



1.1.

**UNIVERSIDAD POLITÉCNICA SALESIANA  
SEDE GUAYAQUIL**

**FACULTAD DE INGENIERÍA**

**CARRERA:**

**INGENIERÍA ELECTRÓNICA**

**Proyecto técnico previo a la obtención del título de:**

**INGENIERO ELECTRÓNICO**

**Título:**

**“Diseño e implementación de un control PID con ganancias programadas utilizando un sistema embebido Launchpad MPS430 para planta balón y viga”**

**Autor:**

**DAVID MARCELO CARPIO HOLGUIN**

**Tutor:**

**ING. LUIS SILVIO CÓRDOVA RIVADENIERA, Msc.**

**AGOSTO DEL 2018**

**GUAYAQUIL- ECUADOR**

## CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA

Yo, DAVID MARCELO CARPIO HOLGUIN autorizo a la **Universidad Politécnica Salesiana** la publicación total o parcial de este trabajo de titulación y su reproducción sin fines de lucro.

Además, se declara que los conceptos y análisis desarrollados y conclusiones del presente trabajo son de exclusiva responsabilidad del autor.

-----  
David Marcelo Carpio Holguín  
Cédula: 0930071568

## CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR

Yo, **DAVID MARCELO CARPIO HOLGUIN**, con documento de identificación N° **0930071568**, manifestó mi voluntad y cedo a la **UNIVERSIDAD POLITÉCNICA SALESIANA** la titularidad sobre los derechos patrimoniales en virtud de que somos autores del trabajo de grado titulado: **“DISEÑO E IMPLEMENTACIÓN DE UN CONTROL PID CON GANANCIAS PROGRAMADAS UTILIZANDO UN SISTEMA EMBEBIDO LAUNCHPAD MSP430 PARA PLANTA BALÓN Y VIGA”** mismo que ha sido desarrollado para optar por el título de **INGENIERO ELECTRÓNICO**, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos antes cedidos.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en condición de autor me reservo los derechos morales de la obra antes citada. En concordancia, suscrito este documento en el momento que se realiza la entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Politécnica Salesiana.

-----  
David Marcelo Carpio Holguín  
Cédula: 0930071568

## CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, **ING. LUIS SILVIO CÓRDOVA RIVADENEIRA, Msc.** Director del proyecto de titulación denominado: **“DISEÑO E IMPLEMENTACIÓN DE UN CONTROL PID CON GANANCIAS PROGRAMADAS UTILIZANDO UN SISTEMA EMBEBIDO LAUNCHPAD MSP430 PARA PLANTA BALÓN Y VIGA”** realizado por el estudiante **DAVID MARCELO CARPIO HOLGUIN**, certifico que han sido orientado y revisado durante su desarrollo, por cuanto se aprueba la presentación de este ante las autoridades pertinentes.

Guayaquil, agosto del 2018

-----  
**Ing. Luis Silvio Córdova Rivadeneira, Msc.**

## **DEDICATORIA**

Este trabajo de titulación está dedicado al Creador por darme salud y fuerzas para seguir avanzando en la trayectoria de toda mi vida estudiantil.

A mi padre y madre por su comprensión, amor y apoyo incondicional.

A mis docentes y compañeros que fueron de gran ayuda para solucionar problemas que transcurrieron a lo largo de mi vida académica.

A mis colegas del club de robótica que siempre fueron un apoyo para obtener nuevos conocimientos y esforzarme en mis metas.

**David Marcelo Carpio Holguín.**

## **AGRADECIMIENTO**

Agradezco principalmente al Creador y a mis padre y madre por su amor incondicional, enseñanza y disciplina que me inculcaron desde criatura ya que fueron bases sólidas para lograr mis metas.

Agradezco a la Universidad Politécnica Salesiana y a los docentes por todas las enseñanzas y conocimiento que pude adquirir de ellos, siempre estando dispuestos a ayudarme en clases y fuera de ellas.

Agradezco al Ing. Luis Córdova, Msc. Tutor de mi trabajo de titulación que sin su guía y ayuda constante para el desarrollo de este proyecto.

**David Marcelo Carpio Holguín.**

## RESUMEN

AÑO	ALUMNOS	DIRECTOR DE PROYECTO TÉCNICO	TEMA DE PROYECTO TÉCNICO
2018	DAVID MARCELO CARPIO HOLGUIN	ING. LUIS SILVIO CÓRDOVA RIVADENEIRA, Msc.	“DISEÑO E IMPLEMENTACIÓN DE UN CONTROL PID CON GANANCIAS PROGRAMADAS UTILIZANDO UN SISTEMA EMBEBIDO LAUNCHPAD MSP430 PARA PLANTA BALÓN Y VIGA”

El presente proyecto técnico tiene como objetivo la implementación de un módulo didáctico balón y viga, mediante el cual los estudiantes de la carrera se familiarizarán con un control PID con ganancias programadas, realizar prácticas básicas y avanzadas para entendimiento de nuevos controladores NATIONAL INSTRUMENTS LAUNCHPAD. Para esto el módulo cuenta con sensores de distancia y elementos básicos para el aprendizaje.

El objetivo principal de este proyecto es que el estudiante aprenda el uso de nuevos sistemas embebidos mediante el manejo de tareas básicas y avanzadas que ayuden al aprendizaje.

Adicionalmente se da la facilidad de un entorno de programación de los embebidos LAUNCHPAD para el sistema de control a implementar.

Palabras claves: LAUNCHPAD, sistemas embebidos, pid, sistemas de control, controles adaptativos, M-430, controladores.

## ABSTRACT

YEAR	STUDENTS	DIRECTOR OF TECHNICAL PROJECT	TECHNICAL PROJECT THEME
2018	DAVID MARCELO CARPIO HOLGUIN	ING. LUIS SILVIO CÓRDOVA RIVADENEIRA, Msc.	" DESIGN AND IMPLEMENTATION OF A PID CONTROL WITH PROGRAMMED GAINS USING A SYSTEM EMBEDDED LAUNCHPAD MSP430 FOR BALL AND BEAM PLANT "

The present technical project aims at the implementation of a training module ball and beam, whereby students become familiar with the Adaptive PID control, perform basic and advanced practices for understanding of new controllers NATIONAL INSTRUMENTS LAUNCHPAD

The main objective of the project is to show the use of new embedded systems through the handling of basic and advanced tasks that help learning.

Additionally, we will check the ease of programming environment of the embedded LAUNCHPAD for the control system to implement.

Key words: LAUNCHPAD, embedded systems, pid, control systems, adaptive controls, M-430, drivers.



## ÍNDICE GENERAL

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA.....	II
CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR .....	III
CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN .....	IV
DEDICATORIA.....	V
AGRADECIMIENTO.....	VI
RESUMEN .....	VII
ABSTRACT .....	VIII
ÍNDICE GENERAL .....	IX
ÍNDICE DE FIGURAS .....	XII
ÍNDICE DE TABLAS .....	XV
INTRODUCCIÓN.....	1
1. EL PROBLEMA .....	2
1.1. Antecedentes .....	2
1.2. Importancia y Alcances .....	2
1.3. Beneficiarios.....	2
1.4. Delimitación.....	2
1.4.1. Temporal .....	2
1.4.2. Espacial.....	2
1.4.3. Académica .....	3
1.5. Innovación.....	3
1.6. Objetivos .....	3
1.6.1. Objetivo general .....	3
1.6.2. Objetivos específicos .....	3
2. FUNDAMENTOS TEÓRICOS .....	4
2.1. Microprocesadores y Microcontroladores .....	4
2.1.1. Microcontroladores.....	4
2.1.2. Microprocesadores.....	4
2.2. Sistemas embebidos .....	4
2.3. Launchpad M-430 .....	5
2.4. Energía Ide (Integrated Development Environment).....	6
2.5. Comunicación Serial RS232 .....	6
2.6. Comunicación I2C.....	6
2.7. Sistemas lineales .....	7
2.8. Sistemas no Lineales .....	7
2.9. PID .....	7
2.9.1. Proporcional (P) .....	8

2.9.2. Integral (I).....	9
2.9.3. Proporcional Derivativo (PD).....	9
2.9.4. Proporcional Integral Derivativo .....	9
2.10. Motores de corriente continua.....	10
2.11. Controlador de Motor (Sabertooth 2x12) .....	11
2.11.1. Modo Entrada Analógica .....	12
2.11.2. Modo Entrada Radio Controlada .....	12
2.11.3. Modo Entrada Serial.....	13
2.11.1. Modo Trama Serial .....	14
2.12. Resistencia variables .....	14
2.13. Sensor de distancia VL53L0X.....	15
2.14. Planta Balón y Viga.....	16
2.14.1. Estación de trabajo bola y varilla .....	16
2.14.2. Balance robótico con una esfera .....	17
2.15. Sistemas de control de lazo abierto. ....	17
2.16. Sistemas de control de circuito cerrado. ....	17
2.17. Sistemas de control multivariable. ....	18
2.17.1. Controlador PID para un motor de CC con retardo de tiempo. ....	18
2.17.2. Diseño de peso de incertidumbre aditiva.....	19
3. MARCO METODOLÓGICO.....	20
3.1. Diseño de la Planta Balón y Viga.....	20
3.2. Etapa de Control .....	20
3.3. Etapa de la Planta.....	21
3.4. Diagrama de control.....	21
3.5. Módulo de Control.....	22
3.6. Módulo de Potencia. ....	23
3.7. Modulo Alimentación.....	23
3.8. Instrumentación.....	24
3.9. Diseño Electrónico .....	24
3.10. Cables de motor .....	25
3.11. Cables de programación .....	26
3.12. Cables de sensores .....	26
3.13. Conexiones internas de la etapa de control.....	27
3.14. Diagrama de conexiones Entradas, salidas Digitales y Analógicas.....	28
3.15. Tarjetas de acondicionamiento de señal del sensor .....	28
3.16. Tarjetas de acondicionamiento de señal de conversión Pwm a voltaje .....	29
4. PRÁCTICAS DE LABORATORIO .....	30
4.1. Práctica #1 .....	30

4.2. Práctica #2 .....	38
4.3. Práctica #3 .....	44
4.4. Práctica #4 .....	48
4.5. Práctica #5 .....	52
4.6. Práctica #6 .....	63
4.7. Práctica #7 .....	69
4.8. Práctica #8 .....	74
RESULTADOS .....	82
CONCLUSIONES.....	86
RECOMENDACIONES .....	87
REFERENCIAS BIBLIOGRÁFICAS .....	88
BIBLIOGRAFÍA .....	88

## ÍNDICE DE FIGURAS

<b>Figura 1:</b> Tarjeta Launchpad M-430 Texas Instrument .....	5
<b>Figura 2:</b> Energía (IDE) .....	6
<b>Figura 3:</b> Comunicación I2C .....	7
<b>Figura 4:</b> Control PID de una planta .....	8
<b>Figura 5:</b> Motor CC .....	10
<b>Figura 6:</b> Motor CC dimensiones.....	10
<b>Figura 7:</b> Controladora de Motor Sabertooth.....	11
<b>Figura 8:</b> Modo Entrada Analógica .....	12
<b>Figura 9:</b> Modo Entrada Radio Controlada .....	13
<b>Figura 10:</b> Modo Entrada Serial Velocidades.....	13
<b>Figura 11:</b> Modo Maestro Esclavo .....	14
<b>Figura 12:</b> Modo Trama Serial.....	14
<b>Figura 13:</b> Potenciómetros Resistencia Variable.....	15
<b>Figura 14:</b> Sensor de distancia VL53L0X.....	15
<b>Figura 15:</b> Diagrama esquemático del VL53L0X.....	16
<b>Figura 16:</b> Estación Balón y varilla .....	16
<b>Figura 17:</b> Un balancín robótico con una esfera.....	17
<b>Figura 18:</b> Diagrama de bloque de control de lazo abierto.....	17
<b>Figura 19:</b> Diagrama de bloque de control de lazo cerrado .....	18
<b>Figura 20:</b> Diagrama de bloque de control multivariable .....	18
<b>Figura 21:</b> Estructura de incertidumbre aditiva básica .....	19
<b>Figura 22:</b> Etapa de control .....	20
<b>Figura 23:</b> Etapa de planta balón y viga .....	21
<b>Figura 24:</b> Diagrama de bloques .....	22
<b>Figura 25:</b> Planta Balón y viga .....	22
<b>Figura 26:</b> Controlador de potencia Sabertooth.....	23
<b>Figura 27:</b> Fuente de alimentación 24-12-5V CC .....	23
<b>Figura 28:</b> Instrumentación posición y ángulo .....	24
<b>Figura 29:</b> Tarjeta del embebido.....	24
<b>Figura 30:</b> Tarjeta del embebido sobre Pcb.....	25
<b>Figura 31:</b> Conector XLR .....	25
<b>Figura 32:</b> Diagrama de conexión motor .....	25
<b>Figura 33:</b> Diagrama de conexión USB .....	26
<b>Figura 34:</b> Conector XLR 4 pines .....	26
<b>Figura 35:</b> Conector XLR 7 pines .....	27
<b>Figura 36:</b> Conexiones internas de la etapa de control .....	27
<b>Figura 37:</b> Diagrama de Conexiones a la tarjeta Launchpad .....	28
<b>Figura 38:</b> Conexión a los periféricos .....	28
<b>Figura 39:</b> Diagrama de tarjeta acopladora .....	28
<b>Figura 40:</b> Tarjeta Acopladora de Pwm a Voltaje .....	29
<b>Figura 41:</b> Conexiones entre planta y caja de control .....	31
<b>Figura 42:</b> Ide Energía de Texas Instrument .....	31
<b>Figura 43:</b> Configuración de puerto Serial .....	32
<b>Figura 44:</b> Configuración de la tarjeta embebida de Texas Instrument.....	32
<b>Figura 45:</b> Creación de archivo nuevo.....	33
<b>Figura 46:</b> Esquema de Conexión Interna creado en Fritzing .....	33
<b>Figura 47:</b> Creación de variable análoga configuración Serial .....	33
<b>Figura 48:</b> Bloque principal del programa para monitoreo.....	33
<b>Figura 49:</b> Acción de Compilar y Cargar en la tarjeta electrónica .....	34
<b>Figura 50:</b> Acción de Monitor Serial en el Ide Energía .....	34
<b>Figura 51:</b> Monitor Serial del Ide Energía.....	34
<b>Figura 52:</b> Viga posición alta y baja.....	35

<b>Figura 53</b>	Parametrización y restricción de rangos .....	35
<b>Figura 54</b>	Datos normalizados y escalados.....	36
<b>Figura 55</b>	Datos del puerto serial.....	36
<b>Figura 56</b>	Comparación Voltaje Vs dato .....	36
<b>Figura 57</b>	Conexión entre Etapa de control y planta .....	39
<b>Figura 58</b>	Conexión entre Etapa de control y planta .....	39
<b>Figura 59</b>	Variables entrada y salida en Práctica 2. ....	40
<b>Figura 60</b>	Configuración Práctica 2 .....	40
<b>Figura 61</b>	Declaración lectura análoga y normalización en Práctica 2.....	40
<b>Figura 62</b>	Declaración de comparación en Práctica 2 .....	41
<b>Figura 63</b>	Planta Balón y Viga 2 .....	41
<b>Figura 64</b>	Resultados del control de posición 2.....	42
<b>Figura 65</b>	Grafica Sensor Vs Referencia .....	42
<b>Figura 66</b>	Declaración Variables-Práctica 3 .....	45
<b>Figura 67</b>	Declaración en lazo configuración-Práctica 3 .....	45
<b>Figura 68</b>	Señal Ancho de Pulso-Práctica 3.....	45
<b>Figura 69</b>	Lectura de ancho de pulso y normalización Práctica 3 .....	45
<b>Figura 70</b>	Lectura e impresión por puerto serial de la entrada Práctica 3 .....	46
<b>Figura 71</b>	Serial Plotter en la Práctica 3 .....	46
<b>Figura 72</b>	Dato en puerto Serial Plotter Práctica 3 .....	46
<b>Figura 73</b>	Dato Pwm vs normalizado Práctica 3.....	47
<b>Figura 74</b>	Declaración en lazo configuración-Práctica 4 .....	49
<b>Figura 75</b>	Declaración del filtro-Práctica 4.....	49
<b>Figura 76</b>	Declaración del lazo principal-Práctica 4.....	49
<b>Figura 77:</b>	Plotter en puerto Serial con filtro aplicado-Práctica 4 .....	50
<b>Figura 78:</b>	Plotter en puerto Serial sin filtro aplicado-Práctica 4.....	50
<b>Figura 79</b>	Dato filtrado vs dato sin filtrar Práctica 4.....	51
<b>Figura 80:</b>	Configuración de Sabertooth en modo análogo Práctica 5.....	53
<b>Figura 81:</b>	Tarjeta de acoplamiento Pwm a voltaje-Práctica 5 .....	53
<b>Figura 82</b>	Declaración de librerías-Práctica 5.....	53
<b>Figura 83</b>	Declaración de variables del encoder canal A-B-Práctica 5.....	54
<b>Figura 84</b>	Declaración de configuración - Práctica 5 .....	54
<b>Figura 85</b>	Declaración de configuración - Práctica 5 .....	54
<b>Figura 86</b>	Declaraciones de Pid de límites-Práctica 5 .....	55
<b>Figura 87</b>	Configuración Modo y tiempo Muestreo -Práctica 5.....	55
<b>Figura 88</b>	Método configuración Práctica 5 .....	55
<b>Figura 89</b>	Método configuración Práctica 5 .....	55
<b>Figura 90</b>	Método para lectura encoder Práctica 5.....	56
<b>Figura 91</b>	Declaración de la referencia Práctica 5.....	56
<b>Figura 92</b>	Declaración de la entrada del PID-Práctica 5.....	56
<b>Figura 93</b>	Función de saturación Práctica 5 .....	57
<b>Figura 94</b>	Función de saturación Práctica 5 .....	57
<b>Figura 95</b>	Función de monitoreo1 Práctica 5.....	57
<b>Figura 96</b>	Función de monitoreo1 Práctica 5.....	58
<b>Figura 97</b>	Matlab-Práctica 5.....	58
<b>Figura 98</b>	Creación de Script Matlab-Práctica 5.....	58
<b>Figura 99</b>	Limpia virtualmente el Puerto USB -Práctica 5 .....	59
<b>Figura 100</b>	Creación de objeto del puerto serial-Práctica 5.....	59
<b>Figura 101</b>	Limpia virtualmente el Puerto USB -Práctica 5 .....	59
<b>Figura 102</b>	Configurar números de muestras-Práctica 5.....	59
<b>Figura 103</b>	Configuración de ventana grafica-Práctica 5.....	60
<b>Figura 104</b>	Configurar los vectores de la ventana grafica-Práctica 5.....	60
<b>Figura 105</b>	configuración de ventana grafica-Práctica 5 .....	60
<b>Figura 106</b>	configuración de ventana grafica-Práctica 5 .....	61

<b>Figura 107</b>	Datos de referencia y variable de entrada Práctica 5.....	61
<b>Figura 108</b>	Datos de referencia y variable de entrada aplicando perturbación Práctica 5 .....	62
<b>Figura 109</b>	Declaración de variables Práctica 6 .....	64
<b>Figura 110</b>	Declaración de variables Pid Angulo y Pid Posición Práctica 6 .....	64
<b>Figura 111</b>	Declaración de variables Práctica 6 .....	64
<b>Figura 112</b>	Declaración de variables Práctica 6 .....	65
<b>Figura 113</b>	Configuración de Pid de ángulo y posición -Práctica 6 .....	65
<b>Figura 114</b>	Declaración de velocidad transmisión-Práctica 6.....	65
<b>Figura 115</b>	Método de lectura del encoder-Práctica 6.....	66
<b>Figura 116</b>	Declaración de método para filtrado y lectura-Práctica 6.....	66
<b>Figura 117</b>	Declaración de referencia y entrada del pid -Práctica 6.....	66
<b>Figura 118</b>	Declaración de los pid de ángulo y posición -Práctica 6 .....	67
<b>Figura 119</b>	Compensación y saturación a la salida del controlador Práctica 6 .....	67
<b>Figura 120</b>	Salida análoga para control del motor Práctica 6.....	67
<b>Figura 121</b>	Método de monitoreo Práctica 6.....	67
<b>Figura 122</b>	Datos de referencia y variable de entrada Práctica 6.....	68
<b>Figura 123</b>	Respuesta de controlador con $K_p=1$ Práctica 7 .....	70
<b>Figura 124</b>	Respuesta de controlador con $K_p=100$ Práctica 7 .....	70
<b>Figura 125</b>	Herramienta Data Cursor Práctica 7 .....	71
<b>Figura 126</b>	Inicio del Ciclo de oscilación Práctica 7.....	71
<b>Figura 127</b>	Final del ciclo de oscilación Práctica 7 .....	71
<b>Figura 128</b>	Respuesta del controlador Práctica 7.....	72
<b>Figura 129</b>	Respuesta con referencia de 20 y proporcional 1 Práctica 8.....	75
<b>Figura 130</b>	Respuesta con referencia de 20 y proporcional 21.75 Práctica 8 .....	75
<b>Figura 131</b>	Respuesta con referencia de 20 Práctica 8.....	76
<b>Figura 132</b>	Respuesta con referencia de 15 y proporcional 1-Práctica 8.....	76
<b>Figura 133</b>	Respuesta con referencia de 15 y proporcional 8.94-Práctica 8.....	77
<b>Figura 134</b>	Respuesta con referencia de 15 Práctica 8.....	77
<b>Figura 135</b>	Método para ganancias programadas Práctica 8.....	78
<b>Figura 136</b>	Método del pid del balón Práctica 8 .....	78
<b>Figura 137</b>	Método del pid del Angulo Viga Práctica 8.....	78
<b>Figura 138</b>	Método de saturación pid del Angulo Viga Práctica 8 .....	79
<b>Figura 139</b>	Método de supervisión del proceso Práctica 8.....	79
<b>Figura 140</b>	Método lazo principal supervisión del proceso Práctica 8.....	79
<b>Figura 140</b>	Método lazo principal supervisión del proceso Práctica 8.....	80
<b>Figura 142</b>	Respuesta del controlador 15 Práctica 8.....	80
<b>Figura 143</b>	Respuesta del controlador 20 Práctica 8.....	81

## ÍNDICE DE TABLAS

<b>Tabla 1:</b> Tipos de motores pololu .....	11
<b>Tabla 2:</b> Modos del controlador Sabertooth .....	12
<b>Tabla 3:</b> Conector XLR 7 .....	27
<b>Tabla 4:</b> Valores Sintonización- .....	72
<b>Tabla 5:</b> Valores Sintonización con ganancia de 100- .....	72
<b>Tabla 6:</b> Valores con referencia de 20. ....	76
<b>Tabla 7:</b> Valores con referencia de 15. ....	77

## INTRODUCCIÓN

Actualmente muchos problemas de control que aparecen en la industria necesitan de un sistema embebido como núcleo ya que tiene grandes ventajas que presentan en aplicaciones donde se requiere altos niveles de procesamiento, dichos sistemas constan con una entrada determinada y su respectiva salida que tiende a ser constante.

El presente proyecto está enfocado a estudiantes de la carrera de Ingeniería Electrónica con mención en sistemas industriales de la universidad Politécnica Salesiana sede Guayaquil para que de esta manera obtengan un conocimiento aplicado en problemas lógicos de programación y manejo de sistemas en los cuales se utilice planta de control no lineales, por consiguiente, el proyecto presenta una variedad de prácticas, las cuales están basadas en equipos de la marca Texas instruments ya que aporta un gran nivel de cómputo frente a otros microprocesadores y sistemas embebidos, además de ser una marca con un sinnúmero de avances tecnológicos que ayudan al aprendizaje y fomentan a la investigación de nuevos procesos de control.

En la sección del problema se encuentra las bases para el planteamiento del proyecto técnico tales como antecedentes, importancias, delimitación y objetivos de este.

En fundamentos teóricos se cuenta con la documentación de los protocolos y periféricos establecidos, así como métodos implementados en el desarrollo del trabajo de titulación.

El marco metodológico posee en detalle el diseño de la planta balón y viga, también como etapas de control, instrumentación para la lectura de las variables físicas en el proceso (distancia y posición angular), diagramas de conexiones para la comunicación entre el control y la planta actuadores

En las prácticas de laboratorio se observa un manual en detalle de diferentes rutinas para el aprendizaje y entendimiento de la instrumentación usada en el proyecto técnico.



## **1. EL PROBLEMA**

### **1.2. Antecedentes**

Los estudiantes de la carrera de ingeniería electrónica se encuentran con limitaciones para realizar actividades que contengan un estudio de sistemas no lineales como la planta (Balón y Viga) y procesos de control donde intervengan la demostración de dichos sistemas, ya que se cuenta con pocas plantas experimentales en el laboratorio de control y automático.

### **1.3. Importancia y Alcances**

El presente trabajo contribuye con el desarrollo de conocimientos referidos a sistemas de control muy utilizados en las carreras de ingeniería y en la práctica cotidiana.

Con el este proyecto el estudiante podrá experimentar y comprobar todos los conocimientos teóricos que le fueron impartidos, de una manera experimental especialmente en procesos de control no lineales utilizando sistemas embebidos Launchpad M-430 de la marca Texas Instrument.

### **1.4. Beneficiarios**

Como beneficiarios del proyecto se tendrá a la Universidad Politécnica Salesiana sede Guayaquil en la cual tanto estudiantes, como Maestros tendrán a disposición el equipo para su respectivo uso en prácticas de laboratorio y fomentar las bases de conocimiento, tan importante para la formación de ingeniería.

### **1.5. Delimitación**

#### **1.5.1. Temporal**

La implementación de este proyecto se realizará en un intervalo de un año a partir de la aprobación de este.

#### **1.5.2. Espacial**

El proyecto está dirigido para ser utilizado en el laboratorio de control automático de la Universidad Politécnica Salesiana sede Guayaquil.

### **1.5.3. Académica**

La implementación del presente trabajo está orientado al estudio de sistemas de control en conjunto con los microcontroladores, sistema embebido, sensores y controladores electrónicos.

El entorno de programación que se utiliza está basado en lenguaje C, el mismo que estará implementado en una Launchpad M-430 de la marca TEXAS INSTRUMENTS.

### **1.6. Innovación**

Para la implementación del proyecto se empleó controladores de la familia de Texas Instruments, el cual poseen procesamientos de 16 bits superando a controladores convencionales.

### **1.7. Objetivos**

#### **1.7.1. Objetivo general**

Diseñar e implementar un control PID con ganancias programadas utilizando un sistema embebido Launchpad M-430 para planta balón y viga.

#### **1.7.2. Objetivos específicos**

- Diseñar una planta balón y viga la cual conste con los elementos necesarios para el control PID.
- Programar el dispositivo embebido Launchpad mediante lenguaje C para el control PID con ganancias programadas.
- Diseñar e implementar circuitería electrónica para integrar los diferentes periféricos de control tales como sensores, motorreductores y controladores de motores para el funcionamiento de esta.
- Realizar prácticas didácticas para comprobar la eficiencia del control PID.

## **2. FUNDAMENTOS TEÓRICOS**

### **2.1. Microprocesadores y Microcontroladores**

#### **2.1.1. Microcontroladores**

Los microcontroladores o MCU se caracterizan por poseer una arquitectura tipo Harvard con memorias de programas independientes, por su simplicidad al momento de acceder a un espacio de memoria (Angulo Usategui, Romero Yesa, & M., 2006).

#### **2.1.2. Microprocesadores**

Los microprocesadores son aquellos que contienen un oscilador de cuarzo que genera los pulsos a un ritmo constante de manera que se pueden generar varios ciclos en un segundo, la velocidad de los microprocesadores se mide en frecuencias y estas pueden ser mega hertzios (MHz) o Giga hertzios (GHz) es decir, miles de millones o millones de ciclos por segundo superando a los microcontroladores (GONZALEZ, 2009).

Poseen subsistemas de entrada/salida y memoria que pueden ser combinados con un subsistema de CPU para formar una computadora o sistema embebido completo. Estos subsistemas se interconectan mediante los buses de sistema formados a su vez por el bus de control, el bus de direcciones y el bus de datos (Heath, 2003).

##### **2.1.2.1. Subsistema de entrada**

El subsistema de entrada acepta datos externos los cuales son procesados y se obtiene una salida del sistema. Lo más habitual es que haya varios subsistemas de entrada y varios de salida (Heath, 2003).

##### **2.1.2.2. Subsistema de memoria**

El subsistema de memoria almacena los comandos que dan el funcionamiento del sistema. La memoria también almacena varios tipos de datos: datos de entrada que aún no han sido procesados, resultados intermedios del procesado y resultados finales en espera de salida al exterior (Heath, 2003).

### **2.2. Sistemas embebidos**

Los sistemas embebidos tienen una respuesta en tiempo real con un intervalo restringido, se clasifican como blandos o duros. Si un sistema de tiempo real

blando no cumple con sus restricciones de tiempo, simplemente se degrada el rendimiento del sistema, pero si el sistema es de tiempo real duro y no cumple con sus restricciones de tiempo, el sistema fallará. Este fallo puede tener posiblemente consecuencias catastróficas (Heath, 2003).

Dichos sistemas usan como núcleo un microprocesador o microcontrolador, FPGA o DSPS, estos tienen sistemas simples y complejos (Cruz & Lutenberg, 2012).

### 2.3. Launchpad M-430

Pertenece a la familia de microcontroladores fabricados por Texas Instruments, consta de un CPU de 16 bits, el cual permite su uso en sistemas de control de complejidad media y avanzada (Barrientos, Antonio, Peñín, Luis Felipe, & Balag, 2007), este dispositivo permite una gran capacidad de funciones tales como:

- PWM
- Temporizador watchdog
- Comunicación USART
- Comunicación bus SPI
- Comunicación bus I<sup>2</sup>C
- Conversores ADC de 16 bits

En la Figura 1 se aprecia el hardware de la tarjeta perteneciente a Texas Instrument.



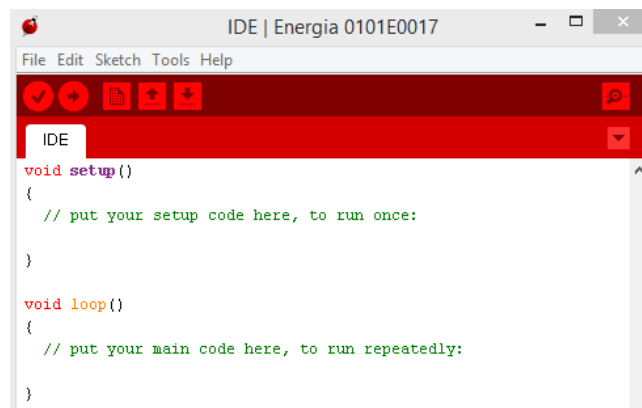
**Figura 1:** Tarjeta Launchpad M-430 Texas Instrument (MSP430, 2014)

Dichos controladores usan una arquitectura von Neuman con direccionamiento simple para las instrucciones y los datos. La memoria se direcciona por bloque de 1 byte, y los pares de byte se combinan en forma de arreglos para hacer palabras de 16 bits. (MSP430, 2014)

## 2.4. Energía Ide (Integrated Development Environment)

Energy es una plataforma de creación de prototipos electrónicos de código abierto iniciada por Robert Wessel en enero de 2012 con el objetivo de llevar el marco Wiring y Arduino al Launchpad basado en Launchpad M-430 de Texas Instruments, posee un entorno de desarrollo integrado (IDE) basado en Processing que en plenitud trabaja en lenguaje de alto nivel CCS (MSP430, 2014).

En la figura 2 se presenta el entorno del Energía (ide) perteneciente a la familia Texas Instruments.



**Figura 2:** Energía (IDE) (MSP430, 2014)

## 2.5. Comunicación Serial RS232

El protocolo de comunicación serial es ampliamente usado ya que posee diferentes tipos de velocidad de comunicación cuya escala es medida en baudios, permitiendo la conexión entre dispositivos también llamada comunicación dúplex.

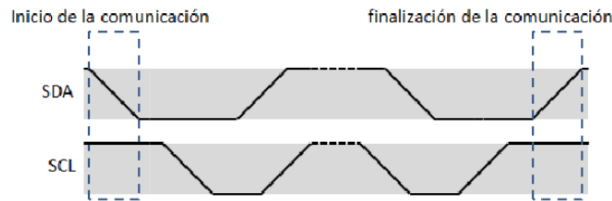
Este protocolo de comunicación está presente en la mayoría de embebidos tales como la familia Texas Instrument M-430, el cual consta de 2 pines de comunicación uno para transmisión y otro de recepción (MSP430, 2014).

Son utilizados en una gran variedad de aplicaciones industriales.

## 2.6. Comunicación I2C

Es un protocolo de comunicación que sirve para acceder a diferentes periféricos que soporte el tipo de comunicación maestro esclavo, multi maestro y modo esclavo. Son utilizados en una gran variedad de aplicaciones en la industria principalmente en controles esclavo – maestro (Breijo, 2009).

La comunicación se realiza a través de dos conductores SCL línea de reloj para sincronización de datos y SDA línea de datos, las cuales van conectadas a VCC mediante resistencias de pull up para garantizar un nivel alto de comunicación como se visualiza en la figura 3.



**Figura 3:** Comunicación I2C (Breijo, 2009)

## 2.7. Sistemas lineales

La linealidad de un sistema ocurre cuando la salida tiene una relación con la entrada del control.

Estos sistemas poseen algunas características tales como:

- Las entradas y salidas son multiplicadas por una variante.
- Presentan el principio de superposición

Poseen una representación por ecuaciones diferenciales lineales (Ramírez, Márquez, Echeverría, & Santiago, 2004).

## 2.8. Sistemas no Lineales

Los sistemas no lineales son ecuaciones diferenciales con coeficientes que son función de la variable dependiente, ecuaciones diferenciales parciales, multiplicación entre variables, funciones senoidales con argumentos en función de la variable dependiente, o cualquier otro tipo de ecuación funcional. (Ramírez, Márquez, Echeverría, & Santiago, 2004)

## 2.9. PID

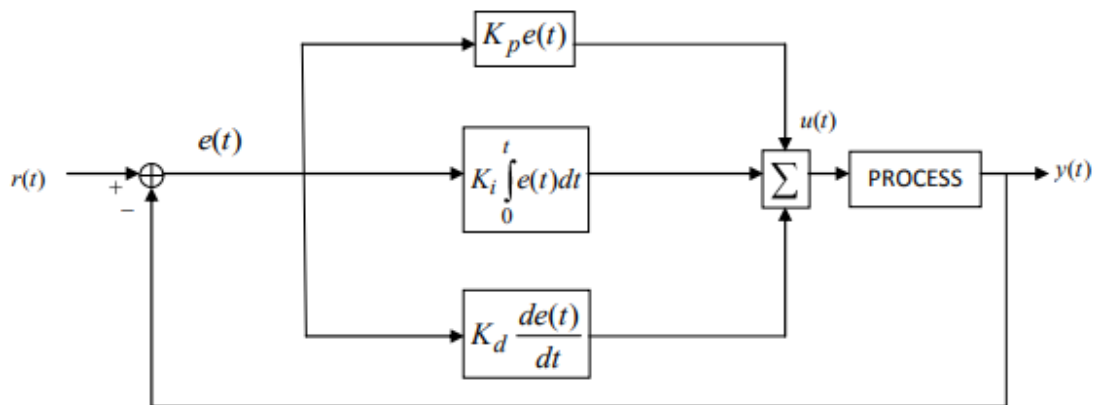
La lógica de control PID es ampliamente utilizada en la industria de control de procesos. Los controladores PID han sido tradicionalmente elegidos por los ingenieros de sistemas de control debido a su flexibilidad y fiabilidad. Un controlador PID tiene términos proporcionales, integrales y derivativos que se pueden representar en forma de función de transferencia como se aprecia en la ecuación 1.

$$K(s) = K_p + \frac{K_i}{s} + K_d s \quad (1)$$

Donde  $K_p$  representa la ganancia proporcional,  $K_i$  representa la ganancia integral, y  $K_d$  representa la ganancia derivativa, respectivamente. Al ajustar estas ganancias del controlador PID, el controlador puede proporcionar una acción de control diseñada para requisitos específicos del proceso.

El término proporcional genera un cambio en la salida que es proporcional al error actual. Este término proporcional se refiere al estado actual de la variable de proceso. El término integral ( $K_i$ ) es proporcional tanto a la magnitud del error como a la duración del error. (Cuando se agrega al término proporcional) acelera el movimiento del proceso hacia el punto de ajuste y, a menudo elimina el error de estado estacionario residual que puede ocurrir con un controlador proporcional único (Dorf & Bishop, 2005).

La tasa de cambio del error de proceso se calcula determinando la pendiente diferencial del error a lo largo del tiempo (es decir, su primera derivada con respecto al tiempo). Esta tasa de cambio en el error se multiplica por la ganancia derivada ( $K_d$ ).



**Figura 4:** Control PID de una planta (Ogata, 2010)

El PID contiene acciones, una proporcional (P), integral (I) y derivativa (D). Se puede realizar diferentes tipos de controladores tales como P, PI, PD y PID (Morales & Zafra Siancas, 2013).

### 2.9.1. Proporcional (P)

La salida del controlador es proporcional al error, como se muestra en la ecuación 2 (Mazzone, 2002).

$$u(t) = K_p \cdot e(t) \quad (2)$$

Donde:

$u(t)$ : salida del controlador.

$e(t)$ : Error del sistema.

$K_p$ : Ganancia proporcional al ajuste.

Dicha función de transferencia del controlador proporcional es la ecuación 3:

$$C_p(s) = K_p \quad (3)$$

Los controladores proporcionales sirven para una planta estable, las limitaciones son un trabajo limitado y error en un régimen permanente (Mazzone, 2002).

### 2.9.2. Integral (I)

La salida del controlador es proporcional al error acumulado como se visualiza en la siguiente fórmula: 4

$$U(t) = K_i \int_0^t e(t) dt \quad (4)$$

Donde:

$K_i$ : Ganancia integral.

### 2.9.3. Proporcional Derivativo (PD)

Se define mediante la ecuación 5:

$$u(t) = K_p \cdot e(t) + K_p \cdot T_d \cdot \frac{d e(t)}{dt} \quad (5)$$

La  $T_d$  es una variable definida como tiempo derivativo, la cual tiene una acción en la respuesta del sistema de ser más rápida, cuyas desventajas son la amplificación de ruidos y saturación sobre el actuador. Esta acción derivativa no se usa sola ya que es funcional durante tiempo transitorios. La educación proporcional derivativa resulta ser la ecuación 6 (Mazzone, 2002):

$$C_{pd}(t) = K_p \cdot e(t) + K_p \cdot T_d \cdot \frac{d e(t)}{dt} \quad (6)$$

Una acción de control derivativa se agrega a un controlador proporcional para obtener un controlador de alta sensibilidad, es decir la velocidad del cambio del error y se logra que el error se vuelva alto de esta manera el error no se vuelve de estado estacionario, agregando un amortiguamiento al sistema permitiendo una precisión estable al proceso (Mazzone, 2002).

### 2.9.4. Proporcional Integral Derivativo

La acción combinada de proporcional integral y derivativo logra un control eficaz de alto nivel con la cual la ecuación resultante que se obtiene es:

$$U(t) = K_p e(t) + K_i \int_0^t e(t) dt \quad (7)$$



La función de transferencia resultantes:

$$C_{pid}(t) = K_p \cdot \left( 1 + \frac{1}{T_{ids}} + T_{ds} \right) \quad (8)$$

## 2.10. Motores de corriente continua

Una definición básica de motor dc, podría estar en la función que este realiza, la cual es convertir la energía eléctrica en energía mecánica rotacional y lo hace siguiendo los fenómenos magnéticos (Poznyak, 2005).

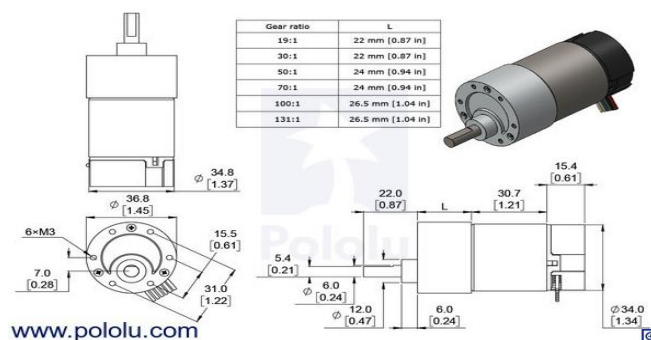
Los motores de corriente continua se manejan mediante la aplicación de voltaje entre sus bornes los cuales comenzaran a girar en un sentido, si se desea cambiar el giro se invertirá la conexión a los bornes (Calderon, 2004).

Este motorreductor que se muestra Figura 5 es un potente motor de CC de 12V con caja de engranajes metálicas 50: 1 y un codificador de cuadratura integrado que proporciona una resolución de 64 conteos por revolución del eje del motor, que corresponde a 3200 conteos por revolución del eje de salida de la caja de engranajes (Pololu, 2006).



**Figura 5:** Motor Dc (Pololu, 2006)

Las dimensiones del motor se muestran en la Figura 6 tanto como la relación de cada motor de esta familia.



**Figura 6:** Motor CC dimensiones (Pololu, 2006)

La siguiente tabla contiene los tipos de relación, torque y velocidad de cada motor.

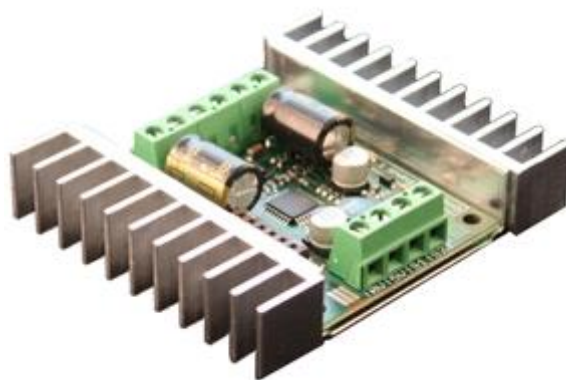
Caja engranajes	Velocidad	Torque
1:1	11,000 RPM	5 oz-in
19:1	500 RPM	84 oz-in
30:1	350 RPM	110 oz-in
50:1	200 RPM	170 oz-in
70:1	150 RPM	200 oz-in
100:1	100 RPM	220 oz-in
131:1	80 RPM	250 oz-in

**Tabla 1:** Tipos de motores pololu (Pololu, 2006)

### 2.11. Controlador de Motor (Sabertooth 2x12)

Un controlador de motor es un pequeño amplificador de corriente; la función de los controladores de motor es tomar una señal de control de baja corriente y luego convertirla en una señal de corriente más alta que pueda conducir un motor (FutureElectronics, 2017).

La controladora Sabertooth 2x12 (Figura 7) es fabricada por la empresa Dimensión Engineering y es uno de los controladores de motores duales más versátiles, eficientes y fáciles de usar, esta tarjeta ofrece control de dos motores Dc brushed (con escobillas), con un máximo de 12 Amperios por cada motor soportando picos de 24 Amperios por cada motor (Celi, 2013).



**Figura 7:** Controladora de Motor Sabertooth (Dimension Engineering, 2004)

Muchos controladores Sabertooth presenta modos mixtos diseñados especialmente para el accionamiento diferencial, donde dos los motores proporcionan tanto dirección como propulsión. También tiene opciones

independientes en todas las operaciones modos. Esto es útil si tienes dos motores para controlar, pero no son necesariamente utilizado para conducir un accionamiento diferencial. Los motores no necesitan ser emparejados o incluso similares, como siempre y cuando ambos estén dentro de los límites operativos de Sabertooth (Dimension Engineering, 2004).

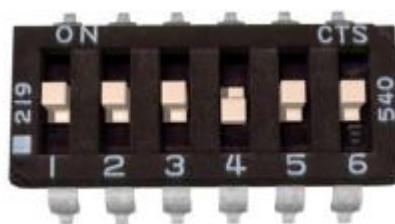
Estos drivers poseen un selector con el cual se configura el modo (Tabla 2):

Modo
Entrada Análoga
Entrada R/C
Serial
Trama Serial

**Tabla 2:** Modos del controlador Sabertooth

### 2.11.1. Modo Entrada Analógica

El modo de entrada analógica como se muestra en la Figura 8 toma una o dos entradas analógicas y las usa para establecer la velocidad y dirección del motor. El rango de entrada válido es de 0v a 5v. Esto hace que el control de Sabertooth usando un potenciómetro, la salida PWM de un microcontrolador (con un filtro RC) o un circuito analógico. Los principales usos incluyen vehículos controlados por joystick o pedal de pie, control de velocidad y dirección para bombas y máquinas, y circuitos de retroalimentación analógicos (Dimension Engineering, 2004).

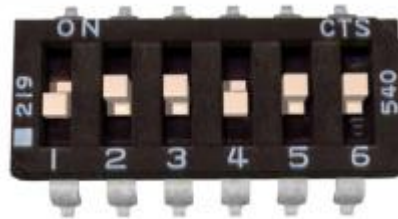


**Figura 8:** Modo Entrada Analógica (Dimension Engineering, 2004)

### 2.11.2. Modo Entrada Radio Controlada

El modo de entrada R / C (Figura 9) toma dos canales de R / C estándar y los usa para establecer la velocidad y dirección del motor. Hay una configuración de tiempo de espera opcional. Cuando el tiempo de espera está habilitado, el controlador del motor apague la pérdida de señal. Esto es por seguridad y para evitar que el robot se escape encuentra interferencia y debe usarse si se usa una radio para controlar el controlador. Si el tiempo de espera está

desactivado, el controlador del motor continuará conduciendo a la velocidad ordenada hasta otro se da el comando. Esto hace que el Sabertooth sea fácil de interconectar con un Sello Básico u otra velocidad baja microcontroladores (Dimension Engineering, 2004).



**Figura 9:** Modo Entrada Radio Controlada (Dimension Engineering, 2004)

### 2.11.3. Modo Entrada Serial

El modo en serie utiliza datos en serie de nivel TTL RS-232 para establecer la velocidad y la dirección del motor. Esto se usa para conectar el Sabertooth a una PC o microcontrolador. Si usa una PC, un nivel convertidor como un chip MAX232 debe ser utilizado. Un convertidor en serie USB-a-TTL también es una opción para la PC. La velocidad en baudios se establece mediante interruptores DIP Figura 10.

2400 Baud: 01x00x	9600 Baud: 01x10x
19200 Baud: 01x01x	38400 Baud: 01x11x

**Figura 10:** Modo Entrada Serial Velocidades (Dimension Engineering, 2004)

Los comandos son de un solo byte. Ahí está también un modo de selección de esclavo que permite el uso de varios Sabertooth 2x12 véase en Figura 11 desde un único puerto serie del microcontrolador (Dimension Engineering, 2004).

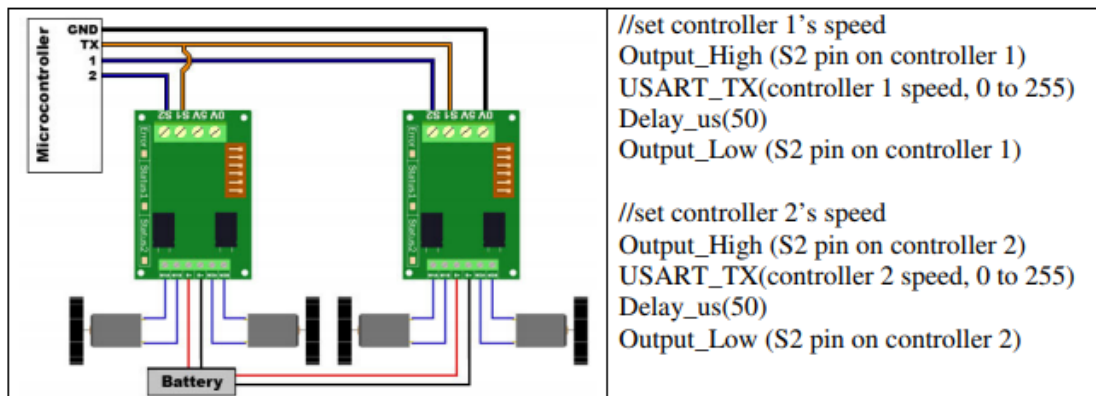


Figura 11: Modo Maestro Esclavo (Dimension Engineering, 2004)

### 2.11.1. Modo Trama Serial

El modo serie en paquete utiliza datos en serie de nivel TTL RS-232 para establecer la velocidad y la dirección del motor. Hay un formato de paquete corto que consiste en un byte de dirección, un byte de comando, un byte de datos y una suma de comprobación de 7 bits. La velocidad en baudios se establece en 9600 por defecto. Esto hace la trama serial, el método preferido para conectar múltiples Sabertooth a una PC o computadora portátil ya que se utiliza el mismo protocolo que nuestros controladores de motor Figura 12 (Dimension Engineering, 2004).

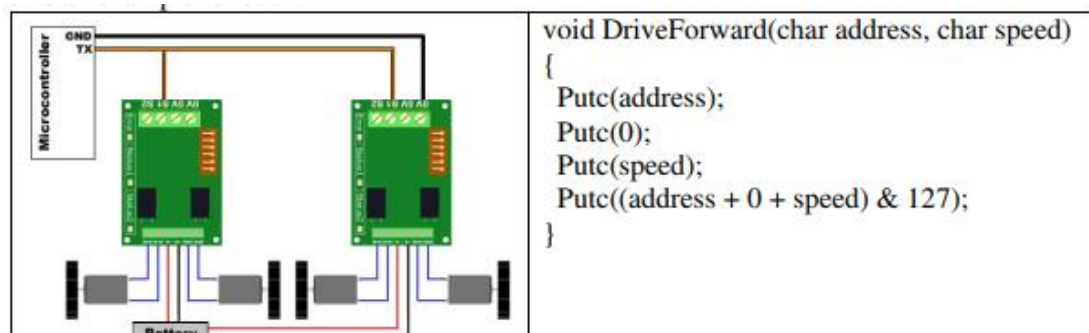


Figura 12: Modo Trama Serial (Dimension Engineering, 2004)

### 2.12. Resistencia variables

Un resistor variable es un resistor lineal sobre el cual desliza un contacto eléctrico capaz de inyectar corriente en un punto intermedio de su elemento resistivo (Areny, 2003).

Elementos que se oponen al paso de la carga eléctrica (Figura 13) en un circuito su función principal será la de marcar el diferencial entre cada nodo son construidas de grafito, película metálica o de alambre, su unidad es el ohmio( $\Omega$ ) (Areny, 2003).

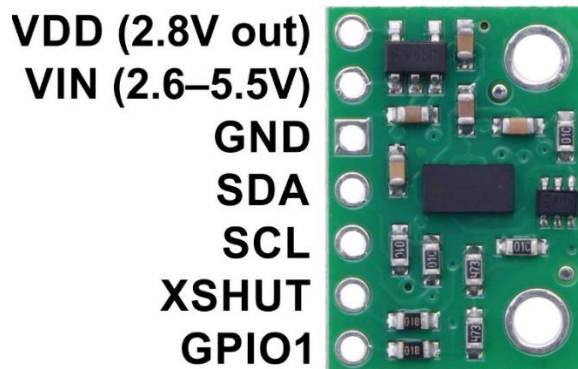


**Figura 13:** Potenciómetros Resistencia Variable (Areny, 2003)

### 2.13. Sensor de distancia VL53L0X

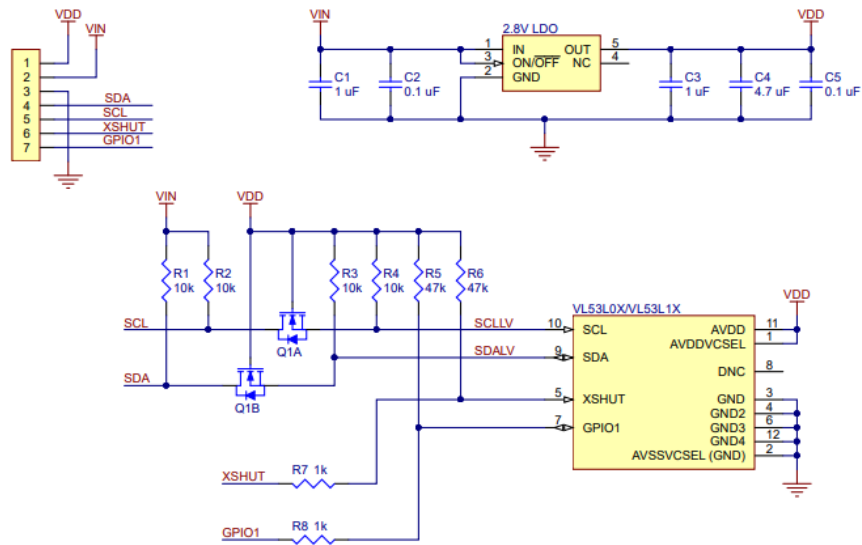
Los sensores de distancia sirven para medir el tiempo de respuesta en la reflexión de un punto a otro desde el emisor hasta el objetivo. Poseen un alcance hasta 2 m. El VL53L0X utiliza mediciones de tiempo de vuelo de pulsos infrarrojos para el rango, lo que le permite obtener resultados precisos del color y la superficie del objetivo. Las mediciones de distancia se pueden leer a través de la interfaz I<sup>2</sup>C digital ( STMicroelectronics, 2016).

La placa tiene un regulador lineal de 2.8 V e indicadores de nivel integrados que le permiten trabajar en un rango de voltaje de entrada de 2.6 V a 5.5 V, y el espaciado de pin de 0.1 "hace que sea fácil de usar con placas de prueba sin soldadura estándar como muestra en la figura 14 (Pololu, 2006).



**Figura 14:** Sensor de distancia VL53L0X (Pololu, 2006)

Las conexiones necesarias para el óptimo funcionamiento para la placa VL53L0X: VIN, GND, SCL y SDA. El pin VIN debe estar conectado a una fuente de 2.6 V a 5.5 V, y GND debe estar conectado a GND referente a la Figura 15. Un regulador de voltaje integrado convierte el VIN en un suministro de 2,8 V para el VL53L0X IC (FutureElectronics, 2017).



**Figura 15:** Diagrama esquemático del VL53L0X (Pololu, 2006)

## 2.14. Planta Balón y Viga

El sistema balón y viga sirve para analizar sistemas de control por su naturaleza inestable. Existen diferentes tipos de modelos comerciales y desarrollados por instituciones privadas.

### 2.14.1. Estación de trabajo bola y varilla - Quanser (2006)

Permite el control de la posición de un cuerpo esférico sobre una superficie plana la cual posee unos rieles que darán un diferencial de voltaje y este servirá de dato de entrada en la tarjeta de adquisición de datos. El cual tendrá como actuador un motor CC para mover la varilla y así controlar la posición del balón como se visualiza en la Figura 16 (Quanser, 2006).



**Figura 16:** Estación Balón y varilla (Quanser, 2006)

### 2.14.2. Balance robótico con una esfera- Lieberman (2009)

Lieberman construyó el sistema que se visualiza en la Figura 17. El sistema usa como base un sensor de posición el cual es un cable con una resistencia variable que esta dispersa en la viga y un sensor de Angulo para la posición del motor el cual posee un controlador el que comandara los cambios de giro (Jeff, 2004).

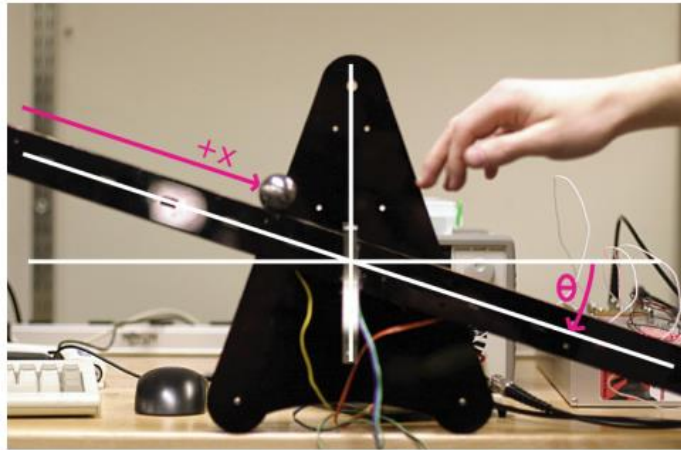


Figura 17: Un balancín robótico con una esfera (Jeff, 2004)

### 2.15. Sistemas de control de lazo abierto.

Un sistema de control de bucle abierto está diseñado para cumplir los objetivos deseados mediante el uso de una señal de referencia que impulsa los actuadores que controlan directamente la salida del proceso. La retroalimentación de salida no está presente en este tipo de sistema. La figura 18 muestra la estructura general de un sistema de control de circuito abierto. Algunos ejemplos de sistemas de control de bucle abierto son tostadoras de pan, hornos, lavadoras y sistemas de rociadores de agua.

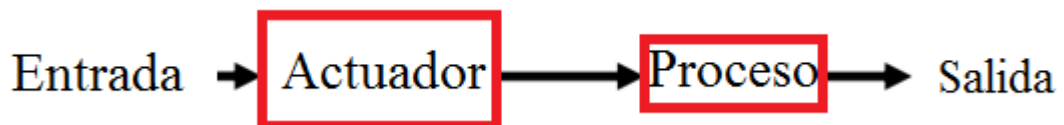


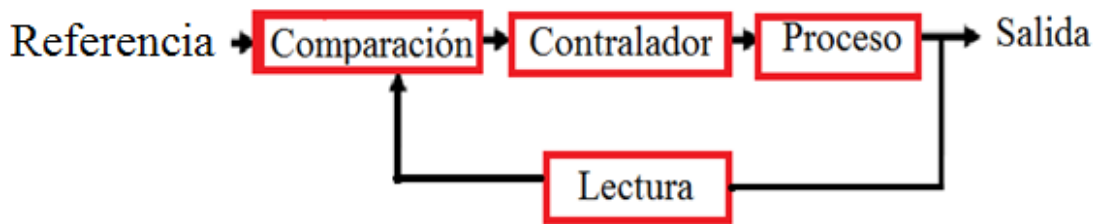
Figura 18: Diagrama de bloque de control de lazo abierto (Ogata, 2010)

### 2.16. Sistemas de control de circuito cerrado.

En los sistemas de control de circuito cerrado, la diferencia entre la salida real y la salida deseada se realimenta al controlador para cumplir con la salida del sistema deseada. A menudo, esta diferencia, conocida como señal de error, se amplifica y alimenta al controlador. La figura 19 muestra la estructura general de un sistema de control de retroalimentación de circuito cerrado.



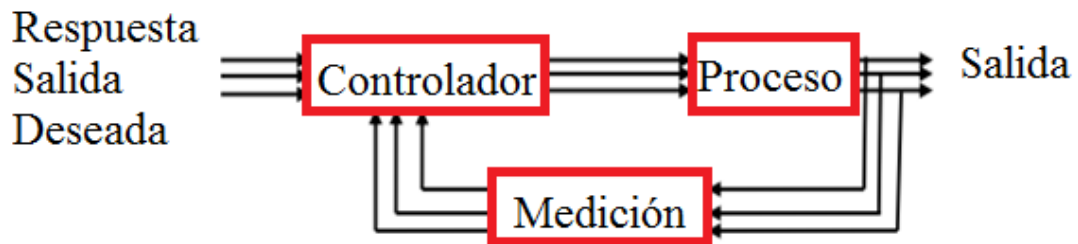
Algunos ejemplos de sistemas de control de retroalimentación son los elevadores, los termostatos y el control de crucero en los automóviles.



**Figura 19:** Diagrama de bloque de control de lazo cerrado (Ogata, 2010)

### 2.17. Sistemas de control multivariable.

El aumento en las complejidades de los sistemas de control involucrados y la interrelación entre las variables de proceso a veces requiere un sistema de control de retroalimentación multivariable. En la figura 2.20 se muestra una estructura general del sistema de control multivariable.



**Figura 20:** Diagrama de bloque de control multivariable (Ogata, 2010)

#### 2.17.1. Controlador PID para un motor de CC con retardo de tiempo.

En esta sección, buscamos controlar un modelo de motor de CC con un retardo de comunicación, donde  $G(s)$  es el motor de CC,  $K(s)$  es el controlador PID y  $e^{-\tau s}$  representa el  $\tau$  segundo retardo de comunicación. El objetivo del diseño es encontrar el conjunto de controladores PID que garanticen la sólida restricción de estabilidad,

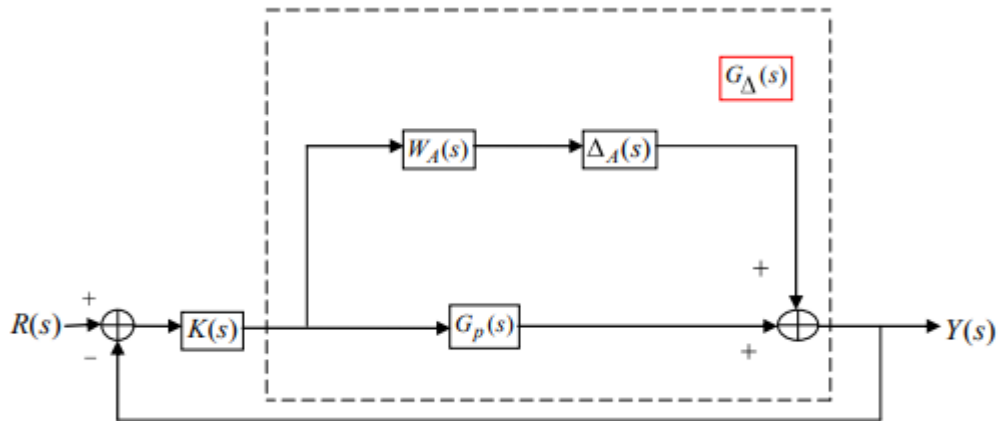
$$|w_a(j\omega)K(j\omega)S(j\omega)|_\infty \leq \gamma \quad (9)$$

se satisface para  $\gamma = 1$ , asegurando así la estabilidad nominal y robusta para la planta perturbada. Aquí  $W_A(j\omega)$ ,  $K(j\omega)$  y  $S(j\omega)$  son el peso de incertidumbre aditivo, el controlador PID y la función de sensibilidad dada en la siguiente ecuación,

$$s(j\omega) = \frac{1}{1 + G_p(j\omega)K(j\omega)} \quad (10)$$

### 2.17.2. Diseño de peso de incertidumbre aditiva.

La incertidumbre dinámica paramétrica y no modelada se puede combinar en una sola perturbación agrupada de una estructura elegida. Aquí, se considera una estructura de incertidumbre aditiva para vincular el rango de incertidumbres presentes en la planta. La figura 21 representa una estructura de incertidumbre aditiva básica utilizada para el modelo incierto.



**Figura 21:** Estructura de incertidumbre aditiva básica (Ogata, 2010)

El modelo de motor de CC incierto se representa como

$$G\Delta(S) = G_p(s) + W_a(s)\Delta A(s) \quad (11)$$

### 3. MARCO METODOLÓGICO

#### 3.1. Diseño de la Planta Balón y Viga

La planta Balón y Viga fue implementada de manera que ayude al entendimiento de procesos básicos y complejos, usando un sistema embebido Launchpad M-430 de la marca Texas Instrument, este sistema embebido consta con entradas y salidas análogas y digital, tanto como interfaces seriales, comunicación I2c y Spi. De esta forma el estudiante tendrá las herramientas para su aprendizaje.

La caja de control y la planta cuentan con los seguros necesarios para que no ocurra una mala conexión, señalado con diferentes tipos de conectores para comunicación y control.

La planta consta de un sistema de engranajes con 32 dientes para distinguir la posición de la viga mediante un potenciómetro acoplado a dicho engranaje de manera que facilite el entendimiento y desarrollo de las prácticas.

La etapa de posicionamiento del balón es medida por un sensor laser el cual será procesado por un microcontrolador y enviado al sistema embebido.

#### 3.2. Tablero de Control

La estructura para control de la planta está hecha en metal. Fue diseñada para tener los diferentes elementos para el funcionamiento tales como el embebido Launchpad, el controlador de motores, las interfaces para el procesamiento de datos del sensor una fuente de poder conmutada de 5-12-24 V y las conexiones véase Figura 22.



Figura 22 Etapa de control

### 3.3. Diseño de la Planta

La planta fue diseñada en SolidWorks donde se tomaron en cuenta las diferentes características de los elementos que se usarían en el módulo tales como motor, sensor, la viga y sus limitaciones para un funcionamiento óptimo, como los engranajes que se acoplarían al actuador o motor para conocer el ángulo de posicionamiento, todos los sensores de posición y distancia estarán conectados mediante unos plug véase Figura 23.

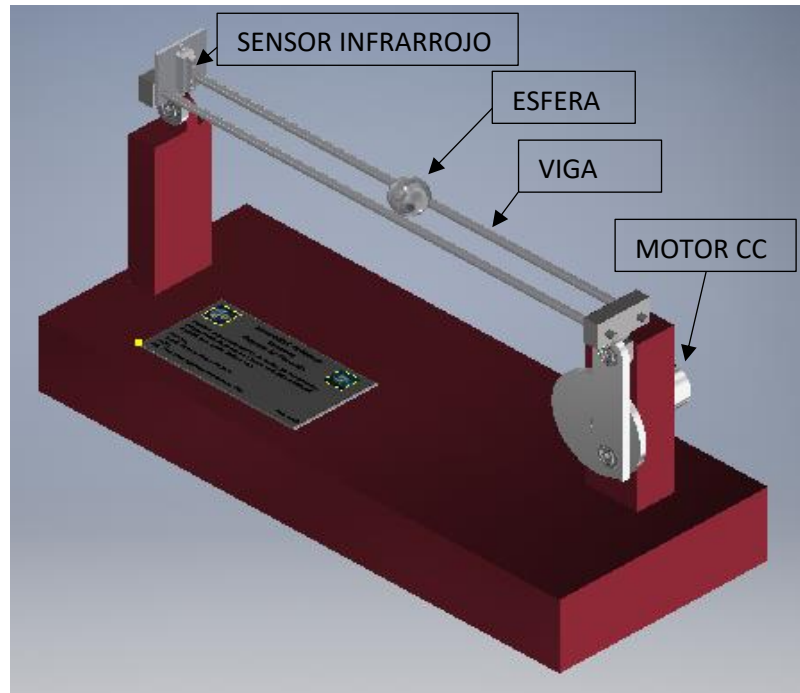


Figura 23 Etapa de planta balón y viga

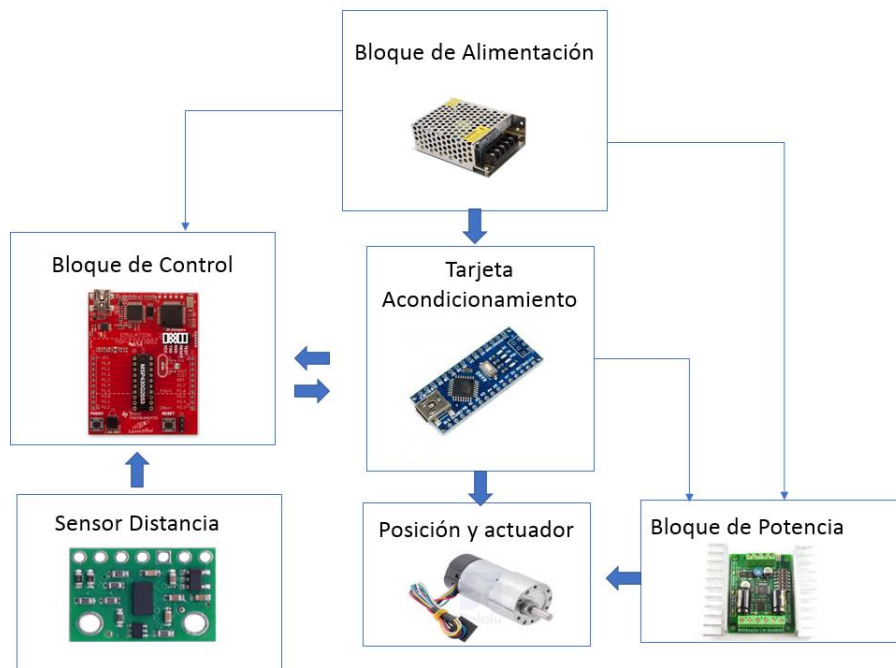
### 3.4. Diagrama de control

El control de la planta consta de un sistema de Lazo Cerrado, que se encargara de la posición del balón sobre la viga enviando el dato de la distancia procesada y comparando con punto de referencia variable aplicando un control PID con ganancias programables para el óptimo funcionamiento.

El bloque de control donde se procesará el ángulo del motor mediante una entrada análoga que será una resistencia variable(potenciómetro) acoplado mediante dos engranajes al eje principal del motor, verificando un dato de entrada que será la posición dada desde el embebido.

El bloque de **Potencia** es la parte del control para el sentido de giro del motor, este tomara un dato serial desde el bloque de control que controlara el driver Sabertooth 2x12 para el motor CC perteneciente a la familia pololu.

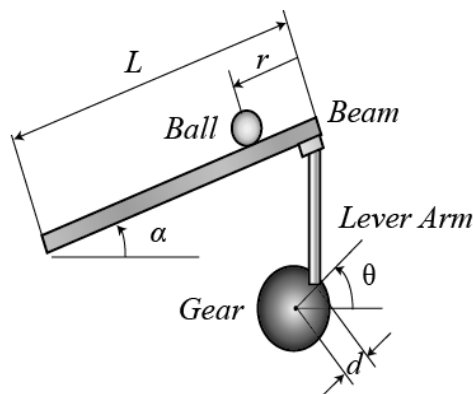
El bloque de Alimentación suministrará la energía necesaria y tendrá sus protecciones para evitar cortos y corrientes parasitas para prevenir un mal funcionamiento de la planta de procesos y control Balón y Viga



**Figura 24** Diagrama de bloques

### 3.5. Módulo de Control

El funcionamiento de la planta está controlado mediante un embebido Launchpad el cual tomará un valor prestablecido para su referencia para colocar una esfera a cierta distancia del sensor mediante un control PID adaptativo que cambiará sus ganancias, dependiendo el valor de referencia, este proceso de cálculo dará una salida estable para el control de posicionamiento véase en Figura 25 la ecuación prestablecida para el control de la planta depende la posición angular de la viga.

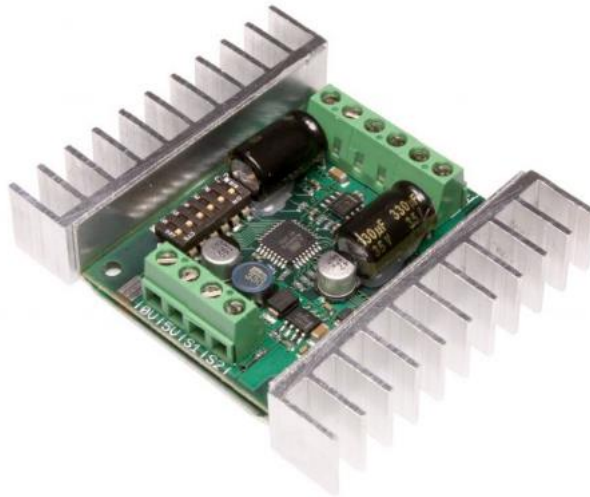


**Figura 25:** Planta Balón y viga (CTMS, 2012)

$$P(s) = \frac{R(s)}{\theta(s)} = - \frac{mgd}{L(\frac{J}{R^2} + m)} \frac{1}{s^2} \left[ \frac{m}{rad} \right] \quad (12)$$

### 3.6. Módulo de Potencia.

En el proyecto se utiliza un controlador SABERTOOTH véase en figura 26 el cual es un controlador robusto que tiene soporte hasta 12 A en corriente continua, con picos de corriente de 30 amperio, como un rango de operación de 4 a 24 voltios. El driver consta como entrada una señal de ancho de pulso y configurando la saber en modo de microcontrolador.



**Figura 26** Controlador de potencia Sabertooth (Dimension Engineering, 2004)

### 3.7. Modulo Alimentación.

En el proyecto se utiliza un módulo de poder conmutada el cual suministrará energía a toda la planta, colocada en la caja de control de la planta de procesos, la cual posee conectores especiales para la comunicación de esta véase Figura 27.



**Figura 27** Fuente de alimentación 24-12-5V CC (Dimension Engineering, 2004)

### 3.8. Instrumentación

Consta con un encoder incremental que posee la configuración de dos canales para lectura de posición mediante el cambio entre alto y bajos para que los grados de la viga tengan una resolución más óptima y sin ningún error en la lectura. En la instrumentación de la lectura de distancia se aplica un filtro por código para la eliminación de ruidos externos.

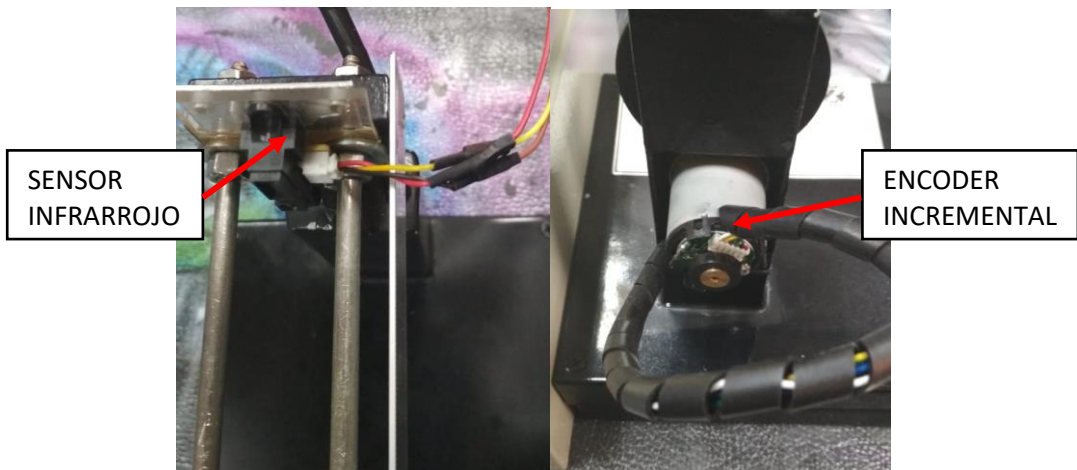


Figura 28 Instrumentación posición y ángulo

### 3.9. Diseño Electrónico

Para el sistema de control consta con una placa embebida la cual posee acoplada a una tarjeta de circuito impreso como se muestra en la Figura 29, para sus conexiones a los demás periféricos de control (Figura 30), como interfaces para lectura del control de posición del balón y viga.

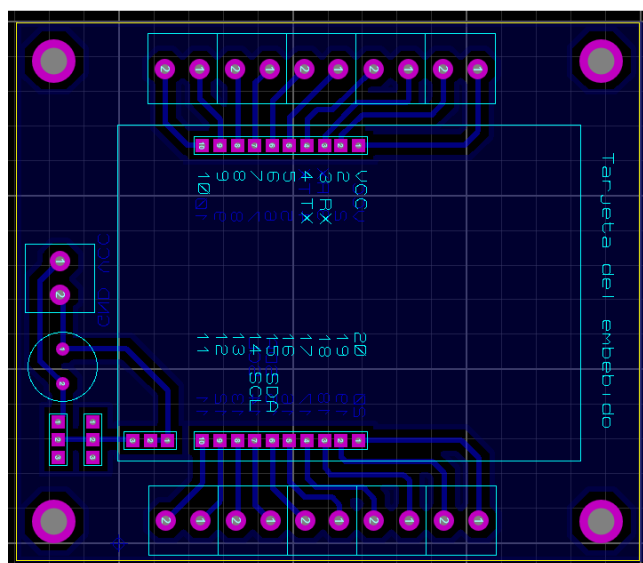
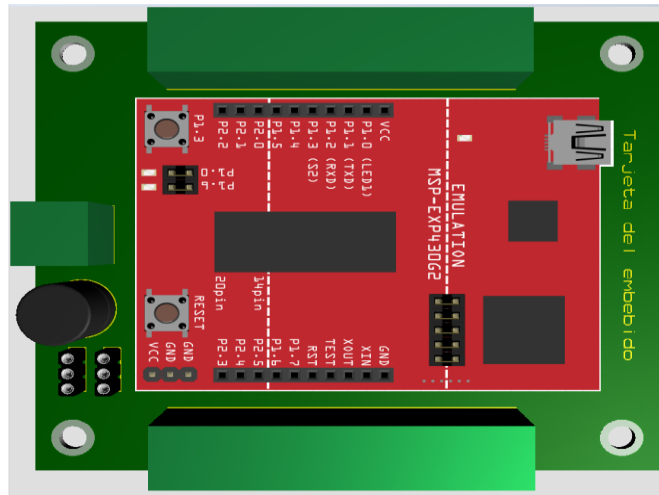


Figura 29 Tarjeta del embebido



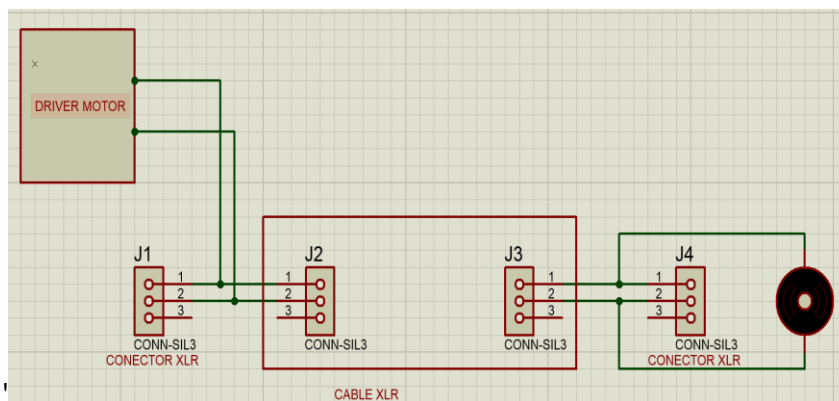
**Figura 30** Tarjeta del embebido sobre Pcb

### 3.10. Cables de motor

Para la conexión del actuador se diseña un conector tipo XLR de 3 pines véase Figura 31 para dejar señalizado el motor para que el estudiante no tenga problemas al momento de conectar el motor con la caja de control en donde se encuentren los demás periféricos. Este plug es industrial para ser usado en diferentes aplicaciones, internamente tiene el diagrama referente a la Figura 32.



**Figura 31** Conector XLR ( (Balbi, 2013)



**Figura 32** Diagrama de conexión motor



### 3.11. Cables de programación

En la etapa de control consta con un cable para que el usuario cargue las Prácticas pertinentes usando en plenitud el embebido (Figura 33) para esto se acoplo un cable USB a un conector XLR de 4 pines referente a la Figura 34 para transmitir la información necesaria al periférico.

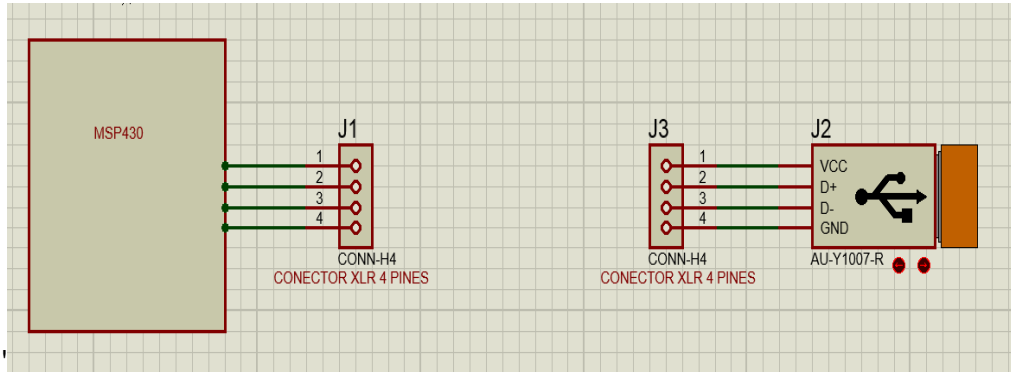


Figura 33 Diagrama de conexión USB

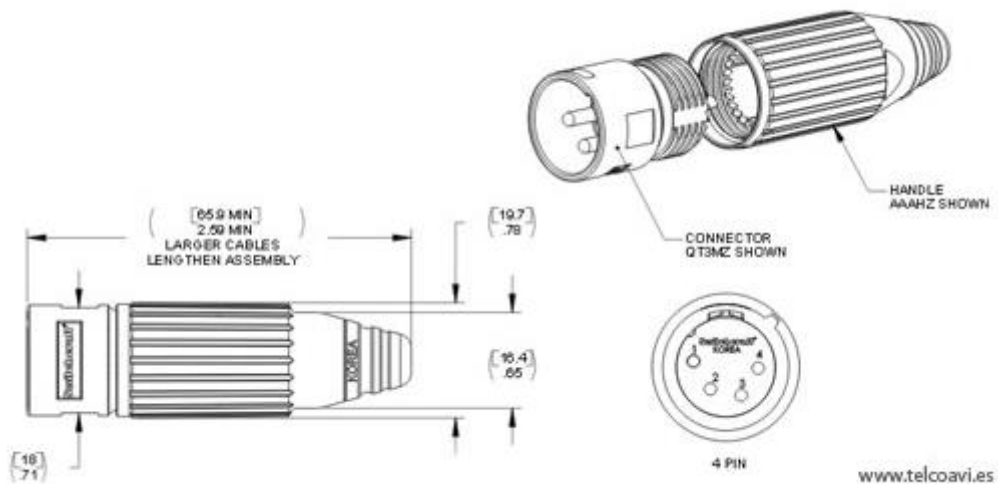


Figura 34 Conector XLR 4 pines (TELCO, 2013)

### 3.12. Cables de sensores

Para la comunicación entre la planta y la etapa de control desde los sensores de distancia y sensor de posicionamiento se debe usar un conector XLR 7 pines (Figura 35) y así tendremos como conectar de manera eficaz y ordenada.



**Figura 35** Conector XLR 7 pines

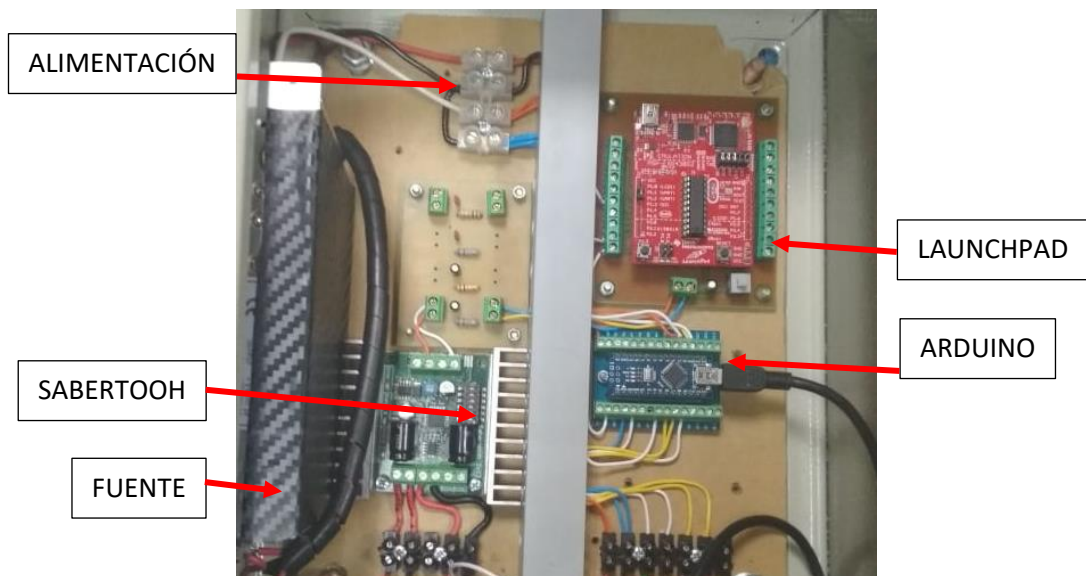
Las conexiones internas del XLR7 se muestran en la tabla 3

Pin	Descripción	Color
1	VCC	Rojo
2	SCL	Azul
3	Signar 1	Café
4	Canal A	Verde
5	Canal B	Blanco
6	GND	Negro
7	No usado	Sin especificar

**Tabla 3:** Conector XLR 7

### 3.13. Conexiones internas de la etapa de control

Dentro del tablero de control posee los diferentes periféricos para el funcionamiento tales como tarjeta Launchpad, circuito de acople del sensor, controlador de motores Sabertooth y una fuente conmutada para la planta balón y viga.



**Figura 36:** Conexiones internas de la etapa de control

### 3.14. Diagrama de conexiones Entradas, salidas Digitales y Analógicas

Las entradas digitales del embebido trabajan a 5V por lo cual se necesita tener driver para controlar y un driver para el acoplamiento de la señal Pwm a un motor, en la Figura 37 se muestra el diagrama se muestra la conexión eléctrica de la parte de control de la planta balón y viga.

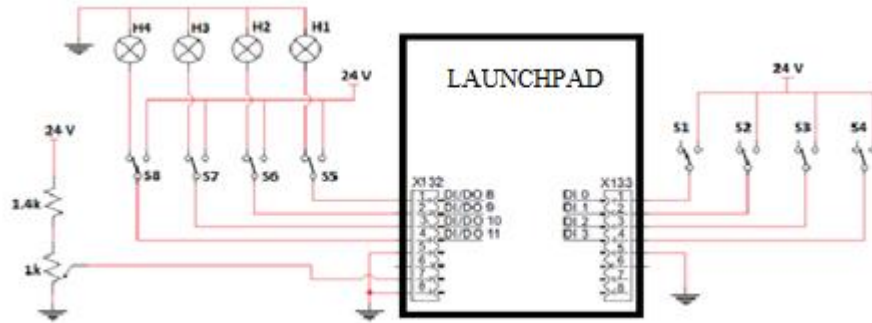


Figura 37: Diagrama de Conexiones a la tarjeta Launchpad

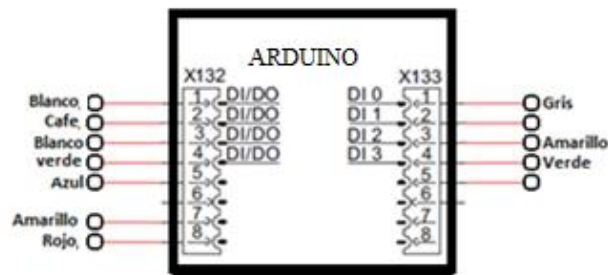


Figura 38: Conexión a los periféricos

### 3.15. Tarjetas de acondicionamiento de señal del sensor

En este proyecto técnico posee dos tarjetas de control en la Figura 39 se muestra la tarjeta para obtener el dato del encoder incremental para controlar el Angulo de inclinación del motor.

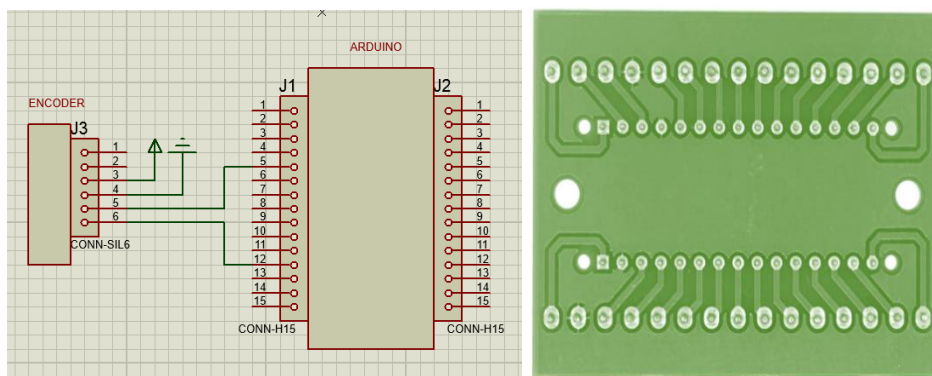
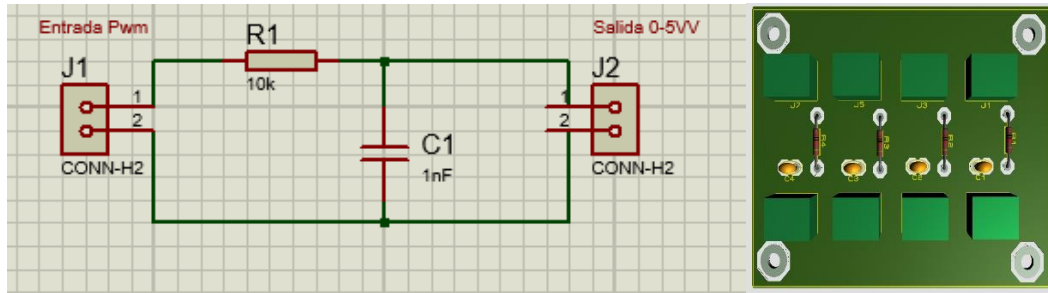


Figura 39 Diagrama de tarjeta acopladora

### 3.16. Tarjetas de acondicionamiento de señal de conversión Pwm a voltaje



En la Figura 40 se muestra la tarjeta que realizara la conversión de ancho de pulso a una señal de voltaje síncrono de 0 a 5 Volts. Par el uso en modo receptor analógico en la tarjeta de control de motor Sabertooth.



**Figura 40:** Tarjeta Acopladora de Pwm a Voltaje

## 4. PRÁCTICAS DE LABORATORIO

### 4.1. Práctica #1

		<b>GUÍA DE PRÁCTICA DE LABORATORIO</b>	
<b>CARRERA:</b> Ingeniería Electrónica		<b>ASIGNATURA:</b> Teoría de Control I	
<b>NRO. PRÁCTICA:</b>	1	<b>TÍTULO PRÁCTICA:</b>	“Programación Energía Ide”
<b>Objetivo:</b>  Adquirir datos del encoder incremental, normalizar y monitoreo  <b>Objetivos específicos:</b> Reconocer los pines del embebido. Realizar una conexión entre la caja de control y planta mediante su respectivo adaptador XLR 7 pines. Configurar del Launchpad M-430 como entrada analógica. Normalizar la entrada análoga y monitoreo de datos en puerto serial. Probar funcionamiento del sensor de posición y diferentes conexiones de la planta.			
<b>INSTRUCCIONES</b>		<b>1. Alimentar la planta de control con 110-220v.</b>	
		<b>2. Verificar que se tienen todos los elementos necesarios para realizar la práctica.</b>	
		<b>3. Seguir la guía paso a paso para realizar la conexión de la parte de control y la planta.</b>	
<b>Nota</b> Antes de hacer la práctica se debe realizar la lectura del tema “3.10,3.11 y3.12 para tener entendimiento de los conectores usados en el proyecto técnico.			
<b>ACTIVIDADES POR DESARROLLAR</b>			
<b>Paso 1.</b>  Verificar que los cables estén conectados con su conector específico y ajustados al módulo de control y a la planta, luego comprobar la conexión entre el cable de alimentación y el cable de comunicación para el sensor como se muestra en la figura 41.			

Proceder a encender el interruptor principal antes de conectar el cable USB xlr 4.



**Figura 41** Conexiones entre planta y caja de control

### Paso 2.

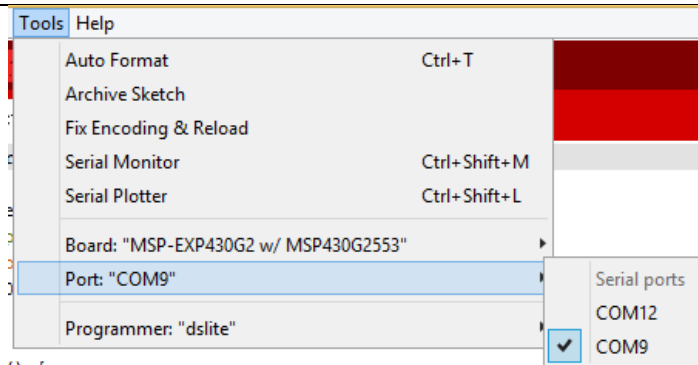
Abrir la interfaz Ide Energía para el embebido Launchpad véase Figura 42.



**Figura 42** Ide Energía de Texas Instrument (MSP430, 2014)

### Paso 3.

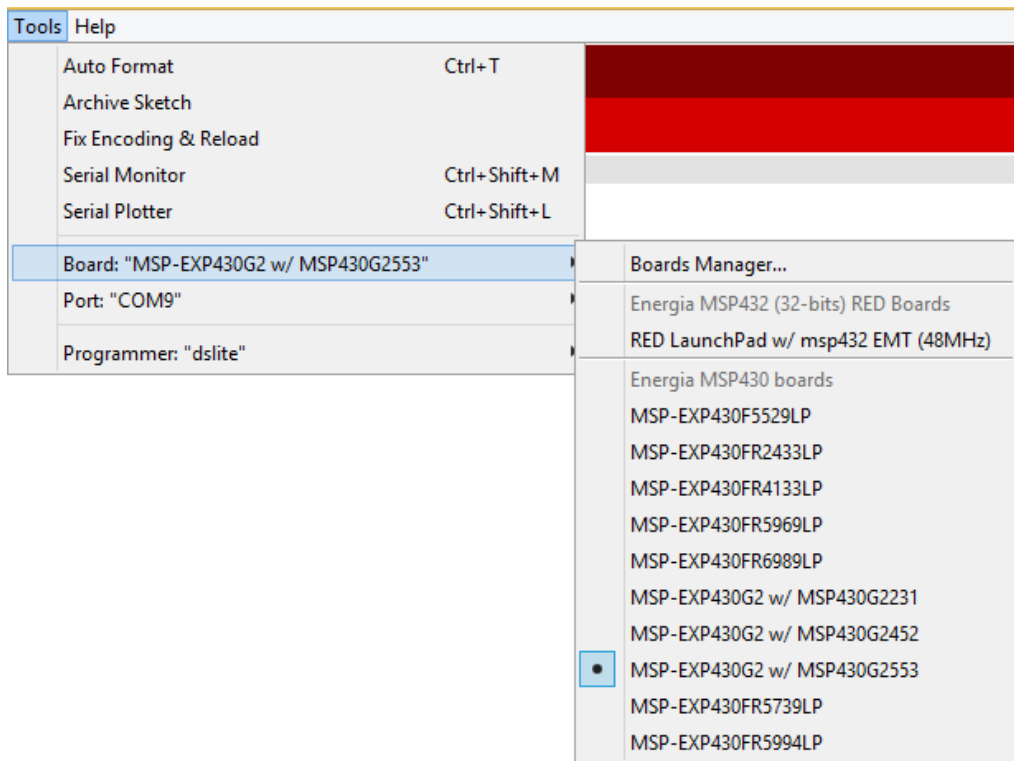
Al ejecutar el ide se debe verificar que el cable USB esté conectado al conector XLR 4 pines como se muestra en la Figura 46. Y que el controlador este correctamente instalado y se visualizara el número del puerto en **Herramientas-Port** véase Figura 43.



**Figura 43** Configuración de puerto Serial

**Paso 4.**

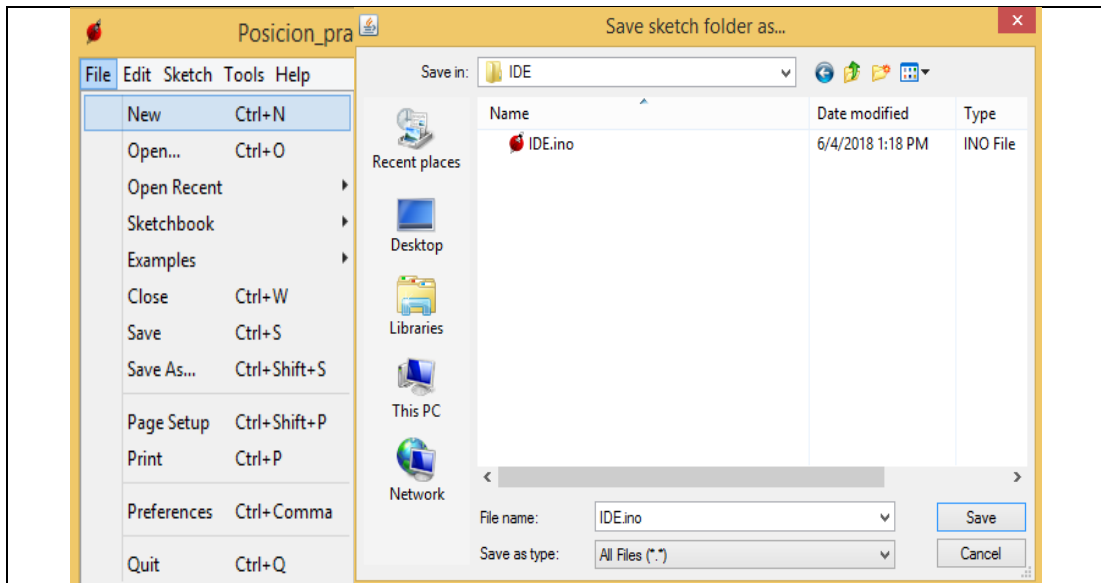
Seleccionar el tipo de tarjeta a usar, la cual es Launchpad M- 430g22553 véase la Figura 44, a velocidad del cristal de Cuarzo es de 16 MHz.



**Figura 44:** Configuración de la tarjeta embebida de Texas Instrument

**Paso 5.**

Crear un nuevo archivo en **File – New** se abre una ventana en la cual se colora el nombre del archivo y se buscara una ubicación para su guardado como se muestra en la Figura 45.



**Figura 45** Creación de archivo nuevo

### Paso 6.

Una vez creado el proyecto se comienza a programar el encoder incremental creando una dirección para la conexión del canal a y b como se muestra en la Figura 46, luego se debe crear un puerto de comunicación serial en el lazo de configuración del ide con una velocidad preestablecida de 9600 baudios para visualizar en la computadora véase Figura 47.

```
#define encodPinA1    2
#define encodPinB1    8
```

**Figura 46** Pines de Conexión del encoder

```
Serial.begin (115200);
```

**Figura 47** Configuración de velocidad de transmisión

### Paso 7.

Se crea un método donde se realiza la lectura del encoder incremental mediante el cambio de estado entre cada canal, como se ve Figura 48

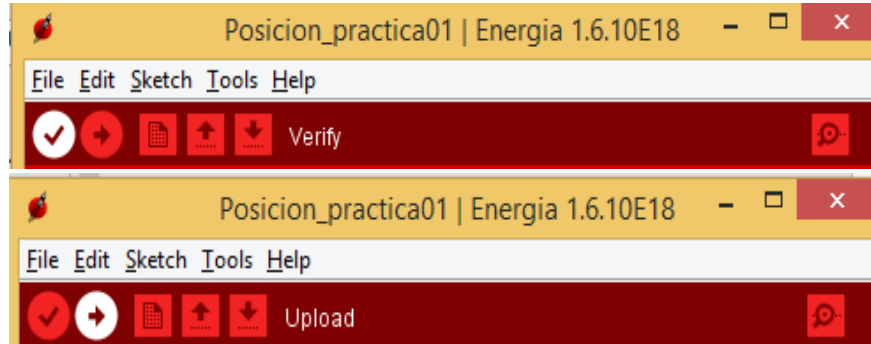
```
void encoder() {
  if (PINB & 0b00000001)  encoderPos++;
  else                    encoderPos--;
}
```

**Figura 48** Bloque del programa para encoder



### Paso 8.

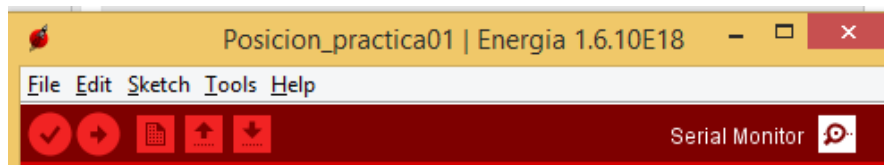
Luego se da **click** da en verificación para comprobar el código y que compile, luego se deberá cargar a la tarjeta como se detalla Figura 49.



**Figura 49** Acción de Compilar y Cargar en la tarjeta electrónica

### Paso 9.

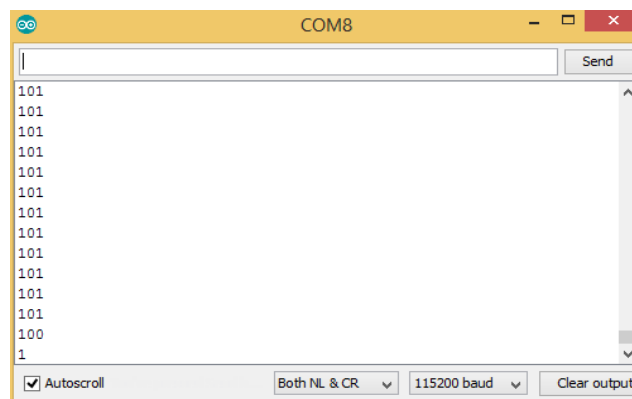
Al subir la programación en la tarjeta se procede a abrir el monitor serial del ide para visualizar el dato adquirido como se muestra en la Figura 50.



**Figura 50** Acción de Monitor Serial en el Ide Energía

### Paso 10.

En el monitor serial se visualiza el valor adquirido del encoder que se usa como sensor de posición. Serial véase en Figura 51.



**Figura 51** Monitor Serial del Ide Energía

### Paso 11.

Al iniciar procede a mover verticalmente la viga para obtener valores de referencia y poder normalizarlos cuando la barra este en posición alta y baja (Figura 52).



**Figura 52** Viga posición alta y baja

### Paso 12.

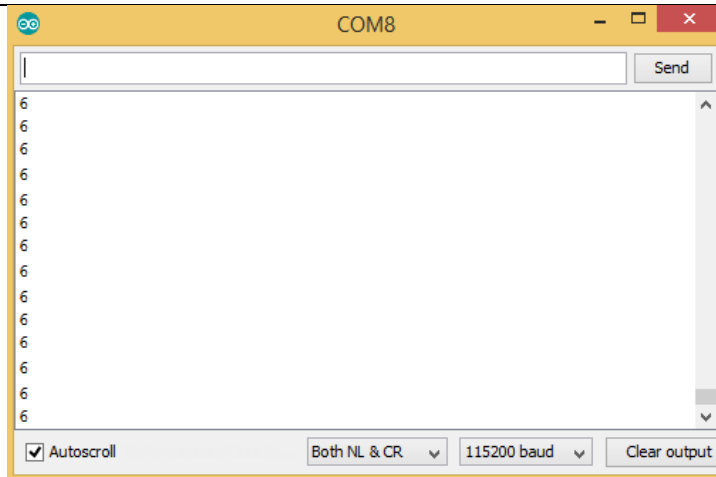
Para controlar la posición en el lazo principal se crea un contador de esta manera el motor gira hacia un sentido durante un tiempo y luego en sentido contrario como se muestra en la Figura 53

```
c++;  
if (c < 500) {  
    analogWrite(6, 170);  
}  
if ((c > 500) && (c < 1000)) {  
    analogWrite(6, 80);  
}  
if (c > 1000) {  
    c = 0;  
}
```

**Figura 53** Parametrización y restricción de rangos

### Paso 13.

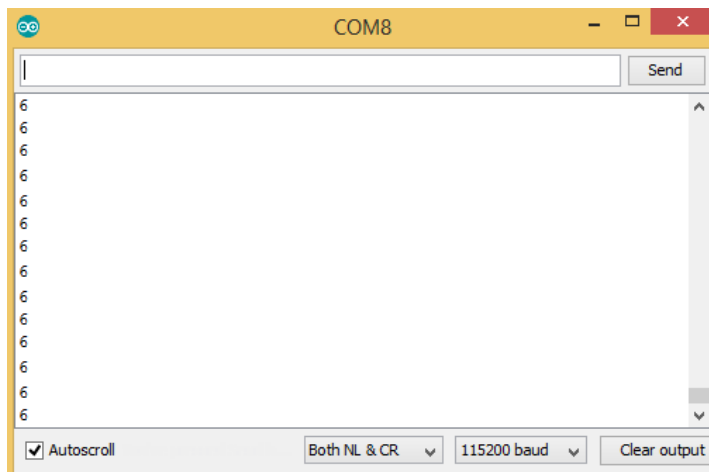
Verificar datos en el puerto serial como se muestra en la Figura 54.



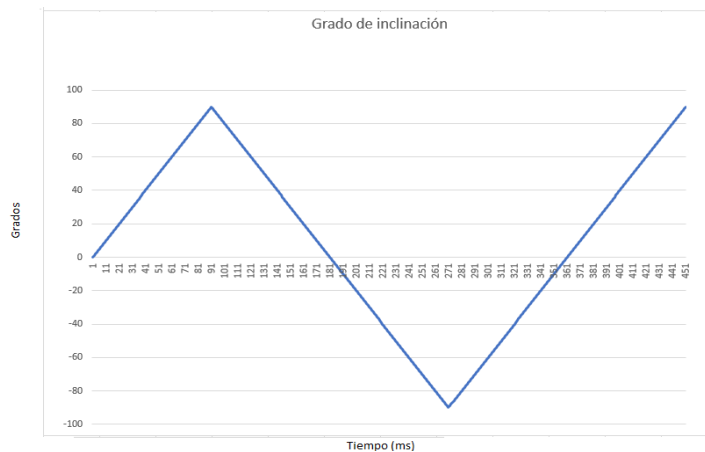
**Figura 54** Datos de encoder

**RESULTADO(S) OBTENIDO(S):**

El valor obtenido en el puerto serial será convertido a voltaje y mostrado en una tabla de Excel mostrando la comparativa ambos.



**Figura 55** Datos del puerto serial






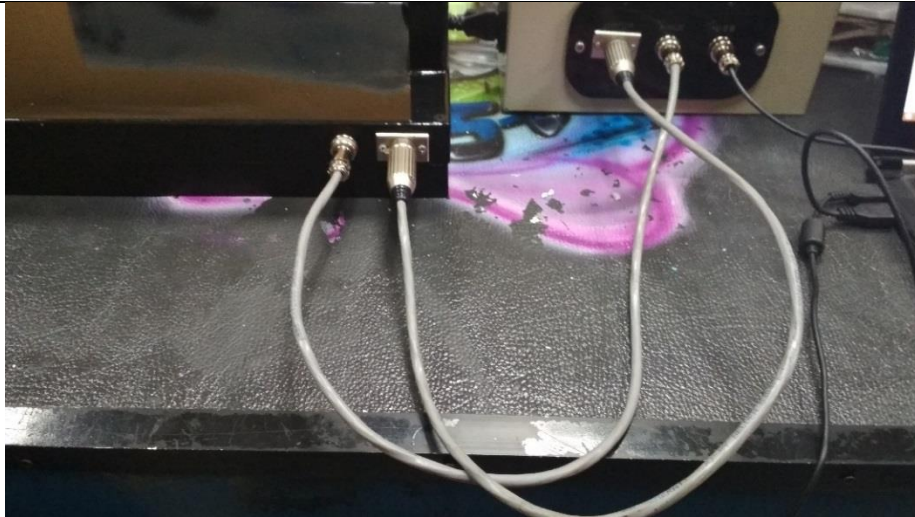
**Figura 56** Comparación Voltaje Vs dato

**CONCLUSIONES:**

1. En esta práctica se aprende a configurar la lectura para encoder incrementales.
2. Se aprende a reconocer las conexiones que tiene la parte de control y la planta.
3. Se expone las grandes ventajas que tiene la normalización de las entradas.

## 4.2. Práctica #2

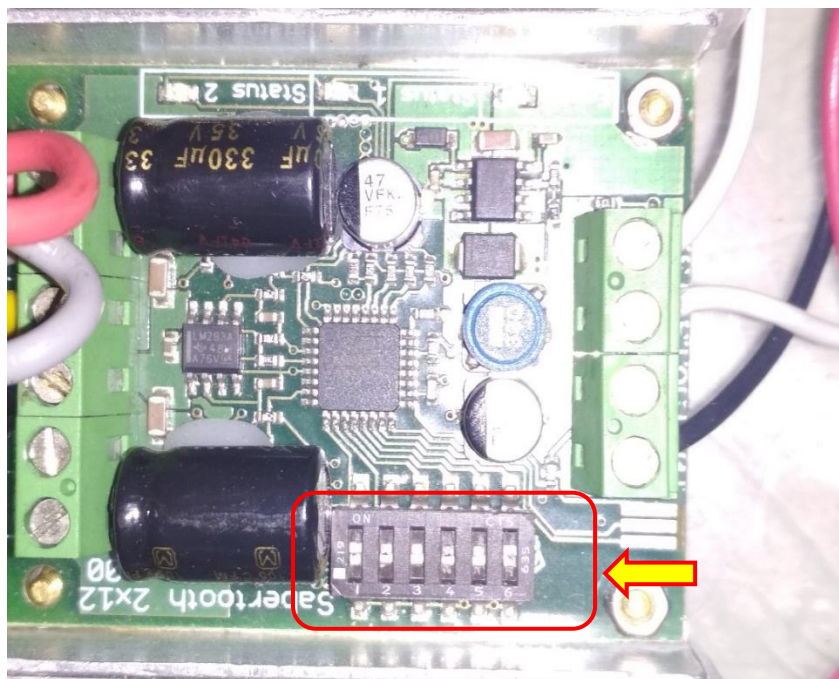
		<b>GUÍA DE PRÁCTICA DE LABORATORIO</b>
<b>CARRERA:</b> Ingeniería Electrónica		<b>ASIGNATURA:</b> Teoría de Control I
<b>NRO. PRÁCTICA:</b>	2	<b>TÍTULO PRÁCTICA:</b> “Parametrizar y configurar la Sabertooth en modo Microcontrolador y control de posición de viga ON/OFF.
<b>Objetivo:</b> Aprender a configurar la Sabertooth en modo microcontrolador e implementar un controlador on/off de posición de la viga.		
<b>Objetivos específicos:</b> Parametrizar un control de posición on/off mediante la tarjeta Launchpad y el controlador de motor Sabertooth usando su modo microcontrolador o ancho de pulso.		
<b>INSTRUCCIONES:</b>		<b>1. Alimentar la parte de control a 110- 220v</b>
		<b>2. Verificar que se tienen todos los elementos necesarios para realizar la práctica</b>
		<b>3. Seguir la guía paso a paso para realizar la conexión de la parte de control y la planta.</b>
<b>ACTIVIDADES POR DESARROLLAR</b>		
<b>Nota</b>		
Antes de hacer la práctica se debe realizar la lectura del tema “Controlador de Motor (Sabertooth)” ubicado en la sección 2.11 del documento general del proyecto de graduación.		
Antes de hacer la práctica se debe realizar la lectura del tema “3.10,3.11 y3.12 para tener entendimiento de los conectores usados en el proyecto técnico.		
<b>Paso 1.</b>		
Conectar la parte de control y la planta véase Figura 57 mediante los conectores XLR de 7 y 3 pines para esto se debe seguir los pasos vistos en la <b>PRÁCTICA #1.</b>		



**Figura 57** Conexión entre Etapa de control y planta

**Paso 2.**

Seleccionar el modo de Microcontrolador en los selectores de la tarjeta Sabertooth, ubicando el selector 1 y 6 en bajo, también la posición 2,3,4,5 en alto véase en la Figura 58.



**Figura 58** Conexión entre Etapa de control y planta

**Paso 3.**

Realizar los pasos 2 hasta el 6 de la **PRÁCTICA #1**.

#### Paso 4.

En la ide se debe declarar las variables a usar, como entradas, y salidas véase Figura 59.

```
int referencia = -40;
const int EntradaPosicion = A0; // Pin analogo
long valorSensado;
long valorNormalizado;
//Variable Almacenamiento
```

**Figura 59** Variables entrada y salida en Práctica 2.

#### Paso 5.

Luego en la parte de configuración del programa se declara el pin de salida que conectara a la Sabertooth enviando un ancho del pulso para romper la inercia del motor, después se declarara la velocidad del puerto serial referente en la Figura 60.

```
void setup() {
  Serial.begin(9600); // Configuracion de Serial
  pinMode(10, OUTPUT); // Pin de salida
}
```

**Figura 60** Configuración Práctica 2

#### Paso 6.

Se debe declarar la lectura análoga de la posición y la normalización del valor para utilizarse en el lazo principal véase Figura 61.

```
valorSensado = analogRead(EntradaPosicion);//entrada analoga
valorSensado = constrain(valorSensado, 580, 950);// restringir a rango de 250 a 950
valorNormalizado = map(valorSensado, 950, 580, -90, 90);//Mapeo de -90 a +90
valorNormalizado = constrain(valorNormalizado, -90, 90);// Restringir a rango -90 a +90
Serial.print(valorSensado);
Serial.print(" ");
Serial.println(valorNormalizado);
```

**Figura 61** Declaración lectura análoga y normalización en Práctica 2

#### Paso 7.

Después se tiene que realizar una comparación con el punto de referencia con el dato de posición y se procederá a realizar una escritura de ancho de pulso por el cual se controlara el motor véase en Figura 62.

```
if (valorNormalizado > referencia) {  
    digitalWrite(10, HIGH); //Bandera en alto  
    delayMicroseconds(1399); //Duracion del pulso  
    digitalWrite(10, LOW); //Bandera en bajo  
}  
else if (valorNormalizado < referencia) {  
    digitalWrite(10, HIGH); //Bandera en alto  
    delayMicroseconds(1599); //Duracion del pulso  
    digitalWrite(10, LOW); //Bandera en bajo  
}  
else {  
    digitalWrite(10, HIGH); //Bandera en alto  
    delayMicroseconds(1500); //Duracion del pulso  
    digitalWrite(10, LOW); //Bandera en bajo  
}  
delay(10);
```

**Figura 62** Declaración de comparación en Práctica 2

### **Paso 8.**

Luego se procede a encender la planta y cargar la tarjeta para su correcto funcionamiento como se vea la Figura 63.

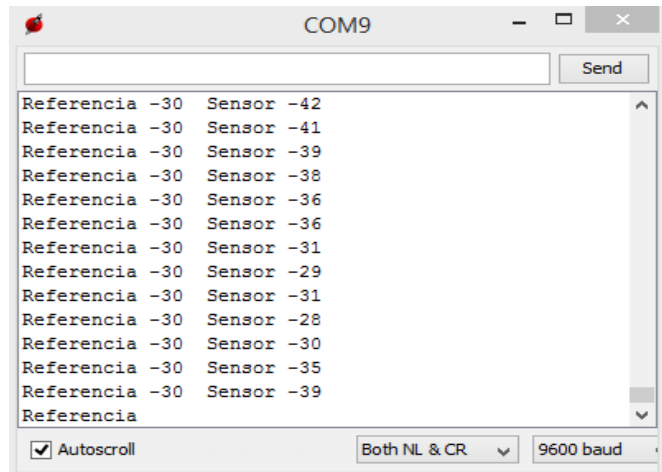


**Figura 63** Planta Balón y Viga 2

### **Paso 9.**

Se abre una ventana del monitor serial para visualizar los datos de entrada y salida de este.





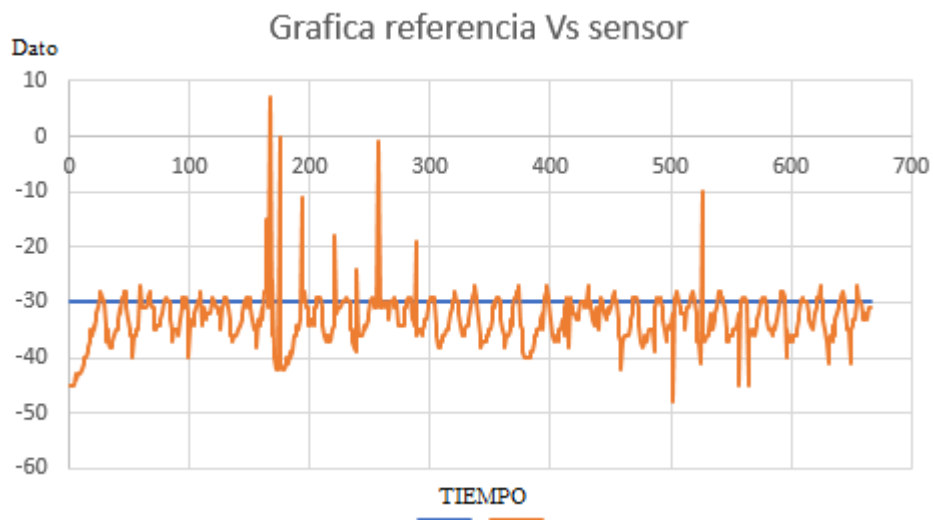
**Figura 64** Resultados del control de posición 2

**Paso 10.**

Luego el motor realiza un control de posición de la viga y se verificara la posición.

**RESULTADO(S) OBTENIDO(S):**

Con los datos que se encuentran en el puerto serial realizar una gráfica comparativa entre el Setpoint y la variable del sensor.





**Figura 65** Gráfica Sensor Vs Referencia

**CONCLUSIONES:**

1. En esta práctica se aprende a configurar un control de posición On / Off usando el embebido Launchpad y un control de motor Sabertooth.
2. Se observa de manera práctica la configuración de la Sabertooth.
3. Mediante las gráficas se puede comprobar la resultante de un control on / off de posición.

#### 4.3. Práctica #3

		<b>GUÍA DE PRÁCTICA DE LABORATORIO</b>
<b>CARRERA:</b> Ingeniería Electrónica		<b>ASIGNATURA:</b> Teoría de Control I
<b>NRO. PRÁCTICA:</b>	<b>3</b>	<b>TÍTULO PRÁCTICA:</b> “Lectura de sensor por medio de ancho de pulsos.
<p><b>Objetivo:</b> Diseñar un programa para la interacción entre 2 microcontroladores usando modulación de ancho de pulso.</p> <p><b>Objetivos específicos:</b> Parametrizar señales de ancho de pulso mediante la función PulseIn. Normalizar variables para aplicaciones de control en sistema embebidos.</p>		
<b>INSTRUCCIONES:</b>		<b>1. Alimentar la parte de control con 110-220v</b>
		<b>2. Verificar que se tienen todos los elementos necesarios para realizar la práctica</b>
		<b>3. Seguir la guía paso a paso para realizar la conexión de la parte de control y la planta.</b>
<p>Antes de hacer la práctica se debe realizar la lectura del tema “3.10,3.11 y3.12 para tener entendimiento de los conectores usados en el proyecto técnico.</p>		
<b>ACTIVIDADES POR DESARROLLAR</b>		
<p><b>Paso 1.</b></p> <p>Como primer paso se debe realizar la conexión entre la parte de control y la planta para esto se deberá seguir los pasos indicados en la <b>PRÁCTICA #1</b> hasta que se realice el paso 5.</p>		
<p><b>Paso 2.</b></p> <p>Una vez que el equipo esté conectado hay que crear en el ide las variables en donde se almacenaran los datos del sensor.</p>		

```
int entrada = 9; // Entrada Fisica del Sensor
unsigned long duration; // variable de lectura
unsigned long dato; // variable normalizada
```

**Figura 66** Declaración Variables-Práctica 3

### Paso 3.

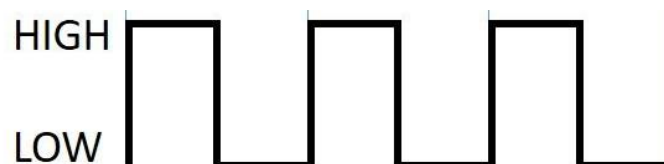
Se declara en el lazo de configuración la velocidad de transmisión y el pin como una entrada digital.

```
void setup()
{
  Serial.begin(9600);
  pinMode(entrada, INPUT);
}
```

**Figura 67** Declaración en lazo configuración-Práctica 3

### Paso 4.

Luego se declara en el lazo principal el método para obtener la señal de ancho de pulso (Figura 68) del sensor mediante el comando PULSEIN. Véase Figura 69.



**Figura 68** Señal Ancho de Pulso-Práctica 3

```
duration = pulseIn(entrada, HIGH);
dato = map(duration, 590, 900, 0, 40);
dato=constrain(dato,0,40);
```

**Figura 69** Lectura de ancho de pulso y normalización Práctica 3

### Paso 5.

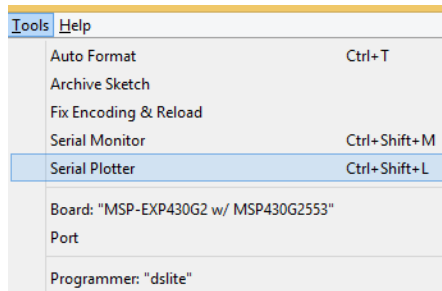
Se designa un puerto serial para visualizar la resultante de la entrada de ancho de pulso. En la Figura 70 se mostrará el código a usarse.

```
void loop()
{
  duration = pulseIn(entrada, HIGH);
  dato = map(duration, 590, 900, 0, 40);
  dato=constrain(dato,0,40);
  Serial.print(duration);
  Serial.print(" ");
  Serial.println(dato);
}
```

**Figura 70** Lectura e impresión por puerto serial de la entrada Práctica 3

### Paso 6.

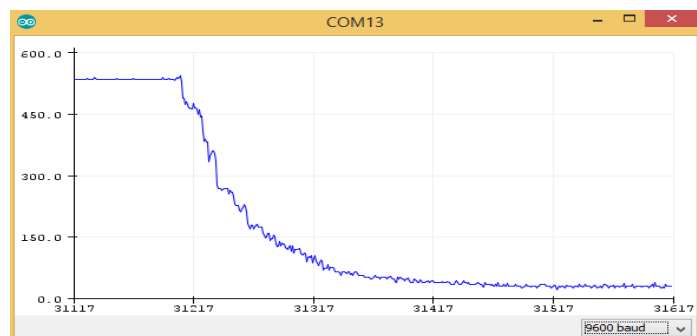
Se abre una ventana donde se ploteará los datos para acceder a esta se tendrá que abrir en **Tools-Serial Plotter** como se visualiza en la Figura 71.



**Figura 71** Serial Plotter en la Práctica 3

### Paso 6.

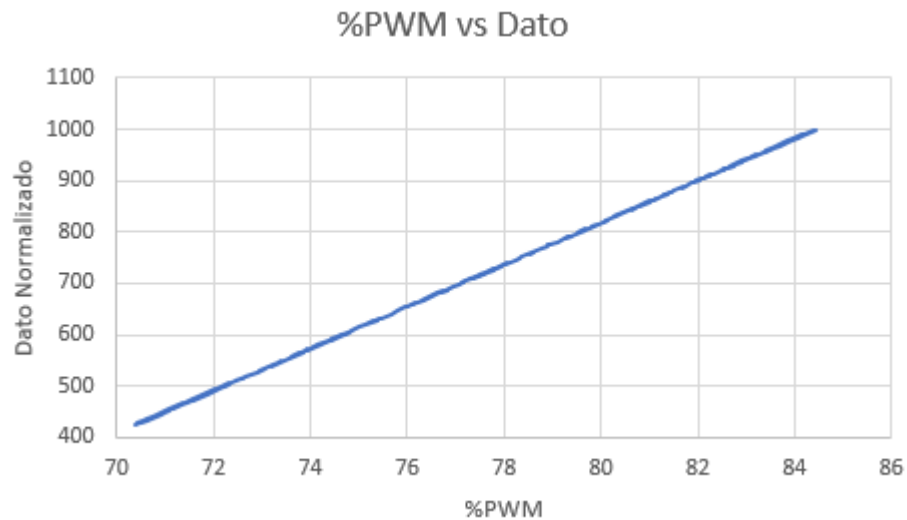
En la Figura 71 se muestra la gráfica correspondiente al dato medido. centímetros.



**Figura 72** Dato en puerto Serial Plotter Práctica 3

## RESULTADO(S) OBTENIDO(S):

Con los datos que se visualizan en el puerto copiar en un Excel y crear una gráfica del dato Pwm vs dato normalizado.





**Figura 73** Dato Pwm vs normalizado Práctica 3

## CONCLUSIONES:

1. Mediante la configuración de diferentes métodos de control logra la adquisición de datos desde otro controlador mediante anchos de pulso.
2. Se podrá probar de manera práctica lo aprendido de manera teórica usando la tarjeta acopladora del módulo i2c para lecturas del sensor.
3. Mediante las rutinas preestablecidas se obtiene un mejor entendimiento para el uso de la tarjeta.

#### 4.4. Práctica #4

		<b>GUÍA DE PRÁCTICA DE LABORATORIO</b>	
<b>CARRERA:</b> Ingeniería Electrónica		<b>ASIGNATURA:</b> Teoría de Control I	
<b>NRO. PRÁCTICA:</b>	4	<b>TÍTULO PRÁCTICA:</b> “Filtro para Eliminación de ruidos.	
<b>Objetivo:</b> Diseñar un programa para la obtención de datos análogas y eliminar ruido mediante un filtro.			
<b>Objetivos específicos:</b>  Mediante un filtro eliminar el ruido de entrada en el dato análogo. Crear rutinas para el entendimiento de filtros.			
<b>INSTRUCCIONES:</b>		<b>1. Alimentar la parte de control con 110-220v</b>	
		<b>2. Verificar que se tienen todos los elementos necesarios para realizar la práctica</b>	
		<b>3. Seguir la guía paso a paso para realizar la conexión de la parte de control y la planta.</b>	
Antes de hacer la práctica se debe realizar la lectura del tema “3.10,3.11 y3.12 para tener entendimiento de los conectores usados en el proyecto técnico.			
<b>ACTIVIDADES POR DESARROLLAR</b>			
<b>Paso 1.</b>  Como primer paso se debe realizar la conexión entre la parte de control y la planta para esto se deberá seguir los pasos indicados en la <b>PRÁCTICA #1</b> hasta que se realice el paso 5.			
<b>Paso 2.</b>  Se declara en el lazo de configuración la velocidad de transmisión y el pin como una entrada digital (Figura 74).			

```

void setup()
{
  Serial.begin(9600);
  pinMode(entrada, INPUT);
}

```

**Figura 74** Declaración en lazo configuración-Práctica 4

### Paso 3.

Se inicia creando el método donde se aplica el filtro que obtendrá una señal promedio del dato medido.

```

int ADC0_promedio(int n)
{
  long suma=0;
  for(int i=0;i<n;i++)
  {
    suma=suma+analogRead(A5);
  }
  return(suma/n);
}

```

**Figura 75** Declaración del filtro-Práctica 4.

### Paso 4.

En la figura 76 se muestra el código para el método e impresión en el puerto serial.

```

//Filtro
int ADC0_promedio(int n)
{
  long suma = 0;
  for (int i = 0; i < n; i++)
  {
    suma = suma + analogRead(A0);
  }
  float adc = suma / n;
  float distancia_cm = 17569.7 * pow(adc, -1.2062);
  // float distancia_cm = 28940.1 * pow(adc, -1.16);
  return (distancia_cm);
}

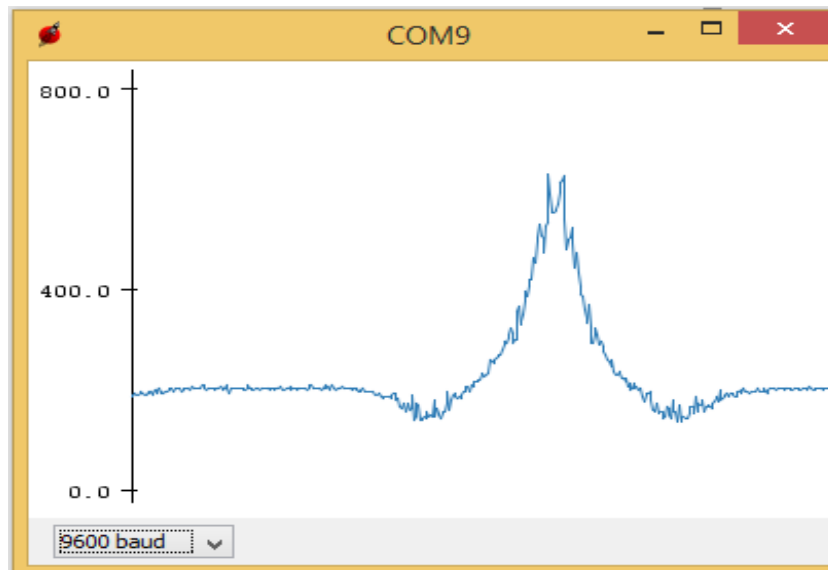
```

**Figura 76** Declaración del lazo principal-Práctica 4.



**Paso 5.**

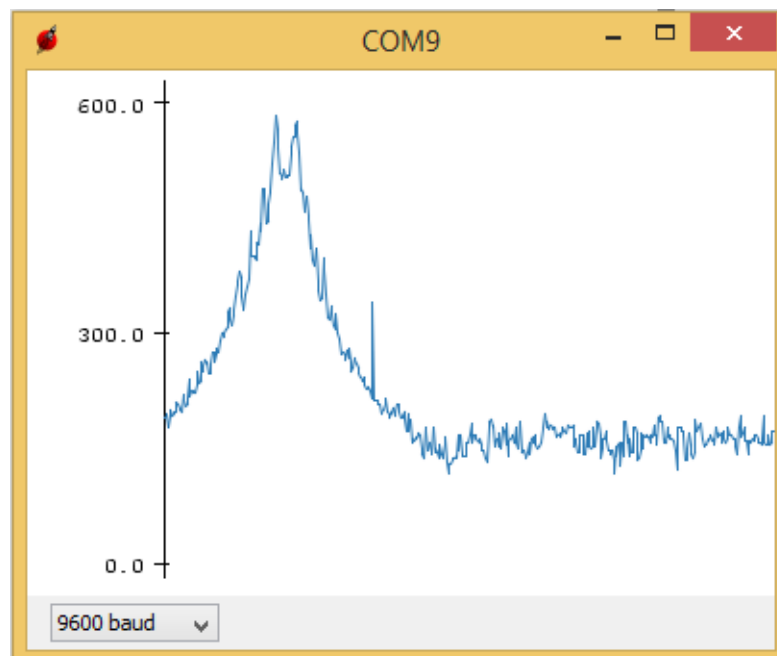
En la figura 77 se abre el Plotter serial para monitoreo de la señal en bits.



**Figura 77:** Plotter en puerto Serial con filtro aplicado-Práctica 4

**Paso 6.**

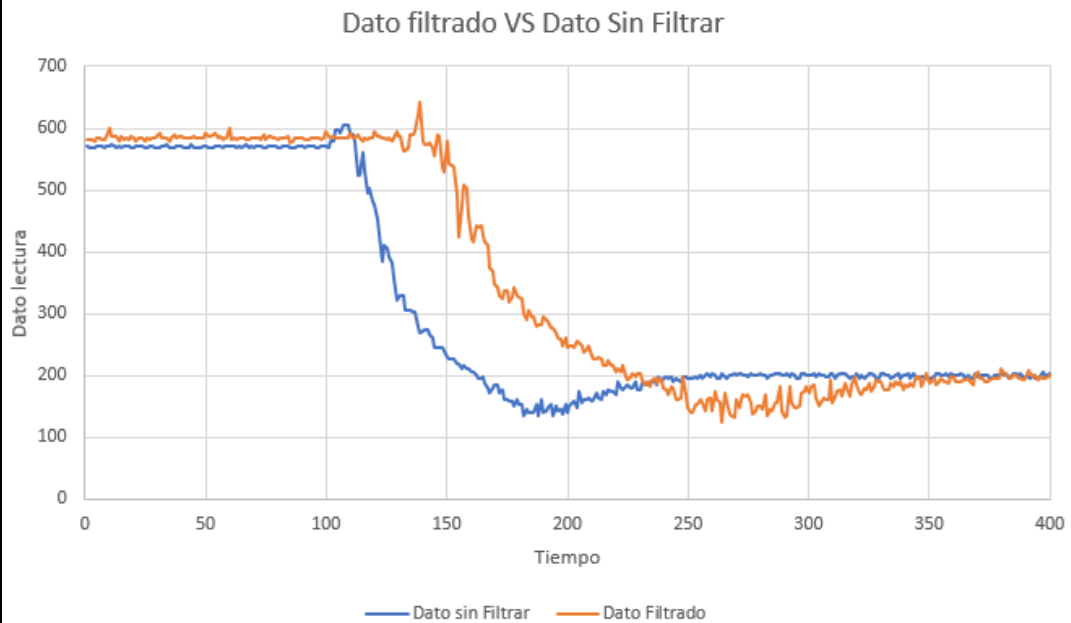
En la figura 78 se monitorea los datos sin la utilización del filtro.



**Figura 78:** Plotter en puerto Serial sin filtro aplicado-Práctica 4

## RESULTADO(S) OBTENIDO(S):

Con los datos que se visualizan en el puerto crear un archivo de Excel en el cual se muestre una comparativa entre el dato filtrado y el dato sin filtrar.





**Figura 79** Dato filtrado vs dato sin filtrar Práctica 4

## CONCLUSIONES:

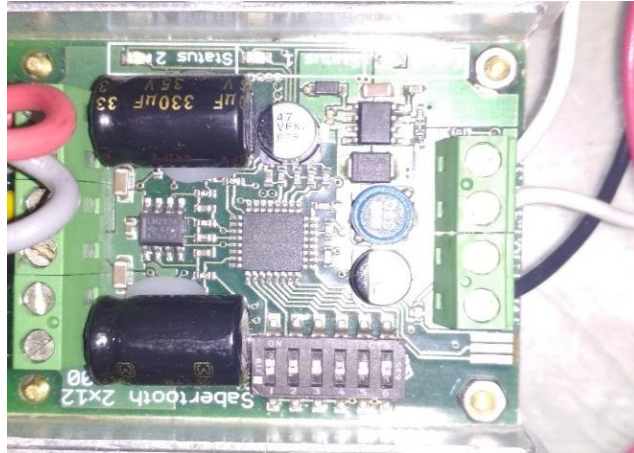
1. En esta práctica se nota la diferencia cuando se aplica un filtrado a una señal y un dato sin filtrar r.
2. Mediante la conversión análoga adquirir un dato para usar en el código.
3. Mediante el uso del filtro se mostrará la eliminación de ruidos en la adquisición de datos.

#### 4.5. Práctica #5

		<b>GUÍA DE PRÁCTICA DE LABORATORIO</b>	
<b>CARRERA:</b> Ingeniería Electrónica		<b>ASIGNATURA:</b> Teoría de Control I	
<b>NRO. PRÁCTICA:</b>	5	<b>TÍTULO PRÁCTICA:</b> “Control PID del ángulo para posición de un motor CC.	
<b>Objetivo:</b>  Implementar un controlador PID para controlar la posición angular de un motor CC.			
<b>Objetivos específicos:</b>  Utilizar encoder incrementales para la lectura de posicionamiento de un motor CC. Diseñar rutinas para el control del método. Usar herramientas de monitoreo tales como Matlab para visualizar datos.			
<b>INSTRUCCIONES:</b>		<b>1. Alimentar la parte de control con 110-220v</b>	
		<b>2. Verificar que se tienen todos los elementos necesarios para realizar la práctica</b>	
		<b>3. Seguir la guía paso a paso para realizar la conexión de la parte de control y la planta.</b>	
Antes de hacer la práctica se debe realizar la lectura del tema “3.10,3.11 y3.12 para tener entendimiento de los conectores usados en el proyecto técnico.			
<b>ACTIVIDADES POR DESARROLLAR</b>			
<b>Paso 1.</b>  Como primer paso se debe realizar la conexión entre la parte de control y la planta para esto se deberá seguir los pasos indicados en la <b>PRÁCTICA #1</b> hasta que se realice el paso 5.			

### Paso 2.

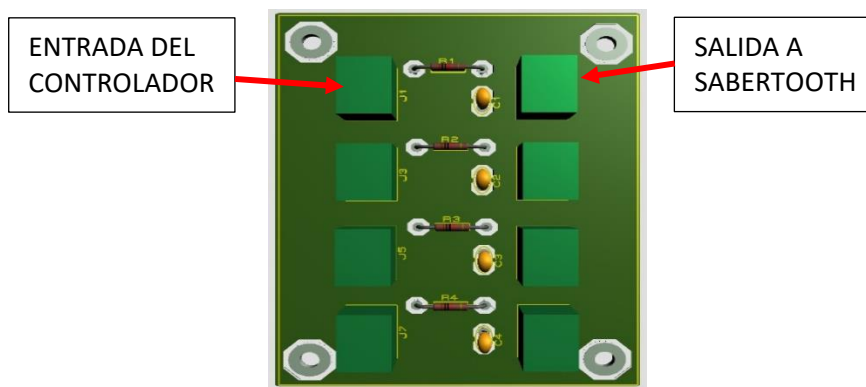
Se debe configurar la Sabertooth en modo análogo como se muestra en la figura 80 seleccionando mediante los selectores de esta.



**Figura 80:** Configuración de Sabertooth en modo análogo Práctica 5

### Paso 3.

Se conecta la placa a una tarjeta de acondicionamiento para convertir la señal Pwm a un valor análogo la cual tendrá como resolución de voltaje de 0 a 5 volts en la salida hacia el controlador Sabertooth. (Figura 81) l.



**Figura 81:** Tarjeta de acoplamiento Pwm a voltaje-Práctica 5

### Paso 4.

Abrir el ide de programación y declarar las librerías a usarse en el código véase en la Figura 82.

```
Practica_05 §  
1 #include <PinChangeInt.h>  
2 #include <PID_v1.h>
```

**Figura 82** Declaración de librerías-Práctica 5

### Paso 5.

Abrir el ide de programación y declarar las variables para el pid de posición referente a la Figura 83 utilizando el encoder incremental como se muestra en la Figura 83.

```
//ENTRADAS DEL ENCODER
#define encodPinA1    2
#define encodPinB1    8
```

**Figura 83** Declaración de variables del encoder canal A-B-Práctica 5

### Paso 6.

En la figura 84 se declara las variables a usarse en el código para su óptimo funcionamiento tales como el proporcional, derivativo, integral, almacenamiento para las lecturas del encoder-y demás sentencias del código.

```
#include <PinChangeInt.h>
#include <PID_v1.h>
//ENTRADAS DEL ENCODER
#define encodPinA1    2
#define encodPinB1    8
//Salida de la saber
int salida = 6;// Salida PwM
int pin = 7;
int duty;
long temp;
unsigned long duration;
volatile long encoderPos = 0;
//VARIABLES VIGA
double kp = 11 , ki = 35, kd = 0.1;
PID VigaPID(&input, &output, &setpoint, kp, ki, kd, DIRECT);
```

**Figura 84** Declaración de configuración - Práctica 5

### Paso 7.

En la configuración se declara el pin del motor como salida y declarar las entradas del encoder canal A-B como se aprecia en la Figura 85.

```
void setup() {
  //Salida del motor
  pinMode(salida, OUTPUT);
  //Encoder
  pinMode(encodPinA1, INPUT_PULLUP); // quadrature encoder input A
  pinMode(encodPinB1, INPUT_PULLUP); // quadrature encoder input B
  attachInterrupt(0, encoder, FALLING); // update encoder position
}
```

**Figura 85** Declaración de configuración - Práctica 5

### Paso 8.

En la parte de la configuración se deberá declarar los límites del pid (Figura 86) para tener sus salidas en un margen aceptable de inclinación para la esfera dichos limites en la salida del motor serán +-255 para el control de cambios de giro del motor.

```
VigaPID.SetOutputLimits(-255, 255);
```

**Figura 86** Declaraciones de Pid de límites-Práctica 5

### Paso 9.

La siguiente parte de la configuración se deberá declarar el método del pid manual o automático, también el tiempo de muestreo para cada iteración del pid como se ve en la Figura 87.

```
VigaPID.SetMode(AUTOMATIC);  
VigaPID.SetSampleTime(1);
```

**Figura 87** Configuración Modo y tiempo Muestreo -Práctica 5

### Paso 10.

La declaración final del método de configuración se tendrá que configurar el modo serial a una velocidad de 9600 varios como se muestra en la Figura 88

```
void setup() {  
  //Salida del motor  
  pinMode(salida, OUTPUT);  
  //Encoder  
  pinMode(encodPinA1, INPUT_PULLUP);  
  pinMode(encodPinB1, INPUT_PULLUP);  
  attachInterrupt(0, encoder, FALLING);  
  //entrada ancho de pulso  
  // PID VIGA  
  VigaPID.SetMode(AUTOMATIC);  
  VigaPID.SetSampleTime(1);  
  VigaPID.SetOutputLimits(-255, 255);  
  Serial.begin(9600);  
}
```

**Figura 88** Método configuración Práctica 5

### Paso 11.

La declaración final del método de configuración se tendrá que configurar el modo serial a 9600 baudios como se muestra en la Figura 89

```
Serial.begin(9600);
```

**Figura 89** Método configuración Práctica 5

### Paso 12

Se declara el método donde se realiza la lectura del encoder mediante interrupción, mediante el cual incrementara o decrementara el valor de encoder para una lectura de la ubicación del motor. (Figura 90)

```
void encoder() {  
  if (PINB & 0b00000001) encoderPos++;  
  else encoderPos--;  
}
```

**Figura 90** Método para lectura encoder Práctica 5

### Paso 13.

En el bucle de repetición o método principal se declarará la referencia a donde sé que desplace el motor (Figura 91)

```
void loop() {  
  
  setpoint = 40;
```

**Figura 91** Declaración de la referencia Práctica 5

### Paso 14.

En la Figura 92 se declarará cual será el bucle de repetición o método principal se declarará la entrada del proceso PID y se dará inicio al controlador para que realice las operaciones pertinentes y calcule la salida (Figura 91)

```
void loop() {  
  
  setpoint = 40;  
  input = encoderPos ;  
  VigaPID.Compute ();|
```

**Figura 92** Declaración de la entrada del PID-Práctica 5

### Paso 15.

Se agrega una función de saturación (Figura 93) cuya variable será **dox** para evitar una zona muerta en el motor y romper el punto de inercia en el mismo. Logrando que el control no tenga perdidas y óptimo funcionamiento.

```

duty = map(output, -255, 255, 0, 255);
if (duty > 130) {
    duty = duty + 30;//180
    duty = constrain(duty, 0, 255);
}
else if (duty < 120) {
    duty = duty - 30;//60
    duty = constrain(duty, 0, 255);
}

```

**Figura 93** Función de saturación Práctica 5

#### Paso 16.

Se agrega la función de saturación cuya variable será **doy** para evitar una zona muerta en el motor y romper el punto de inercia en el mismo. Logrando que el control no tenga perdidas y óptimo funcionamiento (Figura 94).

```

duty = map(output, -255, 255, 0, 255);
if (duty > 130) {    duty = duty + 30;//180
    duty = constrain(duty, 0, 255); }
else if (duty < 120) {    duty = duty - 30;//60
    duty = constrain(duty, 0, 255); }

```

**Figura 94** Función de saturación Práctica 5

#### Paso 17.

Se configura un bloque de monitoreo1(Figura 95) para visualizar los datos en el puerto serial mediante el cual se comunicará con **Matlab** para la visualización de los datos (Figura 96).

```

void monitoreo1() {
    Serial.print(setpoint);
    Serial.print("#");
    Serial.print(input);
    Serial.print("$");
    Serial.println(duty);

}

```

**Figura 95** Función de monitoreo1 Práctica 5



```

void loop() {
  setpoint = 100;
  input = encoderPos ;
  VigaPID.Compute ();

  duty = map(output, -255, 255, 0, 255);

  if (duty > 130) {
    duty = duty + 20;//180
    duty = constrain(duty, 0, 255);
  }
  else if (duty < 120) {
    duty = duty - 20;//60
    duty = constrain(duty, 0, 255);
  }
  analogWrite(salida, duty);
  monitoreo1();
}

```

**Figura 96** Función de monitoreo1 Práctica 5

### Paso 18.

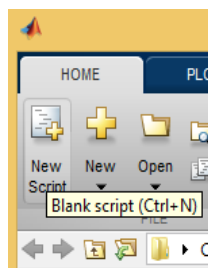
Abrir el programa **Matlab** (Figura 97) para poder visualizar una gráfica en donde se muestren la entrada, salida y referencia del controlador. agrega una función de saturación cuya variable será **duty** para evitar una zona muerta en el motor y romper el punto de inercia en el mismo. Logrando que el control no tenga pérdidas y óptimo funcionamiento (Figura 94).



**Figura 97** Matlab-Práctica 5 (MathWorks, 1984)

### Paso 19.

Crear un nuevo script de **Matlab** (Figura 98), donde se escribirá el código.



**Figura 98** Creación de Script Matlab-Práctica 5

### Paso 20.

En el script se realiza la limpieza del puerto virtual para evitar algún mal funcionamiento (Figura 99).

```
§LIMPIEZA DEL PUERTO
delete(instrfind({'Port'},{'COM4'}));
```

**Figura 99** Limpia virtualmente el Puerto USB -Práctica 5

### Paso 21.

Los datos se deberán adquirir mediante un puerto COM conectado al CPU, en el script se crea la configuración para tener acceso al USB permitiendo a **Matlab**, se crea el objeto del puerto. (Figura 100)

```
§CREAR OBJETO
s = serial('COM5','BaudRate',9600,'Terminator','CR/LF');
warning('off','MATLAB:serial:fscanf:unsuccessfulRead');
```

**Figura 100** Creación de objeto del puerto serial-Práctica 5

### Paso 22.

Al crear el Objeto al ejecutar el programa es necesario abrir el objeto (Figura 101).

```
§LIMPIEZA DEL PUERTO
delete(instrfind({'Port'},{'COM4'}));
```

**Figura 101** Limpia virtualmente el Puerto USB -Práctica 5

### Paso 23.

Se debe acondicionar el número de muestras que se tomara para la gráfica optimizando la lectura para visualizar un dato en el serial (Figura 102).

```
§ parámetros de medidas
tmax = 1000; § Muestras
```

**Figura 102** Configurar números de muestras-Práctica 5

### Paso 24.

Se configura la ventana donde se graficar (Figura 102) los datos tales el nombre de los vectores, los limites en el eje x, y, el titulo en cada vector y

se activa el guardado del dato anterior para mostrar todos los vectores de la lectura.

```
% Configurar la Figura
f = figure('Name','Datos Seriales');
a = axes('XLim',[0 tmax],'YLim',[-256 256]);
```

**Figura 103** Configuración de ventana grafica-Práctica 5

### Paso 25.

Se almacena la cantidad de vectores a almacenar que serán el número de muestras, luego se debe crear un bucle donde se debe realizar cada iteración, se indica los colores de cada señal de entrada y a sus respectivos ejes se designan nombres (Figura 104).

```
l1 = line(nan,nan,'Color','r','LineWidth',2);
l2 = line(nan,nan,'Color','b','LineWidth',2);
l3 = line(nan,nan,'Color','y','LineWidth',2);

xlabel('S')
ylabel('Variables')
grid on
hold on
```

**Figura 104** Configurar los vectores de la ventana grafica-Práctica 5

### Paso 26.

Se declara las variables para el bucle de lectura del dato y almacenado de vectores (Figura 105)

```
v1 = zeros(1,tmax);
v2 = zeros(1,tmax);
v3 = zeros(1,tmax);
i = 1;
t = 0;
```

**Figura 105** configuración de ventana grafica-Práctica 5

### Paso 27.

Declaración del ciclo de repetición cuando  $t < t_{max}$  de esta manera se ejecutará dependiendo el tiempo total de muestreo que se empleó. (Figura 106)

```

tic
while t<tmax
    t = toc;
    a = fscanf(s,'%d,%d');
    v1(i)=a(1);    v2(i)=a(2);
    x = linspace(0,i/rate,i);
    set(l1,'YData',v1(1:i),'XData',x);
    set(l2,'YData',v2(1:i),'XData',x);
    drawnow
    i = i+1;
end

```

Figura 106 configuración de ventana grafica-Práctica 5

### RESULTADO(S) OBTENIDO(S):

Utilizando el Paso 16 al 24 de la práctica 5, graficar el control PID aplicado al posicionamiento angular con la variable de entrada y su referencia

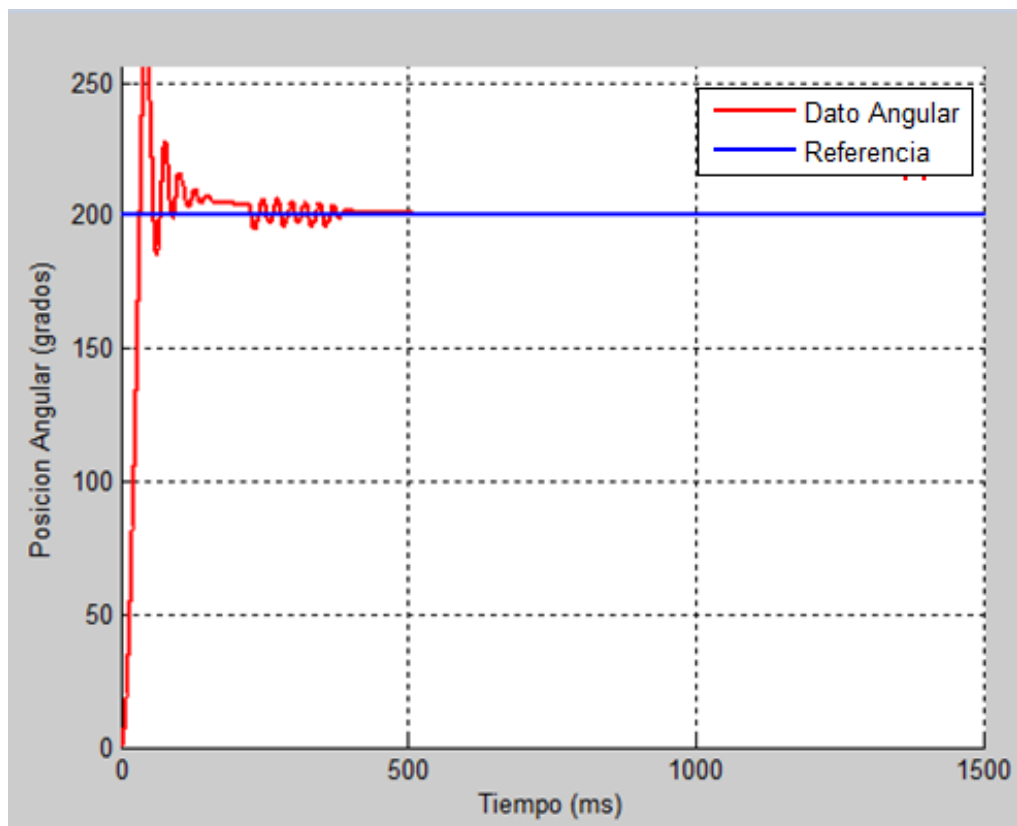
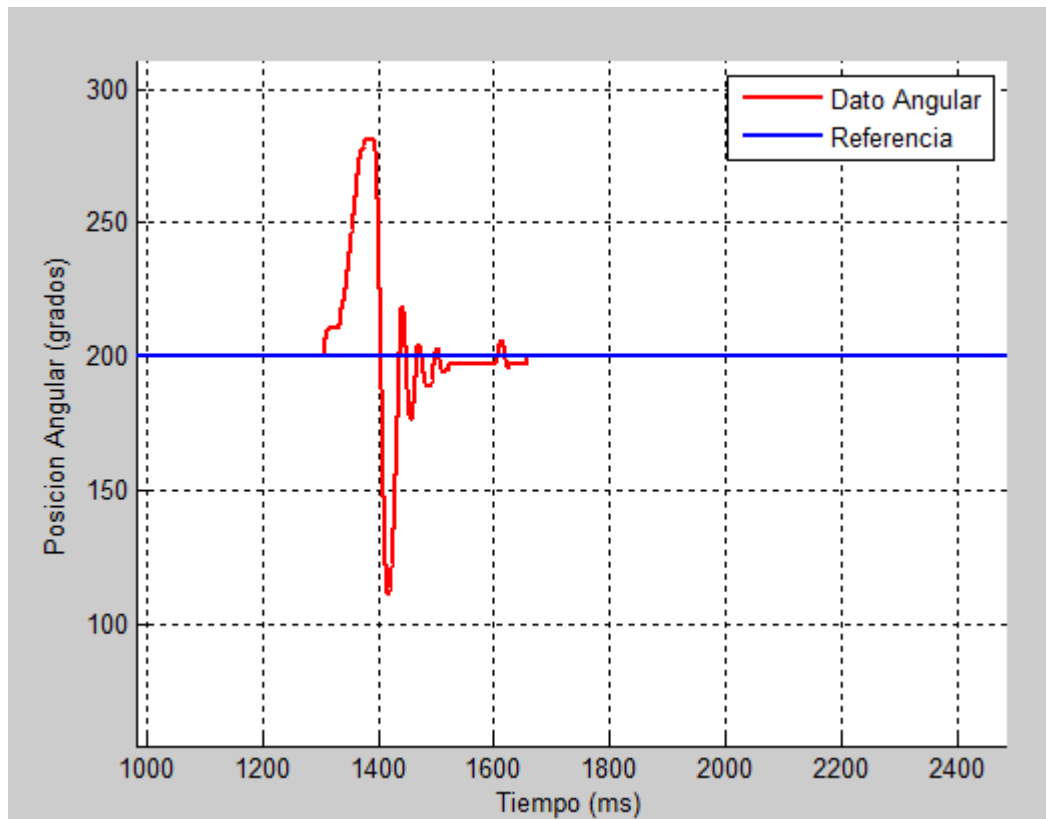


Figura 107 Datos de referencia y variable de entrada Práctica 5

Graficar la respuesta del controlador PID del ángulo aplicando una perturbación externa al motor.





**Figura 108** Datos de referencia y variable de entrada aplicando perturbación Práctica 5

### CONCLUSIONES:

1. En esta práctica se visualizó la estabilidad del control aplicando perturbaciones al motor CC.
2. Mediante la conversión los anchos de pulso de los canales del encoder se logró hacer lectura de posicionamiento.
3. Mediante el uso de la herramienta Matlab se logra obtener mejores visualizaciones de las gráficas.

#### 4.6. Práctica #6

		<b>GUÍA DE PRÁCTICA DE LABORATORIO</b>
<b>CARRERA:</b> Ingeniería Electrónica		<b>ASIGNATURA:</b> Teoría de Control I
<b>NRO. PRÁCTICA:</b>	6	<b>TÍTULO PRÁCTICA:</b> “Control Pid de posición del balón y viga.
<p><b>Objetivo:</b> Diseñar un controlador pid para el control del Posicionamiento.</p> <p><b>Objetivos específicos:</b></p> <p>Crear rutinas para la instrumentación y control de un balón sobre la viga. Creación de métodos para eliminación de ruidos y monitoreo en el puerto serial. Usar herramientas de monitoreo tales como Matlab para visualizar datos.</p>		
<b>INSTRUCCIONES:</b>		<b>1. Alimentar la parte de control con 110-220v</b>
		<b>2. Verificar que se tienen todos los elementos necesarios para realizar la práctica</b>
		<b>3. Seguir la guía paso a paso para realizar la conexión de la parte de control y la planta.</b>
<p>Antes de hacer la práctica se debe realizar la lectura del tema “3.10,3.11 y3.12 para tener entendimiento de los conectores usados en el proyecto técnico.</p>		
<b>ACTIVIDADES POR DESARROLLAR</b>		
<p><b>Paso 1.</b></p> <p>Como primer paso se debe realizar la conexión entre la parte de control y la planta para esto se deberá seguir los pasos indicados en la <b>PRÁCTICA #1</b> hasta que se realice el paso 5.</p>		
<p><b>Paso 2.</b></p> <p>Realizar los pasos 2 y 3 de la <b>PRÁCTICA #5.</b></p>		

### Paso 3.

Declarar librerías a usarse y definir las constantes de entrada para las señales de los encoder A y B, como pin de salida a la Sabertooth (Figura 109).

```
#include <PinChangeInt.h>
#include <PID_v1.h>

//ENTRADAS DEL ENCODER
#define encodPinA1 2
#define encodPinB1 8
volatile long encoderPos = 0; //valor almacenamiento de encoder
//Salida de la saber
int salida = 6;
int duty;
```

Figura 109 Declaración de variables Práctica 6

### Paso 4.

Declarar las variables de control para cada PID Angulo (Proporcional, Integral, Derivativo, Referencia, Salida y Entrada del sistema) y para PID Posición (Proporcional, Integral, Derivativo, Referencia, Salida y Entrada del sistema) véase (Figura 110), para más detalle del método de sintonización revisar **PRÁCTICA #7**.

```
double input2 = 0, output2 = 0, setpoint2 = 0; //Valores del PID Posicion Balon
double kp2 = 14.7, ki2 = 1.25, kd2 = 0.025; //Valores del PID Posicion Balon
double input = 0, output = 0, setpoint = 0; //Valores del PID Angulo Viga
double kp = 11, ki = 38, kd = 0.09; //Valores del PID Angulo Viga
PID VigaPID(&input, &output, &setpoint, kp, ki, kd, DIRECT);
PID PosicionPID(&input2, &output2, &setpoint2, kp2, ki2, kd2, DIRECT);
```

Figura 110 Declaración de variables Pid Angulo y Pid Posición Práctica 6

### Paso 5.

En el método de configuración declaramos un retorno de tiempo antes del inicio del código para evitar cualquier activación no deseada. (Figura 111)

```
void setup() {
  //Salida del motor
  pinMode(salida, OUTPUT);
  analogWrite(salida, 130);
  delay(5000); //Espera antes de inicio
```

Figura 111 Declaración de variables Práctica 6

### Paso 6.

Se configura las entradas del encoder como **INPUT\_PULLUP** para lecturas rápidas, como así su interrupción correspondiente refiérase a Figura 112.

```
//Encoder
pinMode(encodPinA1, INPUT_PULLUP);
pinMode(encodPinB1, INPUT_PULLUP);
attachInterrupt(0, encoder, FALLING);
```

**Figura 112** Declaración de variables Práctica 6

### Paso 7.

Configurar los modos del pid, tiempo de iteración entre cada calculo y los límites de salida máximos y mínimos. (Figura 113)

```
// PID VIGA
VigaPID.SetMode(AUTOMATIC);
VigaPID.SetSampleTime(1);
VigaPID.SetOutputLimits(-255, 255);
// PID POSICION
PosicionPID.SetMode(AUTOMATIC);
PosicionPID.SetSampleTime(1);
PosicionPID.SetOutputLimits(-75, 75);
```

**Figura 113** Configuración de Pid de ángulo y posición -Práctica 6

### Paso 8.

Declarar la velocidad de transmisión serial y un bucle **while** para cuando este activo el puerto. (Figura 114)

```
Serial.begin(9600);
while (!Serial) {
    ; // Espera de conexión USB
}
```

**Figura 114** Declaración de velocidad transmisión-Práctica 6

### Paso 9.

Se declara el método de lectura del encoder mediante la interrupción por cambio de flanco, realiza en cada iteración de tiempo un muestreo a los pines del encoder, y a su vez incrementando o decreciendo, dependiendo el orden de la secuencia de giro. (Figura 115)



```

void encoder() {
    if (PINB & 0b00000001)    encoderPos++;
    else                      encoderPos--;
}

```

**Figura 115** Método de lectura del encoder-Práctica 6

### Paso 10.

Declarar el método de acondicionamiento y filtrado de la señal de entrada para el sensor de distancia. (Figura 116)

```

//Filtro
int ADC0_promedio(int n)
{
    long suma = 0;
    for (int i = 0; i < n; i++)
    {
        suma = suma + analogRead(A0);
    }
    float adc = suma / n;
    float distancia_cm = 17569.7 * pow(adc, -1.2062);
    // float distancia_cm = 28940.1 * pow(adc, -1.16);
    return (distancia_cm);
}

```

**Figura 116** Declaración de método para filtrado y lectura-Práctica 6

### Paso 11.

En el método principal se asigna la distancia de referencia con el cual actuará el controlador, como su respectiva retroalimentación de entrada que será la lectura del sensor de distancia., se manda a realizar los cálculos del pid posición (Figura 117)

```

void loop() {
    setpoint2 = 22;
    input2 = ADC0_promedio(20);
}

```

**Figura 117** Declaración de referencia y entrada del pid -Práctica 6

### Paso 12.

Se asigna a la entrada del pid del ángulo la salida del pid de posición ya que este tendrá dependencia de este y se ejecutará el cálculo de la viga (Figura 118).

```

void loop() {
  setpoint2 = 22;
  input2 = ADC0_promedio(20);
  PosicionPID.Compute();
  setpoint = output2;
  input = encoderPos;
  VigaPID.Compute();
}

```

**Figura 118** Declaración de los pid de ángulo y posición -Práctica 6

### Paso 13.

Se agrega un bloque de compensación y saturación al código para evitar que el motor conste con puntos muertos y de estar manera romper la inercia. (Figura 119).

```

duty = map(output, -255, 255, 0, 255);

if (duty > 130) {
  duty = duty + 30;//180
  duty = constrain(duty, 0, 255);
}
else if (duty < 120) {
  duty = duty - 30;//60
  duty = constrain(duty, 0, 255);
}

```

**Figura 119** Compensación y saturación a la salida del controlador  
Práctica 6

### Paso 14.

Se asigna el dato a una salida analógica para controlar la posición del motor en relación con la posición del balón (Figura 120).

```

analogWrite(salida, duty);

```

**Figura 120** Salida analógica para control del motor Práctica 6

### Paso 15.

Se crea un método para la visualización de los datos de entrada y salida de cada controlador pid (Figura 121).

```

void monitoreo2() {
  Serial.print((int)input);
  Serial.print(",");
  Serial.println((int)output2);
}

```

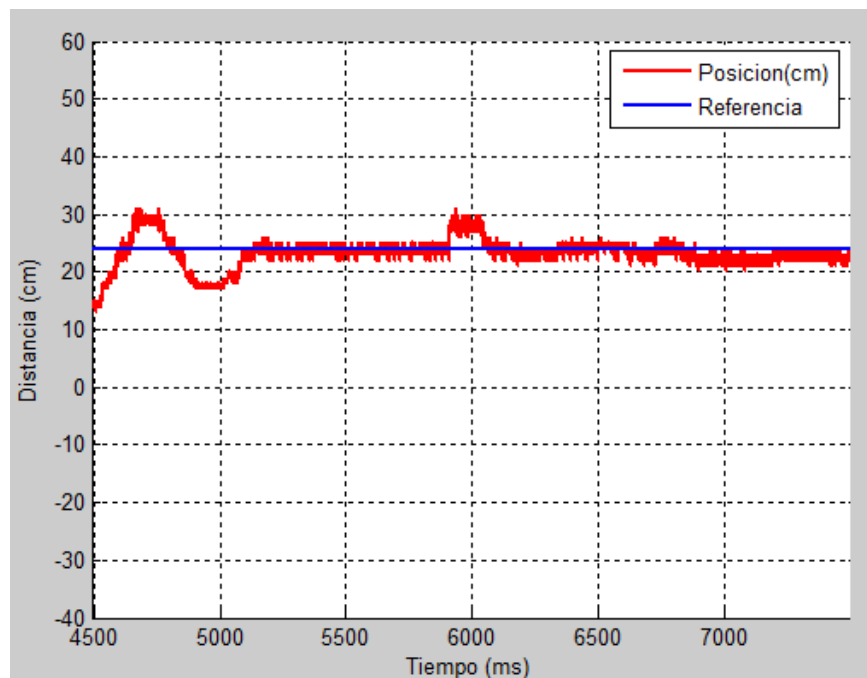
**Figura 121** Método de monitoreo Práctica 6

### Paso 16.

Seguir los pasos 16 al 24 de la **PRÁCTICA #5** para configuración del script de Matlab y poder visualizar el dato de entrada y la referencia del control de posición del balón.

### RESULTADO(S) OBTENIDO(S):

Utilizando el Paso 16 al 24 de la Práctica 5, graficar el control pid aplicado al balón y viga





**Figura 122** Datos de referencia y variable de entrada Práctica 6

### CONCLUSIONES:

1. En esta práctica se visualizó la estabilidad del control de posición con un punto de referencia de 15.
2. Mediante el uso de filtros se eliminó los ruidos y se logró una entrada más estable que facilite el control.
3. Mediante el uso de la herramienta Matlab se logra obtener mejores visualizaciones de las gráficas.

#### 4.7. Práctica #7

		<b>GUÍA DE PRÁCTICA DE LABORATORIO</b>	
<b>CARRERA:</b> Ingeniería Electrónica		<b>ASIGNATURA:</b> Teoría de Control I	
<b>NRO. PRÁCTICA:</b>	<b>7</b>	<b>TÍTULO PRÁCTICA:</b> "Utilización del método de Ziegler-Nichols.	
<p><b>Objetivo:</b></p> <p>Implementar el método de sintonización por ganancia crítica del método Ziegler Nichols.</p> <p><b>Objetivos específicos:</b></p> <p>Usar el método de ganancias críticas para ajustar el control del proporcional integral y derivativo.          Aprender a crear métodos para el control.          Usar herramientas de monitoreo tales como Matlab para visualizar datos.</p>			
<b>INSTRUCCIONES:</b>		<b>1. Alimentar la parte de control con 110-220v</b>	
		<b>2. Verificar que se tienen todos los elementos necesarios para realizar la práctica</b>	
		<b>3. Seguir la guía paso a paso para realizar la conexión de la parte de control y la planta.</b>	
<p>Antes de hacer la práctica se debe realizar la lectura del tema "3.10,3.11 y3.12 para tener entendimiento de los conectores usados en el proyecto técnico.</p>			
<b>ACTIVIDADES POR DESARROLLAR</b>			
<p><b>Paso 1.</b></p> <p>Como primer paso se deber realizar la conexión entre la parte de control y la planta para esto se deberá seguir los pasos indicados en la <b>PRÁCTICA #1</b> hasta que se realice el paso 5.</p>			

### Paso 2.

Seguir los pasos de la **PRÁCTICA #5**, los valores de  $K_p$ ,  $k_i$ ,  $K_d$  se encontrarán usando el método de Ziegler Nichols para valores de 20. Para esto se incrementará la variable  $K_p$  hasta que se aprecie una oscilación mantenida ante cualquier ante cualquier estímulo exterior.

### Paso 3.

Abrir **Matlab** y ejecutar los pasos de la **PRÁCTICA #5** y graficar los datos considere un valor de  $K_p = 1$  y se graficará en la Figura 123.

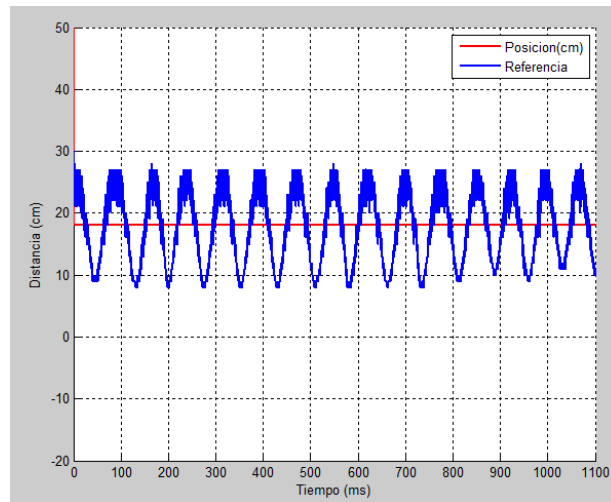


Figura 123 Respuesta de controlador con  $K_p=1$  Práctica 7

### Paso 4.

Graficar los datos cuando ubicara un  $K_p = 100$ ,  $K_p$  sería igual a  $K_{cr}$  y se graficara en la Figura 124.

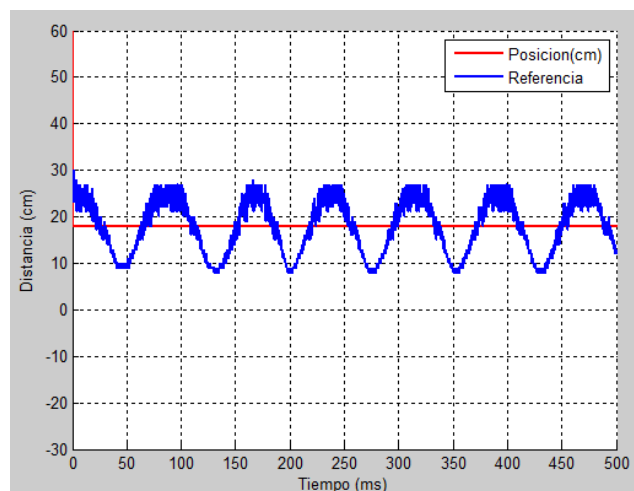
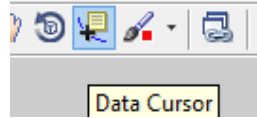


Figura 124 Respuesta de controlador con  $K_p=100$  Práctica 7

**Paso 5.**

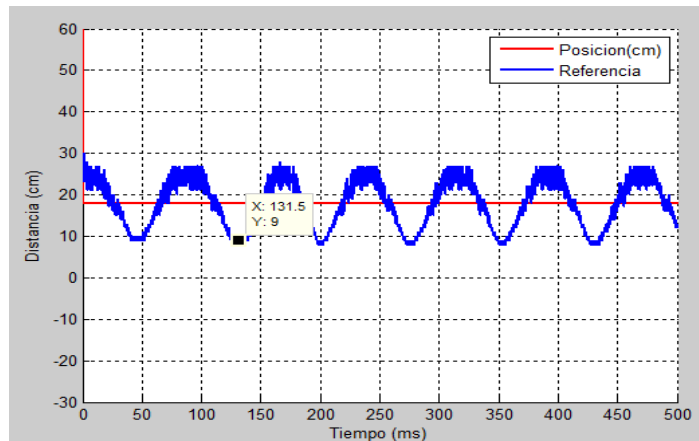
Tomar el valor de periodo de la gráfica con la herramienta **Data Cursor** de esta manera se selecciona dos puntos en la gráfica con los cuales se sabrá el periodo (Figura 125).



**Figura 125** Herramienta Data Cursor Práctica 7

**Paso 6.**

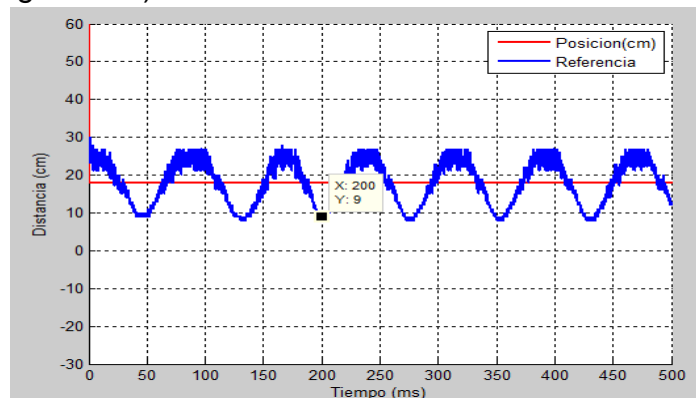
Seleccionar el punto A 131.5 de la gráfica donde se aprecie el inicio de la oscilación (Figura 126)



**Figura 126** Inicio del Ciclo de oscilación Práctica 7

**Paso 7.**

Seleccionar el punto B 200 de la gráfica donde se aprecie el fin de la oscilación (Figura 127)



**Figura 127** Final del ciclo de oscilación Práctica 7

### Paso 8.

Encontrando la diferencia entre cada punto es 68.5 será el periodo o  $T_c$ , este valor se usará para encontrar  $K_p$ ,  $K_d$ ,  $k_i$  usando la tabla 4.

Controlador	$K_p$	$T_i$	$T_d$
P	$0.5 * K_{cr}$		
PI	$0.45 * K_{cr}$	$T_{cr}/1.2$	
PID	$0.6 * K_{cr}$	$T_{cr}/2$	$T_{cr}/8$

**Tabla 4:** Valores Sintonización- (Hagglund, 1995)

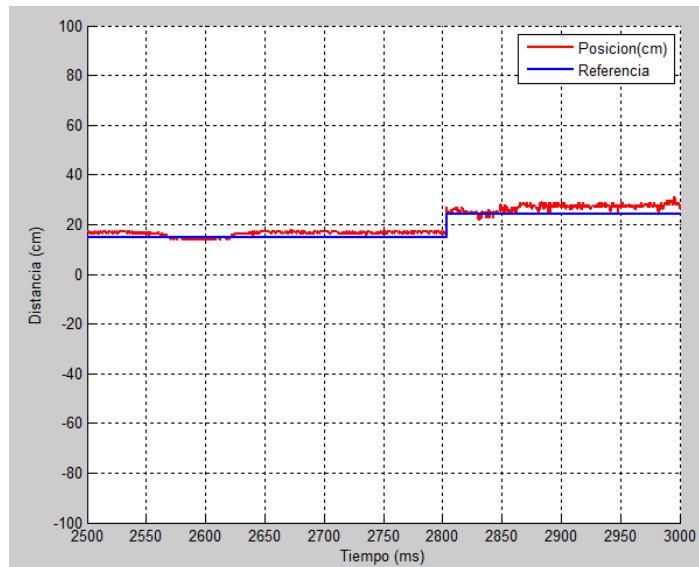
### RESULTADOS

Los resultantes de aplicar un control pid con una referencia de 15 aplicando el método de ganancia critica da como resultante los valores de la Table 5.

Controlador	$K_p$	$T_i$	$T_d$
PID	60	34.25	17.125

**Tabla 5:** Valores Sintonización con ganancia de 100- (Hagglund, 1995)

**La respuesta del sistema se muestra en la Figura 128 Utilizando el Paso 16 al 24 de la Práctica 5, graficar el control pid aplicado al balón y viga**





**Figura 128** Respuesta del controlador Práctica 7

**CONCLUSIONES:**

1. En esta práctica se comprende el método de ganancia críticas para un punto de referencia de 15.
2. Mediante el uso de este método se podrá encontrar los valores para cada punto de referencia preestablecido.
3. Mediante el uso de la herramienta Matlab se logra obtener mejores visualizaciones de las gráficas.



#### 4.8. Práctica #8

		<b>GUÍA DE PRÁCTICA DE LABORATORIO</b>	
<b>CARRERA:</b> Ingeniería Electrónica		<b>ASIGNATURA:</b> Teoría de Control I	
<b>NRO. PRÁCTICA:</b>	8	<b>TÍTULO PRÁCTICA:</b> “Ganancias programadas en planta de control balón y viga.	
<p><b>Objetivo:</b> Implementar un control PID ganancias programadas.</p> <p><b>Objetivos específicos:</b> Usar el método de ganancias críticas para obtener los valores de proporcional integral y derivativo en diferentes puntos de referencia. Crear un método el cual realice una comparación y configure los proporcionales, integral y derivativo dependiendo el punto de referencia. Aprender a crear métodos para el control. Usar herramientas de monitoreo tales como Matlab para visualizar datos.</p>			
<b>INSTRUCCIONES:</b>		<b>1. Alimentar la parte de control con 110-220v</b>	
		<b>2. Verificar que se tienen todos los elementos necesarios para realizar la práctica</b>	
		<b>3. Seguir la guía paso a paso para realizar la conexión de la parte de control y la planta.</b>	
<p>Antes de hacer la práctica se debe realizar la lectura del tema “3.10,3.11 y3.12 para tener entendimiento de los conectores usados en el proyecto técnico.</p>			
<b>ACTIVIDADES POR DESARROLLAR</b>			
<p><b>Paso 1.</b></p> <p>Como primer paso se debe realizar los pasos de la <b>PRÁCTICA #7</b> cambiando el valor de referencia a 20 y una referencia de 15.</p>			

### Paso 2.

En la Figura 129 se aprecia la sintonización con un proporcional de 1 no se genera una oscilación estable.

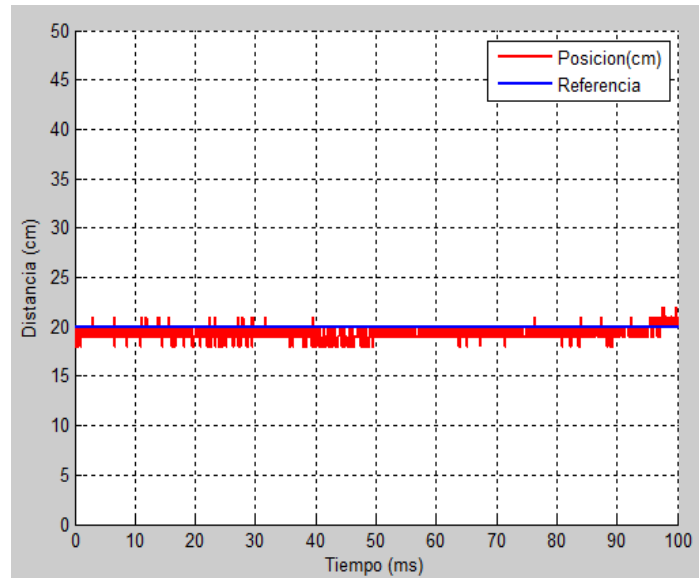


Figura 129 Respuesta con referencia de 20 y proporcional 1 Práctica 8

### Paso 3.

En la Figura 130 se aprecia la sintonización con un proporcional de 21.75 se muestra una oscilación estable de manera que se procede a aplicar la tabla del segundo método ziegler Nichols.

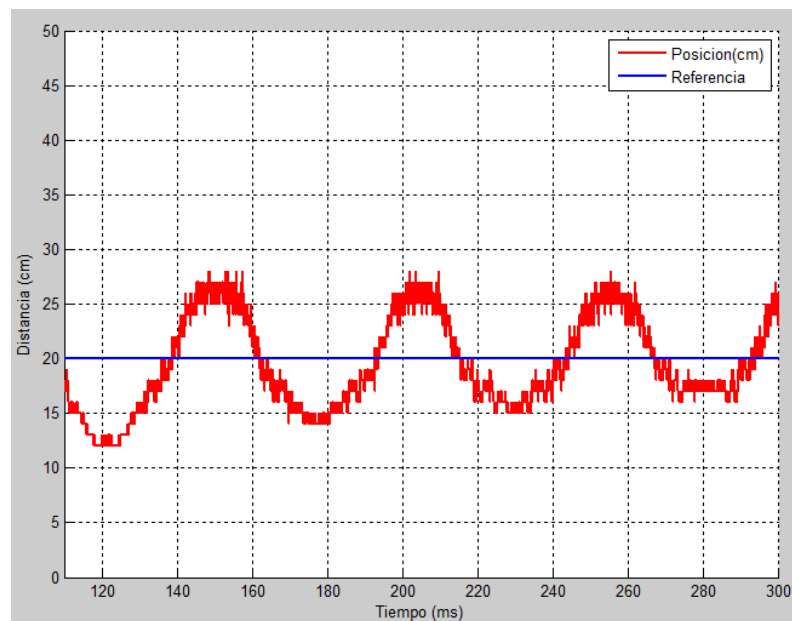


Figura 130 Respuesta con referencia de 20 y proporcional 21.75 Práctica 8

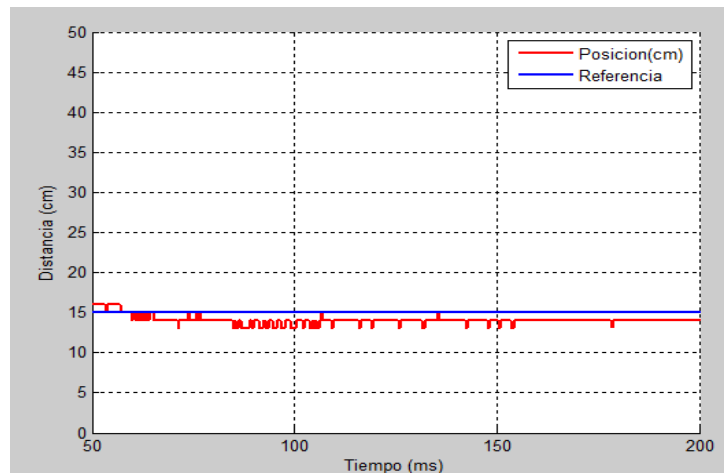
#### Paso 4.

Con los datos en el paso 2 se obtiene los datos  $K_p$ ,  $k_i$ ,  $K_d$  véase en tabla6.

Referencia	$K_p$	$K_i$	$K_d$
20	13.05	1.3	0.42

**Tabla 6:** Valores con referencia de 20.

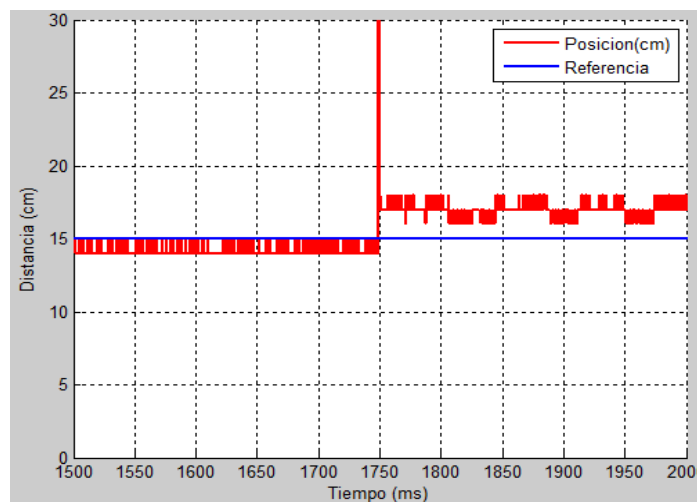
La respuesta en una referencia de con los valores de  $K_p$ ,  $k_i$  y  $K_d$ , tiene el resultado que se muestra en la Figura 131.



**Figura 131** Respuesta con referencia de 20 Práctica 8

#### Paso 5.

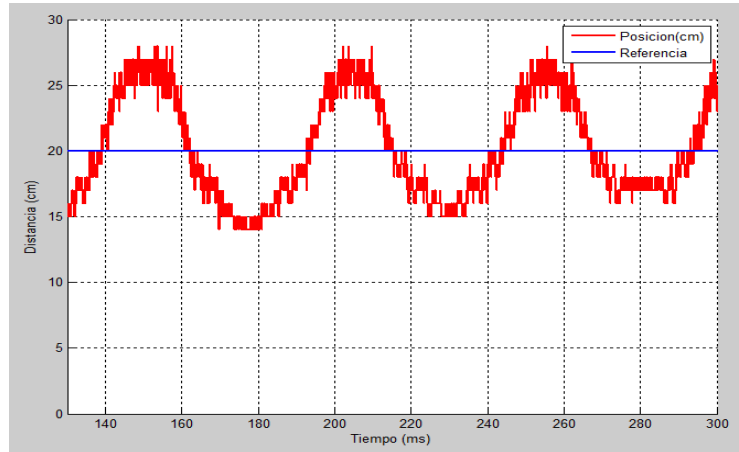
Sintonizar con una referencia de 15 y un proporcional de 1 véase en la Figura 132.



**Figura 132** Respuesta con referencia de 15 y proporcional 1-Práctica 8

### Paso 6.

Sintonizar con una referencia de 15 y un proporcional de 8.94 véase en la Figura 133.



**Figura 133** Respuesta con referencia de 15 y proporcional 8.94-Práctica 8

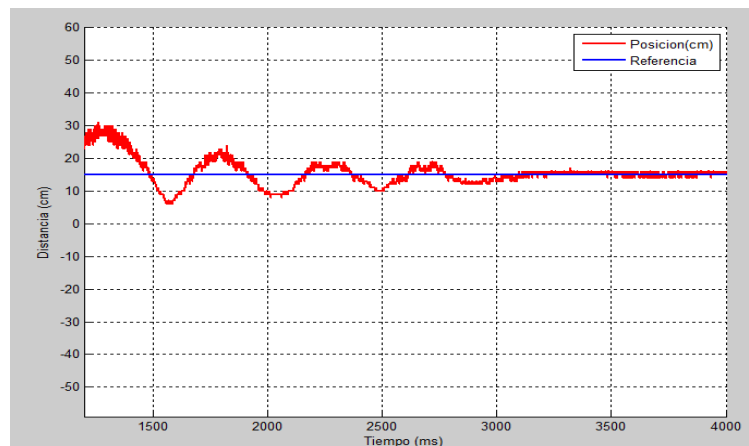
### Paso 7.

Con los datos en el paso 6 se obtiene los de proporcional que general una oscilación estable aplicando el método de ganancia crítica los datos  $K_p$ ,  $k_i$ ,  $K_d$  se muestran en la tabla 7.

Referencia	$K_p$	$K_i$	$K_d$
15	14.9	1.22	0.5

**Tabla 7:** Valores con referencia de 15.

La respuesta en una referencia de con los valores de  $K_p$ ,  $k_i$  y  $K_d$ , tiene el resultado que se muestra en la Figura 134.



**Figura 134** Respuesta con referencia de 15 Práctica 8

### Paso 8.

Se crea un método para procesar el valor de referencia y asignar las ganancias según la referencia establecida como se muestra en la Figura 135.

```
void SET_GANANCIAS() {
  if (setpoint2 == 15)
  {
    kp2 = 14.9 , ki2 = 1.22, kd2 = 0.5;           // Valores Posicion PARA SETPOINT =15
    PosicionPID.SetTunings(kp2, ki2, kd2);

    PosicionPID.SetOutputLimits(-75, 75);
  }
  else if (setpoint2 == 20) {
    {
      kp2 = 13.05 , ki2 = 1.3 , kd2 = 0.42;       // Valores Posicion PARA SETPOINT =20
      PosicionPID.SetTunings(kp2, ki2, kd2);
    }
  }
}
```

**Figura 135** Método para ganancias programadas Práctica 8

### Paso 9.

Se crea un método para procesar el pid del balón como se aprecia en la Figura 136.

```
void BALON() {
  input2 = ADC0_promedio(20);
  PosicionPID.Compute();
}
```



**Figura 136** Método del pid del balón Práctica 8

### Paso 10.

Se crea un método para procesar el pid de la posición en base a la salida del pid del balón y saturando la entrada para no tener ningún dato fuera de la escala como se aprecia en la Figura 137.

```
void VIGA() {
  output2 = constrain(output2, -255, 255);
  setpoint = output2;
  input = encoderPos ;
  VigaPID.Compute();
}
```

**Figura 137** Método del pid del Angulo Viga Práctica 8

### Paso 11.

Se Configura un método para la saturación de la salida del pid del Angulo evitando un punto muerto, para romper la inercia del motor, véase en la Figura 138.

```
void VIGA() {  
    output2 = constrain(output2, -255, 255);  
    setpoint = output2;  
    input = encoderPos ;  
    VigaPID.Compute();  
}
```

**Figura 138** Método de saturación pid del Angulo Viga Práctica 8

### Paso 12.

Crear un método para supervisión mediante el puerto serial de la tarjeta con los valores de la entrada y la referencia (Figura 139).

```
void monitoreo1() {  
    Serial.print((int)input2);  
    Serial.print(",");  
    Serial.println((int)setpoint2);  
}
```

**Figura 139** Método de supervisión del proceso Práctica 8

### Paso 13.

El método principal sería codificado como se muestra en la Figura 140.

```
void loop() {  
    setpoint2 = 20;  
    SET_GANANCIAS();  
    BALON();  
    VIGA();  
    saturacion();  
    analogWrite(salida, duty);  
    monitoreo1();  
}
```

**Figura 140** Método lazo principal supervisión del proceso Práctica 8

### Paso 14.

El método principal sería codificado como se muestra en la Figura 140.

```
void loop() {  
    setpoint2 = 20;  
    SET_GANANCIAS();  
    BALON();  
    VIGA();  
    saturacion();  
    analogWrite(salida, duty);  
    monitoreo1();  
}
```

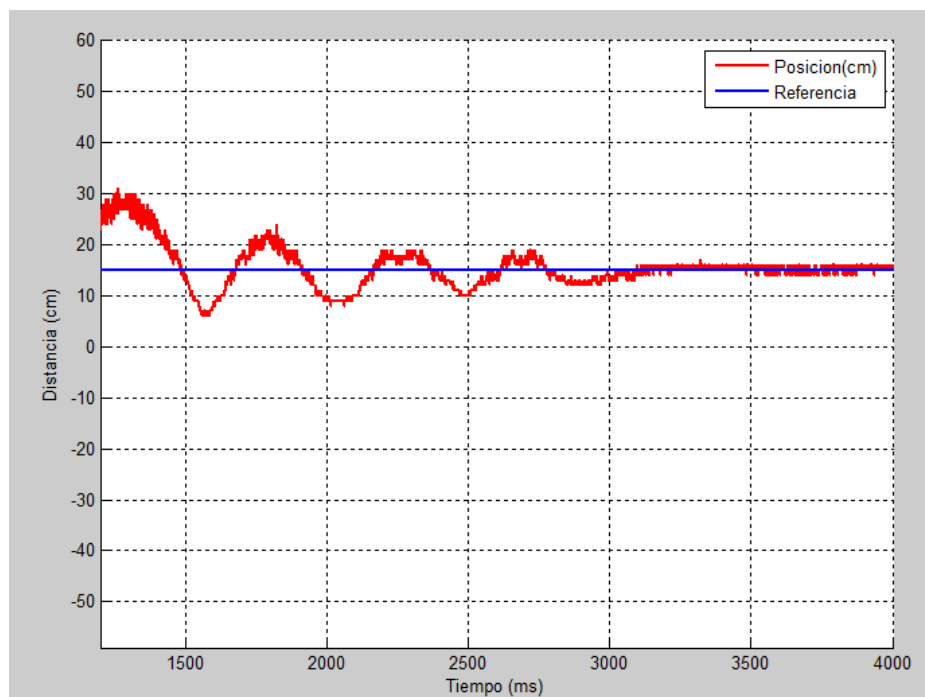
**Figura 141** Método lazo principal supervisión del proceso Práctica 8

### Paso 15.

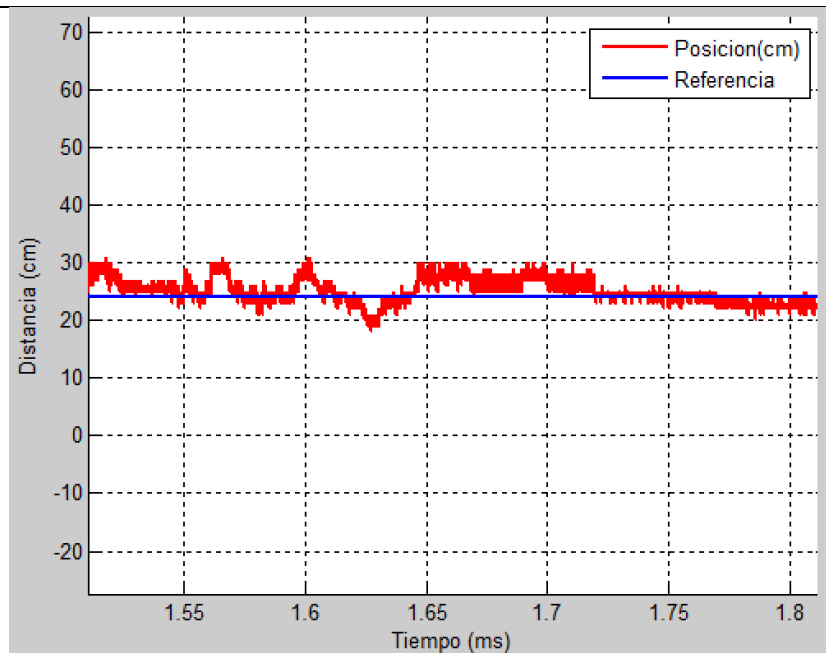
Crear un script en Matlab para la supervisión de los datos y graficarlos siguiendo los pasos de **PRÁCTICA 5 (Paso 18 hasta el 27)**

### RESULTADOS

**Los resultantes de aplicar un control pid con ganancias programadas con referencias de 15 y 20**



**Figura 142** Respuesta del controlador 15 Práctica 8



**Figura 143** Respuesta del controlador 20 Práctica 8

**CONCLUSIONES:**

1. En esta práctica se comprende el método de ganancia críticas para un punto de referencia de 15 y 20.
2. Mediante el uso de este método se podrá encontrar los valores para cada punto de referencia preestablecido.
3. Mediante el uso de la herramienta Matlab se logra obtener mejores visualizaciones de las gráficas.



## RESULTADOS

Para medir el resultado del proyecto técnico se realizó una clase explicativa a los alumnos de la materia Control 1 y Microcontroladores sobre la planta de control balón y viga

<b>1. ¿Qué tal le pareció el diseño de la planta balón y viga?</b>
Excelente Bueno Regular Malo
<b>2. ¿Qué tan útil le parece las diferentes funciones que posee el embebido Launchpad?</b>
Muy útil Útil Inútil
<b>3. ¿Qué tan sencillo le pareció el trabajar una práctica?</b>
Muy difícil Difícil Sencillo Muy sencillo
<b>4. ¿Estaría usted interesado en aprender mucho más de lo explicado en la exposición acerca del Launchpad de la familia Texas Instrument?</b>
Si No
<b>5. ¿Cómo calificaría la exposición dada?</b>
Excelente Buena Regular Mala

**6. En su opinión que se debería mejorar al momento de exponer y que se podría mejorar en el proyecto**

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Luego de realizar las encuestas a los 40 estudiantes asistentes a clases obtuvo los siguientes resultados:

- ¿Con respecto al diseño de la planta balón y viga?  
25 pensaron opinaron que era excelente.  
10 personas opinaron que era buena.  
5 personas opinaron que era Regular.  
0 personas opinaron que era malo.
  
- Con respecto a la utilidad de las funciones que tiene el controlador se obtuvo que:  
  
38 personas opinaron que es Muy útil.  
2-personas opinaron que era útil.  
0 personas opinaron que era Inútil.
  
- Con respecto a la sencillez de trabajar con la planta de control se obtuvo que:  
  
0 pensaron opinaron que era muy difícil.  
7 persona opino que era difícil.  
30 personas opinaron que era Sencillo.  
3 personas opinaron que era Muy sencillo.
  
- Con respecto a si los estudiantes estarían dispuestos a aprender más acerca del controlador Launchpad se obtuvo que:  
  
36 pensaron opinaron que SI.  
4 personas opinaron que NO.
  
- Con respecto a la exposición dada se obtuvo que:  
  
8 pensaron opinaron que era excelente.  
7 personas opinaron que era buena.  
1 persona opino que era Regular.  
0 personas opinaron que era malo.



Exposición de Objetivos del proyecto



Utilidad de la planta balón y viga

## CONCLUSIONES

1. En el desarrollo del proyecto de titulación se trabajó con el Ide Energía de la familia TEXAS INSTRUMENTS por el cual se programa mediante lenguaje C, ya que muestra una ventaja al momento de creación de librerías.
2. El controlador Sabertooth permite realizar múltiples configuraciones para su uso desde el embebido ya que provee múltiples protecciones y conmutación de alta frecuencia para variar la tensión en el motor cc sin tener picos de voltajes que afecten al embebido.
3. El encoder incremental permite realizar lecturas rápidas para saber la velocidad o posición del motor para estas lecturas es necesario un controlador para poder leer los pulsos y usarlos en el control ya que este tipo de periféricos tienen muchas utilidades en la parte industrial.
4. Una de las grandes ventajas que presenta del embebido es el uso de ejecución rápida y debut en sistemas de control y procesamiento de datos, usando las entradas y salidas que posee el mismo tales como modos de comunicación.
5. Los sensores de distancias necesitan un filtro para eliminar toda perturbación y que el proceso PID no tenga ningún tipo de ruido en la entrada para su óptimo funcionamiento de la planta.
6. Como se aprecia la planta de control posee entradas, salidas tanto digitales como análogas y módulos que permiten un trabajo óptimo en la toma de decisiones para el proceso PID.

## RECOMENDACIONES

1. Tener conocimiento básico en lo que programación en lenguaje C y un correcto uso de periféricos electrónicos.
2. El embebido además de realizar el proceso PID necesitara sus debidas precauciones para su correcto uso tales como evitar estática que afecte directamente al sistema
3. Precautelar que los encoder incrementales no tengan ninguna obstrucción que no permita su lectura, que este permanezca siempre con su protección establecida.
4. Se recomienda utilizar la versión de Energía 1.6.8. para evitar problemas de compatibilidad y acceso a código.

## REFERENCIAS BIBLIOGRÁFICAS

### BIBLIOGRAFIA

- STMicroelectronics. (2016). *STMicroelectronics*. Obtenido de STMicroelectronics: <https://www.pololu.com/file/0J1187/VL53L0X.pdf>
- Angulo Usategui, J. M., Romero Yesa, S., & M., I. A. (2006). *Microcontroladores PIC: diseño práctico de aplicaciones*. Madrid: Mc Graw Hill.
- Areny, R. P. (2003). *Sensores acondicionadores de señal* (Vol. 4). Barcelona: Marcombo.
- Balbi. (13 de 12 de 2013). *MEDIAFILE BACKUP*. Obtenido de TIPOS DE CONECTORES XLR - CANNON: <http://mediafilebackup.blogspot.com/2013/12/tipos-de-conectores-xlr-cannon.html>
- Barrientos, Antonio, Peñín, Luis Felipe, & Balag. (2007). *Fundamentos de robótica* (2a. ed.). Madrid: McGraw-Hill España.
- Breijo, G. E. (2009). *Compilador C CCS y Simulador Proteus para Microcontroladores Pic* (Vol. 3). Barcelona: marcombo.
- Calderon, G. G. (2004). *Diseño de tarjeta controladora de motores DC de altas prestaciones*. Sartenejas: Universidad Simon Bolivar.
- Celi, J. A. (2013). *Diseño de un prototipo de sillas de ruedas electrica, con sistema de ascenso y elevacion*. Cuenca: Universidad Politecnica Salesiana.
- Cruz, J. M., & Lutenberg, A. (2012). Introducción General a los Sistemas Embebidos. *Ariel Lutenberg* (pág. 14). Argentina: Universidad de Buenos Aires.
- CTMS. (2012). <http://ctms.engin.umich.edu>. Obtenido de <http://ctms.engin.umich.edu/CTMS/index.php?example=BallBeam&section=SystemModeling>
- Dimension Engineering. (2004). *Dimension Engineering*. Obtenido de Dimension Engineering: <https://www.dimensionengineering.com/products/sabertooth2x12>
- Dorf, R. C., & Bishop, R. H. (2005). *Sistemas de control moderno* (10 ed.). Madrid: Pearson Educacion S.A.
- Fritzing. (2015). *Fritzing*. Obtenido de Fritzing: <http://fritzing.org/home/>
- FutureElectronics. (3 de Marzo de 2017). *FutureElectronics*. Obtenido de <http://www.futureelectronics.com/en/drivers/motor-driver.aspx>
- GONZALEZ, H. U. (2009). *309696 - MODULO DE MICROPROCESADORES & MICROCONTROLADORES*. CHIQUINQUIRA: UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA.
- Hagglund, K. J. (1995). *PID Controllers* (Vol. 2). Us: Instrument Society of America. Obtenido de <http://www.el.bqto.unexpo.edu.ve/tperez/SC1/DIS-ZN.pdf>
- Heath, S. (2003). *Embedded System Design* (Vol. 2). Oxford: Newnes.
- Jeff, L. (2004). A robotic Ball Balancing Beam. *Jeff Lieberman*, <http://www.bea.st/sight/rbbb/rbbb.pdf>.
- Jiménez, M. P. (2014). *Introduction to Embedded Systems*. New York: Springer-Verlag New York.
- MathWorks. (1984). *Mathworks*. Obtenido de <https://www.mathworks.com/products/matlab.html>
- Mazzone, V. (2002). *Controladores PID*. Quilmes: Universidad Nacional de Quilmes.
- Morales, P. M., & Zafra Siancas, H. D. (2013). *Diseño e Implementación de un Módulo Educativo para el control del Sistema Bola y Varilla*. Lima: Pontificia Universidad Catolica del Peru.
- MSP430. (Septiembre de 2014). *MSP430 LaunchPad Value Line Development kit*. Obtenido de <http://www.ti.com/tool/MSP-EXP430G2>
- Ogata, K. (2010). *Ingeniería de control moderna* (Vol. 5). Madrid: Pearson.

- Pololu. (10 de Enero de 2006). *Pololu*. Obtenido de Pololu:  
<https://www.pololu.com/product/2824>
- Poznyak, A. S. (2 de Mayo de 2005). *Modelado Matematico de los sistemas mecanicos, electrico y electromecanicos*. Obtenido de  
[https://www.ctrl.cinvestav.mx/~coordinacion/documents/cursos/modelado\\_matematico.pdf](https://www.ctrl.cinvestav.mx/~coordinacion/documents/cursos/modelado_matematico.pdf)
- Quanser. (2006). *Quanser*. Obtenido de Quanser: <https://www.quanser.com/>
- Ramírez, H. S., Márquez, R., Echeverría, F. R., & Santiago, O. L. (2004). *Control de Sistemas no lineales*.
- TELCO. (6 de 08 de 2013). *Conectores XLR de la serie AAA de Switchcraft*. Obtenido de TELCO: <http://telcoavi.blogspot.com/2013/08/conectores-xlr-de-la-serie-aaa-de.html>