UNIVERSIDAD NACIONAL DE COLOMBIA

Péndulo invertido rotatorio

Sebastián Palacio Betancur
Mariana Hernández Beltrán

Universidad Nacional de Colombia
Facultad de Minas

PAE
Pablo Santiago Rivadeneira Paz

Medellín, Colombia
Septiembre 2023

Índice

1. Introducción

El péndulo invertido rotatorio, también conocido como péndulo de Furuta, es un sistema dinámico no lineal cuyo estudio y control resulta de gran utilidad para diversas aplicaciones prácticas especialmente en el ámbito de la robótica. Algunas de estas aplicaciones incluyen la mejora de la estabilidad en robots bipedales y el diseño de sistemas de transporte autónomo más seguro y eficiente.

En el trabajo se presenta el modelo matemático del péndulo, abordando las ecuaciones de energía cinética y potencial asociadas con los movimientos del brazo y el péndulo, se utiliza la función lagrangiana para obtener las ecuaciones diferenciales del sistema, a partir de las cuales se llega a un modelo en espacio de estados y su representación en funciones de transferencia, que facilitan el análisis y diseño de control.

Después, se realiza una revisión a la documentación de control del péndulo y se compara con el código implementado del mismo, para posteriormente discutir los resultados prácticos y la necesidad de mejorar la fidelidad del diseño con la implementación práctica.

Finalmente, se presentan conclusiones y recomendaciones para futuros trabajos.

2. Modelo del péndulo invertido rotatorio

El esquema permite observar y entender las variables asociadas al brazo giratorio y el péndulo.

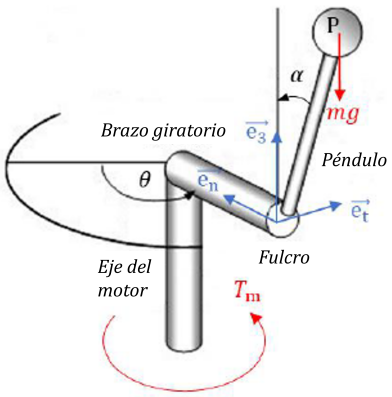


Figura 1: Péndulo Invertido

2.1. Tabla de estados

Variable	Descripción	Unidad
θ	Ángulo del brazo giratorio	(rad)
α	Ángulo del péndulo	(rad)
$\dot{\theta}$	Velocidad angular del brazo giratorio	(rad/s)
$\dot{\alpha}$	Velocidad angular del péndulo	(rad/s)

2.2. Tabla de parámetros

Variable	Descripción	Valor	Unidad
m	Masa del péndulo	0.140	(kg)
l	Distancia del centro de masa del péndulo al punto de apoyo	$\frac{0{,}135}{2}$	(m)
r	Longitud del brazo giratorio	0.156	(m)
J	Momento de inercia cuando el péndulo gira alrededor del centro de masa	2.2775e-4	$(kg \cdot m^2)$
I	Momento de inercia cuando el brazo giratorio gira alrededor del eje del motor	8.6562e-4	$(kg \cdot m^2)$

2.3. Suposiciones

1. El ángulo de inclinación del péndulo es relativamente pequeño ($-10° \leq \alpha \leq 10°$)
2. Se ignoran los efectos de la fricción ya que la velocidad y aceleración con la que trabaja el sistema hace que los pares de fricción no sean significativos.

2.4. Balances de energía

La energía cinética del sistema está dada por la suma de la energía cinética del brazo T_α y la del péndulo T_θ.

$$T = T_\alpha + T_\theta \tag{1}$$

Estas energía cinéticas están descritas por:

$$T_\alpha = \frac{1}{2} I \dot{\theta}^2 + \frac{1}{2} J \dot{\alpha}^2 - \frac{1}{2} m v_P^2 \tag{2}$$

$$T_\theta = \frac{1}{2} I \dot{\theta}^2 + \frac{1}{2} J \dot{\alpha}^2 - \frac{1}{2} m [(\dot{\theta} r - l\dot{\alpha}\cos\alpha)^2 + (l\dot{\alpha}\sin\alpha)^2]^2 \tag{3}$$

Tomando como superficie de energía potencial cero el plano donde se encuentra el brazo giratorio, la energía potencial del sistema (energía potencial del brazo giratorio y energía potencial del péndulo) es:

$$V = mglcos\alpha \tag{4}$$

Tomando las ecuaciones 1, 2, 3 y 4, la función lagrangiana es:

$$L = T - V = \left(\frac{1}{2} I + \frac{1}{2} m r^2\right) \dot{\theta}^2 + \left(\frac{1}{2} J + \frac{1}{2} m l^2\right) \dot{\alpha}^2 - mr\dot{\theta} l \dot{\alpha} \cos\alpha - mgl\cos\alpha \tag{5}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} = Q \tag{6}$$

Donde q son las coordenadas generalizadas, $\dot{q}$ son la velocidades generalizadas, Q es una fuerza conservativa generalizada.

$$q = \begin{bmatrix} \theta \\ \alpha \end{bmatrix} \qquad \dot{q} = \begin{bmatrix} \theta \\ \alpha \end{bmatrix}$$

Así, de la ecuación 6, se una ecuación por cada coordenada generalizada:

$$\begin{cases} (I + mr^2)\ddot{\theta} - mrl\ddot{\alpha}\cos\alpha + mrl\dot{\alpha}^2\sin\alpha = T_m \\ (J + mr^2)\ddot{\theta} - mrl\ddot{\alpha}\cos\alpha - mgl\sin\alpha = 0 \end{cases} \tag{7}$$

Considerando la suposición (1):

$$\begin{cases} \cos\alpha = 1 \\ \sin\alpha = \theta \\ \dot{\alpha}^2 = 0 \end{cases}$$

El sistema queda dado por:

$$\begin{cases} (I + mr^2)\ddot{\theta} - mrl\ddot{\alpha} = T_m \\ (J + mr^2)\ddot{\theta} - mrl\ddot{\alpha} - mgl\alpha = 0 \end{cases} \tag{8}$$

2.5. Modelo del motor DC

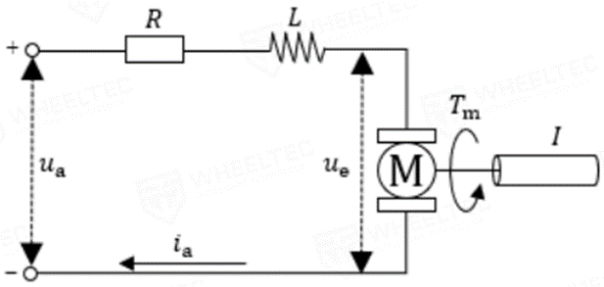


Figura 2: Modelo electromecánico del motor DC

Debido a que la entrada del sistema no es el torque como se tiene en el modelo del péndulo invertido (8), sino el voltaje inducido al motor, se caracteriza el motor de corriente continua para obtener T_m en términos de variables conocidas.

2.5.1. Tabla de variables

Variable	Descripción	Unidad
u_a	Voltaje de entrada al motor	(V)
T_m	Torque de motor DC	(N.m)
ω_m	Velocidad angular del motor	(rad/s)
u_e	Fuerza contraelectromotriz	(V)
i_a	Corriente a través del motor	(A)

2.5.2. Tabla de parámetros

Variable	Descripción	Valor	Unidad
R	Resistencia del motor	4.286	(Ω)
K_e	Coeficiente de fuerza contraelectromotriz	0.2087	(V.s/rad)
K_m	Coeficiente de torque	0.1797	(N.m/A)

2.5.3. Suposiciones

1. Se ignoran los efectos de la fricción y amortiguamiento ya que la velocidad y aceleración con la que trabaja el sistema hace que los pares de fricción no sean significativos.

2. Se ignoran los efectos de la inductancia debido a que sus efectos son despreciables, respecto a los efectos generados por la constante mecánica.

2.5.4. Leyes de Kirchhoff sobre modelo de motor DC

Aplicando una ecuación de malla en el modelo circuital presentado en la figura 2 se obtiene:

$$u_a = Ri_a + L\frac{di_a}{dt} + u_e \tag{9}$$

Si se aplica la suposición (2) esta ecuación se simplifica y resulta:

$$u_a = Ri_a + u_e \tag{10}$$

Además se considera que el motor DC satisface las siguientes relaciones de acople electromecánico:

$$u_e = K_e\omega_m \tag{11}$$

$$T_m = K_m i_a \tag{12}$$

De las ecuaciones 10, 11 y 12, se obtiene:

$$T_m = \frac{-K_m K_e}{R}\omega_m + \frac{K_m}{R}u_a \tag{13}$$

2.6. Modelado en espacio de estados

Con el modelo obtenido para el péndulo invertido rotatorio y la función para T_m se puede expresar el sistema en espacio de estados como:

$$\begin{cases} \ddot{\theta} = \dfrac{m^2l^2rg}{\lambda}\alpha - \dfrac{K_mK_e(J+ml^2)}{\lambda R}\dot{\theta} + \dfrac{K_m(J+ml^2)}{\lambda R}u_a \\ \ddot{\alpha} = \dfrac{mgl(I+mr^2)}{\lambda}\alpha + \dfrac{mrlK_mK_e}{\lambda R}u_a \end{cases} \tag{14}$$

Donde:

$$\lambda = (IJ + Iml^2 + Jmr^2)$$

Y de forma matricial:

$$\begin{pmatrix} \dot{\theta} \\ \ddot{\theta} \\ \dot{\alpha} \\ \ddot{\alpha} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & A_{22} & A_{23} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & A_{42} & A_{43} & 0 \end{pmatrix} \begin{pmatrix} \theta \\ \dot{\theta} \\ \alpha \\ \dot{\alpha} \end{pmatrix} + \begin{pmatrix} 0 \\ B_{21} \\ 0 \\ B_{41} \end{pmatrix} u_a \tag{15}$$

Donde:

$$A_{22} = -\frac{K_mK_e(J+ml^2)}{\lambda R} \qquad A_{23} = \frac{m^2l^2rg}{\lambda}$$

$$A_{42} = -\frac{mlrK_mK_e}{\lambda R} \qquad A_{43} = \frac{mgl(I+mr^2)}{\lambda}$$

$$B_{21} = \frac{K_m(J+ml^2)}{\lambda R} \qquad B_{41} = \frac{mrlK_m}{\lambda R}$$

La salida del sistema está dada por:

$$y = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \theta \\ \dot{\theta} \\ \alpha \\ \dot{\alpha} \end{pmatrix} \tag{16}$$

3. Control del péndulo invertido

3.1. Diseño del controlador LQR

La implementación del controlador LQR requiere la conversión de las matrices A y B del sistema continuo a su forma discreta. Este proceso se lleva a cabo mediante el uso de la función c2d() de Matlab. En este caso, se ha seleccionado un intervalo de muestreo de 0.020 s para realizar dicha discretización. Como resultado se obtienen las matrices discretizadas G y H, las cuales corresponden a las matrices A y B del sistema continuo, respectivamente.

$$A = \begin{bmatrix} 0 & 1{,}000 & 0 & 0 \\ 0 & -4{,}9660 & 89{,}5113 & 0 \\ 0 & 0 & 0 & 1{,}000 \\ 0 & -8{,}4574 & 259{,}4297 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ 23{,}7951 \\ 0 \\ 40{,}5243 \end{bmatrix} \qquad Ts = 0{,}005s \tag{17}$$

$$G = \begin{bmatrix} 1{,}0000 & 0{,}0190 & 0{,}0175 & 0{,}0001 \\ 0 & 0{,}9045 & 1{,}7341 & 0{,}0175 \\ 0 & -0{,}0017 & 1{,}0513 & 0{,}0203 \\ 0 & -0{,}1638 & 5{,}1297 & 1{,}0513 \end{bmatrix} \qquad H = \begin{bmatrix} 0{,}0046 \\ 0{,}4577 \\ 0{,}0079 \\ 0{,}0079 \end{bmatrix} \tag{18}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad D = 0 \tag{19}$$

Así el sistema en espacio de estados discreto resultante queda dado por:

$$x(k+1) = G(k)x(k) + H(k)u(k) \tag{20}$$

$$y(k) = C(k)x(k) + D(k)u(k) \tag{21}$$

Se usa entonces la función dlqr() de Matlab, para hallar los valores de realimentación óptima de la matriz de ganancia K, es fundamental usar las matrices discretizadas que describen el sistema, es decir G, H, C y D, se establecen además las condiciones de la matriz ponderada de coste de estado Q, en la cual se establece un peso y la matriz ponderada de control R.

$$Q = \begin{bmatrix} 50 & 0 & 0 & 0 \\ 0 & 25 & 0 & 0 \\ 0 & 0 & 50 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix} \qquad R = 1000 \tag{22}$$

La matriz de ganancia K obtenida es:

$$K_{calculada} = \begin{bmatrix} -0{,}2067 & -0{,}5500 & 15{,}2365 & 1{,}0702 \end{bmatrix} \tag{23}$$

Es necesario comprobar la estabilidad del sistema realimentado usando la matriz de ganancia K, para lo cual obtienen los eigenvalores de la matriz del sistema en lazo cerrado.

$$eig(G - HK) = \begin{bmatrix} 0{,}9076 & 0{,}9330 & 0{,}9954 & 0{,}9885 \end{bmatrix} \tag{24}$$

Así se puede verificar que el sistema es estable debido a que tiene todos sus valores propios dentro de la circunferencia unitaria del plano complejo.
El archivo de MATLAB con el diseño del control se encuentra en el Anexo 3.

3.2. Controladores PD

Debido a que en este sistema los estados están dados por los ángulos del brazo y del péndulo y sus derivadas temporales, este sistema puede ser equivalente a un sistema de lazo cerrado con dos controladores PD, esto se puede ver claramente si se analiza la acción de control del sistema LQR que está dada por:

$$u = -Kx = -K \begin{pmatrix} \theta & \dot{\theta} & \alpha & \dot{\alpha} \end{pmatrix}' \tag{25}$$

Si se toma $K = \begin{pmatrix} K_1 & K_2 & K_3 & K_4 \end{pmatrix}$, la acción de control resultante sería:

$$u = -K_1\theta - K_2\dot{\theta} - K_3\alpha - K_4\dot{\alpha} \tag{26}$$

Esto se puede plasmar en un diagrama de control con dos controladores PD, así:

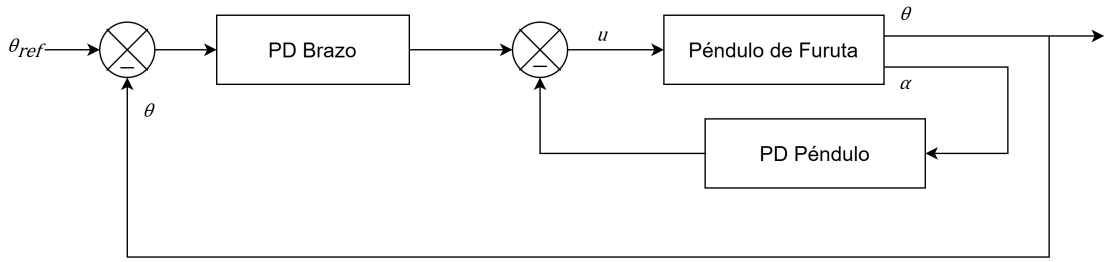


Figura 3: Lazo cerrado con dos controladores PD

4. Simulación

Después de verificar la estabilidad del sistema, se procede a realizar la simulación en Matlab con el objetivo de analizar la respuesta del sistema. En esta simulación, se establecen condiciones iniciales, representadas por el vector x0, que indican que el brazo comienza detenido en su posición de referencia, mientras que el péndulo se encuentra en reposo e inclinado 10°. Para evaluar la respuesta del sistema ante perturbaciones, se introduce una perturbación a los 25 segundos, simulando así un pequeño impacto en el péndulo.

$$x0 = \begin{bmatrix} 0 \\ 0 \\ -0{,}1745 \\ 0 \end{bmatrix} \tag{27}$$

Es importante que el esfuerzo de control no supere los 12V, ya que este valor es el voltaje nominal del motor, los valores de voltaje negativos indican un cambio de dirección del motor.

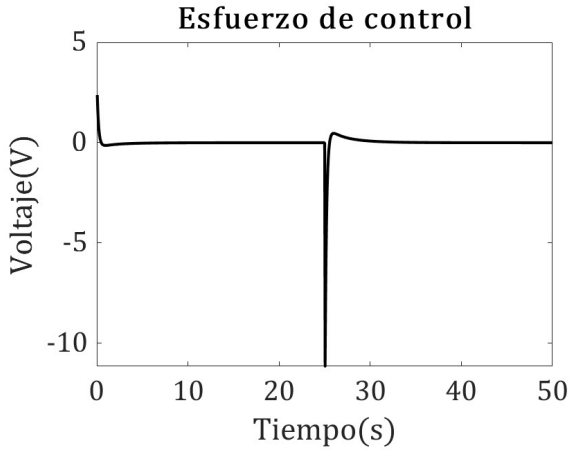
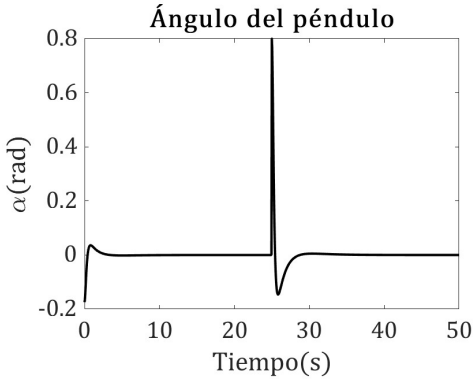
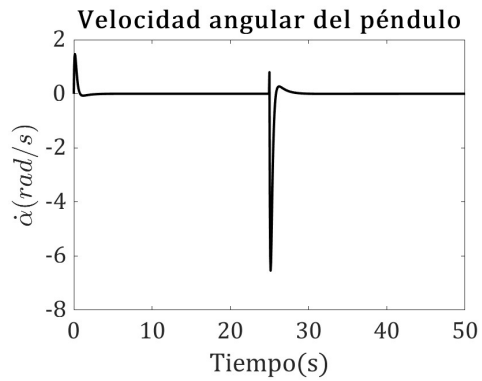


Figura 4: Esfuerzo de control

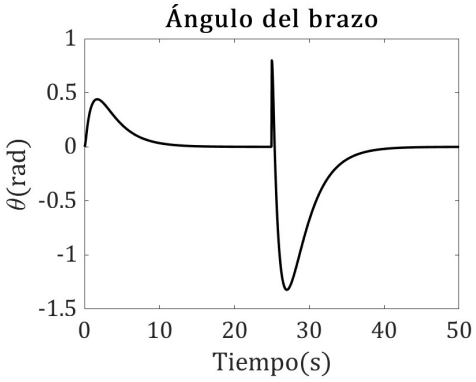


(a) Ángulo del péndulo

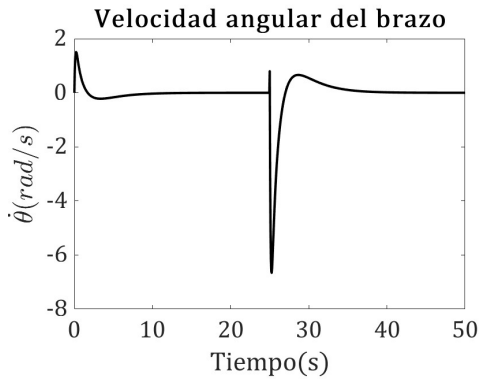


(b) Velocidad del péndulo

Figura 5: Ángulo y velocidad del péndulo



(a) Ángulo del brazo



(b) Velocidad del brazo

Figura 6: Ángulo y velocidad del brazo

Si bien tanto el ángulo del brazo como el del péndulo logran alcanzar el valor de estado estable, se hace evidente que el tiempo de establecimiento es menor para el ángulo del péndulo, estos resultados concuerdan con lo esperado ya para los estados asociados al control del péndulo al ser más críticos les fue asignados un mayor peso en la matriz Q.

5. Revisión de documentación

Antes de implementar el control diseñado, se revisó el código proporcionado por la empresa que comercializa el péndulo (Anexo 2) y se identificaron los siguientes aspectos:

- El control se plantea como un sistema en lazo cerrado con dos controladores PD. Cada controlador se calcula en una rutina propia como se puede ver en las lineas 8 y 9 del código 1, donde se usan las rutinas Balance_Control() y Position_Control(), mostradas en los códigos 2 y 3 respectivamente.
- En la línea 9 del código 1 también se puede ver que el cálculo del control del movimiento del brazo se hace cada cuatro ciclos, lo que equivale a aumentar el tiempo de muestreo a 20 ms.
- En el control del brazo también se añade un filtro de primer orden como se ve en el código 3, que en la documentación se justifica su uso con la finalidad de que el cambio en este movimiento sea suave.
- Adicionalmente se revisaron los valores de la matriz de ganancia K hallados en la documentación, se compararon con aquellos implementados en el control y se encontró discrepancia entre los mismos.

Valores hallados en la documentación:

$$K_{documentación} = \begin{bmatrix} -1{,}6349 & -1{,}7563 & 26{,}8000 & 2{,}3682 \end{bmatrix} \tag{28}$$

Valores implementados en el código de Arduino:

$$K_{implementación} = \begin{bmatrix} 6 & 12 & 10 & 10 \end{bmatrix} \tag{29}$$

```
void control()
{
      static float Voltage_All,Voltage_Count;
      int Temp;
      static unsigned char Position_Count;
      sei();
      Sensor= Get_Adc_Average(5,5);
      Balance_Pwm =Balance_Control(Sensor);
      if(++Position_Count>4) Position_Pwm=Position_Control(Position),
         Position_Count=0;
      Motor=Balance_Pwm-Position_Pwm;
      if(Turn_Off()==0)
      Set_Pwm(Motor);
       Adjust();
       Temp = analogRead(0);
       Voltage_Count++;
       Voltage_All+=Temp;
       if(Voltage_Count==200) Battery_Voltage=Voltage_All*0.05371/2,
          Voltage_All=0,Voltage_Count=0;
 }
```

Código 1: Rutina de control proporcionada

```
float Balance_Control(float sensor)
{
   float Bias;
   static float Last_Bias,D_Bias;
   int balance;
   Bias=sensor-ZHONGZHI;
   D_Bias=Bias-Last_Bias;
   balance=Balance_KP*Bias+D_Bias*Balance_KD;
   Last_Bias=Bias;
   return balance;
}
 }
```

Código 2: Rutina de control del péndulo

```
float Position_Control(int Encoder)
{
   static float Position_PWM,Last_Position,Position_Bias,
      Position_Differential;
   static float Position_Least;
    Position_Least =Encoder-Target;
    Position_Bias *=0.8;
    Position_Bias += Position_Least*0.2;
    Position_Differential=Position_Bias-Last_Position;
```

```
    Last_Position=Position_Bias;                          Position_PWM=
        Position_Bias*Position_KP/100+Position_Differential*Position_KD;
    return Position_PWM;
}
```

Código 3: Rutina de control del brazo

6. Implementación del control

6.1. Hardware

Para implementar este control se utilizó como controlador un Minibalace duino (Arduino UNO modificado) y como planta un péndulo invertido rotatorio comercial equipado con un motor para el movimiento del eje y encoders para la medición de los ángulos del brazo y el péndulo.

Figura 7: Péndulo usado

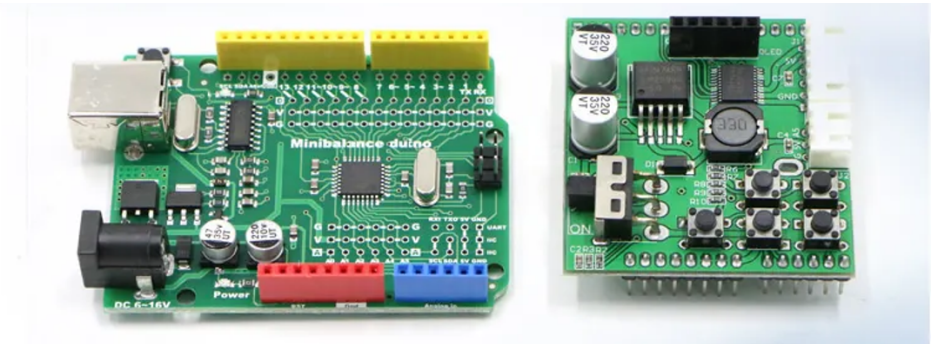


Figura 8: Minibalance duino

Un informe más detallado del hardware y su uso se encuentra en el Anexo 4.

6.2. Código implementado

Se tomó como base el código de la documentación para la implementación del control diseñado, el código modificado se puede encontrar en el Anexo 1, dentro de las modificaciones realizadas se destacan:

- Cambio en la adquisición de los datos para obtener una medición de los ángulos en radianes.
- Cambio de nombres a las variables para facilitar la comprensión del código.
- La rutina de control ahora contiene todos los cálculos y se ejecutan con igual tiempo de muestreo como se puede ver en el código 4.
- Se eliminan funciones que no son usadas para este caso.
- Para trabajar con las últimas versiones de Arduino, se cambia la librería PinChangeInt-master usada en el código original por PinChangeInterrupt, las cuales tienen la misma finalidad y sintaxis, sin embargo, es necesario el cambio de un parámetro para utilizar el pin digital 4 para leer las interrupciones. (Ver línea 141 del código del Anexo 1).
- Se añade la impresión de los datos por la interfaz serial, para permitir el uso de un programa de adquisición de datos.

```
1  void control() {
2    static double Voltage_All, Voltage_Count;
3    int Temp;
4    sei();
5    Angulo_Pendulo_Medido = Get_Adc_Average(5, 5) * 2.0 * pi / 1040.0; //
        Lectura del angulo del pendulo desde el sensor
6    Angulo_Pendulo = Angulo_Pendulo_Medido - Cero_Angulo_Pendulo;
7    V_Angular_Pendulo = Angulo_Pendulo - Angulo_Pendulo_Pasado;
8
9    Angulo_Brazo = Angulo_Brazo_Medido - Cero_Angulo_Brazo; // Lectura del
        angulo del brazo desde el sensor
10   Vel_Angular_Brazo = Angulo_Brazo - Angulo_Brazo_Pasado;
11
12   Motor =-K_Angulo_Pendulo*Angulo_Pendulo-K_Vel_Angular_Pendulo*
        V_Angular_Pendulo-K_Angulo_Brazo*Angulo_Brazo-K_Vel_Angular_Brazo*
        Vel_Angular_Brazo;  // Calculo de accion de control
13
14   Angulo_Pendulo_Pasado = Angulo_Pendulo; // Actualizacion de valores
        pasados
15   Angulo_Brazo_Pasado = Angulo_Brazo;
16
17   if (Turn_Off() == 0)
18     Set_Pwm(Motor * 255 / 12);
19
20   Temp = analogRead(0);
21   Voltage_Count++;
22   Voltage_All += Temp;
23   if (Voltage_Count == 200) Voltaje_Fuente = Voltage_All * 0.05371 / 2,
        Voltage_All = 0, Voltage_Count = 0;
24 }
```

Código 4: Rutina de control implementada

6.3. Recopilación de datos

Para la adquisición de datos, se utiliza el programa PLX-DAQ en el entorno de Microsoft Excel. Una vez el Arduino lee los datos de los sensores, estos se transmiten a través de la comunicación serial al programa PLX-DAQ, que facilita la transferencia y visualización en tiempo real en una hoja de cálculo de Excel.

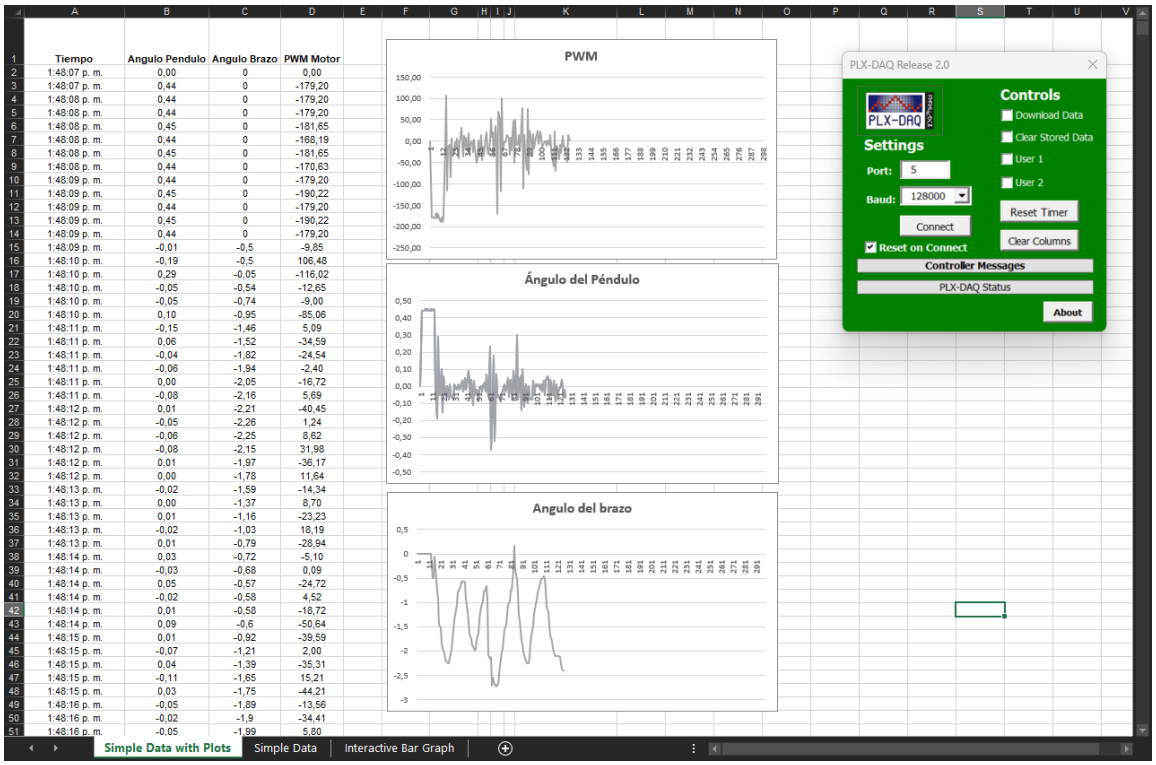


Figura 9: Toma de datos con PLX-DAQ

Un manual de uso de este software y cómo se implementa en el código Arduino se encuentra en el Anexo 5.

6.4. Ajuste de constantes del controlador

Al implementar las constantes derivadas del diseño del controlador no se logró un control exitoso del péndulo. Por ende, se procedió a obtener nuevos valores para la matriz de ganancia K mediante ensayos y ajustes con el objetivo de lograr un control satisfactorio. Los valores obtenidos son:

$$K_{implementada} = \begin{bmatrix} -0{,}3659 & -137{,}6000 & 19{,}0418 & 66{,}7800 \end{bmatrix} \tag{30}$$

Estos valores son los utilizados en el código del Anexo 1.

```
double K_Angulo_Pendulo = 19.0418, K_Vel_Angular_Pendulo = 66.7800,
    K_Angulo_Brazo = -0.3659, K_Vel_Angular_Brazo = -137.6000;
```

Código 5: Valores de K en el código

Debido a las discrepancias entre los valores implementados y los obtenidos durante el diseño, se concluye que pueden existir errores en el modelo o factores que no se están teniendo en cuenta, lo que provoca que el comportamiento de la simulación difiera de la implementación práctica.

7. Resultados prácticos

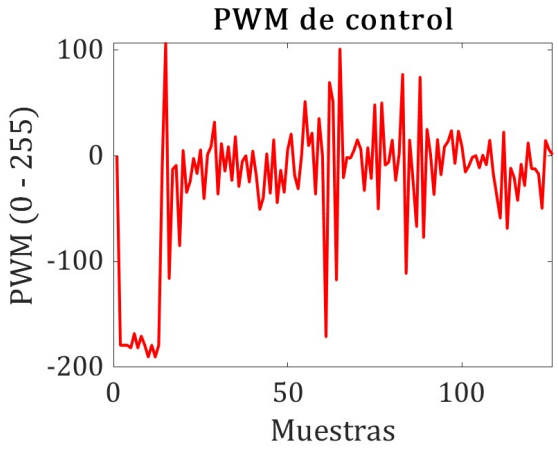
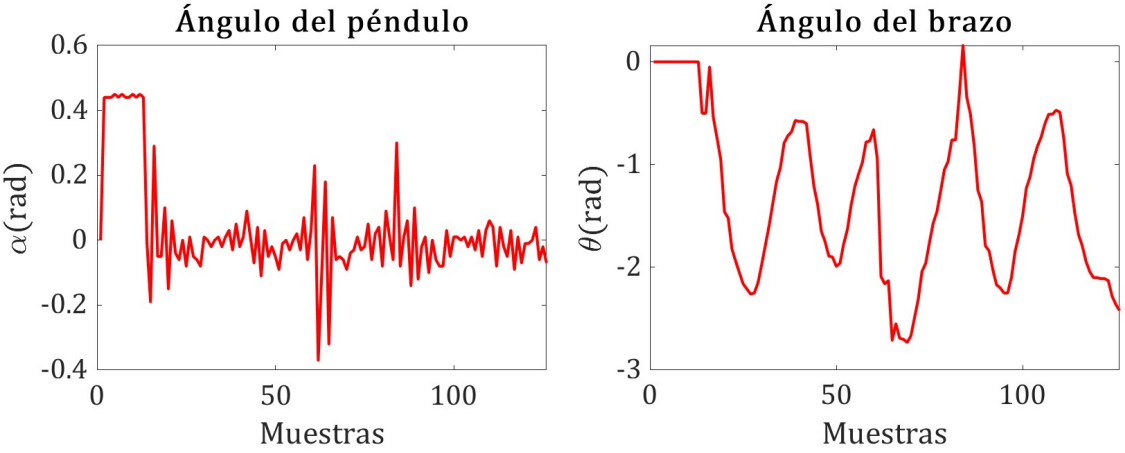


Figura 10: PWM de control



(a) Ángulo del péndulo

(b) Ángulo del brazo

Figura 11: Ángulo del brazo y del péndulo

A pesar de que los resultados prácticos difieren de la simulación, se obtiene un control aceptable que mantiene el ángulo del péndulo cercano a 0 y responde correctamente a las perturbaciones, se espera que con un mejor entendimiento del modelo y un diseño del controlador que corresponda a la realidad este resultado mejore y se obtengan resultados similares a los obtenidos en la simulación.

8. Conclusiones y recomendaciones

- Para futuros trabajos, es crucial que el modelo se ajuste fielmente a la realidad para que la fase de diseño del controlador y la simulación sean efectivas. Además, se recomienda la calibración precisa de los encoders para obtener mediciones más exactas y mejorar la calidad de los datos utilizados en el control.
- Es crucial considerar los ajustes necesarios, como cambios de escala y factores de conversión al pasar del modelo teórico a la implementación práctica. Estos ajustes son fundamentales para garantizar resultados efectivos y precisos en la aplicación práctica del modelo.

- Mantener un registro detallado de las actividades y los cambios realizados puede resultar muy útil en este tipo de trabajos. Esto no solo previene la repetición de errores, sino que también sirve como documentación las acciones llevadas a cabo.
- Es esencial tener presente que a lo largo del tiempo algunas piezas físicas pueden experimentar desgaste debido a su uso continuo. Este desgaste, natural en cualquier sistema mecánico, puede tener repercusiones directas en el funcionamiento del control.
- Es importante verificar las versiones del software utilizado, ya que las actualizaciones pueden conllevar cambios en las bibliotecas y funciones empleadas en el código.
- La recopilación de datos resulta útil para entender el desempeño del código, evaluar las modificaciones realizadas y comparar los resultados con el modelo teórico u otros códigos de control. Este proceso facilita la evaluación práctica y refinamiento continuo del control.

9. Anexos

Anexo 1: Código Implementado

Anexo 2: Código Original Proporcionado

Anexo 3: Diseño del controlador en MATLAB

Anexo 4: Informe del hardware del péndulo

Anexo 5: Manual de uso del péndulo

Anexo 6: Manual de uso de PLX-DAQ

10. Bibliografía

[1] Playfultechnology, "Playfultechnology/PID-invertedpendulum," GitHub, https://github.com/playfultechnology/pid-invertedpendulum (accessed Dec. 6, 2023).

[2] H. Guo, J. Wu, Z. Yin, and W. Zhang, "Comparative study of furuta pendulum based on LQR and PID Control," Journal of Physics: Conference Series, vol. 2562, no. 1, p . 012075, 2023. doi:10.1088/1742-6596/2562/1/012075