P o I en la ausencia de las acciones de control respectivas. Los controladores PI son particularmente comunes, ya que la acción derivativa es muy sensible al ruido, y la ausencia del proceso integral puede evitar que se alcance al valor deseado debido a la acción de control [6],[7],[8],[9] y [10].

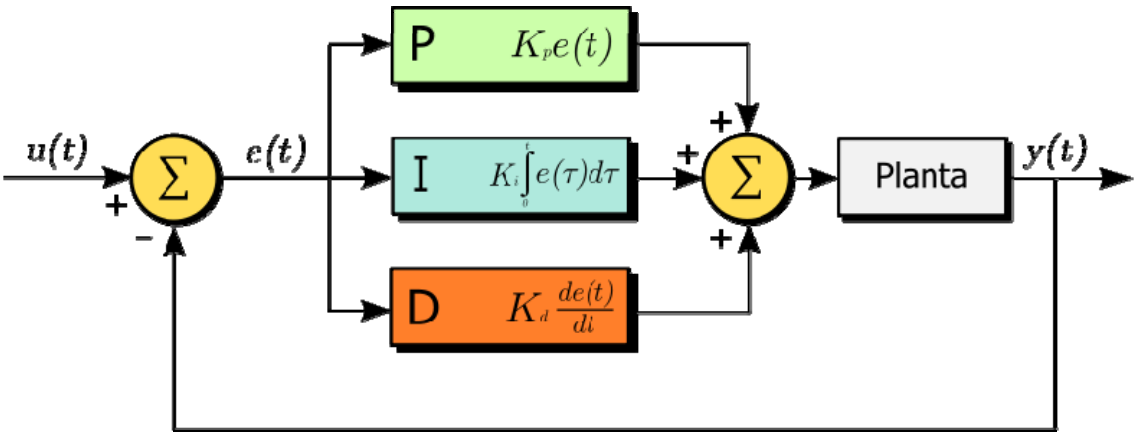


Figura 66 Diagrama en bloques de un control PID

5.2.1. Funcionamiento

Para el correcto funcionamiento del controlador PID que se usa en este proyecto se necesita:

- Los encoder propios del motor, que determinen la posición del mismo.
- Un controlador, que genere la señal que gobierna al motor.
- Un motor, que modifique al sistema de manera controlada.

Las tres componentes de un controlador PID son: parte Proporcional, acción Integral y acción Derivativa. El peso de la influencia que cada una de estas partes tiene en la suma final, viene dado por la constante proporcional, el tiempo integral y el tiempo derivativo,

respectivamente. Se pretenderá lograr que el bucle de control corrija eficazmente y en el mínimo tiempo posible los efectos de las perturbaciones.

5.2.1.1. PROPORCIONAL

La parte proporcional consiste en el producto entre la señal de error y la constante proporcional como para que hagan que el error en estado estacionario sea casi nulo, pero en la mayoría de los casos, estos valores solo serán óptimos en una determinada porción del rango total de control, siendo distintos los valores óptimos para cada porción del rango. Sin embargo, existe también un valor límite en la constante proporcional a partir del cual, en algunos casos, el sistema alcanza valores superiores a los deseados. Éste fenómeno se llama sobreoscilación y, por razones de seguridad, no debe sobrepasar el 30%, aunque es conveniente que la parte proporcional ni siquiera produzca sobreoscilación. Hay una relación lineal continua entre el valor de la variable controlada y la posición del elemento final de control. La parte proporcional no considera el tiempo, por lo tanto, la mejor manera de solucionar el error permanente y hacer que el sistema contenga algún componente que tenga en cuenta la variación respecto al tiempo, es incluyendo y configurando las acciones integral y derivativa (ver Figura 67).

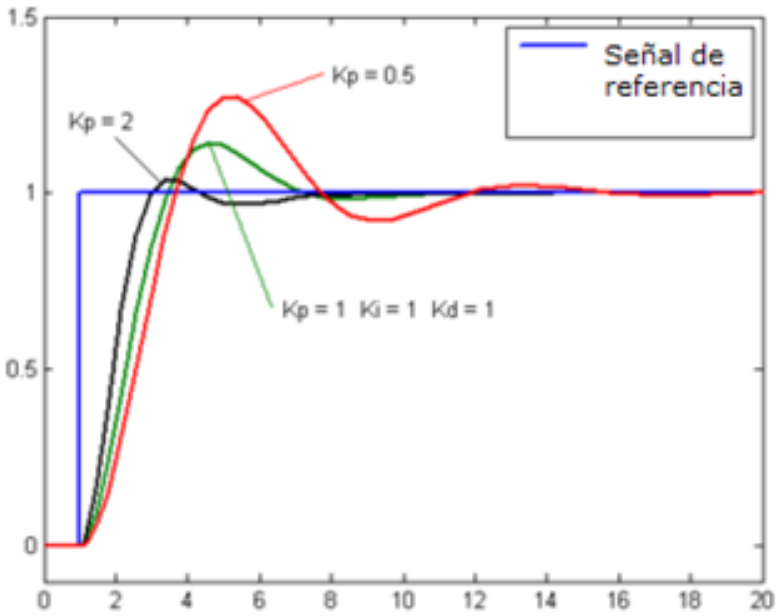


Figura 67 Acción Proporcional de un PID

5.2.1.2. INTEGRAL

El modo de control Integral tiene como propósito disminuir y eliminar el error en estado estacionario, provocado por el modo proporcional. En la Figura 68 se muestra como al modificar el valor de esta constante varía el error de la señal.

El control integral actúa cuando hay una desviación entre la variable y el punto de consigna, integrando esta desviación en el tiempo y sumándola a la acción proporcional. El error es integrado, lo cual tiene la función de promediarlo o sumarlo por un período determinado; Luego es multiplicado por una constante I. Posteriormente, la respuesta integral es adicionada al modo Proporcional para formar el control P + I con el propósito de obtener una respuesta estable del sistema sin error estacionario.

El modo integral presenta un desfasamiento en la respuesta de 90º que sumados a los 180º de la retroalimentación negativa acercan al proceso a tener un retraso de 270º, por lo que solo será necesario que el tiempo muerto contribuya con 90º de retardo para provocar la oscilación del proceso. Se caracteriza por el tiempo de acción integral en minutos por repetición.

El control integral se utiliza para obviar el inconveniente del offset (desviación permanente de la variable con respeto al punto de consigna) de la banda proporcional.

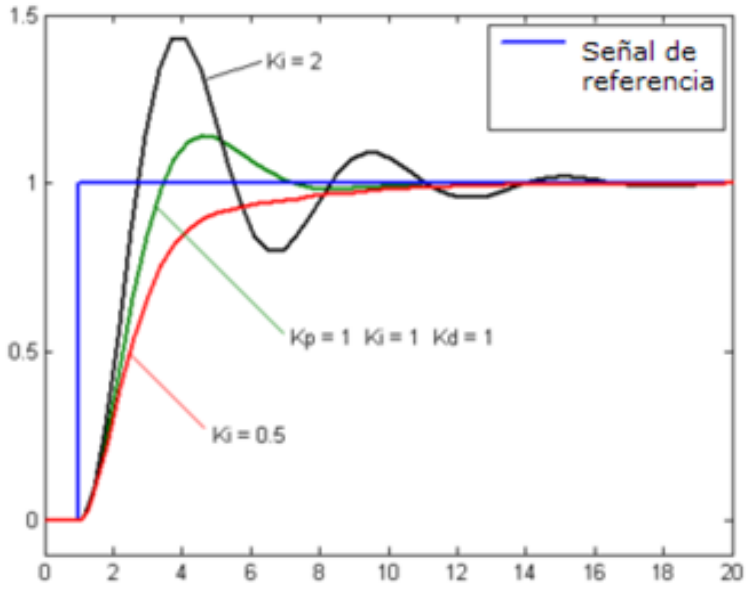


Figura 68 Acción Integral de un PID

5.2.1.3. DERIVATIVO

La acción derivativa se manifiesta cuando hay un cambio en el valor absoluto del error; (si el error es constante, solamente actúan los modos proporcional e integral).

El error es la desviación existente entre el punto de medida y el valor consigna.

La función de la acción derivativa es mantener el error al mínimo corrigiéndolo proporcionalmente con la misma velocidad que se produce, de esta manera evita que el error se incremente (ver Figura 69).

Se deriva con respecto al tiempo y se multiplica por una constante D y luego se suma a las señales anteriores (P+I). Es importante adaptar la respuesta de control a los cambios en el sistema ya que una mayor derivativa corresponde a un cambio más rápido y el controlador puede responder acordemente.

La acción derivada es adecuada cuando hay retraso entre el movimiento de control y su repercusión a la variable controlada.

Cuando el tiempo de acción derivada es grande, hay inestabilidad en el proceso. Cuando el tiempo de acción derivada es pequeño la variable oscila demasiado con relación al punto de consigna. Suele ser poco utilizada debido a la sensibilidad al ruido que manifiesta y a las complicaciones que ello conlleva.

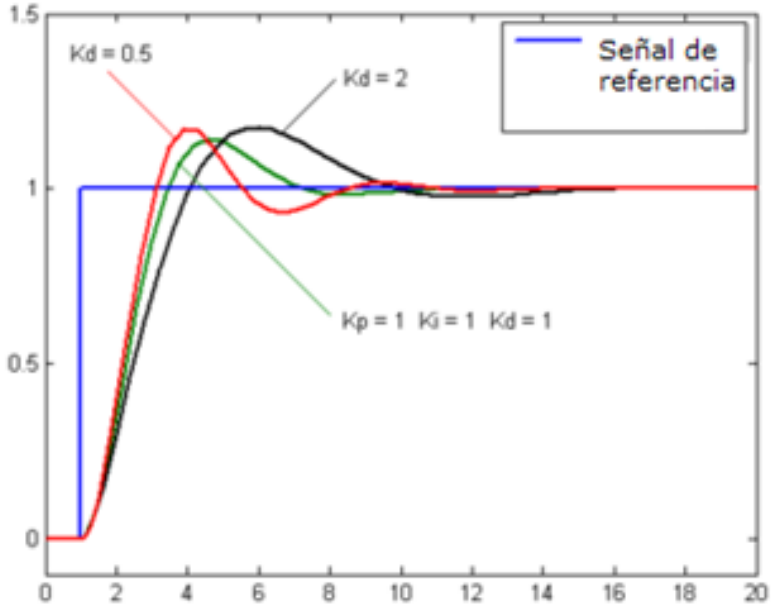


Figura 69 Acción Derivativa de un PID

La acción derivada puede ayudar a disminuir el rebasamiento de la variable durante el arranque del proceso. Puede emplearse en sistemas con tiempo de retardo considerables, porque permite una repercusión rápida de la variable después de presentarse una perturbación en el proceso.

5.2.2. Significado de las constantes

- P constante de proporcionalidad: se puede ajustar como el valor de la ganancia del controlador o el porcentaje de banda proporcional.
- I constante de integración: indica la velocidad con la que se repite la acción proporcional.
- D constante de derivación: hace presente la respuesta de la acción proporcional duplicándola, sin esperar a que el error se duplique. El valor indicado por la constante de derivación es el lapso de tiempo durante el cual se manifestará la acción proporcional correspondiente a 2 veces el error y después desaparecerá.

Tanto la acción Integral como la acción Derivativa, afectan a la ganancia dinámica del proceso. La acción integral sirve para reducir el error estacionario, que existiría siempre si la constante Ki fuera nula.

5.2.3. Ajuste de parámetros del PID

El objetivo de los ajustes de los parámetros PID es lograr que el bucle de control corrija eficazmente y en el mínimo tiempo los efectos de las perturbaciones; se tiene que lograr la mínima integral de error. Si los parámetros del controlador PID (la ganancia del proporcional, integral y derivativo) se eligen incorrectamente, el proceso a controlar puede ser inestable, por ejemplo, que la salida de éste varíe, con o sin oscilación, y está limitada solo por saturación o rotura mecánica. Ajustar un lazo de control significa ajustar los parámetros del sistema de control a los valores óptimos para la respuesta del sistema de control deseada. El comportamiento óptimo ante un cambio del proceso o cambio del "setpoint" varía dependiendo de la aplicación. Generalmente, se requiere estabilidad ante la respuesta dada por el controlador, y éste no debe oscilar ante ninguna combinación de las condiciones del proceso y cambio de "setpoints". Algunos procesos tienen un grado de no-linealidad y algunos parámetros que funcionan bien en condiciones de carga máxima

no funcionan cuando el proceso está en estado de "sin carga". Hay varios métodos para ajustar un lazo de PID. El método más efectivo generalmente requiere del desarrollo de alguna forma del modelo del proceso, luego elegir P, I y D basándose en los parámetros del modelo dinámico. Los métodos de ajuste manual pueden ser muy ineficientes. La elección de un método dependerá de si el lazo puede ser "desconectado" para ajustarlo, y del tiempo de respuesta del sistema. Si el sistema puede desconectarse, el mejor método de ajuste a menudo es el de ajustar la entrada, midiendo la salida en función del tiempo, y usando esta respuesta para determinar los parámetros de control. A continuación se describe como realizar un ajuste manual.

5.2.3.1. AJUSTE MANUAL

El primer paso es el ajuste de la banda proporcional. Si el controlador tiene ajustes para la parte integral y derivada, habrá que ponerlos en cero. El ajuste de la banda proporcional selecciona la velocidad de respuesta (a veces llamada ganancia) que necesita un control proporcional para conseguir la estabilidad del sistema.

La banda proporcional debe ser más ancha, en grados, que las oscilaciones normales del sistema, pero no demasiado ancha como para amortiguar la respuesta del sistema. Comience con la banda proporcional lo más angosta posible. Si existen oscilaciones se debe aumentar la banda proporcional en pequeños incrementos, esperando cada vez varios minutos para que el sistema se estabilice, hasta el punto en el cual la caída comienza a aumentar. En éste punto las variables del proceso deberán estar en un estado de equilibrio en algún punto por debajo del setpoint.

El paso siguiente es el ajuste de la acción integral o de reset. Si el control tiene un ajuste manual, se lo ajusta hasta que la caída del proceso se ha eliminado. El problema con el ajuste manual es que cada vez que se cambia el setpoint de valor, posiblemente tengamos una caída otra vez y haya que ajustarlo nuevamente.

Si el control tiene reset automático, se ajusta el mismo a su inicio de modo que halla el mínimo número de repeticiones por minuto para permitir el equilibrio del sistema. En otras palabras se ajusta el auto reset en pequeños pasos, permitiendo que el sistema se equilibre después de cada paso, hasta que empiecen pequeñas oscilaciones. Luego se retrocede con el ajuste hasta que las oscilaciones se detengan y se restablezca el equilibrio. En éste punto el sistema se ajustará automáticamente para los errores de caída.

El último parámetro de control para ajustar es la función derivada. Siempre se debe ajustar esta función a lo último. Si este ajuste se hace antes del reset, éste se irá de límites, y habrá que comenzar todo el proceso nuevamente.

La función del ajuste derivado es reducir en todo lo posible cualquier sobrepaso de temperatura. El ajuste derivado es uno basado en el tiempo medido en minutos sintonizado para trabajar con el tiempo de respuesta del conjunto del sistema.

El ajuste inicial deberá ser la mínima cantidad de minutos posible. Se aumenta el ajuste en muy pequeños incrementos. Después de cada ajuste se debe esperar hasta que se equilibre. Luego se incrementa el setpoint en una magnitud moderada. Vigile la acción del control cuando se llega al setpoint. Si existe un sobrepaso, se aumenta la acción derivada en una pequeña cantidad y se repite el procedimiento hasta que el sobrepaso se elimina. Algunas veces el sistema se hace lento y nunca llega al setpoint. Si esto ocurre, disminuya el ajuste derivado hasta que el proceso llega al setpoint.

5.3. DISEÑO DE UN REGULADOR PID CON MATLAB

Esta nueva fase consiste principalmente en diseñar un PID externo al sistema (Figura 70). En el apartado 5.2 se detallan en profundidad los métodos utilizados para el diseño. Al querer comprobar la robustez mecánica del tobillo y la validez o no de los motores, no se realiza un cálculo exacto del PID, siendo éste el motivo por el que se usa un método de *"prueba-error".*

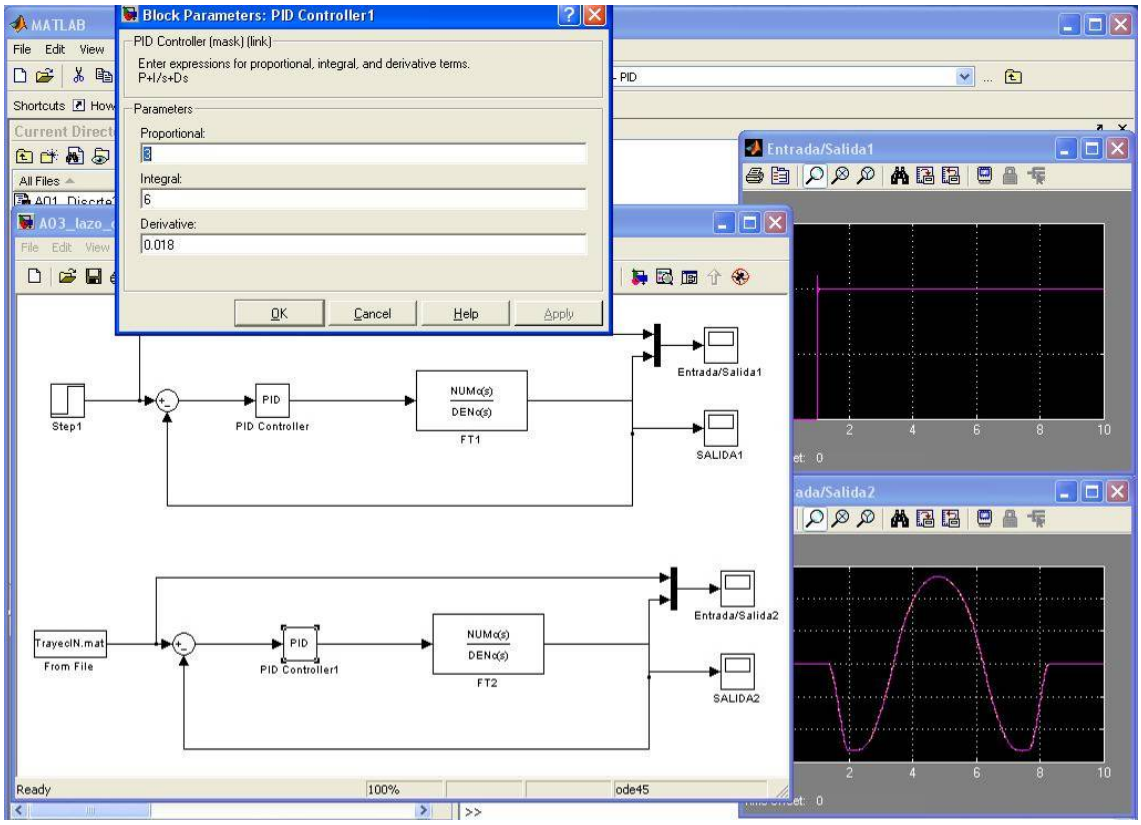


Figura 70 Diseño de PID con Matlab

5.3.1. Ajuste PID externo

La Figura 71 muestra la ventana de Simulink que sirve para ajustar manualmente el PID externo que se desea diseñar. Siguiendo las técnicas estudiadas en el apartado 5.2 se llega a un PID como el que se muestra. Se han representado también las salidas generadas por los dos sistemas ante una entrada escalón y una trayectoria real.

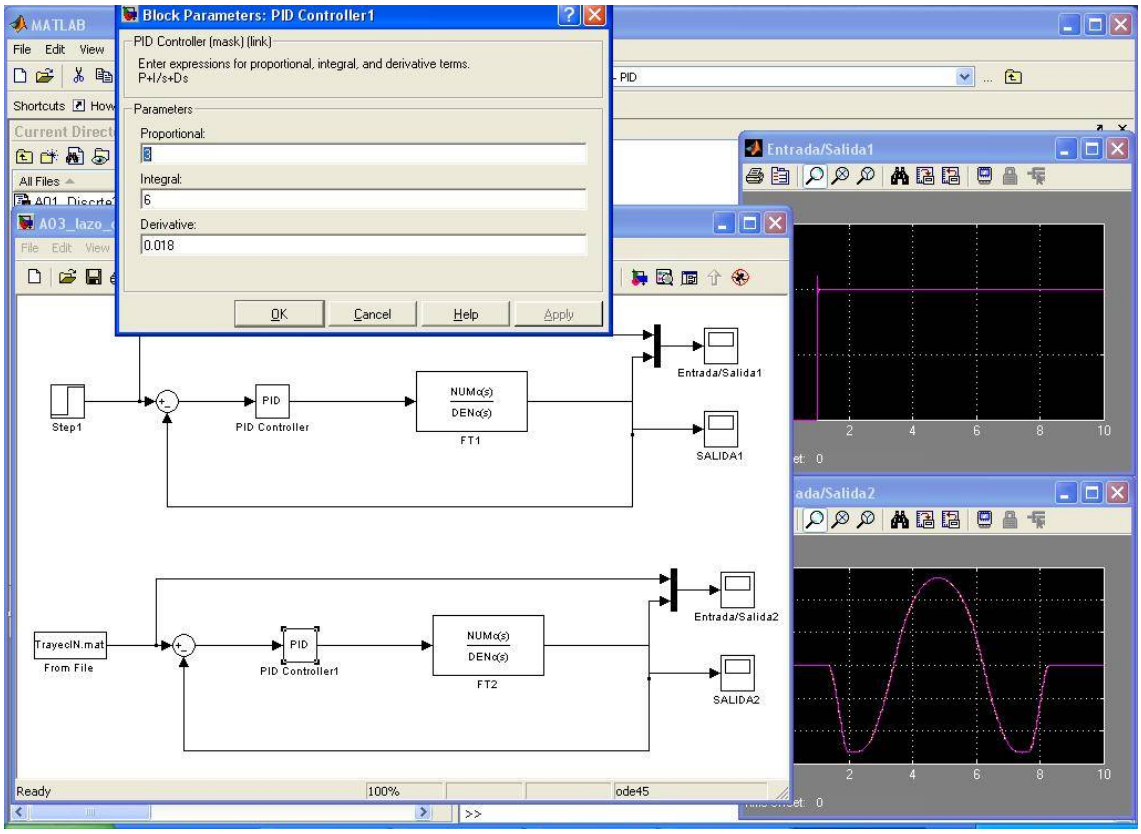


Figura 71 Diseño PID externo

Se puede observar que la salida en lazo cerrado del tobillo tiene un error menor que en el caso de lazo abierto, pudiéndose implementar éste en caso de necesitar una mayor precisión.

Al igual que en el apartado anterior, se incluye un flujograma con los pasos a seguir para realizar correctamente el control en lazo cerrado (ver Figura 72)

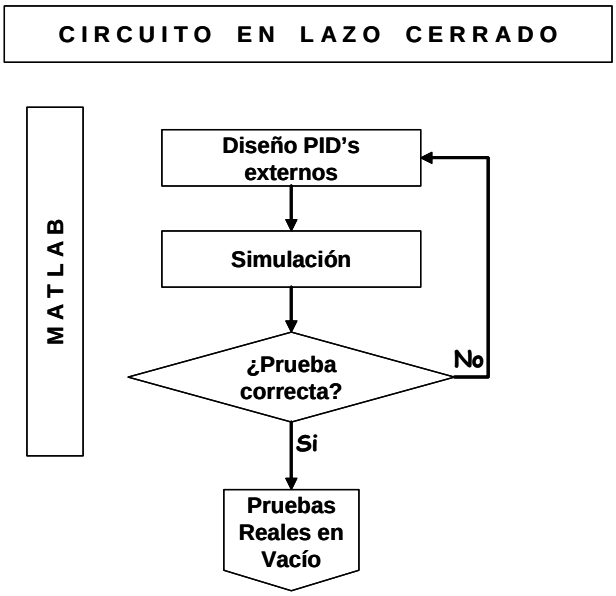


Figura 72 Diagrama de flujo del modelado y control simulado en lazo cerrado

Capítulo 6

RESULTADO EXPERIMENTALES

6.1. ENSAYOS REALES EN VACÍO

6.1.1. Introducción

Una vez obtenido un PID adecuado para nuestro sistema, ya sea interno o externo, comienzan unos procesos, si caben, más importantes que los anteriores. Éstos consisten en realizar pruebas reales tanto en vacío (únicamente con el peso propio de la estructura del tobillo) como a plena carga, comprobando que la estructura no queda comprometida y será fiable durante su trabajo.

Es conveniente seguir el flujograma de la Figura 73 para asegurar que el sistema se comporta de manera adecuada. Como se puede comprobar, el primer paso sería realizar las conexiones necesarias para poner en funcionamiento los motores, inicialmente desconectados de la estructura para no exponerla a esfuerzos innecesarios provocados por una mala conexión o una mala alimentación, así como para comprobar el correcto funcionamiento de éstos. Una vez se esté seguro de que los motores funcionan perfectamente y que las conexiones son acertadas, se integrarán a la estructura del robot, realizando pruebas de trayectorias y velocidades individualmente para cada motor, pasando a poner en funcionamiento los dos motores a la vez únicamente cuando se esté seguro de conocer las limitaciones de trayectorias y velocidades de los mismos.

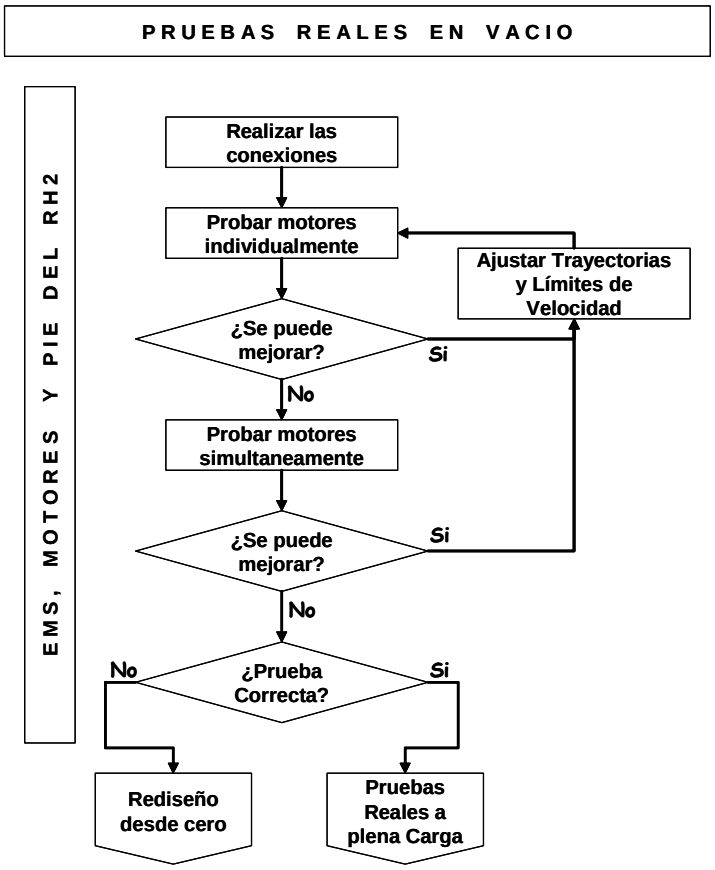


Figura 73 Diagrama de flujo del control en vacío

Al finalizar esta fase nos podemos encontrar antes dos posibilidades:

- La primera será que el diseño teórico inicial no sea factible, por las limitaciones que presentan los motores o la propia estructura, por lo que habría que hacer una reingeniería desde el principio del proyecto. Este inconveniente puede ser debido a un mal estudio previo, por lo que es muy aconsejable dedicar el máximo tiempo posible a este estudio e intentar conocer a la perfección el sistema completo que se va a tratar.
- Cabe esperar que esta fase se ajuste a las necesidades específicas del robot por lo que el paso siguiente es realizar las pruebas pertinentes a plena carga.

Es conveniente realizar estos experimentos basándose en trayectorias límite. Para este modelo es conveniente conocer los máximos movimientos de cada uno de los dos grados de libertad del tobillo permitidos por la estructura. Como ya se ha comentado, estos límites son diferentes para cada eje, por lo que es muy importante elegir adecuadamente la

trayectoria a seguir. Siendo la más comprometida aquella cuyo movimiento del vértice superior de la estructura forma una elipse si se observa el moviendo desde arriba, plano de planta.

Al ser éste el primer experimento real que se realiza, se hará en vacío, es decir sin colocar más carga que la propia del tobillo, alejándose considerablemente del peso real de la estructura y sin tener en cuenta los centros de gravedad que serán de suma importancia en pruebas posteriores.

Tras estos ensayos, pueden observarse dos tipos de errores totalmente diferenciados pero, a la vez, íntimamente relacionados, cuyo impacto y tiempo necesario para llegar a una solución puede ser muy dispar:

- Debido a una mala elección en el sistema de control diseñado, cabe la posibilidad de que se observe un error posicional tras un largo tiempo de funcionamiento y por tanto que se generen movimientos incontrolados.
- La estructura podría no ser lo suficientemente rígida, provocándose una fractura que lo inutilizaría. Algo que podría ocurrir también si el diseño del sistema de control no es adecuado al provocar en éste vibraciones o movimientos bruscos del conjunto.

Es muy importante conocer el tipo de error y su origen, siendo aconsejable vigilar la estructura durante un periodo prudencial de tiempo. Así se podrían impedir fracturas evitables al observar movimientos no programados y por tanto incontrolados.

Si se detecta a tiempo un error posicional debido a un mal diseño de los controladores, el ensayo ha de ser interrumpido inmediatamente y así salvaguardar la integridad de la estructura. En este caso es necesario rediseñar nuevamente los PID o valorar la posibilidad de integrar en el sistema un PID externo al driver.

En definitiva, hay que tener especial cuidado en no provocar una rotura por un error de posicionamiento, ya que es posible que, sin saberlo, también estemos ante una estructura frágil para las especificaciones de funcionalidad.

Si no se detectan errores de movimientos y aún así la estructura rompe, habría que volver a fabricar el tobillo utilizando, entre otras cosas, nuevos materiales o redistribuyendo los centros de gravedad de manera distinta.

6.1.2. Procedimiento utilizado en los ensayos

En el DVD se adjuntan todas las trayectorias y pruebas realizadas durante la fase de diseño y pruebas.

Como ya se ha comentado, al tener que controlar 2 motores debido a los 2 grados de libertad del pie, es necesario crear 2 proyectos en EMS, uno para cada giro:

6.1.2.1. PROYECTO 1. GIRO EN EL EJE Y

Éste primer proyecto controlará el giro en el eje Y, por diseño tiene una libertad de movimiento de unos 140º (Figura 74).

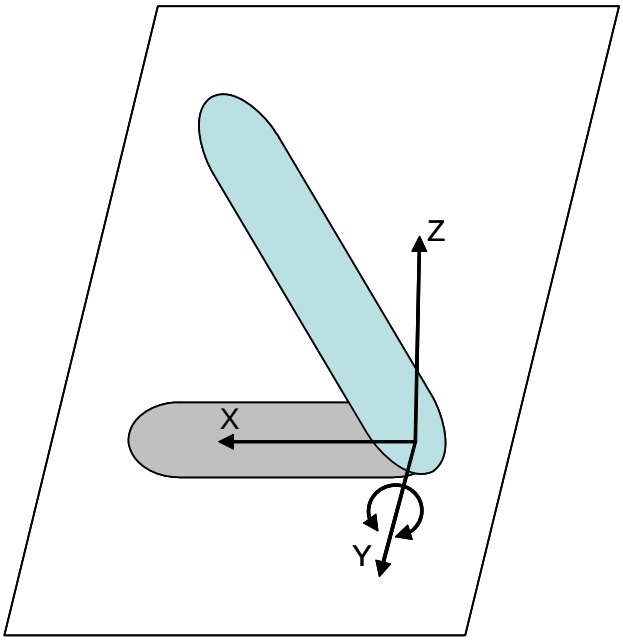


Figura 74 Esquema Giro en Y

La trayectoria utilizada para generar el movimiento límite del motor 1 se muestra en la Figura 75.

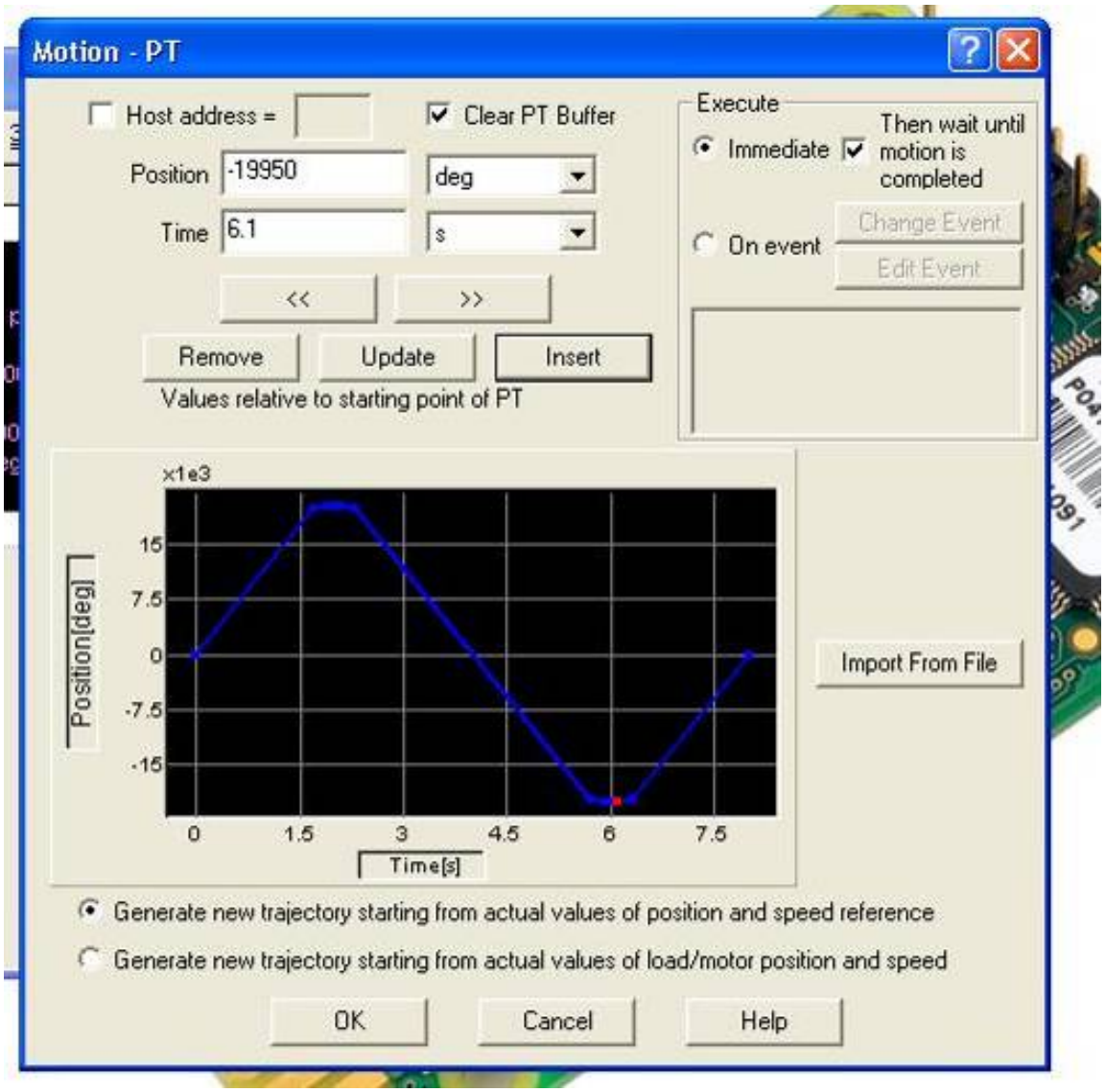


Figura 75 Trayectoria motor 1

6.1.2.2. PROYECTO 2. GIRO EN EL EJE X

Con este segundo proyecto se seguirán los mismo pasos que en el parto anterior, con la diferencia de que controlará el giro en el eje X, éste tiene una libertad de movimiento de 40º (ver Figura 76).

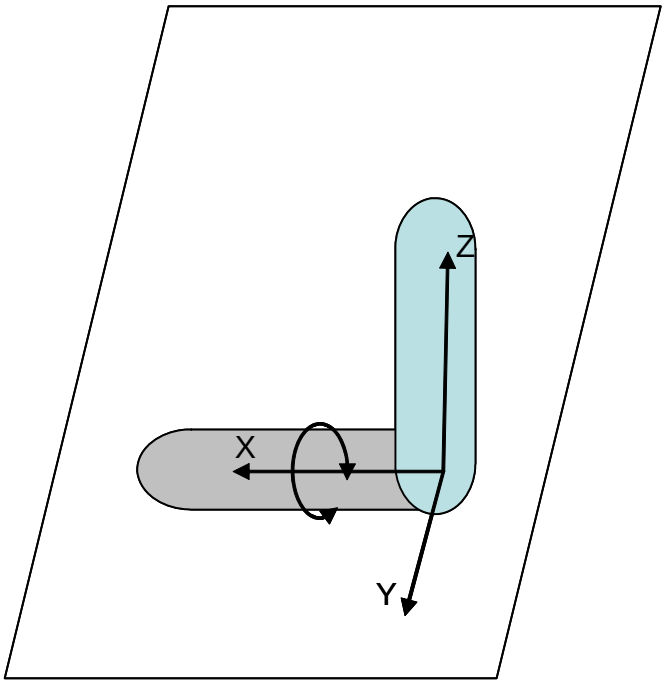


Figura 76 Esquema Giro en X

La trayectoria utilizada para generar el movimiento límite del motor 2 se muestra en la Figura 77.

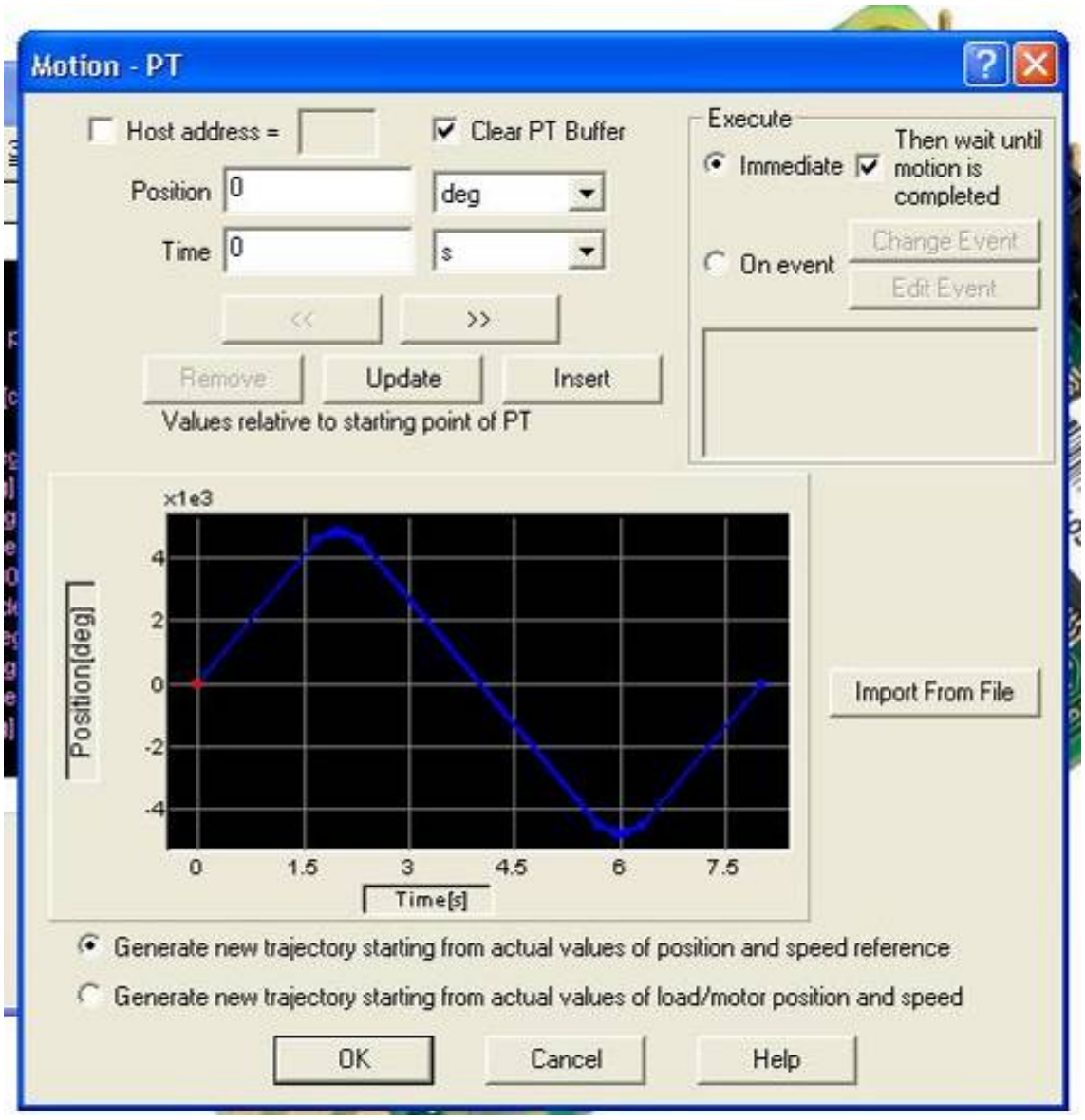


Figura 77 Trayectoria motor 2

6.2. ENSAYOS REALES A PLENA CARGA

Una vez llegado a este último punto, podemos estar seguros de que el sistema electrónico y eléctrico ha sido diseñado correctamente y que al menos, la estructura diseñada soporta los movimientos sin carga.

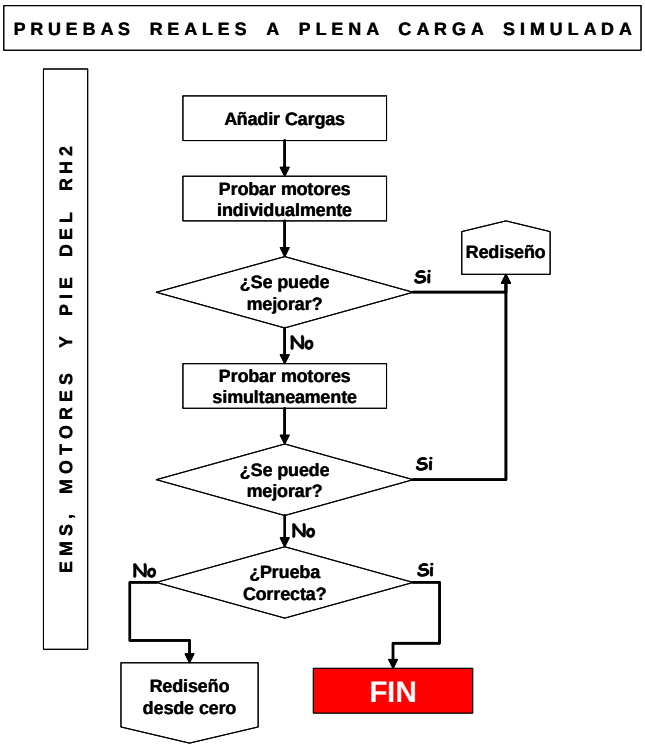


Figura 78 Diagrama de flujo de las pruebas reales a plena carga simulada

Es muy importante realizar esta fase (ver Figura 78) con unas cargas lo más parecidas posibles a la implementación final del robot humanoide. Para ello se ha fabricado una estructura consistente en una barra metálica de una altura similar a la final, en ésta se acoplarán una pesas cuyo peso total simulará al de RH2.

Se procederá de manera similar al apartado 6.1, es decir, inicialmente probando separadamente los motores y haciéndolos funcionar simultáneamente una vez realizadas estas pruebas iniciales.

A priori, las conclusiones de esta fase no se pueden predecir hasta haber realizado el experimento. Nos podemos encontrar ante la posibilidad de que los motores no soporten tanta carga o simplemente que la estructura mecánica rompa, por lo que habría que realizar nuevamente una reingeniería. De echo, tras realizar el ensayo y llevar varias horas el sistema en funcionamiento, se observó que los motores soportaron perfectamente la carga, sin embargo la estructura se rompió, por el harmonic drive,.

En la Figura 79 se muestra una comparación entre las respuestas generadas en Matlab en lazo abierto y en lazo cerrado, observándose que la respuesta en lazo cerrado se ajusta mucho más a lo deseado. Aunque como se ha comentado a la largo de la memoria,

el objetivo de este proyecto no es obtener el PID definitivo que se implementará en el RH2, si no comprobar que los elementos que se usarán en dicho proyecto son adecuados.

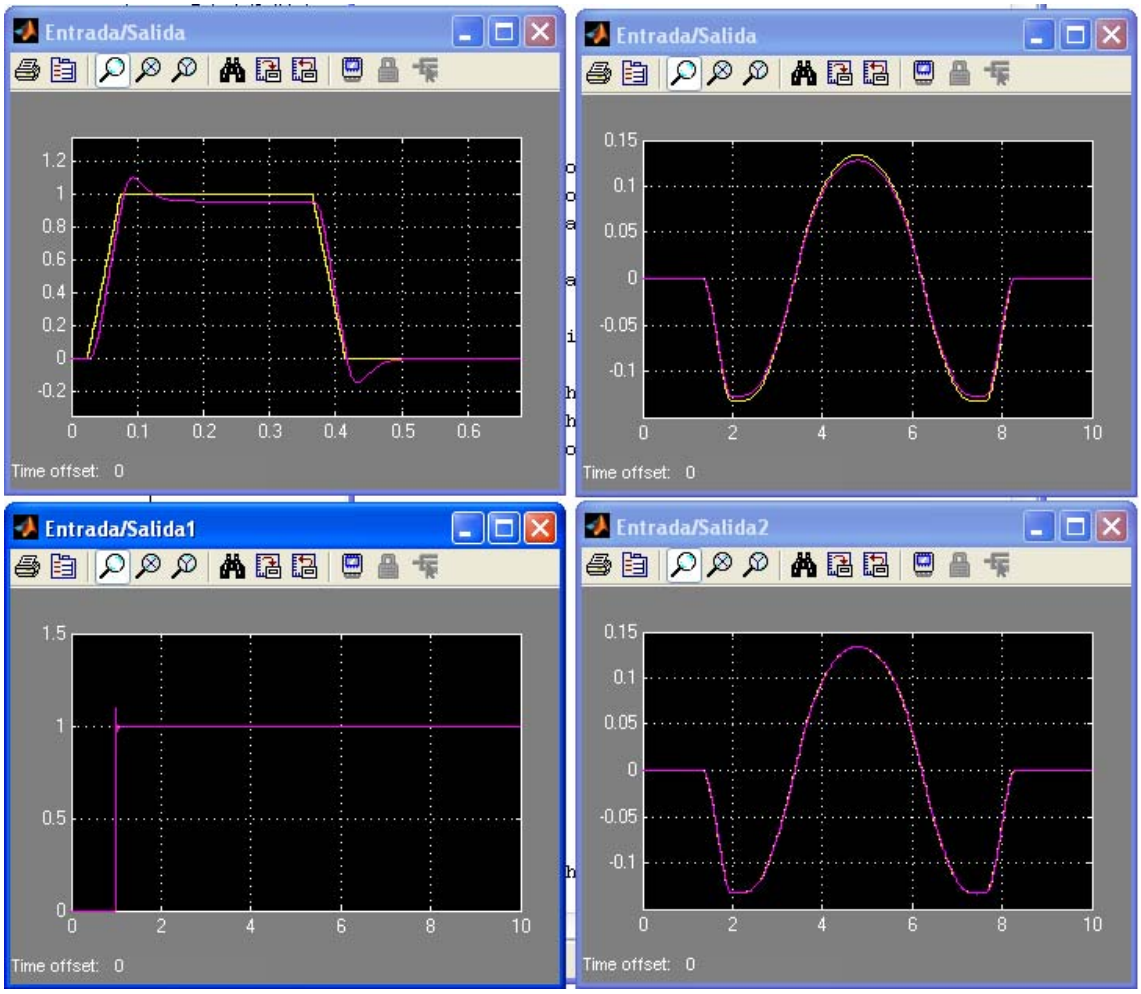


Figura 79 Comparación entre lazo abierto y lazo cerrado del tobillo

6.3. GRÁFICAS DE LOS DOS GRADO DE LIBERTAD.

Como ya se ha comentado, para poner en funcionamiento los dos motores a la vez, es necesario abrir dos veces el software Easy Motion Studio e indicar en cada pantalla uno de los puertos a usar, los cuales serán el COM 1 y el COM 4. Esta configuración de realiza a través de la siguiente ruta: "*Comunication-> Setup"*. El último paso será cargar las dos pantallas con la misma trayectoria y ejecutar primero uno y luego el otro.

En las Figura 80, Figura 81, Figura 82 y Figura 83 se observa la posición y velocidad en el plano frontal y sagital, respectivamente, del tobillo cuando usamos los dos motores a la vez durante la trayectoria programada.

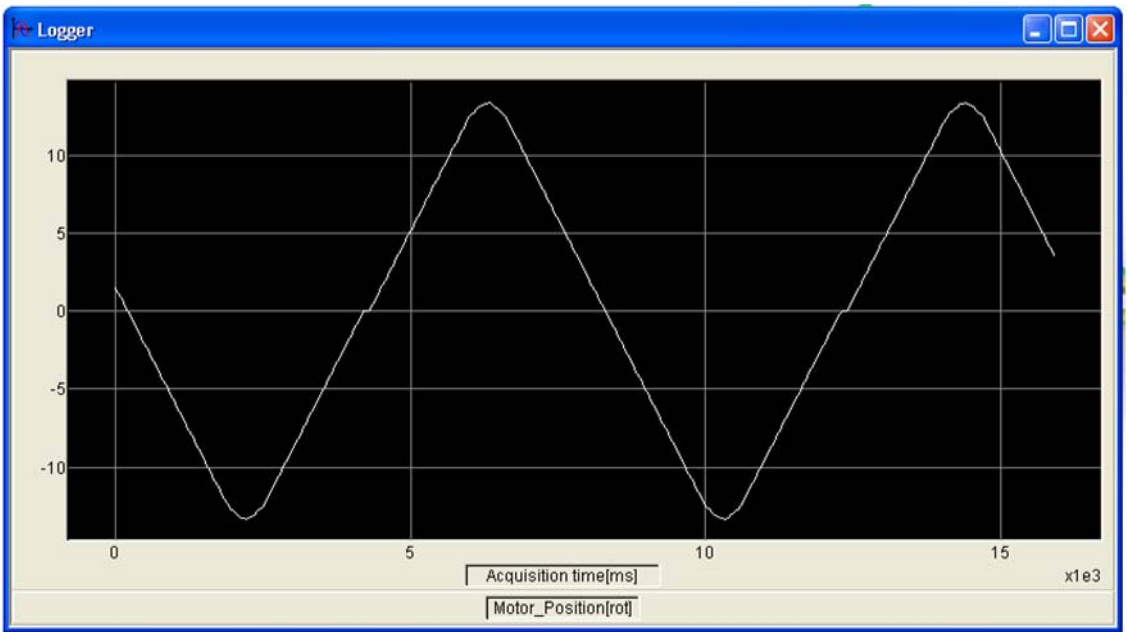


Figura 80 Posición de los motores en el plano frontal

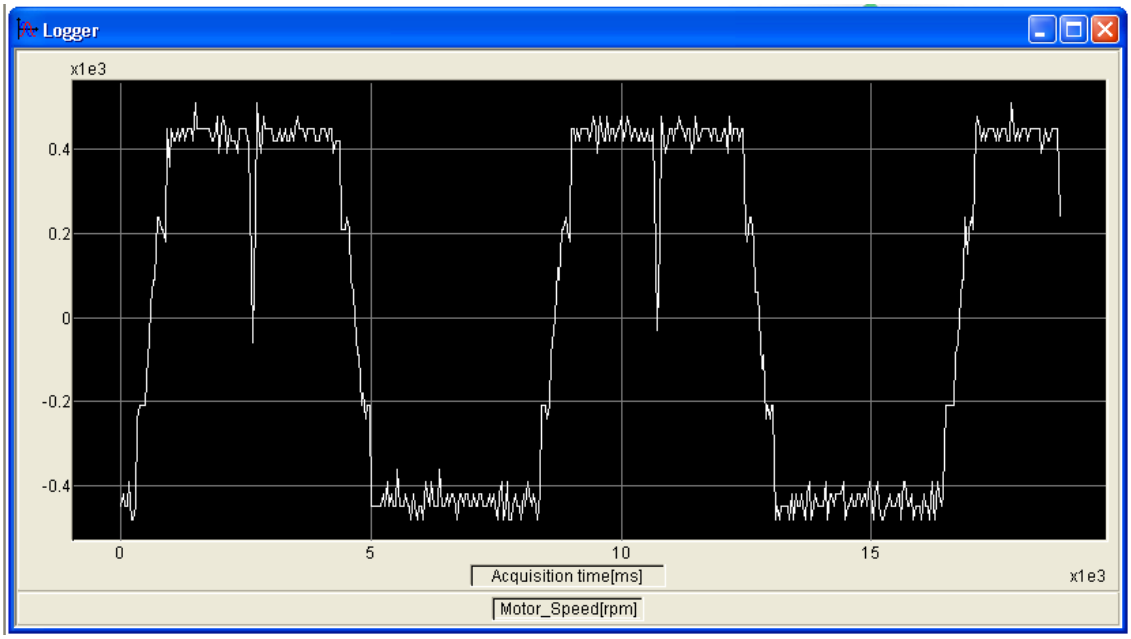


Figura 81 Velocidad de los motores en el plano frontal

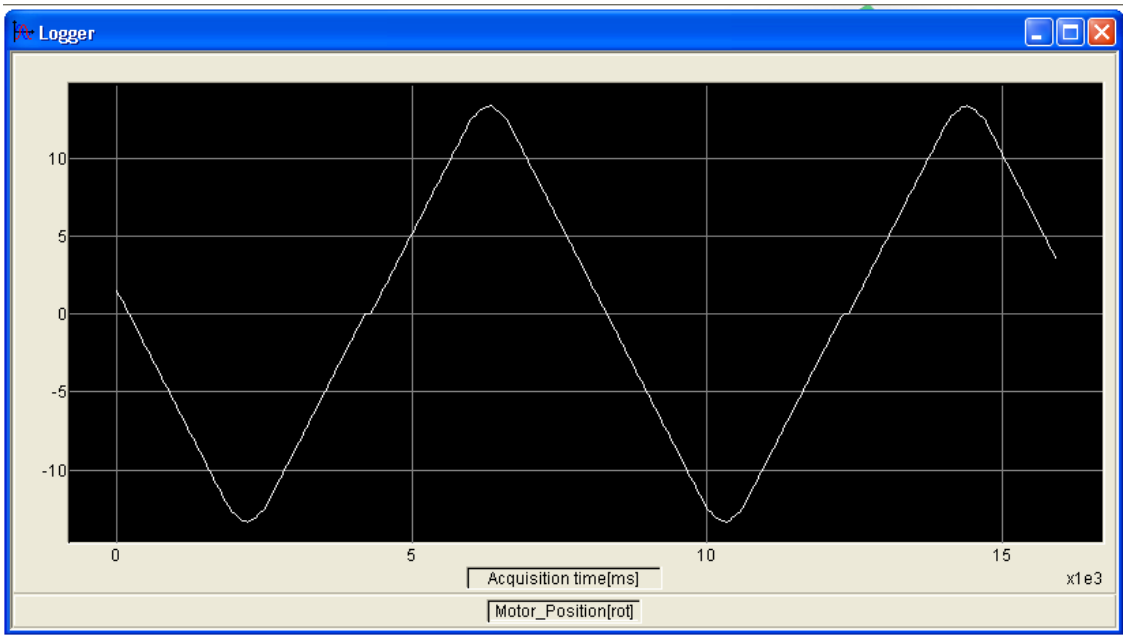


Figura 82 Posición de los motores en el plano sagital

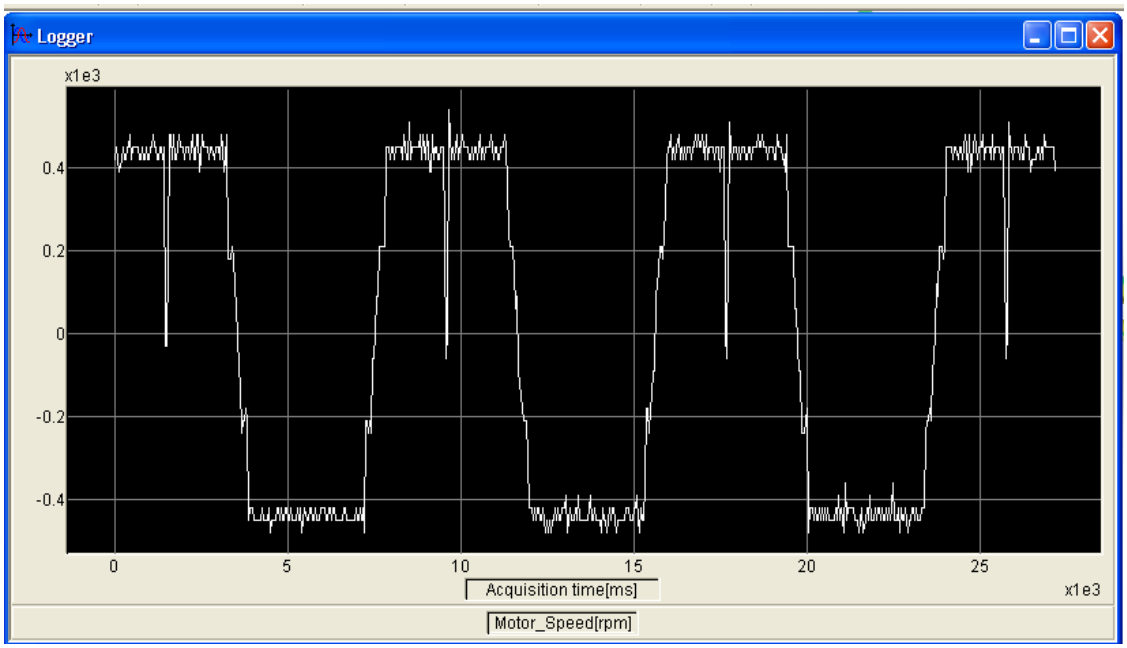


Figura 83 Velocidad de los motores en el plano sagital

Capítulo 7

CONCLUSIONES Y TRABAJOS FUTUROS

7.1. CONCLUSIONES

En este proyecto se ha estudiado la caracterización y control del tobillo del novedoso RH-2. Para ello se han utilizado las herramientas Easy Motion Studio y MATLB que han permitido obtener una representación matemática del sistema, consiguiendo así diseñar un PID que permite controlar la posición de los motores que gobiernan el movimiento del tobillo.

Se han implementado dos PID, uno interno y otro externo al motor, utilizándose experimentalmente el primero para ajustar los lazos de control que permiten el seguimiento de las trayectorias de la caminata del robot.

Para comprobar la robustez mecánica y el correcto diseño del PID se mantuvo funcionando el sistema en vacío y a plena carga varias horas, utilizándose una trayectoria límite para ambos motores.

Tras las pruebas en vacío no se observó ningún error de posición ni ningún daño en la estructura mecánica. Con las pruebas a plena carga no hubo tanta suerte y finalmente la estructura mecánica se rompió por el Harmonic Drive.

7.2. TRABAJOS FUTUROS

La estructura no soportó el ensayo a plena carga, por tanto se propone como posibles trabajos futuros los siguientes:

- Rediseñar la estructura de tal modo que soporte mayor peso y realizar estas mismas pruebas.
- Implementar un control en lazo cerrado.
- Utilizar motores más pequeños pero con la misma funcionalidad.
- Probar distintas estrategias de control para la estabilidad de la caminata del robot humanoide.

-

Bibliografía y Referencias

[1] Historia del Robot Humanoide, disponible en la dirección: http://www.uc3m.es/portal/page/portal/actualidad_cientifica/actualidad/reportajes/archivo_reportajes/robot_humanoide_uc3m

[2] Monje, C. *Control de.la camimnata del robot humanoide RH-1.*

[3] Starter Kit for ISCMxx05 Intelligent Servo Drive – Brushless Motor "USER MANUAL".

[4] Manual del Toolbox IDENT de Matlab, disponible en la dirección: http://www.mathworks.com/products/sysid/

[5] Manual del Toolbox Simulink de Matlab, disponible en la dirección: http://www.mathworks.es/access/helpdesk/help/pdf_doc/simulink/sl_gs.pdf

[6] Sistemas de control, disponible en la dirección: http://syscontrol2.blogspot.com/2007/10/informacin-terica-previa.html

[7] Teoría de PID, disponible en la dirección: http://www.automatas.org/hardware/teoria_pid.htm

[8] PID, disponible en la dirección: http://es.wikipedia.org/wiki/Proporcional_integral_derivativo

[9] Apuntes de Control, disponible en la dirección: http://www.alumnos.usm.cl/~ignacio.morande/descargas/apuntes_de_control_pid.pdf

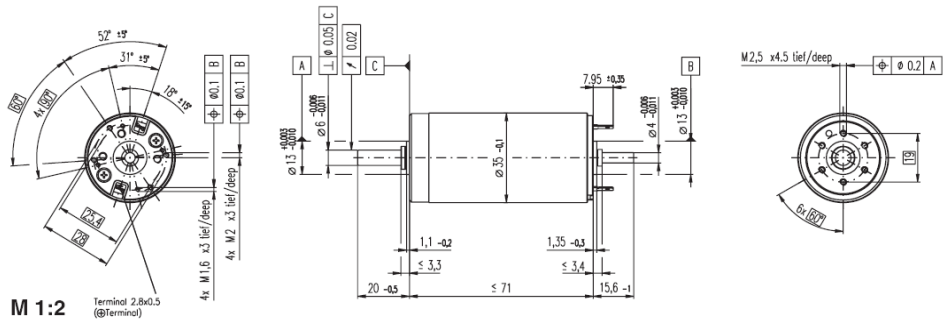
[10] Luís Moreno, Santiago Garrido y Carlos Balaguer. *Ingeniería de control. Modelado, análisis y control de sistemas.* Ariel (ISBN: 8434480557 ISBN-13: 9788434480551). Octubre 2010.

Anexos

ANEXO I HOJA DE CARACTERÍSTICAS DEL MOTOR MAXON

maxon DC motor

RE 35 ∅35 mm, Escobillas de grafito, 90 Vatios



Programa Stock
Programa Estándar
Programa Especial (previo encargo)

Números de Referencia

Medidas conforme con el dibujo	273752	323890	273753	273754	273755	273756	273757	273758	273759	273760	273761	273762	273763	
Versión con eje corto (4 en lugar 15,6 mm)	285785	323891	285786	285787	285788	285789	285790	285791	285792	285793	285794	285795	285796	
Datos del motor														
Valores a tensión nominal														
1 Tensión nominal	V	15.0	24.0	30.0	42.0	48.0	48.0	48.0	48.0	48.0	48.0	48.0	48.0	48.0
2 Velocidad en vacío	rpm	7070	7670	7220	7530	7270	6650	5960	4740	3810	3140	2570	2100	1620
3 Corriente en vacío	mA	245	168	123	92.7	77.3	68.7	59.7	44.7	34.2	27.1	21.6	17.2	12.9
4 Velocidad nominal	rpm	6270	6910	6420	6770	6490	5860	5150	3920	2970	2280	1710	1220	732
5 Par nominal (máx. par permanente)	mNm	73.2	93.3	92.4	97.7	96.5	98.2	98.8	102	105	105	105	104	104
6 Corriente nominal (máx. corriente en continuo)	A	4.00	3.36	2.50	1.95	1.63	1.51	1.36	1.12	0.915	0.752	0.621	0.503	0.391
7 Par de arranque	mNm	874	1160	949	1070	967	878	766	613	493	394	320	253	194
8 Corriente de arranque	A	45.0	39.7	24.4	20.3	15.5	12.9	10.1	6.43	4.16	2.74	1.83	1.18	0.704
9 Máx. rendimiento	%	81	84	84	86	85	85	84	83	82	80	79	77	74
Características														
10 Resistencia en bornes	Ω	0.334	0.605	1.23	2.07	3.09	3.72	4.75	7.46	11.5	17.5	26.2	40.5	68.2
11 Inductancia en bornes	mH	0.085	0.191	0.340	0.620	0.870	1.04	1.29	2.04	3.16	4.65	6.89	10.3	17.1
12 Constante de par	mNm / A	19.4	29.2	38.9	52.5	62.2	68	75.8	95.2	119	144	175	214	276
13 Constante de velocidad	rpm / V	491	328	246	182	154	140	126	100	80.5	66.4	54.6	44.7	34.6
14 Relación velocidad / par	rpm / mNm	8.43	6.79	7.76	7.16	7.62	7.67	7.89	7.85	7.84	8.08	8.19	8.46	8.55
15 Constante de tiempo mecánica	ms	5.97	5.60	5.50	5.40	5.38	5.38	5.39	5.38	5.37	5.38	5.39	5.39	5.41
16 Inercia del rotor	gcm²	67.6	78.7	67.6	72.0	67.4	67.0	65.2	65.4	65.5	63.6	62.8	60.8	60.4

Especificaciones

Datos térmicos

17	Resistencia térmica carcasa/ambiente	6.2 K / W
18	Resistencia térmica bobinado/carcasa	2.0 K / W
19	Constante de tiempo térmica del bobinado	30 s
20	Constante de tiempo térmica del motor	1050 s
21	Temperatura ambiente	-30 ... +100°C
22	Máx. temperatura del bobinado	+155°C

Datos mecánicos (rodamiento a bolas)

23	Máx. velocidad permitida	12000 rpm
24	Juego axial	0.05 - 0.15 mm
25	Juego radial	0.025 mm
26	Carga axial máx. (dinámica)	5.6 N
27	Máx. fuerza de empuje a presión (estática)	110 N
	(idem, con eje sostenido)	1200 N
28	Carga radial máx. a 5 mm de la brida	28 N

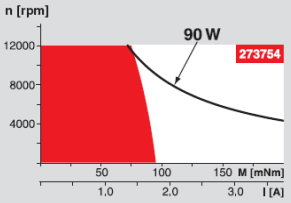
Otras especificaciones

29	Número de pares de polos	1
30	Número de delgas del colector	13
31	Peso del motor	340 g

Los datos de la tabla son valores nominales.
Explicación del diagrama en página 49.

Opción
Eje hueco disponible en versión especial
Rodamiento a bolas pretensado

Rango de funcionamiento



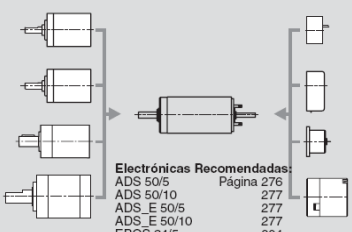
Leyenda

Funcionamiento continuo
Teniendo en cuenta los valores de resistencia térmica antes mencionados (líneas 17 y 18). El rotor alcanzará la máxima temperatura durante funcionamiento continuo a 25°C de temperatura ambiente = límite térmico.

Funcionamiento intermitente
El motor puede ser sobrecargado durante cortos períodos (cíclicamente).

Potencia nominal asignada

Sistema Modular maxon Esquema general en página 16 - 21



Reductor planetario
∅32 mm
0.75 - 4.5 Nm
Página 239

Reductor planetario
∅32 mm
1.0 - 6.0 Nm
Página 240

Reductor planetario
∅32 mm
8 Nm
Página 242

Reductor planetario
∅42 mm
3 - 15 Nm
Página 244

Electrónicas Recomendadas:

ADS 50/5	Página 276
ADS 50/10	277
ADS_E 50/5	277
ADS_E 50/10	277
EPOS 24/5	294
EPOS2 50/5	295
EPOS P 24/5	297
Notas	**18**

Encoder MR
256 - 1024 ppv,
3 canales
Página 259

Encoder HED_ 5540
500 ppv,
3 canales
Página 262 / 264

Tacodinamo DCT
∅22 mm
0.52 V
Página 271

Freno AB 28
∅40 mm
24 VDC, 0.4 Nm
Página 308

Edición de octubre de 2008 / Sujeto a modificaciones

ANEXO II ISCM 8005 INTELLIGENT SERVO DRIVE

Your Next Intelligent Move

TECHNOSOFT
MOTION TECHNOLOGY

ISCM4805 / 8005 Intelligent Servo Control Modules 200W

Digital Motor Control for Brushless, DC Brush, Linear and Step Motors

The ISCM4805 and ISCM8005 are new Technosoft high-performance intelligent servo modules, combining motion controller, drive and PLC functionality in a single compact unit.
The ISCM modules are flexible, cost effective and compact solutions, particularly adapted for distributed and co-ordinated control of brushless, DC, linear or step motors of powers up to 240W, with voltages up to 80V.
Typical applications include distributed motor control with possibilities of gearing and electronic CAM functions in a CAN network operation.
Targeted for medium to high volume applications, the ISCM hardware structure is based on a cost optimised design integrating all the basic motor control functions on one double-sided credit card format. A series of I/O signals, both digital and analogue, are available for easy interfacing with the application.
A complete set of high-level Technosoft Motion Language (TML) instructions permit to define and start complex motion sequences from your host, PC, or to execute pre-stored motion sequences selected from I/O lines, in a stand-alone mode.
The Embedded Intelligence of the ISCM facilitates the configuration and programming of the module through a high level graphical interface as the EasyMotion Studio.

Your Next Intelligent Move
"Motion Control at the Click of a Mouse"

The configuration, tuning and programming of the ISCM intelligent servo modules is easy using the powerful graphical Technosoft EasyMotion Studio.
System **configuration** and **parameterization** are performed by the selection and test of system structure, motor and sensors type and control mode.

P091.047.052.ISCM.LFT.0509

Features

- Fully digital servo drive with embedded intelligence and PLC functionality
- Suitable for brushless DC, brushless AC (vector control) DC brush, linear and two-phase step motors
- Compact open frame design (70x50 mm) credit card format; DIN-rail version also available
- Various control modes as:
  - Torque, speed or position control
  - Electronic gearing, contouring, profiling
  - Step motor emulation (step and direction input)
  - External variables control capabilities (pressure, flow, temperature etc.)
- Powerful Technosoft Motion Language (TML) instruction set for definition and execution of motion sequences in:
  - Single or multi axis control (master or slave mode)
  - Standalone operation with Stored Motion sequences
- RS-232 serial communication
- CAN-Bus 2.0B up to 1 Mbit/s / CANopen
- Programmable digital input / outputs and analog inputs
  - 5 to 8 programmable inputs / outputs
  - Differential quadrature encoder and digital Halls interface
  - Linear Hall sensors interface
  - 2 analog inputs, 0...5V; +/- 10V range
- Motor power supply 48V - ISCM4805, or 80V - ISCM8005
- High current capability (5A continuous, 16A peak current)
- Protection for over current, short circuit, earth fault, over- and under-voltage, I²t, control error
- 2.54 mm pitch edge connector
- Custom hardware and firmware options available (PMCM series)

Typical Applications

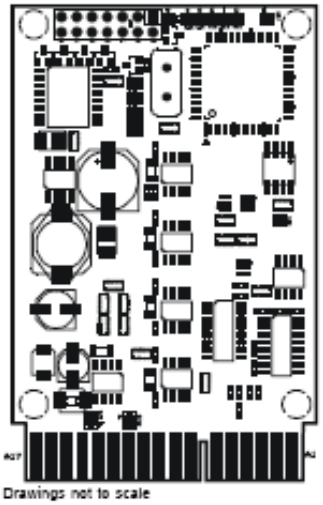
- Systems with distributed motor control intelligence
- Packaging equipment
- Printing
- Textile
- Medical
- Automotive
- Pick and place
- Factory automation

Application notes with ready to run Motion Language program examples are available at www.technosoftmotion.com

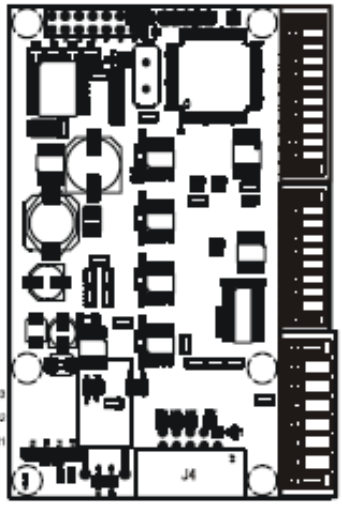
DIMENSIONS, SPECIFICATION, ORDERING INFORMATION

ISCM4805 / ISCM8005

ISCM4805 / ISCM8005 v1.3



ISCM4805 / ISCM8005 - DIN v1.3



Drawings not to scale

EASYMOTION STUDIO

The high level graphical development environment EasyMotion Studio, supports the configuration, parameterization and programming of the drive, through

- Motion system set-up wizard
- Tuning assistance
- Definition, programming and testing of motion sequences

MOTION CONTROL LIBRARIES

The TML_LIB Motion Control Libraries can be used to implement a motion control application on a PC from Visual C / C++, C#, Visual Basic, Delphi or LabVIEW under Windows or Linux operating systems.
If a PLC is used as host, implementations of the TML_LIB observing the IEC 61131 standard are available for Siemens and Omron PLCs.

ISCM STARTER KIT

Complete evaluation packages for the ISCM drives, containing the servodrive, motor, I/O board, EasyMotion Studio software that are supported by a collection of application notes and documentation.

P091.047.052.ISCM.LFT.0509

ISCM4805 / ISCM8005 INTELLIGENT SERVO MODULES

Electrical Specifications	ISCM4805	ISCM8005
DC supply voltage: logic	24V	
motor	12-48V	12-80V
Maximum continuous current	5A	
Peak current (100 ms. max.)	15A	
Minimal load inductance	200 microHenry*	330 microHenry*
Nominal switching frequency	20kHz	
Operating ambient temperature	0°C-40°C	

*at 48V (ISCM4805) / 80V (ISCM8005) and 20kHz switching frequency

Ordering Information

P047.001.E201	ISCM4805	Servo Module, 48V, 5A, CAN
P047.001.E211	ISCM4805	Servo Module, 48V, 5A, CANopen
P047.001.E301	ISCM8005	Servo Module, 80V, 5A, CAN
P047.001.E311	ISCM8005	Servo Module, 80V, 5A, CANopen
P052.001.E201	ISCM4805	Servo Module, 48V, 5A, DINrail, CAN
P052.001.E211	ISCM4805	Servo Module, 48V, 5A, DINrail, CANopen
P052.001.E301	ISCM8005	Servo Module, 80V, 5A, DINrail, CAN
P052.001.E311	ISCM8005	Servo Module, 80V, 5A, DINrail, CANopen
P047.001.E084	ISCM4805	Starter Kit for Brushless Motor
P047.001.E085	ISCM4805	Starter Kit for Stepper Motor
P047.001.E184	ISCM4805	I/O board
P034.001.E002		EasyMotion Studio software
P040.001.Exxx		TML_LIB Motion Libraries**

**ask for existing libraries types

FLEXIBILITY

Standard control schemes supported by the ISCM Drives

Motor Types	Torque control	Speed control	Position control
Brushless DC / AC	✓	✓	✓
DC Brush	✓	✓	✓
Linear	✓	✓	✓
Step	✓	✓	✓

Headquarters

SWITZERLAND
Tel.: +41 32 732 5500
Fax: +41 32 732 5504
sales@technosoftmotion.com

GERMANY
Cell: +49 (0)1718 077 664
Tel.: +49 (0)7231 424 919
Fax: +49 (0)7231 424 920
sales.de@technosoftmotion.com

BENELUX
Tel.: +32 (0)14 21 18 21
Fax: +32 (0)14 21 18 23
sales.be@technosoftmotion.com

EASTERN EUROPE
Tel.: +40 (0)21 425 90 95
Fax: +40 (0)21 425 90 97
sales.ro@technosoftmotion.com

UNITED STATES
Tel.: +1 784 667 5275
Fax: +1 784 667 5276
sales.us@technosoftmotion.com

www.technosoftmotion.com

ANEXO III HOJA DE CARACTERÍSTICAS IDENT

System Identification Toolbox 7

Create linear and nonlinear dynamic models from measured input-output data

System Identification Toolbox lets you construct mathematical models of dynamic systems from measured input-output data. This data-driven approach helps you describe systems that are not easily modeled from first principles or specifications, such as chemical processes and engine dynamics. It also helps you simplify detailed first-principle models, such as finite-element models of structures and flight dynamics models, by fitting simpler models to their simulated responses.

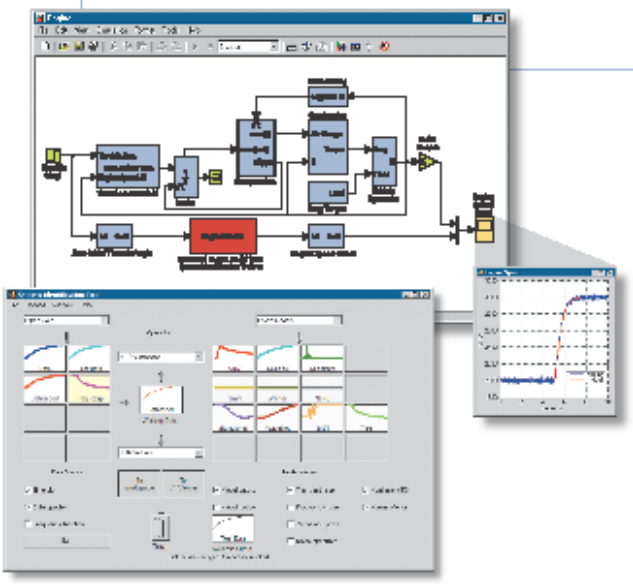
Models obtained with System Identification Toolbox are well suited for simulation, prediction, and control system design using products such as Simulink®, Control System Toolbox, and Model Predictive Control Toolbox (all available separately).

System Identification Toolbox lets you fit linear and nonlinear models to data, a process known as black-box modeling. Available model structures include low-order process models, transfer functions, state-space models, linear models with static nonlinearities at the inputs or outputs, and nonlinear autoregressive models. If you have a mathematical model of the system dynamics, you can tune its parameters to better match experimental data, a process known as grey-box modeling.

The principal architect of the toolbox is Professor Lennart Ljung, a recognized leader in the field of system identification.

KEY FEATURES

- Identifies linear and nonlinear models from time- and frequency-domain data
- Simplifies identification of first-, second-, and third-order continuous-time models
- Analyzes measured data and advises on data quality, required preprocessing, and presence of feedback or nonlinearities
- Provides data processing tools for detrending, filtering, and reconstructing missing data
- Provides Simulink blocks for simulating identified models and transferring data to and from the MATLAB® workspace
- Provides an interface for using linear models in Control System Toolbox (available separately)



Using System Identification Toolbox with Simulink. The toolbox provides blocks that let you bring identified models into Simulink.

The MathWorks
Accelerating the pace of engineering and science

Working with System Identification Toolbox

System Identification Toolbox facilitates the multistep process of identifying models from data. The toolbox enables you to:

- Analyze and process data
- Determine suitable model structure and order
- Estimate model parameters and validate model accuracy
- View the model responses and their uncertainties

You can perform these tasks by using either command-line functions or a graphical user interface (GUI).

You can convert the models into linear time-invariant (LTI) objects for use with Control System Toolbox. You can incorporate most identified models into Simulink models using blocks provided by the toolbox.

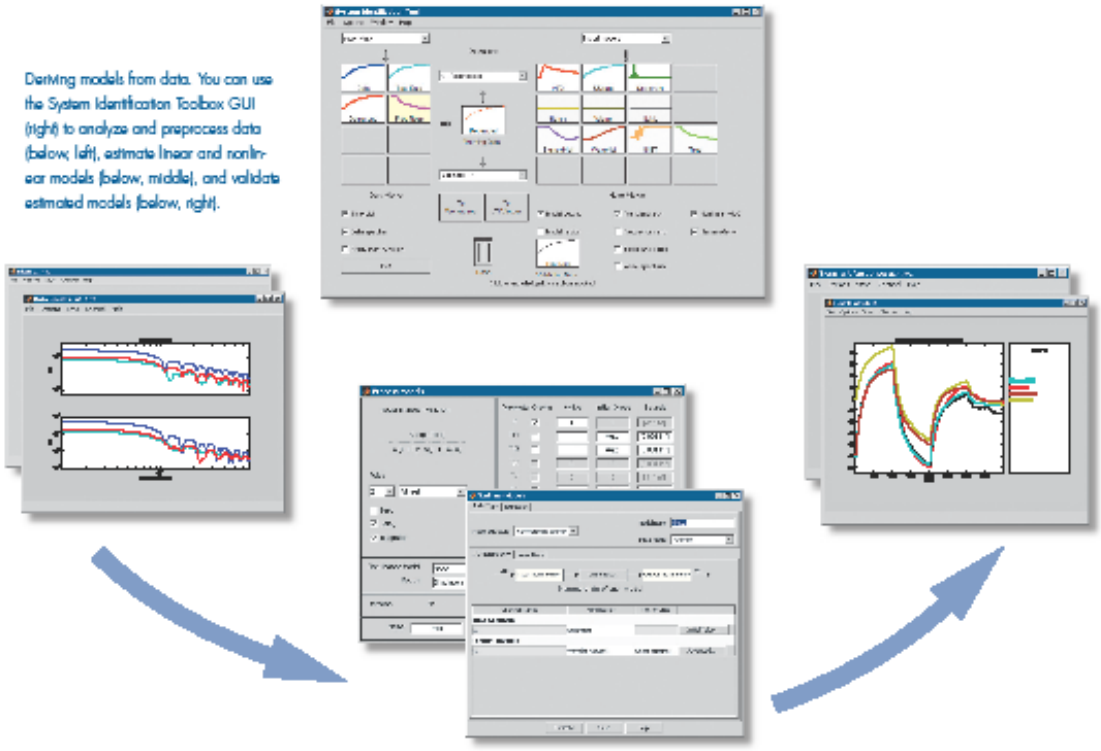
Working with Measured Data

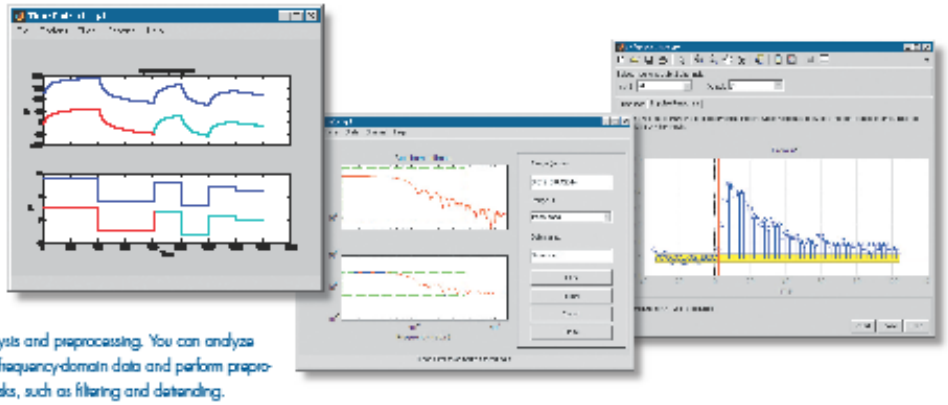
When preparing data for identification of models, you need to specify information such as input-output channel names, sampling interval, and intersample behavior.

The toolbox lets you attach this information to the data using data objects. The data objects facilitate easy visualization of data, domain conversion, and various preprocessing tasks.

Measured data often has offsets, slow drifts, outliers, missing values, and other anomalies. The toolbox removes such anomalies by performing operations such as detrending, filtering, resampling, and reconstruction of missing data. The toolbox can analyze the suitability of data for identification and provide diagnostics regarding persistence of excitation, existence of feedback loops, intersample behavior, and presence of nonlinearities.

Deriving models from data. You can use the System Identification Toolbox GUI (right) to analyze and preprocess data (below, left), estimate linear and nonlinear models (below, middle), and validate estimated models (below, right).





Data analysis and preprocessing. You can analyze time- and frequency-domain data and perform preprocessing tasks, such as filtering and detrending.

The toolbox produces estimates of step and frequency responses of the system directly from measured data. Using these responses, you can analyze system characteristics, such as time constants, input delays, and resonant frequencies. You can use this information to configure the parametric models during estimation.

Estimating Parametric Models

Parametric models, such as transfer functions or state-space models, use a small number of parameters to capture system dynamics. The toolbox estimates model parameters and their uncertainties. You can analyze these models, or their linear equivalents, using time- and frequency-response plots such as step, impulse, bode plots, and pole-zero maps.

Estimating Linear Black-Box Models

You can identify polynomial and state-space models using various estimation routines offered by the toolbox. These routines include autoregressive models (ARX, ARMAX), Box-Jenkins (BJ) models, Output-Error (OE) models, and state-space parameterizations. Estimation techniques include maximum likelihood, prediction-error minimization schemes, and such subspace methods as CVA, MOESP, and N4SID. You can also estimate a model of the noise affecting the observed system.

In cases where you only need a low-order continuous-time model, the toolbox provides special capabilities to simplify the estimation process and obtain results quickly. These models are expressed as simple transfer functions involving three or fewer poles, and optionally, a zero, a time-delay, or an integrator.

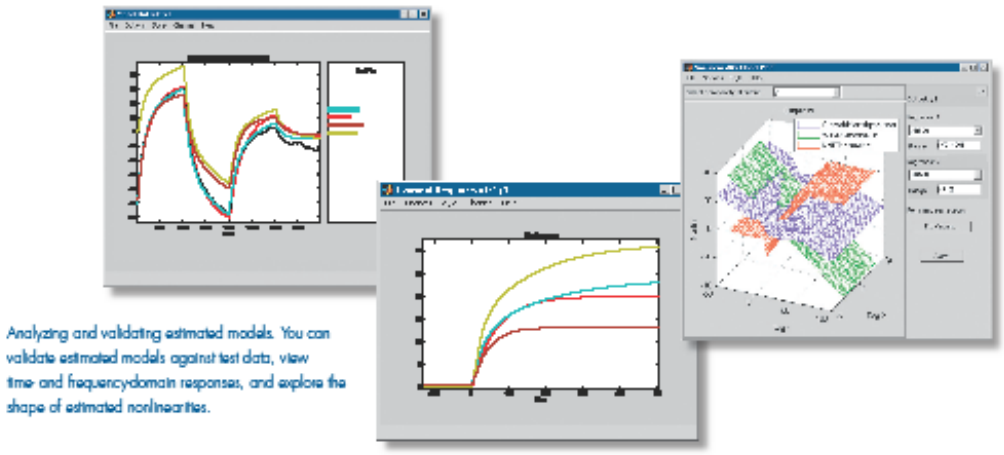
Estimating Nonlinear Black-Box Models

When linear models are not sufficient to capture system dynamics, you can estimate nonlinear models, such as nonlinear ARX and Hammerstein-Wiener models. Nonlinear ARX models enable you to model nonlinearities using wavelet networks, tree-partitioning, sigmoid networks, and neural networks (with Neural Network Toolbox, available separately). Using Hammerstein-Wiener models, you can estimate static nonlinear distortions present at the input and/or output of an otherwise linear system. For example, you can estimate the saturation levels affecting the input current into a DC motor, or capture a complex nonlinearity at the output using a piecewise linear nonlinearity.

Estimating Parameters in User-Defined Models

A user-defined (grey-box) model is a set of differential or difference equations with some unknown parameters. The toolbox lets you specify the model structure and estimate its parameters using nonlinear optimization techniques. For linear models, you can explicitly specify the structure of state-space matrices and impose constraints on identified parameters. For nonlinear models, you can specify differential equations as C, FORTRAN, or M-code.

www.mathworks.com



Analyzing and validating estimated models. You can validate estimated models against test data, view time- and frequency-domain responses, and explore the shape of estimated nonlinearities.

Validating Results

System Identification Toolbox helps validate the accuracy of identified models using independent sets of measured data from a real system. For a given set of input data, you can compare simulated output of the identified model with the measured output from the real system. You can also view the prediction error and produce time- and frequency-response plots with confidence bounds to visualize the effect of parameter uncertainties on model responses.

Required Products

MATLAB

Related Products

Simulink. Simulation and Model-Based Design

Control System Toolbox. Design and analyze control systems

Neural Network Toolbox. Design and simulate neural networks

Signal Processing Toolbox. Perform signal processing, analysis, and algorithm development

Simulink® Parameter Estimation. Estimate model parameters using test data

For a complete list of related products, visit *www.mathworks.com/products/sysid*

Platform and System Requirements

For platform and system requirements, visit *www.mathworks.com/products/sysid* ■

Resources

VISIT
www.mathworks.com

TECHNICAL SUPPORT
www.mathworks.com/support

ONLINE USER COMMUNITY
www.mathworks.com/matlabcentral

DEMOS
www.mathworks.com/demos

TRAINING SERVICES
www.mathworks.com/training

THIRD-PARTY PRODUCTS AND SERVICES
www.mathworks.com/connections

WORLDWIDE CONTACTS
www.mathworks.com/contact

E-MAIL
info@mathworks.com

The MathWorks
Accelerating the pace of engineering and science

8000v07 03/07

ANEXO IV PROGRAMAS PARA LA TRANSFORMACIÓN DE DATOS

```
/*
Software para adaptar los datos obtenidos en EMS a formato reconocido por Matlab
Autor:Guillermo Gallardo Fernández
*/
#include <stdio.h>
main(){
char c;
char x; // Variable para copiar.
int linea=1;
int num_lineas;
int tab=0; // hemos pasado la tabulación si TAB=1;
FILE *fentEMS;
FILE *fsalMLAB;
fentEMS = fopen("fent_EMS.txt", "r");
fsalMLAB = fopen("fsal_MLAB.txt", "w");
if (fentEMS == NULL){
   printf("Error, no se puede abrir el archivo\n");
   exit(0);
}
if (fsalMLAB == NULL){
   printf("Error, no se puede crear el archivo\n");
   exit(0);
}
/* se procesa el archivo */
/*Mostrar por pantalla hasta que encuentra tabulador*/
while(!feof(fentEMS)){
   if (c == '\n'){ //Calcula en nº de linea y dice q ya hemos terminado la fila X.
      //linea=linea+1;
      tab=0;
   }
   if (tab == 1){ //Si Hemos pasado el tabular empezamos a COPIAR.
      putchar(c = getc(fentEMS));
      //fscanf(fentEMS, "%c", &c); //Otra forma de hacerlo.
      //Copia si no hay '\n' y si hay '\n' mete un espacio.
      if (c == '\n'){
      fprintf(fsalMLAB,"%c", ' ');}
      else{
      fprintf(fsalMLAB, "%c", c);}
   }
   else{ //Si estamos antes del tabulador solo actualiza la dirección del puntero
      c = getc(fentEMS);
      }
   if (c == 9){ //Si estamos en el tabulador (9) pone tab a 1 para saber q lo hemos pasado
      //printf("%d -> ",linea); //imprime el nº de linea para ver si está bien
      tab = 1;
   }
}
fclose(fentEMS);
fclose(fsalMLAB);

printf("\n");
system("PAUSE");

exit(0);
}
```