Universidad del País Vasco
Euskal Herriko Unibertsitatea

ZIENTZIA ETA TEKNOLOGIA FAKULTATEA
FACULTAD DE CIENCIA Y TECNOLOGÍA

Trabajo Fin de Grado

Grado en Ingeniería Electrónica

Implementación de maqueta de motor DC y aplicación de diferentes tipos de control utilizando Arduino

Autor:

Borja González Perancho

Director:

Dr. Ibón Sagastabeitia Buruaga

Leioa, 21 de Junio de 2019

eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea
ZIENTZIA ETA TEKNOLOGIA FAKULTATEA
FACULTAD DE CIENCIA Y TECNOLOGÍA

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA ELECTRÓNICA

Implementación de maqueta DC y aplicación de diferentes tipos de control utilizando Arduino

MEMORIA PRESENTADA POR BORJA GONZÁLEZ PERANCHO

FECHA DE MATRICULACIÓN Y DEFENSA: JUNIO Y JULIO 2019

DIRECTOR: DR. IBON SAGASTABEITIA BURUAGA

DEPARTAMENTO: CONTROL AUTOMÁTICO

Leioa, 21 de Junio de 2019

Índice

CAPÍTULO 1: INTRODUCCIÓN

En el presente trabajo de fin de grado, se pretende diseñar un dispositivo hardware de simulación de los diferentes controles estudiados en las asignaturas de Control Automático I y Control Automático II [1]. Se trata de un sistema formado por una varilla capaz de desplazarse angularmente [2]. La idea es sacar a la varilla de su posición de equilibrio mediante una perturbación externa. El sistema de control diseñado, actuará entonces para hacer que vuelva a la posición inicial con la mejor dinámica posible con la ayuda del actuador del sistema, un motor de corriente continua, y un controlador implementado en la placa Arduino Uno [3]. Además de lo anterior, con el objetivo de una mejor visualización de la dinámica que introducen los controles, se aplican cambios de referencia sacando a la varilla de la posición de equilibrio.

Debido a que la placa Arduino trabaja con señales digitales, se tendrán que discretizar los controles. Además, se deberá elegir un periodo de muestreo adecuado para el correcto funcionamiento del sistema y evitar problemas como puede ser el Aliasing [4]. También la elección de los componentes tiene un papel fundamental a la hora de conseguir los mejores resultados. Características como la resolución, la zona muerta y la saturación serán importantes para conocer las limitaciones que tendrán en el sistema.

La memoria se divide en varios capítulos, ordenados cronológicamente según la realización del proyecto. En ellos se detallan de forma concisa, los procedimientos que se han llevado a cabo para diseñar y evaluar cada una de las partes que componen este proyecto.

La primera parte está dedicada a la elección de los elementos que lo forman y la elaboración de la estructura que los soporte. El trabajo continúa con la obtención del modelo matemático del sistema, crucial para que finalmente se pueda concluir el proyecto con el diseño e implementación de los diferentes controladores y comprobar su funcionamiento en el sistema.

CAPÍTULO 2: PROTOTIPO

Para crear el dispositivo de control angular de posición, es importante seguir una serie de pautas de selección de componentes y de construcción, antes de empezar el diseño de los controles. Este capítulo está dedicado a los componentes que formarán el sistema y al papel que juegan en él.

2.1. ELECCIÓN DEL SENSOR

En este proyecto, el sensor es considerado algo esencial. Es el que se encarga de registrar la posición de la varilla cada periodo de muestreo, permitiendo al sistema de control un correcto funcionamiento.

La primera decisión que se toma es la elección del sensor, que se encargará de recoger el desplazamiento rotatorio, cuando se saca la varilla de la posición de equilibrio. La característica a tener en cuenta para dicha elección, es la resolución. En este caso, al ser un desplazamiento angular, la medida será en grados sexagesimales. Por lo tanto, será importante conocer el menor cambio en la magnitud de entrada (posición angular) que se aprecia en la magnitud de salida (tensión) de cada sensor.

En esta parte del capítulo, se analizan dos posibles candidatos para ser utilizados como sensores del sistema.

2.1.1. POTENCIÓMETRO

Una de las opciones es utilizar un potenciómetro. El potenciómetro es excelente para medir ángulos de giro. Cuando se realiza una rotación de su eje, se cambia la posición del cursor que tiene en su interior, variando así el divisor de tensión que crea entre sus tres patillas, permitiendo conocer el ángulo girado en función de la salida de voltaje que se tiene.

Para conocer la resolución del potenciómetro, es necesario obtener su curva característica que, para este caso, es posición angular frente a voltaje.

Se ha creado un dispositivo de medición formado por el potenciómetro, un transportador de ángulos, una alimentación de $+5V$ con la que se ha alimentado el potenciómetro y un multímetro.

El movimiento rotatorio del potenciómetro, hace cambiar el valor de la tensión que se registra en el multímetro, los grados correspondientes a dichos valores se adquieren a través del transportador y un indicador unido a la parte móvil del potenciómetro.

Los datos obtenidos vienen registrados en la siguiente tabla y, a su derecha, se presenta su curva característica (Figura 1).

Grados (°)	Voltios (V)
40	1,55
90	2,44
80	2,24
75	2,17
70	2,08
65	2,00
50	1,73
95	2,52
100	2,60
105	2,69
110	2,78
120	2,97
130	3,16
140	3,35

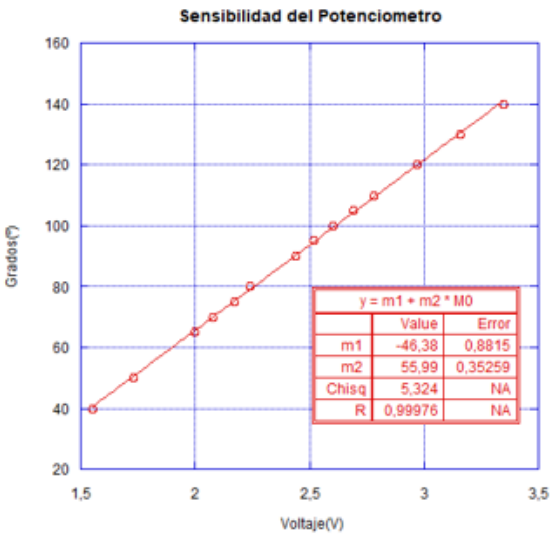

Figura 1. Tabla de los diferentes valores de voltaje asociados a los grados medidos y curva característica del potenciómetro obtenidos.

De dicha curva se obtiene la ganancia del potenciómetro, que es la pendiente de la recta.

$$G_{potenciómetro} = 55.99°/V = 0.97\ rad/V \tag{2.1}$$

Para obtener la resolución en grados/bit, basta con multiplicar la resolución que tiene el Arduino Uno (V/Bit), por la ganancia del potenciómetro (Ecuación (1.2).

$$Resolución = G \times \frac{5\ V}{1024} = 55.99 \times \frac{5\ V}{1024} = 0.27°/bit \tag{1.2}$$

2.1.2. ENCODER

La otra opción de medir ángulos es utilizar un encoder incremental. Es un transductor rotativo que, mediante una señal eléctrica (normalmente un pulso de tensión), nos indica el ángulo girado. El número de pulsos que manda por vuelta, indican la resolución del encoder.

La resolución del encoder utilizado en el laboratorio (600EN-128-CBL) se obtiene de la correspondiente hoja de características y se hace una pequeña conversión para ajustar las unidades (Ecuación (2.2). De la hoja de características se toma el valor "*Pulse Per Revolution*".

$$Resolución_{encoder} = \frac{1\ Rev}{128\ Pulsos} \times \frac{360\ °}{1\ Rev} = 2.81°/pulso \tag{2.2}$$

Los valores de la resolución de los sensores no pueden compararse directamente, debido a que los métodos de lectura de ellos a través del Arduino, son diferentes.

Interpretando los datos, por cada $0.27°$ de giro del potenciómetro, corresponde un bit del *conversor Analógico/Digital* del Arduino Uno. Por otro lado, por cada $2.81°$, el encoder emite un pulso. Esto implica que el potenciómetro, no es capaz de corregir errores inferiores a $0.27°$, ni el encoder valores inferiores a $2.81°$. Por ello, se ha decido que el candidato más apropiado para ser utilizado como sensor del sistema, es el potenciómetro.

2.2. ESTRUCTURA

En esta parte del trabajo, se muestra cómo se ha diseñado y montado la maqueta del proyecto. Con el objetivo de conseguir una estructura robusta y versátil, se decidió utilizar el programa "Autodesk Fusion 360" para realizar el diseño.

Las dos piezas que se han modelado son, por un lado, una estructura donde se colocan el motor y el sensor, a la cual se le añaden unas *barreras* para evitar la rotura del sensor en caso de fallo en el control. Por otro lado, se ha creado un acople que une el motor y el sensor no rígido para reducir al máximo el par producido por la posible desalineación de los ejes.

En la Figura 2 se muestra el diseño de estos dos elementos a través del programa mencionado.

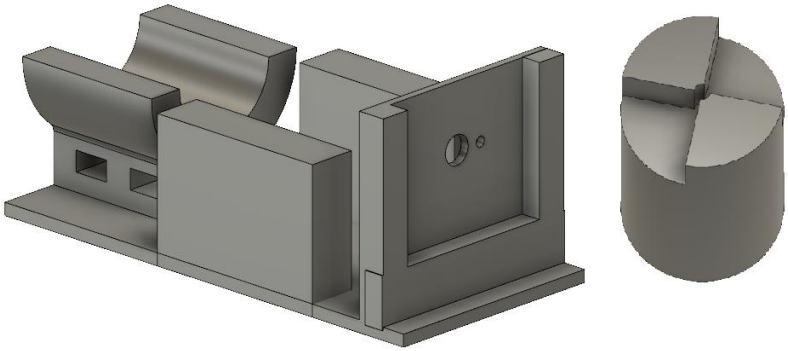

Figura 2. Diseño en el programa "Fusion" de las piezas que conformarán la maqueta, soporte y eje de unión.

El sistema real completo y montado se muestra en la Imagen 1.

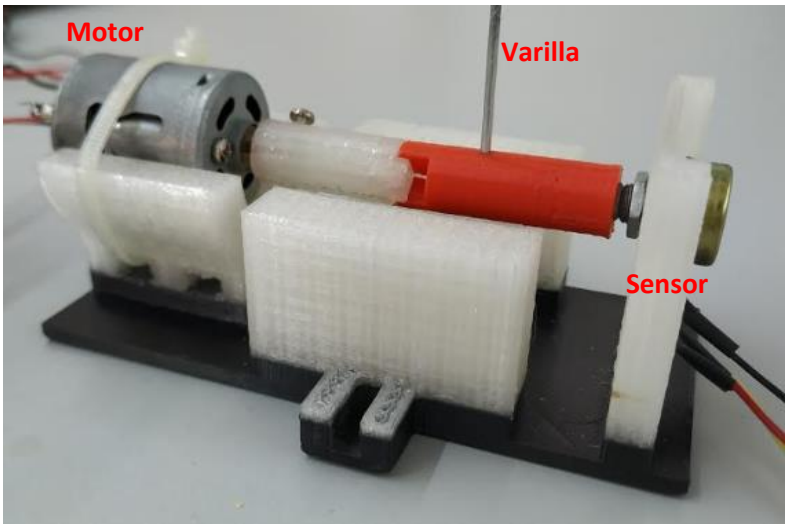

Imagen 1. Maqueta real del sistema con todos los elementos montados: motor, varilla y potenciómetro.

2.3. CIRCUITO ELÉCTRICO

El diseño del circuito eléctrico del sistema es otra de las partes importantes del proyecto. La elección correcta de la circuitería y de los componentes que lo forman, permitirá un mejor diseño posterior de los controladores. Este epígrafe da a conocer los componentes utilizados y cómo funcionan. Para entender el porqué de las elecciones tomadas, a continuación, se explica brevemente cómo funciona el sistema.

El potenciómetro realiza una lectura de la posición de equilibrio y ésta se almacena en una variable en el programa de Arduino. Cuando se saca la varilla de la posición de equilibrio, el sistema actúa para corregir el error a través del controlador que se ha implementado en dicho programa, creando una señal de control que se introduce al motor a través del driver.

Se presentan, a continuación, los elementos que se utilizan y su principio de funcionamiento.

2.3.1. DRIVER “POLOLU DUAL MC33926”

Es el componente del sistema que se encarga de convertir la señal de control que se crea en el programa de Arduino, una señal digital, en una señal de corriente continua, adecuada para el rango de voltajes con los que trabaja el motor. Esta señal de entrada al driver puede ser positiva o negativa, por lo que tiene que ser capaz de mandar al motor tensiones de ambos signos.

2.3.1.2. “PULSE WIDTH MODULATION” (PWM)

La idea de este proyecto es controlar la tensión con la que se alimenta el motor DC. El rango de valores con los que trabaja este motor es $[-15V, +15V]$. Es necesario, entonces, que en cada periodo de muestreo se envíen señales de voltaje dentro de esos valores. Sin embargo, el Arduino no es capaz de generar tensiones superiores a 5 voltios. Aquí es donde entra en función el driver. Gracias a la capacidad que tiene para crear señales “Pulse Width Modulation” (PWM), puede conseguirse el rango de señales deseados para alimentar el motor.

Esta técnica consiste en modificar el ciclo de trabajo de una señal periódica cuadrada de una determinada frecuencia. La señal que interpreta el motor corresponde a una tensión de voltaje continua [5].

Para que el driver pueda crear señales dentro del rango deseado, debe alimentarse con la tensión equivalente al valor máximo de tensión que se quiere conseguir, en este caso, $+15V$.

2.3.1.2. PUENTE H

Como ya se ha mencionado antes, las señales de control pueden tener valores negativos, lo que implica que el motor también pueda trabajar con tensiones negativas.

El puente H está constituido por 4 transistores que funcionan como interruptores y, dependiendo de cómo se accionen éstos, puede invertirse la polarización con la que se alimenta el motor [6]. Esta capacidad permite al motor trabajar con tensiones de ambos signos. El driver tiene implementados dos *puentes H* que permiten controlar la polarización de hasta dos motores de corriente continua.

El driver incluye una librería para el programa Arduino mediante la cual, utilizando el comando *setM1Speed ()* y un rango de valores digitales entre $[-400, +400]$, permite introducir señales de voltaje al motor de $[-15\mathrm{V}, +15\mathrm{V}]$. La conversión que se utiliza para conseguir el rango de valores adecuados para el comando, se presenta en el siguiente capítulo.

2.3.2. ARDUINO UNO

Es una plataforma de hardware libre, basada en una placa con un microcontrolador y un entorno de desarrollo. Se ha creado un programa en Arduino **(Anexo 1),** que se carga en la placa, en el que se ha implementado el funcionamiento del sistema y los controles que se diseñarán en el capítulo 4.

Como todas las señales con las que trabaja el Arduino son digitales, la implementación tanto del sistema, como de los controles, se tiene que realizar en el dominio discreto. Esto hace necesario elegir un periodo de muestreo adecuado para un correcto funcionamiento. En el capítulo 3 se obtiene y se explica la elección del mismo.

2.3.3. POTENCIÓMETRO

Como se ha mencionado al principio del capítulo, se ha elegido el potenciómetro como sensor del sistema. El potenciómetro se encarga de mandar el valor de la posición en la que se encuentra la varilla en cada periodo de muestreo al Arduino, para poder generar así, una señal de control en forma de voltaje con la que es alimentado el motor. En la Figura 3 se muestra el resultado del circuito eléctrico montado.

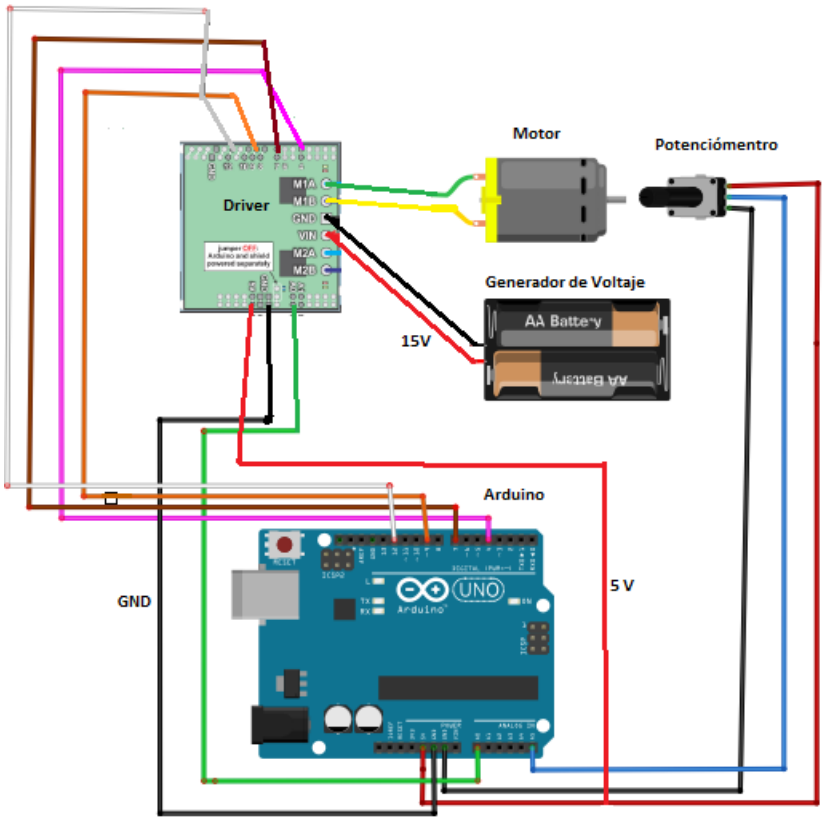

Figura 3. Circuito eléctrico completo formado por el driver, el motor, el potenciómetro y el Arduino Uno.

CAPÍTULO 3: MODELO MATEMÁTICO

En los sistemas de control la modelización matemática del sistema es imprescindible, dado que el diseño del control se va a realizar a partir del modelo obtenido. Por ello, como el sistema de la varilla es un sistema a controlar, se debe obtener su modelo matemático.

El elemento a controlar es la posición angular de la varilla, la cual está acoplada al eje del motor, el actuador del sistema, que se encarga de corregir su posición para devolverla a su punto de partida.

Para modelizar matemáticamente el sistema, hay que hallar las ecuaciones diferenciales que rigen su dinámica. Dentro de este sistema, el único elemento que necesita modelizarse es el motor de corriente continua. Debido a que se desconocen los parámetros que lo caracterizan, la obtención de su modelo se realiza de manera experimental.

En una primera fase del proyecto, se obtuvo el modelo matemático del sistema en lazo abierto, utilizando como sensor el encoder rotativo del capítulo anterior. De este modo, se eligió el periodo de muestreo que se utiliza a lo largo de todo el proyecto. Una vez se obtuvo el modelo, a partir del diagrama de Bode del sistema en lazo abierto, se eligió una frecuencia de muestreo que estuviera una década a la derecha de la frecuencia de ganancia crítica. Con ello se obtuvo el periodo de muestreo de trabajo para el sistema, cuyo valor es $T_m = 0.013$ segundos.

Sin embargo, el modelo matemático obtenido inicialmente no fue el definitivo, porque se observó que la alineación entre el sensor y el eje del motor, no era perfecta y el sistema de anclaje generaba un rozamiento considerable. Lo más conveniente, era lograr el modelo matemático una vez que se hubiera realizado el montaje de la maqueta final, consiguiendo así tener unos rozamientos que se mantuvieran lo menos variantes a lo largo de todo el proyecto.

3.1. DIAGRAMA DE BLOQUES DEL SISTEMA

En lo referente al sistema a diseñar, el diagrama de bloques es una manera gráfica de representar las relaciones entre las variables de un sistema, donde la función de cada uno de sus componentes, se representa en forma de su función de transferencia. Es un modelo matemático a través del cual se caracterizan las relaciones de entrada y salida de sistemas. Los diagramas de bloques se describen mediante ecuaciones diferenciales lineales invariantes en el tiempo. En la Figura 4 se muestra el diagrama de bloques del sistema final en lazo cerrado.

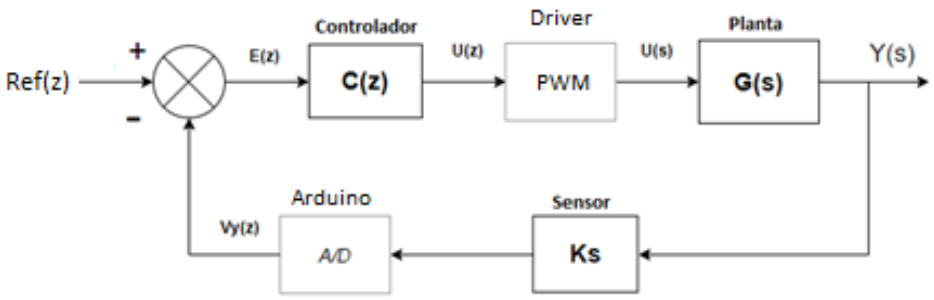

Figura 4. Diagrama de bloques del sistema en lazo cerrado.

K_s y $G(s)$ son las funciones de transferencia del sensor y del motor (en el diagrama llamados planta y sensor), respectivamente, y *C(z)* es el controlador, al que se dedicará un capítulo para sus diferentes diseños. $Ref(z)$ es la entrada que se introduce al sistema. Es una señal digital que tiene el mismo rango de valores del conversor A/D de Arduino [0-1023]. Para la medición experimental de la dinámica del sistema con los diferentes controles, se utilizará una entrada de 69 bits, correspondiente a un valor de 18.88°. $E(z)$ es el error de la posición angular de la varilla, esto es, la diferencia entre la posición en la que se encuentra la varilla y la referencia, medida en bits. $U(z)$ es la señal de control que se manda al motor, para corregir el error en la posición de la varilla (Ecuación (3.1)).

$$U(z) = C(z)E(z) \tag{3.1}$$

Finalmente, para convertir esta señal digital en una señal de corriente continua $U(s)$, se utiliza el driver descrito en el capítulo anterior, utilizando para ello la Ecuación (3.2).

$$U(z)_{ajustado} = U(z)\frac{5\ V}{1023\ bits}\frac{400\ bits}{15\ V} = \frac{400}{3069}\ U(z)\ bits \tag{3.2}$$

Se consigue adecuar el rango de valores para el comando de la librería del driver *setM1Speed ()* y generar así valores de tensión en el rango deseado $[-15\text{V}, +15\text{V}]$, a través del PWM.

3.2. OBTENCIÓN DEL MODELO MATEMÁTICO DEL MOTOR

Como se ha mencionado antes, la obtención del modelo matemático definitivo se realiza con el sistema final ya montado. Debido a que el sensor que se utiliza en el montaje final, es el potenciómetro y su rotación máxima es de 300°, no permitiendo una rotación completa, para evitar su rotura, la única forma de modelizar el motor, es mediante la respuesta del sistema en lazo cerrado, utilizando el diagrama de bloques de la Figura 4.

Los sistemas de control en lazo cerrado, son aquellos en los que existe una realimentación de la señal de salida o, lo que es lo mismo, aquellos en los que la señal de salida tiene efecto sobre la acción de control. Se trabaja con el error entre la referencia y la posición de la varilla, por lo tanto, se pueden realizar giros controlados de unos cuantos grados. Una vez se ha montado el sistema final en lazo cerrado, se pasa a modelizar el motor.

Conociendo las ecuaciones que rigen la dinámica de un motor de corriente continua, es posible deducir su función de transferencia. Haciendo una serie de aproximaciones, el motor se puede modelizar cómo una ganancia proporcional y un polo real.

La entrada al motor es una tensión de corriente continua, y la salida, la velocidad angular con la que gira su rotor. Como se desea controlar la posición angular, se introduce un integrador al modelo (un polo en el origen) ya que, como se sabe, la integral de la velocidad angular es la posición angular. Se obtiene entonces el modelo del *motor más integrador*, que en adelante se llamará *planta del sistema*, y que se expresa mediante la Ecuación (3.3).

$$G_p(s) = \frac{k}{(s + polo)} \times \frac{1}{s} \tag{3.3}$$

3.2.1. SISTEMA DE ADQUISICIÓN DE DATOS

Debido a que la transmisión serie que permite guardar los datos en Arduino, requiere un tiempo superior al periodo de muestreo escogido, se decidió utilizar una tarjeta de adquisición NI USB 653 y la herramienta *Labview* para obtener los datos. Se ha desarrollado un programa en Labview que permite obtener, a través de la tarjeta de adquisición, la evolución del sistema a lo largo del tiempo de forma visual.

Con este programa, a partir de la elección de un periodo de muestreo elegido de tal forma que se cumpla el criterio de Nyquist [4] y así evitar el aliasing, se obtienen una cantidad de muestras. Mediante estas muestras se pueden observar las partes más interesantes del sistema, como el transitorio y el estado estacionario, lo que permite conocer cómo de bueno es el controlador.

El programa recoge la señal de voltaje del sensor en cada periodo de muestreo, representándose dichos valores en una gráfica y obteniendo así la evolución temporal del sistema. En realidad, son muestras muy próximas entre sí que dan un efecto de respuesta temporal.

El programa también incorpora un “trigger”, el cual sincroniza la adquisición cuando se ejecuta el programa en Arduino, donde se ha diseñado todo el sistema de control. El “trigger” no es más

que un flanco de subida que se manda a través de uno de los pines digitales de la placa Arduino Uno y que es leído por tarjeta, activando así la etapa de adquisición.

En la Figura 5 se muestra el diagrama de bloques y el interfaz gráfico donde se obtienen las gráficas en el programa Labview.

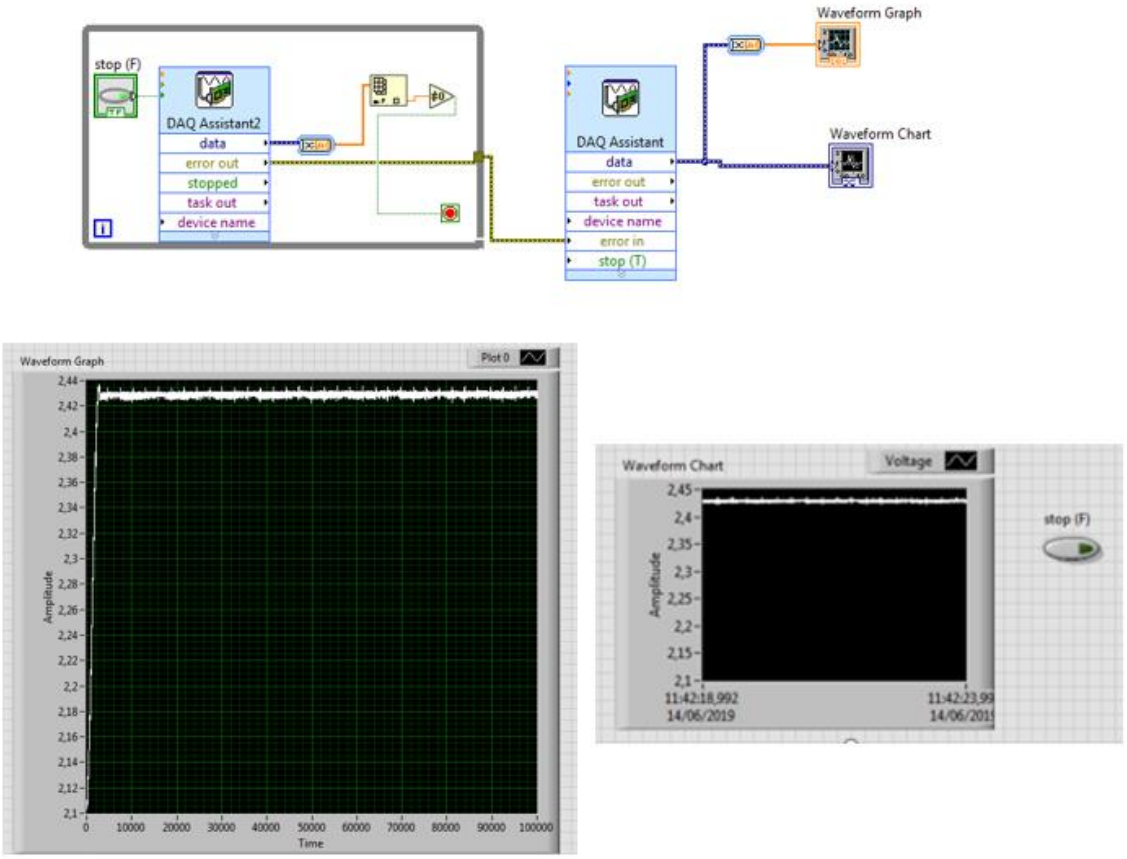

Figura 5. Programa adquisición de datos diseñado en Labview. En la parte superior está el diagrama de bloques y en la parte inferior donde se visualizan los resultados.

3.2.2. OBTENCIÓN DEL MODELO MATEMÁTICO

A partir de la maqueta diseñada en el capítulo anterior, con la forma que se ha elegido para obtener el modelo matemático y el programa diseñado para la adquisición, se utiliza un control proporcional con diferentes valores de su ganancia K_C para modelizar la planta del sistema. Se introduce una entrada escalón de 18.88°, correspondiente a una entrada digital de 69 bits, sacando la varilla de la posición inicial y se observa cómo evoluciona la dinámica con los diferentes valores de las ganancias. En la Figura 6 se muestra un esquema del movimiento.

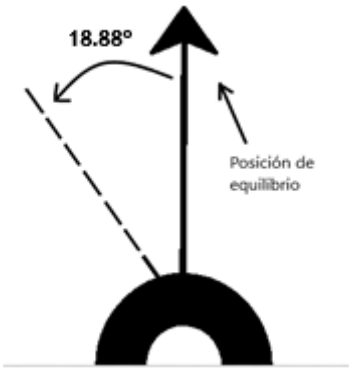

Figura 6. Imagen del movimiento para obtener el modelo matemático del sistema.

Los valores correspondientes de K_C para cada una de las gráficas de la Figura 7 son los mostrados en la Tabla 1, se incluye además el color que tiene cada una de ellas para su mejor comprensión.

K_C	Color de la gráfica
40	Rojo
50	Verde
60	Azul
100	Amarillo

Tabla 1. Parámetros del control proporcional.

Con el objetivo de una buena interpretación y un correcto entendimiento, las gráficas muestran la salida del sistema en grados, en vez de en voltios, que es la lectura que se obtiene. Las conversiones que se han hecho para llegar a estos resultados han sido las siguientes. En primer lugar, se ha eliminado el *offset* de la señal de voltaje. Debido a que el sistema se ha montado de tal forma que la varilla en la posición de equilibrio esté en la mitad del rango de voltaje del potenciómetro ($2.5\ V$), hay que restarle a cada una de las muestras, el valor obtenido de la primera de ellas. En segundo y último lugar, se multiplica el valor de voltaje por la ganancia del sensor, convirtiéndolo así en grados.

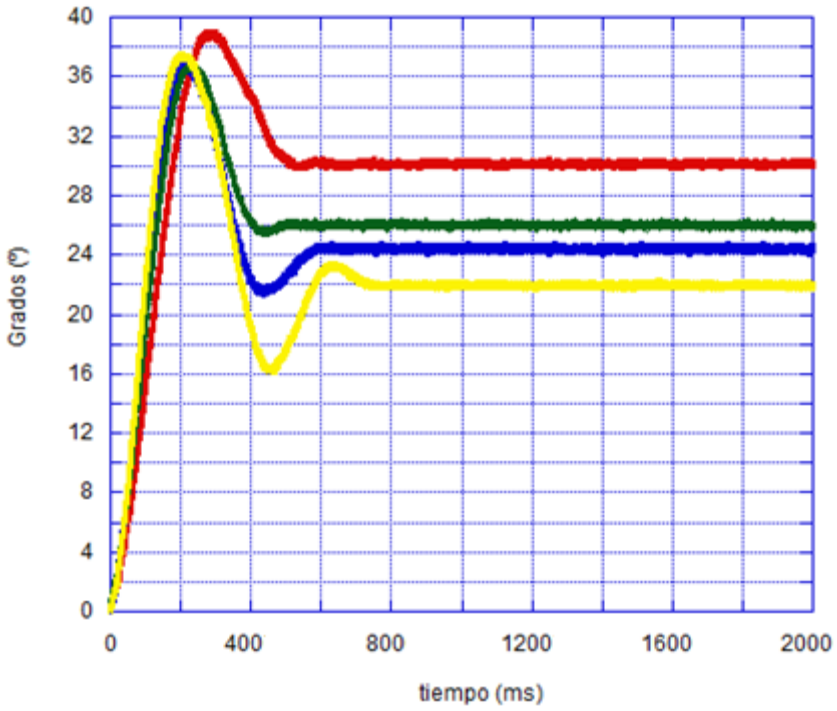

Figura 7. Respuesta del sistema en lazo cerrado para los diferentes valores de K_C.

Como se puede apreciar en las gráficas, el sistema en lazo cerrado tiene la forma de un sistema típico de segundo orden, por lo tanto, es posible obtener su modelo mediante la función de transferencia que lo caracteriza.

Analizando las gráficas, también se pueden apreciar dos factores que producen un error en el estado estacionario y un desplazamiento vertical de ellas. Son, por un lado, la zona muerta del motor, que posteriormente se calcula y, por el otro, el peso de la varilla.

Estos dos factores en conjunto cambian la forma de la dinámica del sistema, ya que, como hemos visto antes, el modelo, al tener un polo en el origen, debería tener un error en estado estacionario nulo. A pesar de ello, se ha decidido no tenerlos en cuenta para la obtención del modelo, ya que no se han considerado muy importantes y se ha seguido adelante.

Se sabe de la teoría [7], que la forma de la función de transferencia de un sistema puramente de segundo orden, es la que se presenta en la Ecuación (3.4);

$$G(s) = \frac{k{w_n}^2}{s^2 + 2\delta w_n \cdot s + {w_n}^2} \tag{3.4}$$

siendo K la ganancia del sistema, δ la relación de amortiguamiento de los polos complejos conjugados y w_n la frecuencia de oscilación natural de los mismos.

A partir de las expresiones matemáticas de las especificaciones del rebose y el tiempo de pico de un sistema típico de segundo orden, y mediante los datos obtenidos de la respuesta del sistema (Figura 7), se pueden obtener los valores para las Ecuaciones (3.5) y (3.6), la de la relación de amortiguamiento y la de la frecuencia natural.

$$\delta = \sqrt{\frac{Ln\frac{100}{\%Rebose}^{2}}{\pi^2 + Ln\frac{100}{\%Rebose}^{2}}} \tag{3.5}$$

$$w_n = \frac{\pi}{T_1\sqrt{1-\delta^2}} \tag{3.6}$$

En la Tabla 2 se recogen todos estos parámetros para los diferentes valores de la ganancia del control proporcional K_C.

K_C	Rebose (%)	T_1 (ms)	δ	w_n(rad/s)
40	29.41	0.29	0.35	11.59
50	42.22	0.22	0.26	13.57
60	51.16	0.21	0.20	14.74
100	69.23	0.20	0.11	15.80

Tabla 2. Parámetros temporales obtenidos para los diferentes valores de la ganancia.

Del diagrama de bloques de la Figura 4 y suponiendo todo el sistema en el dominio *s* al completo por su mayor simplicidad, la función de transferencia en lazo cerrado del sistema se expresa según la Ecuación (3.6). La elección del domino *s* puede hacerse porque el periodo de muestreo es suficientemente bueno.

$$G_{LC}(s) = \frac{K_c\, K_s\, G_p(s)}{1 + G_p(s)\, K_s} \tag{3.7}$$

Despejando $G_p(s)$ de la Ecuación (3.6), la función de transferencia de la planta queda se expresa en la Ecuación (3.7).

$$G_p(s) = \frac{G(s)_{LC}}{K_c K_s\big(1 + G_{LC}(s)\big)} \tag{3.8}$$

En la Tabla 3 se presentan los diferentes modelos obtenidos para las diferentes ganancias.

K_C	Modelo
40	$G_p(s) = \frac{3.28}{s \times (8.26 + s)}$
50	$G_p(s) = \frac{3.60}{s \times (7.18 + s)}$
60	$G_p(s) = \frac{3.54}{s \times (6.15 + s)}$
10	$G_p(s) = \frac{2.24}{s \times (3.67 + s)}$

Tabla 3. Modelos matemáticos obtenidos para los diferentes valores de K_C.

Se ha decidido usar como modelo matemático de la planta el resultado de la media de los cuatro modelos obtenidos, quedando la planta de la siguiente forma:

$$G_p(s) = \frac{3.22}{s \times (6.31 + s)} \tag{3.9}$$

Para finalizar, se presenta una tabla con todas las funciones de transferencia que conforman el diagrama de bloques del sistema.

Sensor: K_s	Controlador: C(z)	Planta: $\mathrm{G(s)_p}$
1.023 (V/rad)	Se diseñará en el siguiente capítulo	$G_p(s) = \frac{3.22}{s \times (6.31 + s)}$

Tabla 4. Funciones de transferencia de los diferentes elementos del sistema.

3.3. ZONA MUERTA

Se entiende como zona muerta a la mínima señal de entrada necesaria para que el actuador del sistema responda [8]. Se ha podido observar, en las gráficas obtenidas para sacar el modelo matemático, cómo ésta afecta a la dinámica del sistema para determinados valores de voltaje en los que el motor no responde.

Se ha elaborado un método para obtener la zona muerta del motor. Consiste en introducir al motor señales de control de diferentes valores y de forma ascendente empezando por valores pequeños, hasta encontrar el límite para el cual el motor comienza a actuar. Utilizando el multímetro, se obtiene el voltaje con los que está siendo alimentado.

Se ha obtenido que, para señales de control positivas, el motor no actúa hasta que la tensión de entrada es superior a $+3.45\ \mathrm{V}$ y que, para señales de control negativas, la tensión necesaria es de $-2.68\ \mathrm{V}$. En la Figura 8 se representa de manera visual esta situación.

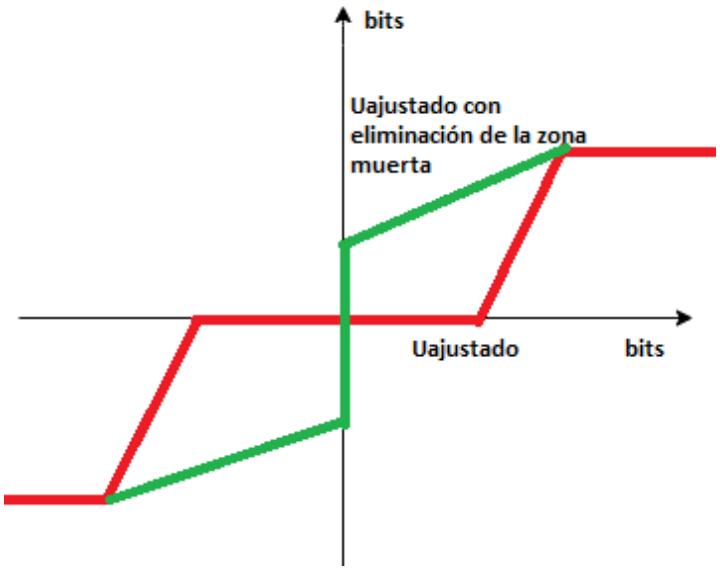

Figura 8. Modelo del motor incluyendo la zona muerta y la corrección de la dicha.

Dado que la zona muerta en este sistema abarca aproximadamente un 20% del rango de tensiones con la que se alimenta el motor, se ha visto necesario introducir a través del programa Arduino **(Anexo 1)**, un *offset* y un reajuste de la recta de carga del motor con el objetivo de mejorar la respuesta de los controladores que se diseñarán en el próximo capítulo.

Ha de mencionarse que, tras hacer un análisis de la señal de error a través del monitor serie de Arduino, se ha determinado que este valor no es nulo para la posición de equilibrio de la varilla. Su valor oscila entre valores de $\pm 1\ bit$. Por ello se ha decidido, con el fin de no amplificar dicho error e inestabilizar el sistema, limitar la corrección de la zona muerta.

CAPÍTULO 4: CONTROL

Cuando se saca el sistema de la posición de equilibrio mediante una perturbación externa, se quiere que el sistema vuelva a dicha posición. Para que esto sea posible, es necesario introducir un sistema de control que sea capaz de mandar una señal de voltaje al motor para que corrija dicha desviación de la manera más rápida y con la mejor dinámica posible. Utilizando las técnicas de control tradicionales, se pretende conseguir dicho objetivo.

Este capítulo trata las diferentes técnicas de diseño de controladores, estudiadas en la asignatura de Control Automático I y II del grado de Ingeniería Electrónica. Por un lado, el diseño de controladores en el dominio temporal utilizando la herramienta *Lugar de las Raíces*. Por otro lado, el diseño de redes de compensación de fase en el dominio de la frecuencia, utilizando el diagrama de Bode como principal herramienta.

Recordando el diagrama de bloques del sistema:

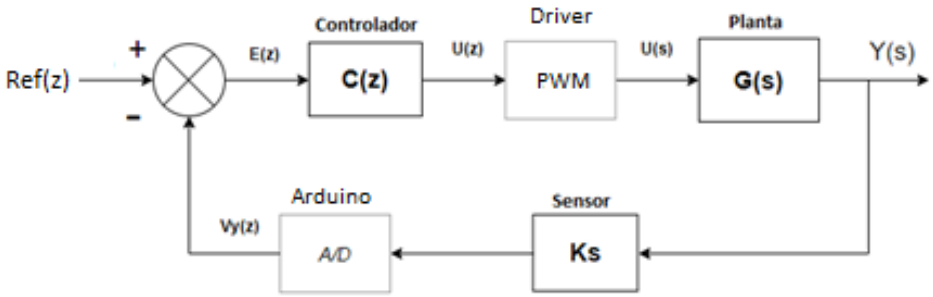

Figura 9. Diagrama de Bloques del sistema.

Como se puede ver en la Figura 9, el controlador es discreto, ese hecho obliga a utilizar técnicas discretas para su diseño. Debido a su sencillez, se ha decidido utilizar las técnicas de diseño de controladores en el dominio *s*, y una vez diseñado, se discretiza al dominio *z* mediante la transformada de Euler.

4.1. DISEÑO DE CONTROLES A TRAVÉS DEL LUGAR DE LAS RAÍCES.

La dinámica de un sistema de control en lazo cerrado, está ligada con la situación de los polos de su función de transferencia en lazo cerrado. Este hecho proporciona un método tanto de análisis como de diseño de controladores.

La idea fundamental, trata de estudiar la influencia de alguno de los parámetros del controlador en la dinámica del sistema en lazo cerrado y ajustar el valor de dichos parámetros para obtener el comportamiento deseado del sistema. Para ello, es útil determinar el cambio de situación en el plano *s* de las raíces de la ecuación característica cuando varía el valor de algún parámetro del sistema de control, dando de esta forma, una medida de la sensibilidad de las raíces con respecto a variaciones en el parámetro que se considere. El parámetro utilizado es la ganancia proporcional que puede variar desde cero hasta infinito.

El *Lugar de las Raíces* se obtiene a partir de la función de transferencia en lazo abierto, que en el dominio *s* queda según la Ecuación (4.1).

$$G_{LA}(s) = G_p(s) \times K_s = \frac{3.29}{s \times (6.31 + s)} \tag{4.1}$$

En la Figura 10 se muestra el *Lugar de las Raíces* del sistema en lazo abierto.

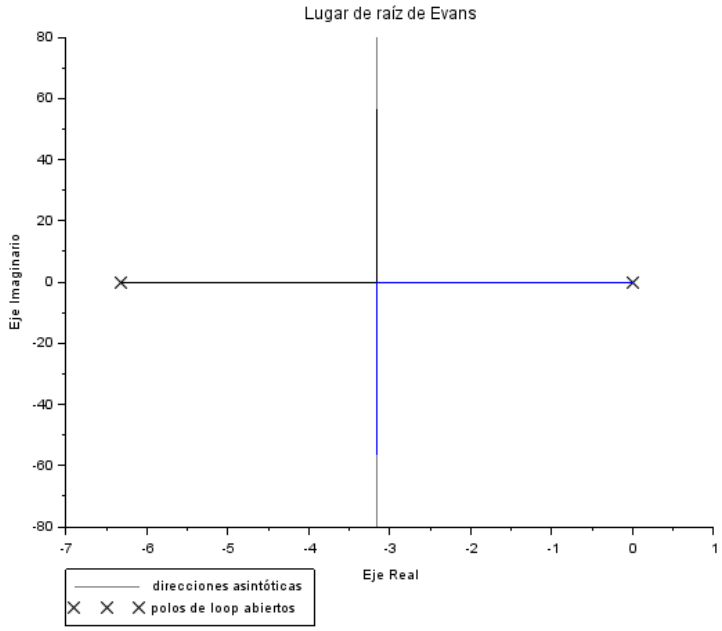

Figura 10. Lugar de las raíces del sistema.

Como se puede comprobar, el hecho de aumentar la ganancia proporcional del controlador, desplaza los polos del sistema por las líneas de su *Lugar de las Raíces*, dando como resultado diferentes dinámicas en función de su valor. El sistema puede ser subamortiguado para $K > 3$, criticamente amortiguado para $K = 3$ y sobreamortiguado para $K < 3$.

4.1.1. CONTROL PROPORCIONAL DERIVATIVO

La idea fundamental para el diseño de este controlador, es buscar una pareja de polos complejos conjugados, que cumpla las especificaciones temporales que se desean para la dinámica del sistema, es decir, aquellos que permitan que cuando se saque el sistema del equilibrio, vuelva al mismo punto de una manera rápida y con el menor rebose posible. Se quiere, entonces, que esta pareja de polos sean los polos dominantes de nuestro sistema en lazo cerrado y la dinámica dependa únicamente de ellos. Para ello, se introduce al sistema un control PD (Ecuación (4.2)).

$$PD(s) = K_d(1 + T_d s) \tag{4.2}$$

Mediante el diseño de sus parámetros se consigue que esta pareja de polos pertenezca al sistema y sean los que aporten la mayor parte de la dinámica. Las condiciones a cumplir son:

- La condición del argumento, a partir de la cual se obtiene el valor del parámetro T_d. Se presentan a continuación las ecuaciones a resolver para obtenerlo.

$$Arg\big(FTLA(s_1)\big) = Arg\big(PD(s_1)\big) + Arg(G_p(s_1)K_s) = (2q + 1)\pi \tag{4.3}$$

$$Arg\big(PD(s_1)\big) = Arg(1 + T_d s_1) = \varphi_1 \tag{4.4}$$

  Se tiene que conseguir un valor de T_d, tal que la suma de los argumentos del controlador y de la función de transferencia en lazo abierto del sistema, sea un número impar de veces π.

- El otro parámetro es K_d, la ganancia que hay que añadir para que los polos pertenezcan al sistema. Se obtiene mediante la condición del módulo, una vez se haya obtenido el parámetro T_d.

$$| PD(s_1)G_p(s_1)K_s | = 1 \tag{4.5}$$

Los parámetros temporales que se han elegido para la obtención de la pareja de polos se muestran en la Tabla 5.

Rebose (%)	**Tiempo de pico (T_1) (s)**	**Tiempo de establecimiento (T_p) (s)**
5	0.39	0.5

Tabla 5. Parámetros deseados para el comportamiento del sistema en lazo cerrado.

La pareja de polos complejos conjugados para estos valores es $s_{1,2} = -8 \pm 6i$. El controlador PD que se obtiene con estos parámetros, queda de la siguiente forma.

$$PD(s) = 30\left(1 + \frac{0.09\ s}{\left(1 + \frac{0.09}{10}\ s\right)}\right) \tag{4.6}$$

Se ha introducido un filtro pasa bajo, de tal forma que no modifique la dinámica del sistema, y que haga al controlador causal. También es capaz de atenuar el ruido de alta frecuencia.

Ha de recordarse que debido a que el controlador se implementa en la placa Arduino Uno, se debe discretizar. Para ello, se ha hecho uso de la transformada de Euler: $s = \frac{1}{T}\frac{z-1}{z}$, en todos los controladores del proyecto.

Para comprobar el funcionamiento del sistema con este control, se le aplica una entrada escalón de $18.88°$ y se ve cómo evoluciona la dinámica del sistema a lo largo del tiempo utilizando como herramienta el programa diseñado el Labview.

En la Tabla 6, se recogen los parámetros del controlador diseñado. Junto a ellos, con el fin de observar la dinámica que aporta este tipo de control, se añaden el resultado para distintos parámetros de los controladores también diseñados.

Color de la gráfica	T_d	k_d
Verde	0.09	30
Azul	0.03	20
Rojo	0.8	10

Tabla 6. Controles PD diseñados.

En la Figura 11 se muestra el comportamiento del sistema con los diferentes controles de la Tabla 6.

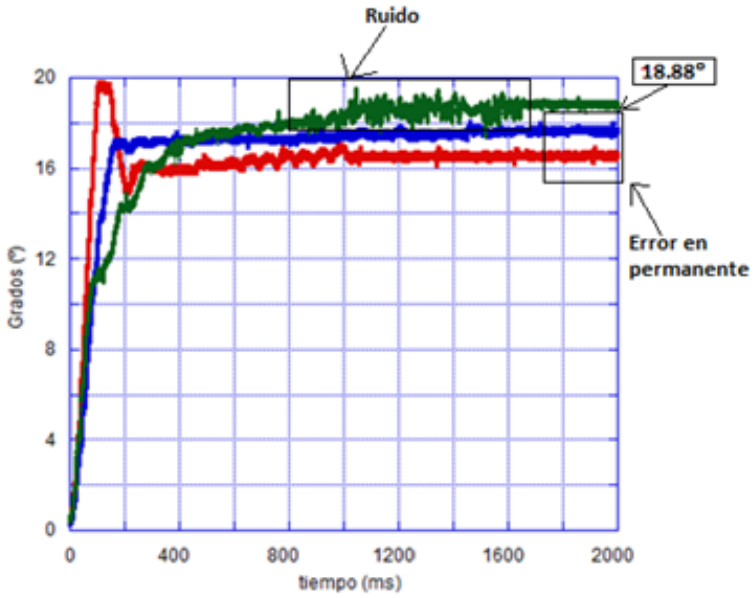

Figura 11. Gráficas de los diferentes controladores PD implementados.

Lo primero que ha de mencionarse, es que los parámetros obtenidos a partir de los polos deseados en lazo cerrado, no consiguen la dinámica de la Tabla 4. Esto implica que el modelo matemático obtenido del motor, no es el más adecuado.

Analizando las posibles mejoras, se tendría que haber evaluado la zona muerta del motor y haberla corregido antes de obtener su modelo. También se podría haber caracterizado el rozamiento, el par producido por el eje y el peso de la varilla. A pesar de ello se pasa a analizar el comportamiento del sistema, con este tipo de controlador.

Como se puede observar, el control derivativo aporta rapidez y amortiguamiento. Las gráficas muestran que cuanto más grande es la acción derivativa más rápido es el sistema, pero en contra partida, el rebose se hace más grande.

Por otro lado, en estado estacionario, se puede observar que el menor error se consigue con el controlador de la gráfica verde. Esto es gracias a su mayor ganancia, que es capaz de superar mejor la zona muerta. Sin embargo, tal como predice la teoría, a costa de reducir el error en estado estacionario, el ruido que tiene el propio sistema es amplificado.

En conclusión, el control proporcional derivativo consigue una dinámica rápida del sistema, a costa de un mayor rebose. Las ganancias bajas no son capaces de superar la zona muerta para señales de error pequeñas, por lo que existe un error en estado estacionario. Al intentar corregir el error aumentando el valor de la ganancia, el ruido del sistema se amplifica.

4.1.2. CONTROL PROPORCIONAL DERIVATIVO INTEGRAL

Con el objetivo de eliminar el error sin tener que aumentar mucho la ganancia, se introduce al sistema un control integral, formando así un controlador PID. La función de transferencia que se añade a la del controlador PD ya usado es:

$$PI(s) = \left(1 + \frac{1}{T_i\, s}\right) \tag{4.7}$$

La condición que hay que tener en cuenta a la hora de introducirlo, es que la dinámica del sistema conseguida con el control PD se vea afectada lo menos posible. Para ello T_i tiene que cumplir la siguiente condición.

$$T_i = \frac{-10}{Re(s_1)} = 1 \tag{4.8}$$

La ganancia del controlador PID, K_{PID}, se obtiene aplicando la misma ecuación que la que se ha aplicado en el caso del control PD, con la condición del módulo.

$$|\, PD(s_1)PI(s_1)G_p(s_1)K_s| = 1 \tag{4.9}$$

El controlador PID que se obtiene queda de la siguiente manera:

$$PID(s) = K_{PID}(1 + T_d s)\left(1 + \frac{1}{T_i s}\right) = \frac{32.390 \;+\; 43.627s \;+\; 3.923s^2}{1.25s} \tag{4.10}$$

Este tipo de controlador PID diseñado, se conoce como interactivo o serie. Los cambios en los parámetros T_i y T_d influyen en todas las acciones de control.

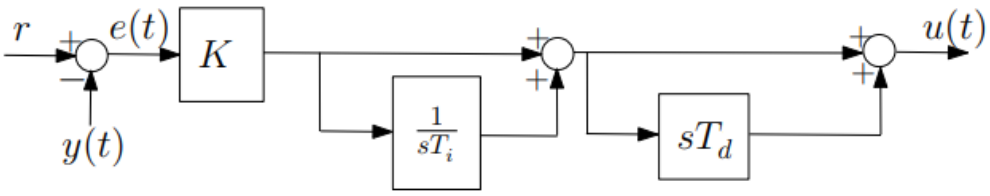

Figura 12. Diagrama de bloques de un PID interactivo.

El modelo de controlador que más se utiliza en la actualidad, se conoce como "PID no interactivo".

$$PID(s)_{no\ interactivo} = K'_{PID}\left(1 + \frac{1}{T'_i s} + T'_d s\right) \tag{4.11}$$

En este caso, los cambios en T'_i y T'_d no influyen en las otras acciones de control.

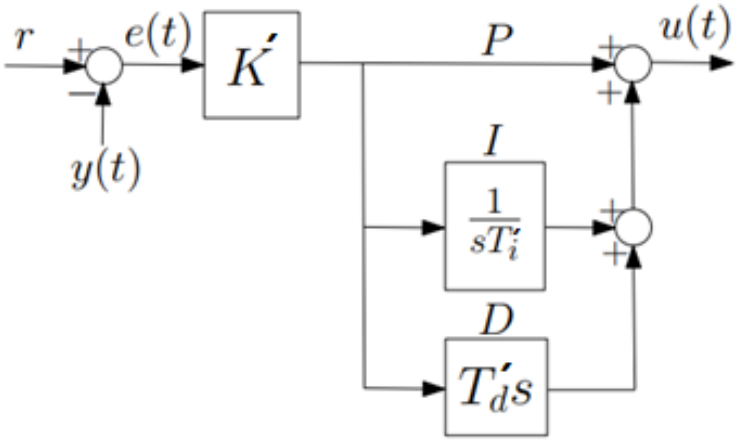

Figure 9. Control PID no interactivo.

Este modelo además permite introducir mejoras al controlador, como puede ser un sistema *antiwindup*. Por ello, se ha decidido convertir el PID interactivo diseñado en su modelo no interactivo.

Ha de mencionarse, que a la parte derivativa se le introduce un filtro para disminuir el ruido de alta frecuencia y hacer el sistema causal.

$$PID(s)_{no\ interactivo} = K'_{PID}\left(1 + \frac{1}{T'_i s} + \frac{T'_d s}{1 + \frac{T'_d}{10} s}\right) \tag{4.12}$$

Igualando las Ecuaciones (4.12) y (4.10), se pasa del control interactivo a su modelo no interactivo.

$$K'_{PID} = K_{PID}\frac{T_i + T_d}{T_i} = 1.15 \tag{4.13}$$

$$T'_i = T_i + T_d = 1.0834 \tag{4.14}$$

$$T'_d = \frac{T_i T_d}{T_i + T_d} = 0.0769 \tag{4.15}$$

Al igual que se ha hecho para los controladores PD, en la Tabla 7 se muestra el control diseñado a partir de las especificaciones descritas a principio de capítulo de la Tabla 5, junto con otros controladores, a fin de dar a conocer la dinámica que se consigue con este tipo de controlador.

Color de la gráfica	K'_{PID}	T'_i	T'_d
Rojo	1.15	1.0834	0.0769
Verde	15	0.5	0.03
Azul	15	1	0.03

Tabla 7. Parámetros PID.

Una vez obtenidos los parámetros, se aplica una entrada escalón al sistema de $18.88°$. Los resultados se muestran en la Figura 10 y 11. La primera gráfica corresponde al comportamiento del sistema con el control teórico diseñado, y la segunda con los parámetros escogidos.

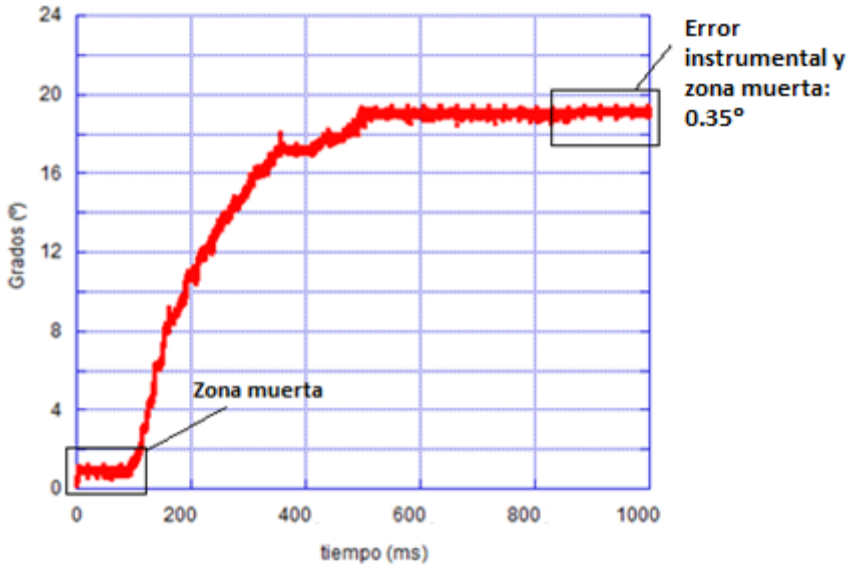

Figura 10. Respuesta del sistema con el control PID diseñado a partir de la teoría.

Se puede observar, que debido a que se tiene una ganancia proporcional muy pequeña, no se consigue superar la zona muerta en los primeros instantes, por lo cual el sistema no actúa. Sin embargo, tras unas milésimas de segundo, el error acumulado en la parte integral comienza a corregir el error. Cuanto mayor sea esta acción más rapido se corregirá dicho error.

El sistema tiene una dinámica muy lenta, pero el error en estado estacionario es prácticamente nulo, teniendo el cuenta el error instrumental del propio potenciómetro y la zona muerta que, como se ha mencionado antes, no se ha corregido completemente para evitar problemas de inestabilidad.

Con el objetivo de mejorar la respuesta se aumenta la ganancia proporcional a $K'_{PID} = 15$, se aumenta también el valor del parametro T_d y se juega con los valores del parámetro T_i. La

siguiente gráfica muestra el comportamiento del sistema con dichos controladores ante una entrada escalón de 18.88°.

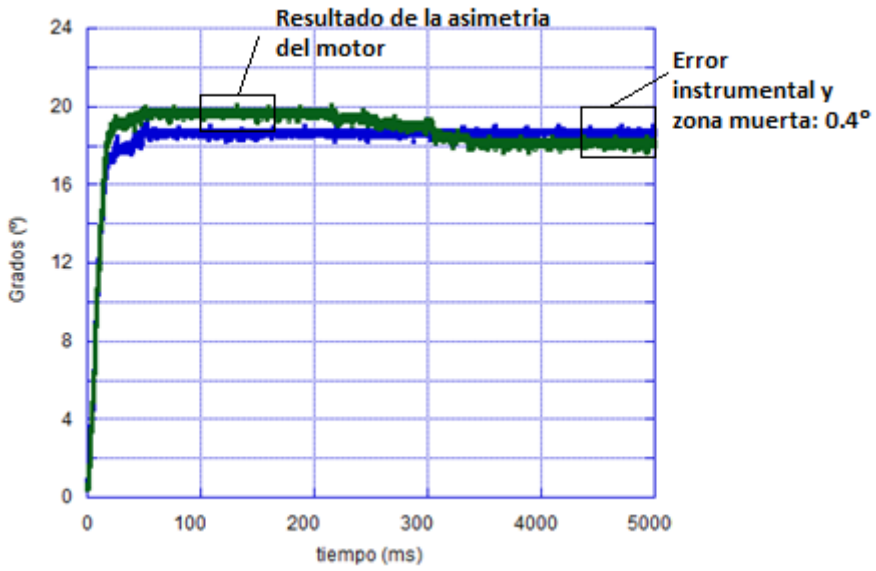

Figure 11. Respuesta del sistema para dos controles ante entrada escalón.

Como se puede apreciar, gracias a un control proporcional más grande, se supera la zona muerta del principio. Unido esto, con los controles derivativo e integral, se consigue una dinámica más rápida y más adecuada.

Conviene mencionar que, debido a una posible mala alineación o problemas propios del motor, existe una actuación asimetrica del sistema, es decir, la dinámica de un lado es más rápida que del otro, como se puede apreciar en el transitorio de la gráfica verde. Se observa también que sigue existiendo un error en estado estacionario debido al error instrumental y la zona muerta.

En resumen, se puede concluir que al introducir un control integral, eliminamos en gran medida el error en estado estacionario y no se amplifica el ruido, ya que las ganancias que se utilizan no son muy grandes. Como inconveniente, la dinámica del sistema se hace más lenta que en el caso de un control proporcional derivativo.

4.1.3. ANTIWINDUP

Todos los actuadores tienen un rango de funcionamiento limitado. Si la señal de control que entra al actuador supera sus límites, se dice que el actuador está saturado y en saturación, el sistema de control deja de funcionar en lazo cerrado.

Si el controlador tiene parte integral, el error, que no se corrige, se va integrando y el término integral se hace cada vez más grande, produciendo una acumulación de energía (windup).

La consecuencia es que el control del sistema no funciona y se producen grandes transitorios si el actuador se satura ya que, aunque el error instantáneo llegue a ser nulo, la señal de control acumulada puede ser grande y el controlador sigue inyectando señal.

En el caso de este sistema el actuador es el motor, y como se ha podido conocer en el capítulo 2, satura para señales de control de $\pm 400\ bit$, correspondientes a un voltaje de alimentación del motor de $\pm 15\ V$.

Con el fin de evitar este fenómeno, se ha diseñado un sistema *antiwindup*. El método utilizado consiste en hacer un seguimiento de la señal integral y corregirla si crece por encima del valor de saturación del actuador. Es un método basado en la realimentación de una señal de error, la diferencia entre la salida real del actuador y la señal de control que debería dar teóricamente. A continuación, se muestra cómo es gráficamente este sistema.

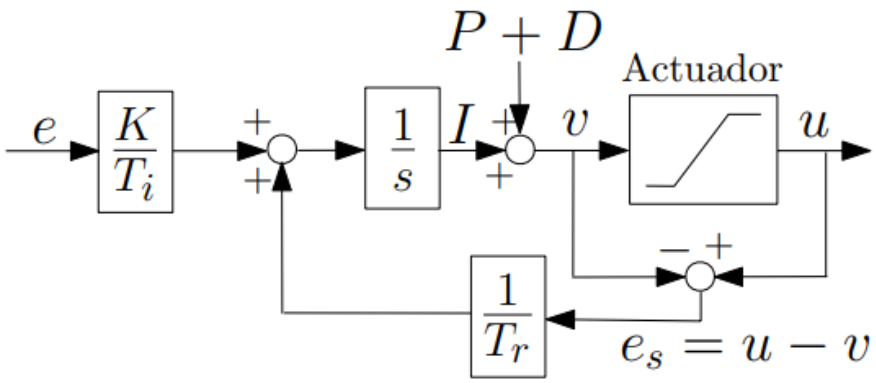

Figura 2. Sistema "Antiwindup".

La acción *antiwindup* puede controlarse a través de la ganancia ($\frac{1}{T_r}$). El parámetro T_r debería ser pequeño para que la corrección de *windup* sea enérgica **(Anexo1).**

4.2. DISEÑO EN EL DOMINIO DE LA FRECUENCIA

Otra forma de diseñar controladores está basada en el estudio del sistema en el dominio de la frecuencia, utilizando como herramienta principal el diagrama de Bode. En este capítulo se diseñan los controles conocidos como redes de compensación: la red de adelanto y la red de atraso.

De la misma manera que en el capítulo anterior, el diseño de las redes se hará en el dominio *s* y posteriormente se discretizará utilizando la transformada de Euler. En el dominio de la

frecuencia también se trabaja con la función de transferencia del sistema en lazo abierto Ecuación (4.1).

$$G_{LA}(s) = G_p(s) \times K_s = \frac{3.29}{s \times (6.31 + s)} \tag{4.1}$$

4.2.1. GANANCIA PROPORCIONAL

Antes de diseñar la red de compensación, se debe escoger una ganancia proporcional *K* para el controlador, cuya elección está basada en un error en estado estacionario ess, lo más pequeño posible.

Dado que el sistema tiene un integrador en la cadena directa, como se ve en el capítulo 3, el error elegido será ante entrada rampa, es decir, un error estacionario en la velocidad.

La ecuación que relaciona la señal de error con la ganancia proporcional es la siguiente:

$$E(s) = \frac{1}{1 + KG_p(s)K_s} Ref(s) \tag{4.16}$$

Para una entrada rampa: $R(s) = {}^{r}/_{s^2}$, donde r es el valor de la referencia y aplicando el teorema del valor final a la ecuación (4.16), se obtiene la ecuación que relaciona la ganancia proporcional con el error estacionario:

$$K = \frac{\frac{1}{{}^{ess}/_{r}}}{\lim_{s\to\infty} sG(s)K_s} \tag{4.17}$$

Se elige un error en la velocidad en estado estacionario del 2%, esto implica que $100\frac{e_{ss}}{R} = 2$. Sustituyéndolo en la ecuación anterior, se obtiene el siguiente valor de la ganancia proporcional:

$$K = 50 \tag{4.18}$$

4.2.2. DIAGRAMA DE BODE

Como se ha mencionado al principio del capítulo, para diseñar estas redes es necesario realizar un análisis del diagrama de Bode del sistema. El análisis del diagrama consiste en conocer dos de sus parámetros: el margen de ganancia y el margen de fase.

El margen de ganancia (GM), se usa para medir la estabilidad relativa del sistema de control. Se define como la cantidad de ganancia en decibelios (dB) que se puede añadir al sistema en lazo abierto antes de que el sistema en lazo cerrado se vuelva inestable. Por otro lado, el margen de fase (PM), se define como el retardo que puede añadirse al sistema en lazo abierto antes de que el sistema en lazo cerrado sea inestable.

El diagrama de Bode que se analiza, es el de la función de transferencia en lazo abierto compensada en ganancia, es decir, introduciendo la ganancia obtenida en el anterior apartado. El diagrama de Bode para nuestro sistema se muestra a continuación.

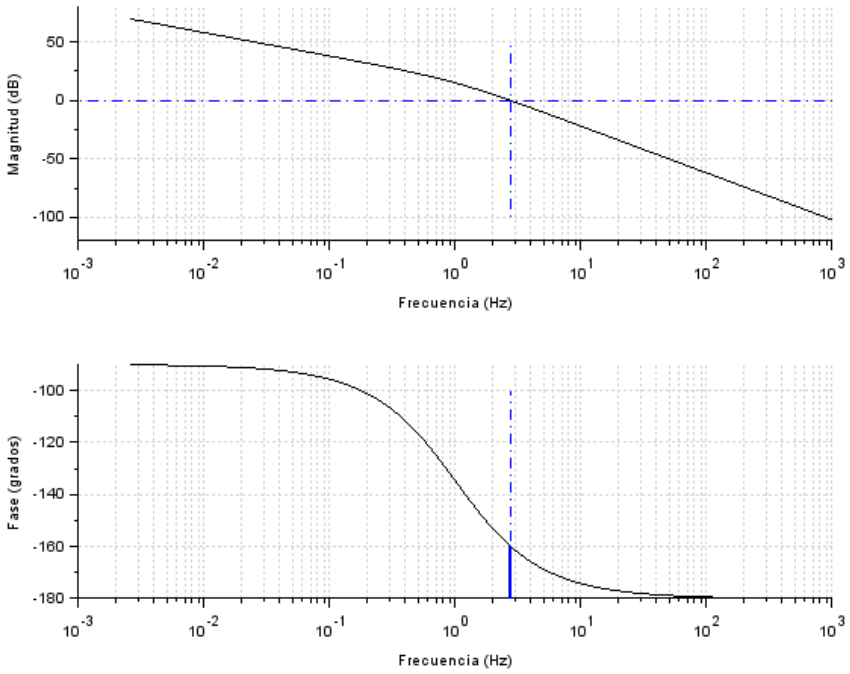

Figura 3. Diagrama de Bode de la función de transferencia en lazo abierto. Se representa tanto la amplitud como la fase.

Como se puede observar en la Figura 12 el margen de fase para todas las frecuencias es positivo, lo que demuestra, como ya se conocía anteriormente, que el sistema en lazo cerrado con un control proporcional es estable. Por otro lado, se tiene que el margen de ganancia es infinito, lo que implica teóricamente que el valor de la ganancia se puede incrementar al infinito sin que ocurra la inestabilidad.

El margen de fase es de 20.37°, es decir, se le puede añadir dicha fase al sistema sin inestabilizarlo.

4.2.3. RED DE COMPENSACIÓN: RED DE ADELANTO

Para realizar el diseño, es necesario tener unas especificaciones sobre rapidez y rebose en el dominio de la frecuencia. Las especificaciones relacionadas son el ancho de banda y el máximo de resonancia. En lazo abierto, se traducen en especificaciones sobre la frecuencia de ganancia crítica y el margen de fase.

Para un sistema en lazo cerrado de segundo orden como el que se tiene, el margen de fase correspondiente en lazo abierto, depende únicamente del coeficiente de amortiguamiento. Por otro lado, el rebose del sistema y el máximo de resonancia solo depende del valor del coeficiente de amortiguamiento, por tanto, existe una relación directa entre el margen de fase y el rebose del sistema ante entrada escalón.

La especificación deseada para el rebose es del 5%, lo que lleva a un margen de fase que se obtiene mediante la siguiente aproximación.

$$PM \approx 100\delta = 100 \times 0.69 = 69° \tag{4.19}$$

El valor de δ se ha obtenido aplicando la Ecuación (3.9), del capítulo 3.

El procedimiento, es utilizar la compensación serie, para introducir un avance de fase alrededor de la frecuencia de ganancia crítica del sistema sin compensar, con el fin de lograr el PM deseado. El procedimiento de diseño tiene como objetivo colocar la red de forma que el máximo avance de fase, se produzca en la frecuencia de ganancia crítica del sistema compensado. Dado que la red introduce una cierta ganancia, la frecuencia de ganancia crítica, se desplazará ligeramente a la derecha del sistema sin compensar.

A continuación, se muestra cómo es la red de adelanto.

$$AD(s) = \frac{1 + aTs}{1 + Ts};\ con\ a > 1 \tag{4.20}$$

El diagrama asintótico de Bode de la red de adelanto es:

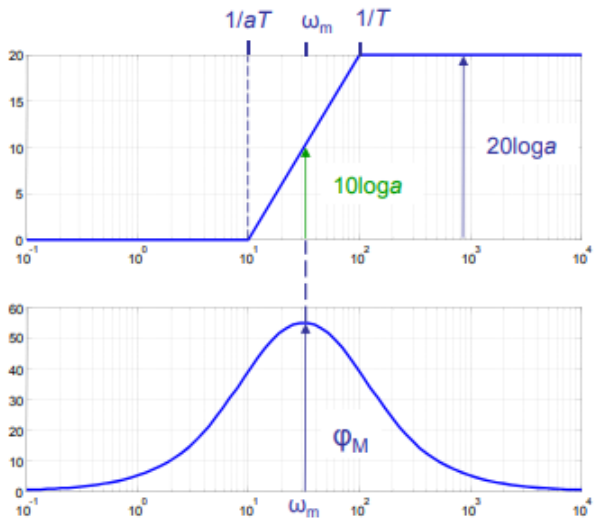

Figura 4. Diagrama de Bode correspondiente a la red de adelanto.

Para obtener el parámetro a, primero se obtiene la fase que se quiere introducir con la red: $\varphi = Fase_{Deseada} - PM + 8° = 69° - 20{,}37° + 8° = 56{,}62°$, y posteriormente aplicamos la siguiente ecuación.

$$a = \frac{1 + sin(^{\varphi\pi}/_{180})}{1 - sin(^{\varphi\pi}/_{180})} = 11{,}128 \tag{4.21}$$

El parámetro T, se obtiene a través de la ecuación.

$$T = \frac{1}{f_m\, 2\pi\, \sqrt{a}} = \frac{1}{11.2 \times 2\pi\sqrt{a}} = 0{,}004 \tag{4.22}$$

Donde la frecuencia f_m es la frecuencia correspondiente y donde la ganancia del diagrama de Bode vale $10\ log\ a = 24{,}094\ dB.$

Se muestra a continuación, el controlador con la ganancia proporcional incluida.

$$AD(s) = \frac{95.644\ +\ 4.533s}{1\ +\ 0.004s} \tag{4.23}$$

Al igual que se ha hecho para los controles diseñados en el *Lugar de las Raíces*, se discretizan los controles utilizando la transformada de Euler, para poder ser implementados en Arduino **(Anexo 2)**.

Como se viene haciendo a lo largo del capítulo, se introduce una entrada escalón del mismo valor y se analiza cómo evoluciona la dinámica.

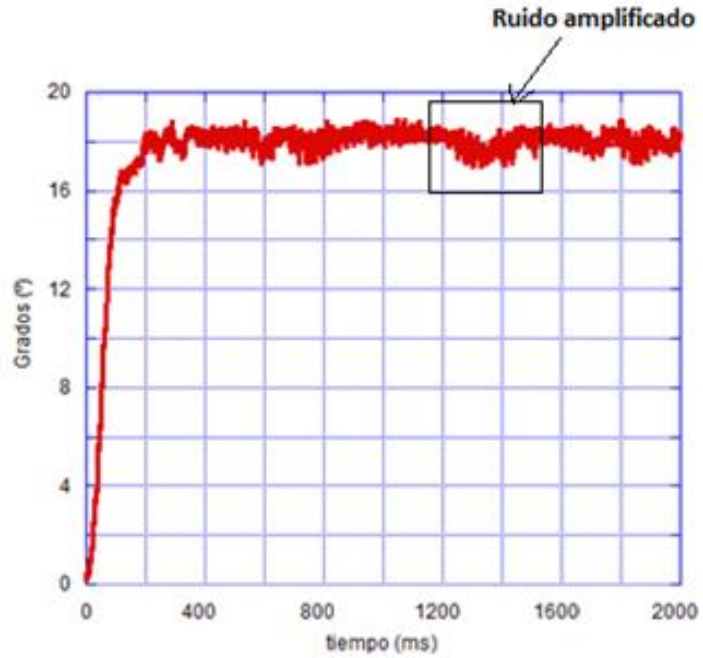

Figura 5.Respuesta del sistema con la red de adelanto como controlador ante una entrada escalón.

Como se puede apreciar en la gráfica, la red de adelanto hace que el sistema actúe de una manera muy rápida, a costa de ello, amplifica mucho el ruido que tiene el propio sistema. Por lo tanto, no es conveniente utilizar este controlador.

4.2.3. RED DE COMPENSACIÓN: RED DE ATRASO

La filosofía del diseño, es utilizar la atenuación que introduce la red, para disminuir la frecuencia de ganancia crítica hacia valores con mayor margen de fase. La red se debe situar de forma que el retardo de fase que introduce, se produzca lejos de las frecuencias de interés, donde se desea situar la nueva frecuencia de ganancia crítica. Se muestra a continuación, cómo es la red de adelanto:

$$AT(s) = \frac{1 + aTs}{1 + Ts};\ con\ a < 1 \tag{4.24}$$

El diagrama asintótico de Bode correspondiente se muestra a continuación.

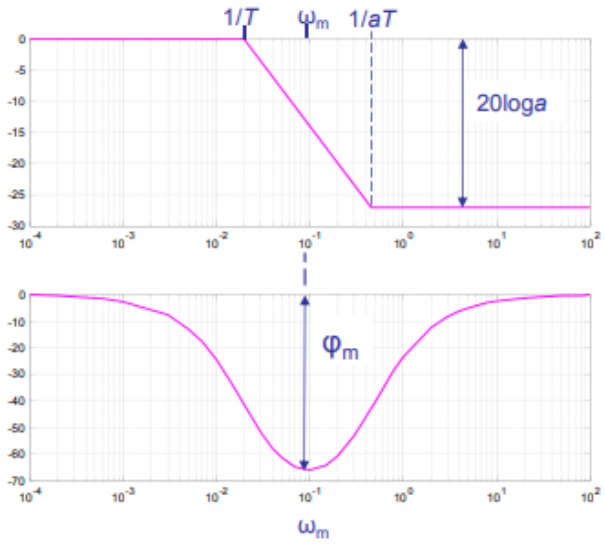

Figura 6. Diagrama de Bode de la red de atraso, en ella se muestra el diagrama de amplitud y de fase.

Para obtener los valores de la red, en primer lugar, se comprueba si se satisfacen las especificaciones sobre el PM. Como se ha podido comprobar, estas no se cumplen con el sistema compensado en ganancia. Se comprueba entonces si existe algún valor de frecuencia, para el que la fase toma dicho valor deseado.

La fase que se busca es: $180° - F - 8° = 69° \rightarrow F = -103°$, la frecuencia correspondiente a este valor es: $f_{gc'} = 0{,}388\ Hz$.

De la curva del módulo se obtiene la atenuación que debe proporcionar la red para que $f_{gc'}$ sea la nueva frecuencia de ganancia crítica. De ahí se obtiene el valor de a. La atenuación que se tiene que introducir es de 25,62 dB, por lo tanto, el valor de a es:

$$20\ log\ a = -30{,}3; \quad a = 0.052 \tag{4.25}$$

El parámetro *T* se elige de tal forma que el retardo de fase no afecte:

$$T = \frac{10}{f_{gc'} 2\pi a} = \frac{10}{0{,}388 \times 2\pi b} = 78.883 \tag{4.26}$$

Al probar estos valores con el control discretizado **(Anexo 2)**, no se consiguió que el sistema actuara correctamente, por ello se decidió aumentar la ganancia proporcional hasta un valor de 150 y así, aumentar el MF deseado para conseguir una dinámica más rápida del sistema. En la siguiente tabla se muestran los diferentes parámetros calculados.

Color de la gráfica	*K*	*MP*	$f_{gc'}$	*a*	*T*
Rojo	150	-125°	0.707	0.11	20.46
Azul	150	-135°	1.0	0.17	9.36
Verde	150	-140°	1.17	0.22	6.18

Tabla 8. Parámetros de las diferentes redes de atraso diseñadas, junto al color correspondiente de su gráfica

En las siguientes figuras se muestra la respuesta del sistema ante una entrada escalón de 18.88°.

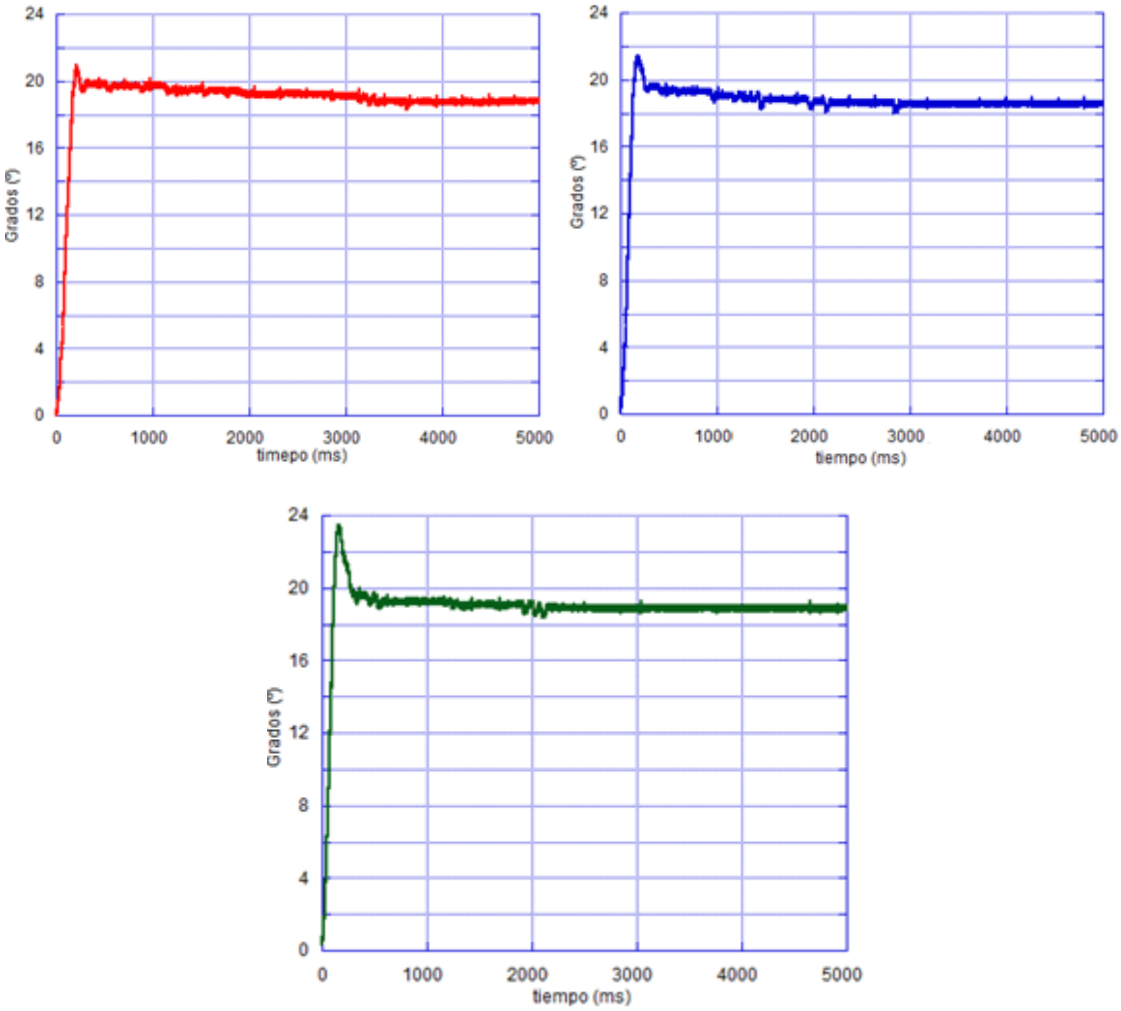

Figura 12.Respuesta del sistema ante entrada escalón de las diferentes redes diseñadas.

Como se aprecia en las gráficas, al contrario que en la red de adelanto, la red de atraso no amplifica el ruido del sistema, una cualidad fundamental de este tipo de redes.

Por otro lado, la rapidez de la dinámica del sistema aumenta conforme se reduce el margen de fase, en contrapartida, se aumenta el rebose. El error en estado estacionario, es prácticamente nulo, teniendo siempre en cuenta el error instrumental y la zona muerta.

En conclusión, todos los controles diseñados en este capítulo siguen la teoría de control automático [1]. El hecho de tener un control puramente derivativo no elimina el error en estado estacionario. Al introducir un integrador a este sistema, se reduce dicho error, pero el sistema se hace más lento. Por otro lado, en el dominio de la frecuencia se puede ver cómo la red de adelanto amplifica mucho el ruido. La red de atraso tiene una dinámica más lenta pero no amplifica el ruido, se podría decir que tiene una gran similitud al comportamiento del PID.

CAPÍTULO 5: CONCLUSIONES

Se ha conseguido realizar una maqueta funcional que soporta todo el sistema y se ha obtenido el modelo matemático del mismo, tanto en lazo abierto como en lazo cerrado. A partir de los diferentes problemas que han ido surgiendo, se han ideado métodos para solucionarlos.

A la hora de obtener el modelo matemático, se ha tenido en cuenta que los rozamientos cambian en función de la maqueta, por los diferentes anclajes y alineaciones de los ejes, por lo que se decidió obtener el modelo matemático con la maqueta final ya montada.

Se ha caracterizado y corregido en la medida de lo posible la zona muerta, con el objetivo de conseguir los mejores resultados para la dinámica de los controladores.

Se ha diseñado un programa de adquisición de datos, con el objetivo de obtener el modelo matemático del sistema y caracterizar los diferentes controles implementados.

Se ha creado un sistema funcional, para el control de la posición angular de la varilla y se han implementado los diferentes controles que se han estudiado a lo largo de las asignaturas de Control Automático, permitiendo comprobar que lo estudiado en la teoría se cumple en los sistemas reales, teniendo en cuenta la no idealidad del sistema.

Como posibles mejoras del proyecto, se propone la obtención de un modelo matemático más preciso, teniendo en cuenta los diferentes efectos no lineales que han ido apareciendo como son: los rozamientos, la zona muerta y otros efectos, como el peso de la varilla o la mala alineación de los ejes. También se podría utilizar otra plataforma que tenga mejores características que el Arduino Uno, en cuanto a precisión y error que el propio sistema introduce.

Se propone realizar un proceso de selección adecuado del actuador, en base a la zona muerta y la saturación del motor, y el par que es capaz de producir para mover la varilla con el objetivo de conseguir un sistema de control mucho más sofisticado y preciso. Se ha considerado que éste es uno de los puntos con más margen de mejora.

La resolución del sensor para este proyecto, se ha considerado suficiente, pero para un sistema más sofisticado, es un posible punto a mejorar ya que se podría reducir aún más el error en el estado estacionario.

Por último, cabe mencionar la posible mejora de la adquisición de datos, mediante la automatización del proceso con el objetivo de reducir el tiempo que se ha dedicado para tomarlos.

REFERENCIAS BIBLIOGRÁFICAS

[1] K. J. (Karl J. Åström, T. Hägglund, and K. J. (Karl J. Åström, *PID controllers*. International Society for Measurement and Control, 1995.

[2] K. Sailan and K. Kuhnert, "DC motor angular position control using PID controller for the purpose of controlling the hydraulic pump," *Proc. Int. Conf. Control. Eng. …*, vol. 1, no. May, pp. 22–26, 2013.

[3] M. Banzi and M. Shiloh, *Introducción a Arduino : edición 2016*. Anaya Multimedia, 2015.

[4] A. V. Oppenheim, A. S. Willsky, S. Hamid Nawab, G. Mata Hernández, and A. Suárez Fernández, *Señales y sistemas*. Prentice Hall, 1998.

[5] V. K. Verma, C. R. Baird, and V. K. Aatre, "Pulse-width-modulated speed control of d.c. motors," *J. Franklin Inst.*, vol. 297, no. 2, pp. 89–101, Feb. 1974.

[6] S. Kivrak, T. Özer, and Y. Oguz, "H Bridge DC Motor Driver Design and Implementation with Using dsPIC30f4011," *Int. J. Innov. Res. Sci. Eng. Technol.*, vol. 6, no. May, 2017.

[7] M. Leon, "Obtención de la Función de Transferencia de Sistemas mediante la Identificación Paramétrica a partir de datos Experimentales." .

[8] Q. Wei-dong and L. Bin-xia, "Operating characteristics analysis and deadband compensation for electric power system motor," *2009 ISECS Int. Colloq. Comput. Commun. Control. Manag.*

ANEXO 1: Programa Arduino con los controles diseñados en el Lugar de las Raíces.

```
#include "DualMC33926MotorShield.h"

DualMC33926MotorShield md;  // Creamos un objeto de este tipo(para poder uso de la libreria del driver

//Tiempo de muestreo
float Tm=13; //en milisegundos
float T=0.013; //ensegundos
unsigned long Tpasado=0;
unsigned long Tahora;
unsigned long Tcambio;

//Sistema de control
const int Potenciometro=A0;  //realimentacion
float sensorValor;
float Re;        //referencia
float Ref;       //referencia mas entrada escalón
float error;     //señal de error
float U;         //señal de control
float UAjustado; //señal de control ajustada
float UA;        //señal de control sin saturar para el sistema antiwindup
float errorA1=0; //señal de error anterior
float UA2=0;     //señal de control anterior del control integral
float UA3=0;     //señal de control anterior del control derivativo

//Parametros del pid
float kp=15;
float Ti=0.5;
float Td=0.3;
float U1=0;
float U2=0;
float U3=0;
float n=10.0; //parámetro del filtro del control derivativo

//Antiwindup
float es=0;
float Tr=0.1;
float es2=0;



void setup() {
  //Trigger para la adquisición de datos
  pinMode(3,OUTPUT);
  digitalWrite(3,HIGH);
  //Configuración del driver
  md.init();
  //Posición de equilibrio
  Re=analogRead(Potenciometro);
  Serial.begin(9600);
  //Enatrada escalón de 18.88 grados
  Ref=Re+69;}
```

```
void loop() {

  Tahora=millis();
  Tcambio=Tahora-Tpasado;
  //Ejecución de lo que hay en su interior cada periodo de muestreo
  if(Tcambio>=Tm){
    sensorValor=analogRead(Potenciometro);

     error=Ref-sensorValor;

     //PID no interactivo
     //parte proporcional
     U1=kp*error;
     //parte integral
     U2=(kp*T/Ti)*error+UA2+(T/Tr)*es2;
     //parte derivativa
     U3=(kp*n*Td*(error-errorA1)+Td*UA3)/(n*T+Td);
     //Suma de todas las señales de control
     U=U1+U2+U3;

     //conversion para el driver
     UAjustado=(2000.0*U/15345.0);

     //Reajuste de la pendiente para evitar la zona muerta
     if(UAjustado>20){
      UAjustado=(295.0/400.0)*UAjustado+115;}
     if(UAjustado<-20){
      UAjustado=(295.0/400.0)*UAjustado-115;}

      //Señal de control sin saturar
      UA=UAjustado;

     //Saturación
     if(UAjustado>400){
      UAjustado=400;}
     if(UAjustado<-400){
      UAjustado=-400;}

     //antiwindup
     es=UAjustado-UA;

     //Desnormalización para el antiwindup
     es2=(15345.0*es/2000.0);



  //Se establece la velocidad del motor a través del driver
  md.setM1Speed(UAjustado);

  //Actualización de los parametros anteriores
  Tpasado=Tahora;
  errorA1=error;
  UA2=U2;
  UA3=U3;}}
```

ANEXO 2: Programa Arduino con los controladores diseñados en el dominio de la frecuencia.

```
#include "DualMC33926MotorShield.h"

DualMC33926MotorShield md;  // Creamos un objeto de este tipo(para poder uso de la libreria)

//Tiempo de muestreo
float Tm=13; //en milisegundos
float T=0.013; //ensegundos
unsigned long Tpasado=0;
unsigned long Tahora;
unsigned long Tcambio;

//Sistema de control
const int Potenciometro=A0;  //realimentacion
float sensorValor;
float Re;        //referencia
float Ref;       //referencia mas entrada escalón
float error;     //señal de error
float U;         //señal de control
float UAjustado; //señal de control ajustada
float UA;        //señal de control sin saturar para el sistema antiwindup
float errorAl=0; //señal de error anterior
float UAl=0;     //señal de control anterior de la red


//parametros de las redes
float k=150.844314;
float a=0.22;
float Tl=6.18;


void setup() {
  //Trigger para la adquisición de datos
  pinMode(3,OUTPUT);
  digitalWrite(3,HIGH);
  //Configuración del driver
  md.init();
  //Posición de equilibrio
  Re=analogRead(Potenciometro);
  Serial.begin(9600);
  //Enatrada escalón de 18.88 grados
  Ref=Re+69;}
```

```
void loop() {

  Tahora=millis();
  Tcambio=Tahora-Tpasado;
  //Ejecución de lo que hay en su interior cada periodo de muestreo
  if(Tcambio>=Tm){
    sensorValor=analogRead(Potenciometro);

    error=Ref-sensorValor;

    //Red de atraso/adelanto
    U=(k*(T+a*T1)*error-k*(a*T1*errorA1)+T1*UA1)/(T+T1);

    //conversion para el driver
     UAjustado=(2000.0*U/15345.0);

     //Reajuste de la pendiente para evitar la zona muerta
     if(UAjustado>20){
      UAjustado=(295.0/400.0)*UAjustado+115;}
     if(UAjustado<-20){
      UAjustado=(295.0/400.0)*UAjustado-115;}

     //Saturación
     if(UAjustado>400){
      UAjustado=400;
      }
     if(UAjustado<-400){
      UAjustado=-400;
      }
     //Se establece la velocidad del motor a través del driver
     md.setM1Speed(UAjustado);

     //Actualización de los parametros anteriores
     Tpasado=Tahora;
     errorA1=error;
     UA1=U;}}
```