Actividad Práctica Nº2 - Sistemas de Control II

Diseño de controladores considerando la dinámica del error y la magnitud de la acción de control en sistemas no lineales multivariables

Gonzalez, Bruno

9 de junio de 2025

Índice

1. Implementación del PID al motor **3**
1.1. Resultados con constantes iniciales sugeridas 4
1.2. Ajuste de parámetros del PID 4
1.3. Limitación de la acción de control 5

2. Ítem 1: Control del ángulo del motor en espacio de estados **6**
2.1. Modelo del sistema y diseño del controlador LQR 6
2.1.1. Sistema ampliado con integrador del error 6
2.1.2. Cálculo del controlador LQR 7
2.2. Simulación 7
2.2.1. Definición de referencias y perturbaciones 8
2.2.2. Resultados 8
2.3. Análisis de sensibilidad del parámetro R en LQR 8
2.4. Comparación entre control PID y control LQR implementados 9
2.5. Implementación de observador 10
2.5.1. Modelo del sistema y controlador 10
2.5.2. Diseño del observador de Luenberger 10
2.5.3. Simulación comparativa 11
2.6. Efecto de no linealidad en acción de control 12

3. Ítem 2: Comparación entre distintos parámetros físicos del motor (Identificado vs Real) **14**

4. Ítem 3: Sistema no lineal de cuatro variables de estado - Carro-péndulo en equilibrio estable **15**
4.1. Modelado del sistema 15
4.2. Controlador LQR con sistema ampliado 15
4.3. Simulación del sistema 16
4.4. Efecto de la zona muerta 17

5. Conclusiones **19**

6. Correcciones **20**
6.1. Ajuste en la acción de control 20

7. Enlaces y recursos **21**

1. Implementación del PID al motor

El objetivo de esta sección es implementar un controlador PID digital para controlar el ángulo del eje de un motor de corriente continua. Para ello, se emplea un modelo del motor con integración de Euler, y se aplica una perturbación externa realista a través de una señal de torque obtenida experimentalmente para evaluar el comportamiento del sistema.

El controlador PID se implementa en su forma discreta, mediante los coeficientes A_1, B_1 y C_1, que permiten calcular la acción de control $u[k]$ directamente en función de los errores actuales y pasados:

$$u[k] = u[k-1] + A_1 \cdot e[k] + B_1 \cdot e[k-1] + C_1 \cdot e[k-2]$$

```
def modmotor(t_etapa, xant, accion, torque_ext):
    Laa = 0.0047
    J = 0.00233
    Ra = 2.27
    B = 0.00131
    Ki = 0.25
    Km = 0.25
    h = 1e-7
    omega, wp, theta = xant
    steps = int(t_etapa / h)
    for _ in range(steps):
        wpp = (-wp * (Ra * J + Laa * B) - omega * (Ra * B + Ki * Km) + accion * Ki -
            torque_ext) / (J * Laa)
        wp += h * wpp
        omega += h * wp
        theta += h * omega
    return [omega, wp, theta]
```

Listing 1: Modelo del motor con integración de Euler

```
# Coeficientes PID
Kp = 10
Ki = 10
Kd = 0.01
Ts = t_etapa
A1 = ((2*Kp*Ts)+(Ki*(Ts**2))+(2*Kd)) / (2*Ts)
B1 = (-2*Kp*Ts + Ki*(Ts**2) - 4*Kd) / (2*Ts)
C1 = Kd / Ts

# Inicializacon de variables
X = [0.0, 0.0, 0.0]
e = np.zeros(N + 3)
acc = np.zeros(N)
u = 0

# Simulacion del sistema con PID
torque_aplicado_arr = np.zeros(N)
for ii in range(N):
    k = ii + 2
    t_actual = ii * t_etapa
    torque_aplicado = float(torque_data(t_actual))
    torque_aplicado_arr[ii] = torque_aplicado
    X = modmotor(t_etapa, X, u, torque_aplicado)
    e[k] = wRef - X[2]
    u += A1 * e[k] + B1 * e[k-1] + C1 * e[k-2]
    # u = np.clip(u, -12, 12)  # Limitacion de la accion de control opcional
    acc[ii] = u
```

Listing 2: Implementación del PID digital y simulación del sistema

1.1. Resultados con constantes iniciales sugeridas

Se utilizó inicialmente la sintonía sugerida por la consigna: $K_p = 0{,}1$, $K_i = 0{,}01$, $K_d = 5$. El resultado se muestra a continuación:

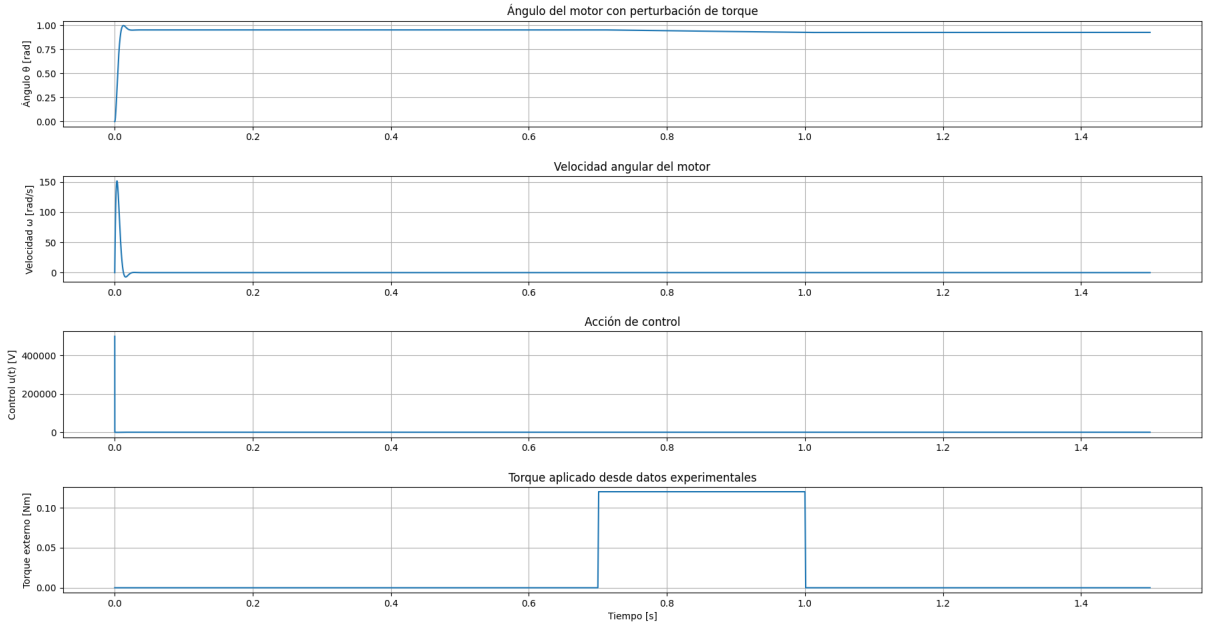

Figura 1: Respuesta del sistema con PID inicial: $K_p = 0{,}1$, $K_i = 0{,}01$, $K_d = 5$

Se observa que el sistema no logra eliminar el error en estado estable y la acción de control alcanza valores superiores a los 400000V, lo que evidencia que esta configuración es inviable físicamente para un sistema real. Esto motivó un ajuste de parámetros.

1.2. Ajuste de parámetros del PID

Se procedió a sintonizar nuevamente el PID buscando una respuesta más suave y estable, con una acción de control acotada. Se establecieron: $K_p = 10$, $K_i = 10$, $K_d = 0{,}01$.

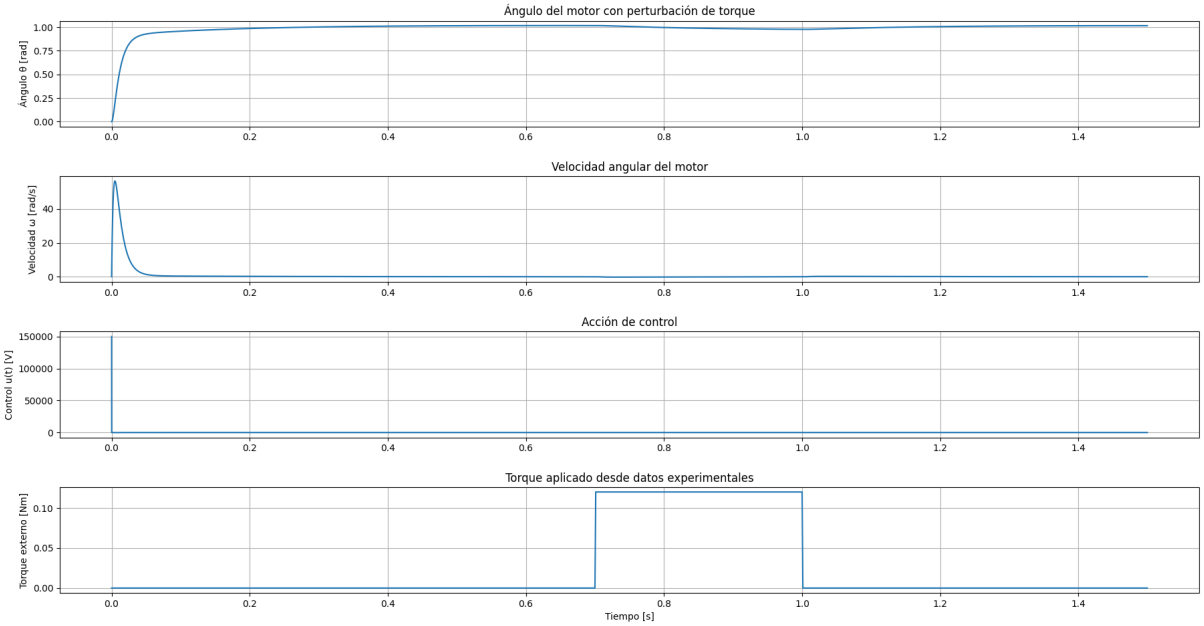

Figura 2: Respuesta con parámetros ajustados: $K_p = 10$, $K_i = 10$, $K_d = 1$

Con esta configuración se logró alcanzar satisfactoriamente la referencia de 1 rad, y la acción de control si bien se redujo en comparación con la iteración anterior, continúa en valores inviables durante el transitorio inicial. Además, frente a la perturbación se observa cómo la salida del sistema se ve levemente afectada; esto puede ser corregido reajustando la acción derivativa, pero penaliza directamente en la magnitud de la acción de control.

1.3. Limitación de la acción de control

Finalmente, se incorporó una saturación de la acción de control a un máximo absoluto de 5V para simular condiciones más realistas.

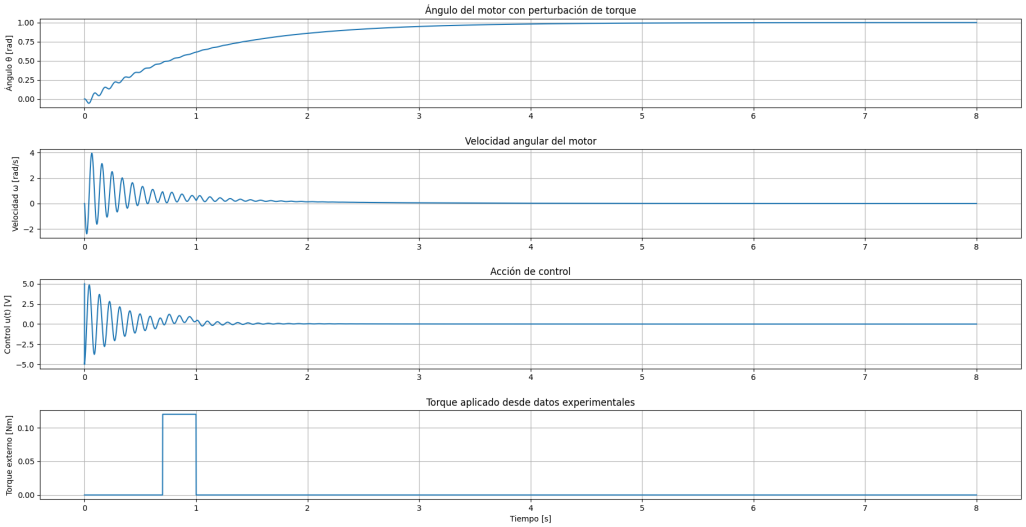

Figura 3: Respuesta del sistema con acción de control limitada a ±5V

Aplicando la restricción de la acción de control para tener una solución físicamente viable y reajustando las constantes del PID, el controlador fue capaz de regular el ángulo

del motor a costa de un aumento sustancial del tiempo de estabilización del sistema.

2. Ítem 1: Control del ángulo del motor en espacio de estados

El objetivo de este ítem es diseñar un controlador en espacio de estados que regule el ángulo del eje del motor frente a cambios alternantes en la referencia, y además observar el comportamiento del sistema frente a una perturbación externa que representa un torque de carga. Se parte del modelo del motor en variables de estado, y se implementa un controlador LQR.

2.1. Modelo del sistema y diseño del controlador LQR

Se consideró un modelo lineal del motor eléctrico, donde las variables de estado son: corriente de armadura i_a, velocidad angular ω, y posición angular θ:

$$x(t) = \begin{bmatrix} i_a(t) \\ \omega(t) \\ \theta(t) \end{bmatrix}, \quad \dot{x}(t) = Ax(t) + Bu(t)$$

Las matrices del sistema se definen a partir de parámetros físicos reales del motor:

```
Ra = 2.27
La = 0.0047
Ki = 0.25
Km = 0.25
J = 0.00233
B = 0.00131

# x = [ia, omega, theta]

A = np.array([
    [-Ra / La,    -Km / La,     0],
    [Ki / J,     -B / J,        0],
    [0,           1,            0]
])
Bmat = np.array([
    [1 / La],
    [0],
    [0]
])
C = np.array([[0, 0, 1]])  # salida:  ngulo
D = np.array([[0]])
```

Listing 3: Definición de matrices del sistema

El controlador se diseña mediante LQR. Pero dado que se desea realizar un seguimiento de una referencia distinta de cero, es necesario agregar un integrador del error. Además se diseñó el controlador penalizando el error en el ángulo ya que es la variable de interés y regulando el esfuerzo de control para cumplir con la consigna.

2.1.1. Sistema ampliado con integrador del error

Se define una nueva variable de estado $z(t)$ como la integral del error:

$$\dot{z}(t) = r(t) - y(t) = r(t) - Cx(t)$$

Agregando esta dinámica al sistema original, se obtiene el modelo ampliado:

$$A_e = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix}, \quad B_e = \begin{bmatrix} B \\ 0 \end{bmatrix}$$

```
Ae = np.block([
[A, np.zeros((3, 1))],
[-C, np.zeros((1, 1))]
])
Be = np.vstack([Bmat, np.zeros((1, 1))])
```

Listing 4: Construcci'on del sistema ampliado

2.1.2. Cálculo del controlador LQR

Se define la funcional de costo:

$$J = \int_0^\infty (x_a Q x_a + uRu)\, dt$$

donde $x_a = [i_a, \omega, \theta, z]^T$ es el vector de estados ampliado. Se seleccionan penalizaciones que den mayor peso al ángulo y al integrador del error:

```
Q = np.diag([0.1, 0.1, 100, 10000])
R = np.array([[5]])
P = solve_continuous_are(Ae, Be, Q, R)
K = np.linalg.inv(R) @ Be.T @ P
Kx = K[:, :3]
Ki_int = K[:, 3]
```

Listing 5: Resoluci'on de Riccati y separaci'on de ganancias

2.2. Simulación

Para realizar la simulación se determinó el tiempo de integración de Euler a partir del polo más rápido del sistema

$$tr = \frac{ln(0{,}95)}{\lambda_{\text{más rápido}})}$$

Y el tiempo de simulación se basó en el período de la señal de referencia para poder evaluar el comportamiento total.

Se simuló el sistema completo con saturación de control en ±5V para no exceder los límites dispuestos en la acción de control:

```
for k in range(N - 1):
ia, omega, theta = X[:, k]
torque_ext[k] = torque_periodico(t[k])
error = ref[k] - theta
e_int[k + 1] = e_int[k] + h * error
u_k = (-Kx @ X[:, k] - Ki_int * e_int[k + 1]).item()
u_k = np.clip(u_k, -5, 5)  #Limitacion de la accion de control
u[k] = u_k

ia_dot = (-Ra * ia - Km * omega + u_k) / La
omega_dot = (Ki * ia - B * omega - torque_ext[k]) / J
theta_dot = omega

X[0, k + 1] = ia + h * ia_dot
X[1, k + 1] = omega + h * omega_dot
X[2, k + 1] = theta + h * theta_dot
```

Listing 6: Simulación con sistema ampliado y controlador LQR

2.2.1. Definición de referencias y perturbaciones

Se definió una referencia angular que cambia cada 5 segundos entre $+\frac{\pi}{2}$ y $-\frac{\pi}{2}$, utilizando una señal cuadrada. Además, se agregó un torque perturbador periódico de 5s, que toma el valor 0.12 Nm durante 1 segundo por ciclo:

```python
def torque_periodico(t):
    t_mod = t % 5
    return 0.12 if 2 <= t_mod < 3 else 0.0
```

Listing 7: Definición del torque perturbador

2.2.2. Resultados

A continuación se muestran los resultados obtenidos:

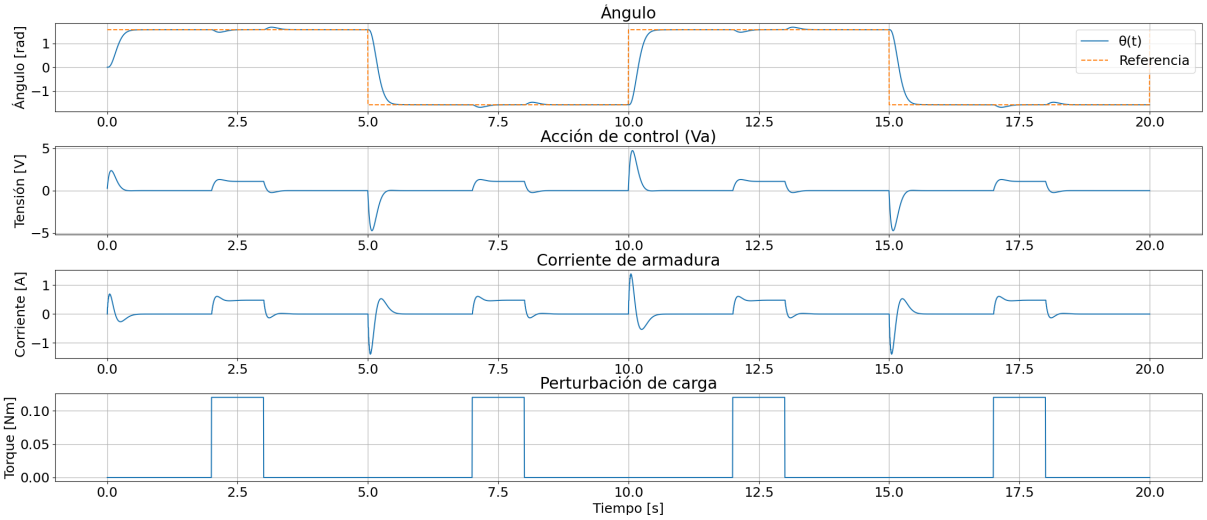

Figura 4: Respuesta del sistema con controlador LQR frente a referencia alternante y torque de carga.

La incorporación del sistema ampliado en conjunto con el controlador LQR permite dotar al sistema de una acción integral sobre el error. Esto garantiza error nulo en régimen permanente y robustez frente a perturbaciones. El controlador resultante ajusta simultáneamente la dinámica del sistema y la magnitud del error acumulado, generando una acción de control eficiente y físicamente realizable.

2.3. Análisis de sensibilidad del parámetro R en LQR

A continuación se evalúa cómo influye la elección del parámetro R en el desempeño del sistema con controlador LQR. Se mantuvieron constantes los pesos $Q = \text{diag}(1, 1, 40)$ y se simularon las respuestas para $R = 2, 5, 10, 20, 30$. Las figuras siguientes muestran la evolución del ángulo del motor y la acción de control $u(t)$ para cada caso frente a cambios en la referencia y de la presencia de un torque de carga como perturbación. Cabe destacar que en este caso no se amplió al sistema, dejando la salida a lazo abierto debido a la falta del integrador de error.

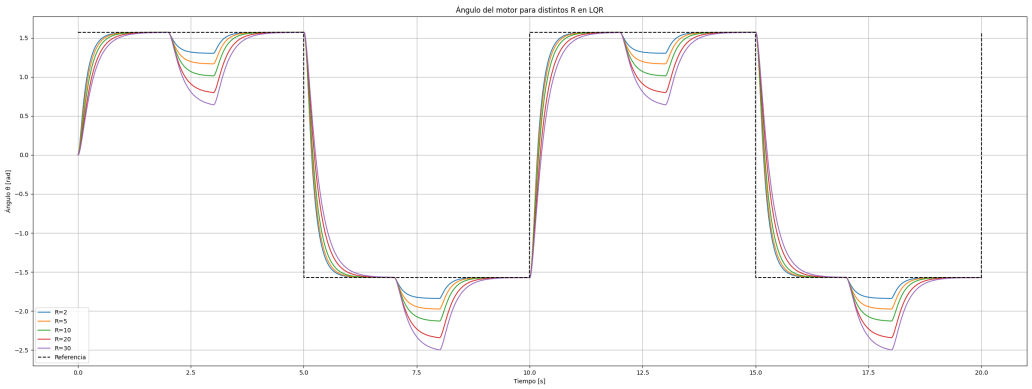

Figura 5: Comparación del ángulo del motor para diferentes valores de R en el controlador LQR.

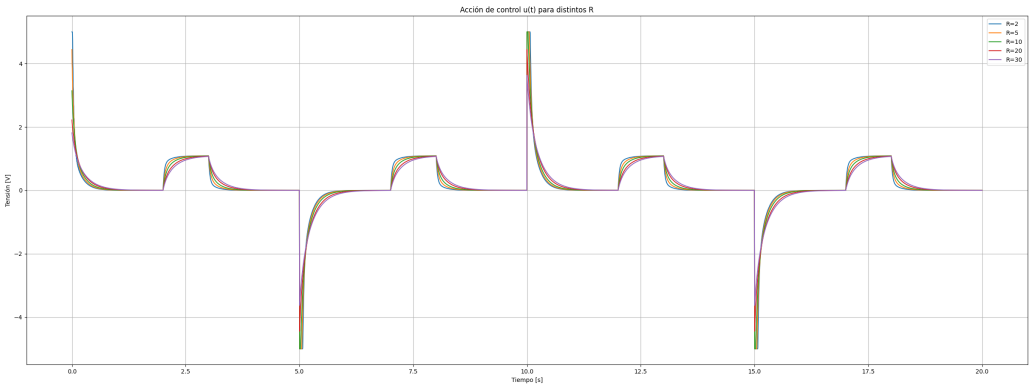

Figura 6: Acción de control $u(t)$ para distintos valores del parámetro R.

Como se observa:

- Valores bajos de R generan una acción de control más agresiva, con mejor seguimiento pero mayor esfuerzo.
- Valores altos de R suavizan la señal de control pero deterioran la capacidad de seguir la referencia rápidamente.

Esto permite seleccionar R en función de los requerimientos de precisión vs. consumo de energía o limitaciones físicas del actuador.

2.4. Comparación entre control PID y control LQR implementados

Ambos controladores, PID y LQR, permitieron regular el ángulo del eje del motor en presencia de perturbaciones externas. Sin embargo, existen diferencias importantes en cuanto al desempeño y la naturaleza del control aplicado:

- El controlador PID opera únicamente sobre el error entre la referencia y la salida, sin considerar directamente la evolución de otras variables internas del sistema como la velocidad angular o la corriente de armadura. Esto limita su capacidad de anticipación frente a perturbaciones o dinámicas complejas, especialmente si no se realiza una sintonización fina de sus parámetros.

- En contraste, el LQR se basa en el estado completo del sistema y utiliza retroalimentación de todas las variables de estado.
- El LQR permite un diseño sistemático mediante la elección de matrices Q y R, lo que brinda un mayor control sobre los compromisos entre esfuerzo de control y precisión de seguimiento, mientras que el PID implementado depende más de prueba y error para su correcta sintonía.

En conclusión, el controlador LQR demostró ser una mejor opción al aprovechar información completa del sistema para lograr un control más equilibrado y preciso gracias a la presencia del funcional de costo y la posibilidad de controlar las relaciones de costo-compromiso entre precisión (con el parámetro Q) y magnitud de la acción de control (con el parámetro R).

2.5. Implementación de observador

En este ítem se plantea la situación en la que no es posible medir directamente la corriente de armadura del motor, pero sí pueden medirse la velocidad angular y la posición del eje. Bajo esta restricción, se propone un esquema de control basado en el diseño de un **observador de Luenberger**, el cual permite estimar el estado completo del sistema a partir de las salidas disponibles.

2.5.1. Modelo del sistema y controlador

Se mantiene el modelo del motor en espacio de estados y el controlador diseñado en el ítem anterior. Es decir, se trabaja con el **sistema ampliado** que incorpora un integrador del error, y con el **controlador LQR** previamente calculado:

```
Q = np.diag([0.1, 0.1, 100, 10000])
R = np.array([[5]])
P = solve_continuous_are(Aamp, Bamp, Q, R)
K = np.linalg.inv(R) @ (Bamp.T @ P)
Kx = K[:, :3]
Ki_int = K[:, 3]
```

Listing 8: Controlador LQR sobre sistema ampliado

2.5.2. Diseño del observador de Luenberger

Para estimar las variables no medibles del sistema (en particular la corriente), se diseña un observador de Luenberger. Dado que la salida medida es el ángulo, se utiliza la **transposición del sistema** para aplicar el diseño LQR en la forma dual:

```
Ao = A.T
Bo = C.T
Qo = np.diag([1, 0.1, 0.1])
Ro = np.array([[5]])
Po = solve_continuous_are(Ao, Bo, Qo, Ro)
Ko = np.linalg.inv(Ro) @ (Bo.T @ Po)
```

Listing 9: Diseño del observador mediante LQR dual

La ganancia K_o del observador permite estimar el vector de estado completo $\hat{x}(t)$ a partir de la salida $y(t)$ y su estimación $\hat{y}(t) = C\hat{x}(t)$.

2.5.3. Simulación comparativa

Se comparan dos escenarios:

- **Control sin observador:** se asume disponibilidad total de los estados.
- **Control con observador:** la corriente es estimada, y se usa $\hat{x}(t)$ para el control.

Ambos sistemas se simulan bajo la misma referencia alternante para el ángulo, utilizando un integrador de Euler con paso fijo. La señal de control y la estimación de la corriente se actualizan en tiempo real.

A continuación se muestra la evolución del ángulo, la corriente de armadura (real y estimada), la acción de control en ambos casos y el error de observación de la corriente:

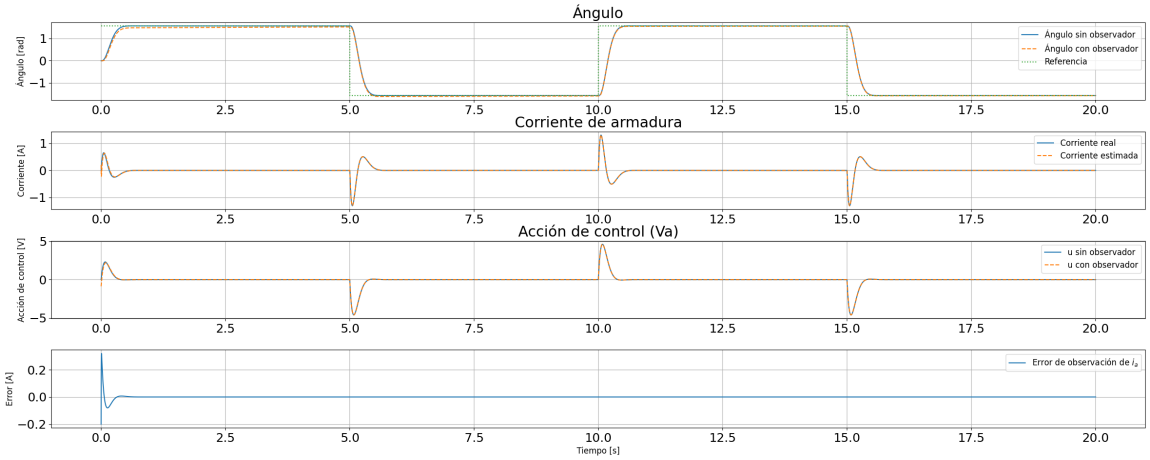

Figura 7: Comparación entre el sistema con medición completa del estado y el sistema con observador.

Se observa que el observador logra reconstruir con alta precisión la corriente, incluso a pesar de no tener acceso directo a dicha medición. La acción de control también se ajusta de forma adecuada a partir del estado estimado, y la salida del sistema (ángulo) presenta un comportamiento casi idéntico al obtenido en el caso ideal.

En la Figura 8 se presentan de forma mas detallada las curvas comparativas entre el sistema controlado utilizando los estados reales (sin observador) y el sistema que emplea un observador de Luenberger para estimar los estados a partir de la salida medida.

En la parte superior se observa la evolución de la **corriente de armadura**, comparando la señal real con la estimada por el observador. Puede apreciarse que, aunque inicialmente hay una diferencia debido al error de estimación inicial, el observador converge rápidamente y logra reconstruir la señal con alta precisión.

En la parte inferior se muestra la **acción de control** aplicada en ambos casos. La diferencia entre ambas señales es leve y transitoria, lo cual confirma que el observador permite mantener un desempeño casi idéntico al esquema que dispone de medición completa.

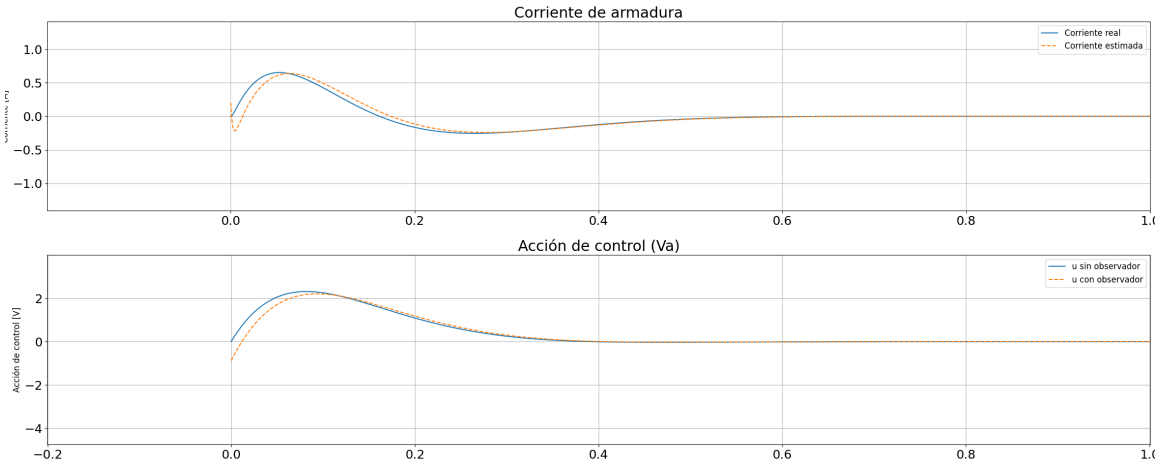

Figura 8: Comparación entre la corriente y acción de control en el sistema con observador y sin observador.

El uso de un observador de Luenberger permite recuperar las variables de estado no medibles. El error de observación es pequeño y tiende a cero rápidamente, lo cual valida la elección del observador y permite una implementación robusta incluso ante limitaciones de sensado.

2.6. Efecto de no linealidad en acción de control

Con el objetivo de analizar el comportamiento del sistema frente a no linealidades del actuador, se introdujo una zona muerta en la acción de control. La presencia de esta no linealidad puede conducir a la aparición de un ciclo límite, es decir, una trayectoria cerrada en el plano de fase que representa una oscilación permanente.

En este caso, se utilizó un controlador LQR con una fuerte penalización sobre el error angular para forzar a la generación del ciclo límite. La acción de control resultante, como se muestra en la Figura 10, presenta un comportamiento conmutado permanente, manteniéndose dentro de la banda establecida por la zona muerta.

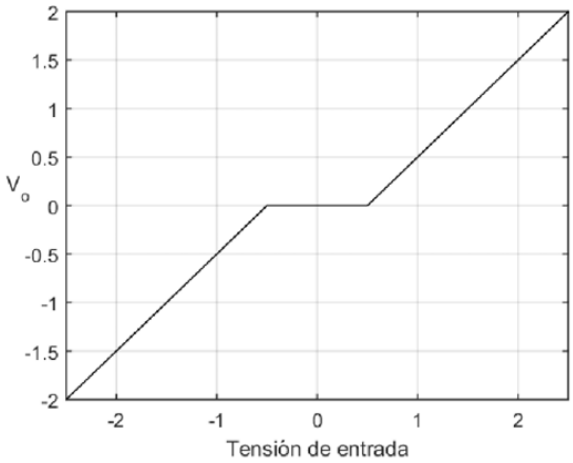

Figura 9: Acción de control con zona muerta.

La respuesta temporal del sistema también evidencia oscilaciones persistentes en el ángulo del eje, como se aprecia en la Figura 10.

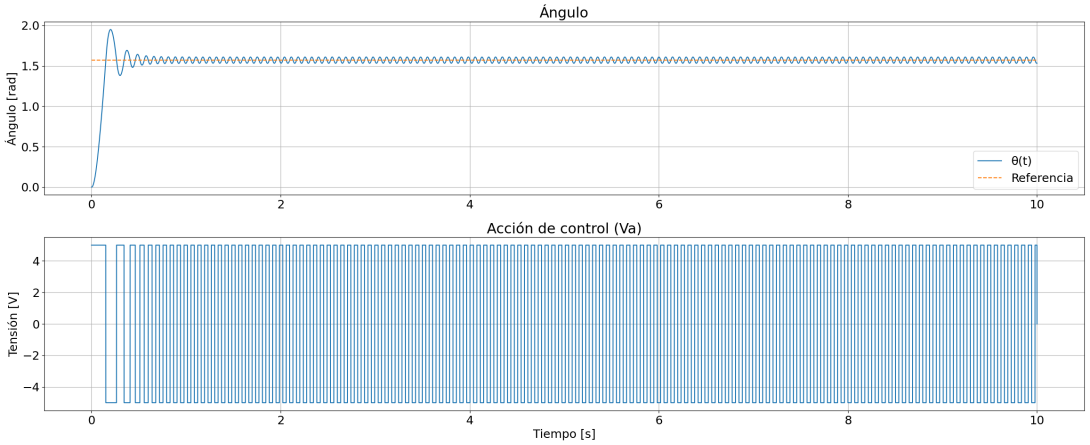

Figura 10: Respuesta del sistema ante una referencia fija con zona muerta en la acción de control.

Finalmente, en el plano de fase θ vs. ω se observa una trayectoria cerrada, típica de un **ciclo límite**. Esta trayectoria se mantiene oscilando en torno al punto de equilibrio, sin converger al mismo, lo cual confirma la influencia directa de la no linealidad sobre la dinámica del sistema.

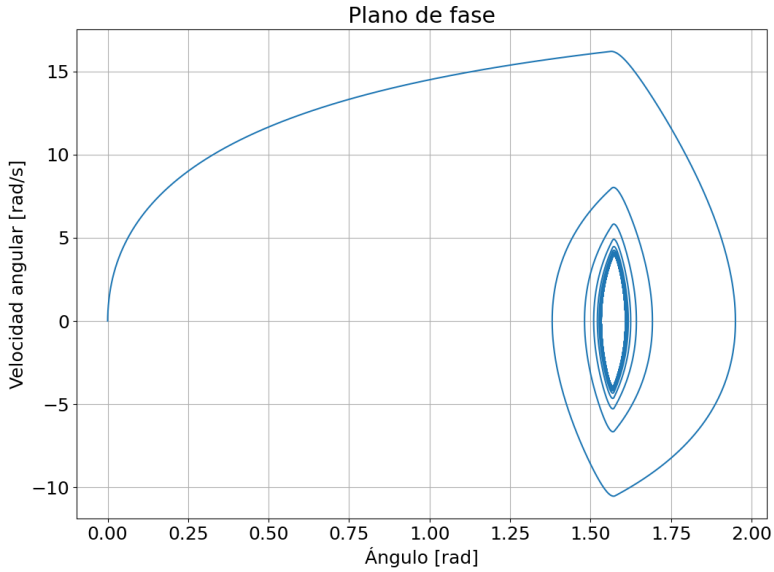

Figura 11: Plano de fase con trayectoria cerrada. Aparición de un ciclo límite debido a la zona muerta.

3. Ítem 2: Comparación entre distintos parámetros físicos del motor (Identificado vs Real)

Para realizar el ítem anterior se realizó las simulaciones utilizando los parámetros físicos reales del motor. A continuación, se procede a simular el mismo controlador pero con dos juegos de parámetros físicos diferenetes, motor 1 corresponde al real mientras que motor 2 corresponde al motor identificado en la actividad práctica 1.

```
# Parámetros físicos de dos motores distintos
params_motor_1 = (2.27, 0.0047, 0.25, 0.25, 0.00233, 0.00131, "Motor 1")
params_motor_2 = (2.25, 0.005, 0.259, 0.25, 0.00284, 0.0014, "Motor 2")
```

A continuación se observan los resultados:

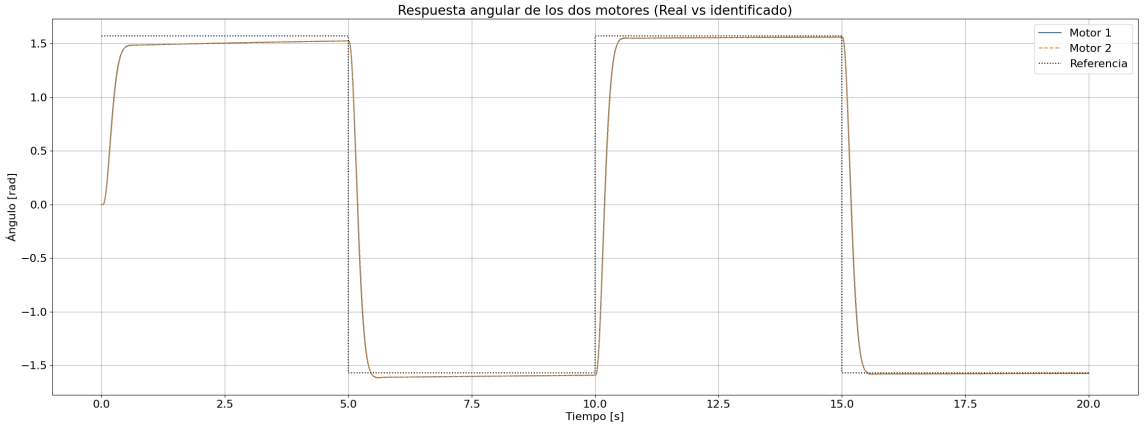

Figura 12: Respuesta transitoria para ambos motores

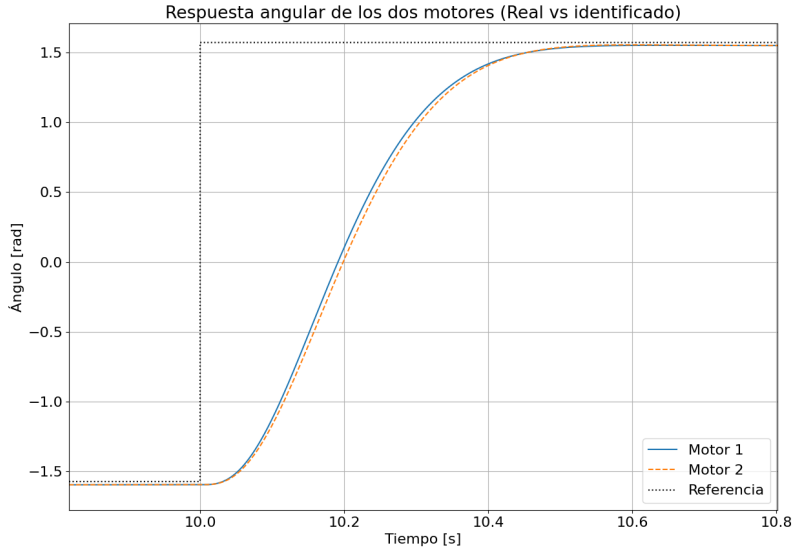

Figura 13: Detalle de la respuesta transitoria para ambos motores

4. Ítem 3: Sistema no lineal de cuatro variables de estado - Carro-péndulo en equilibrio estable

Este caso de estudio aborda el modelado y control de un sistema no lineal compuesto por un carro móvil sobre el cual se articula un péndulo. La dinámica del sistema se describe mediante cuatro variables de estado: desplazamiento δ, velocidad del carro $\dot{\delta}$, ángulo del péndulo ϕ y velocidad angular $\dot{\phi}$. El objetivo de control consiste en desplazar el carro hasta una posición deseada mientras se estabiliza el péndulo en la posición de equilibrio estable, correspondiente a $\phi = \pi$ (péndulo colgando hacia abajo).

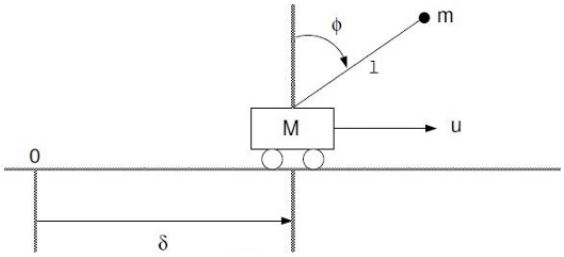

Figura 14: Sistema del péndulo.

4.1. Modelado del sistema

Se linealiza el sistema en torno al equilibrio $\phi = \pi$ para obtener el modelo en espacio de estados. Las matrices del sistema en tiempo continuo quedan definidas como:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{F}{M} & -\frac{mg}{M} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{F}{lM} & -\frac{g(m+M)}{lM} & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{lM} \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

donde m es la masa del péndulo, M la masa del carro, l la longitud del péndulo, F el coeficiente de fricción, y g la aceleración gravitatoria.

4.2. Controlador LQR con sistema ampliado

Con el fin de asegurar el seguimiento de referencias de desplazamiento, se implementa un controlador óptimo LQR sobre un sistema ampliado que incluye un integrador del error:

$$A_a = \begin{bmatrix} A & \mathbf{0} \\ -C_\delta A & 0 \end{bmatrix}, \quad B_a = \begin{bmatrix} B \\ -C_\delta B \end{bmatrix}$$

donde C_δ es la primera fila de C, correspondiente a la salida de desplazamiento.

Dado que sólo se dispone de mediciones de δ y ϕ, se diseña un observador de Luenberger para estimar las variables no medidas:

$$\dot{\hat{x}} = A\hat{x} + Bu + K_o(y - \hat{y})$$

4.3. Simulación del sistema

Se realizan dos tipos de simulaciones:

1. Con referencia constante de 10 m, comparando el desempeño del sistema con y sin observador.

2. Caso completo: el sistema se desplaza 10 m, luego se incrementa la masa del péndulo a 1 kg para simular la recogida de una carga, y se retorna a la posición inicial con la nueva carga.

Para la simulación del sistema controlado en tiempo continuo, se utilizó un paso de integración $\Delta t = 1x10^{-4}$ s. Este valor fue seleccionado considerando el polo más rápido del sistema linealizado, con el objetivo de garantizar estabilidad.

El tiempo total de simulación fue de $T = 45$ s, lo cual permitió observar el comportamiento completo del sistema en ambos escenarios: movimiento hasta los 10 m, aplicación de la carga de 1 kg, y retorno al origen.

Rango posible para el tiempo de muestreo en implementación digital Dado que el controlador LQR fue diseñado inicialmente en tiempo continuo, para su implementación digital en un microcontrolador debe discretizarse adecuadamente. El tiempo de muestreo T_s debe ser significativamente menor que el tiempo característico del sistema, en particular del polo que describe la dinámica más rápida del sistema.

Una regla práctica consiste en elegir T_s tal que:

$$T_s < \frac{1}{10 f_{\max}}$$

donde $f_{\max}$ es la frecuencia asociada al polo más rápido del sistema.

Las siguientes gráficas muestran la evolución temporal del desplazamiento, ángulo del péndulo y acción de control, comparando el comportamiento real del sistema con la estimación provista por el observador.

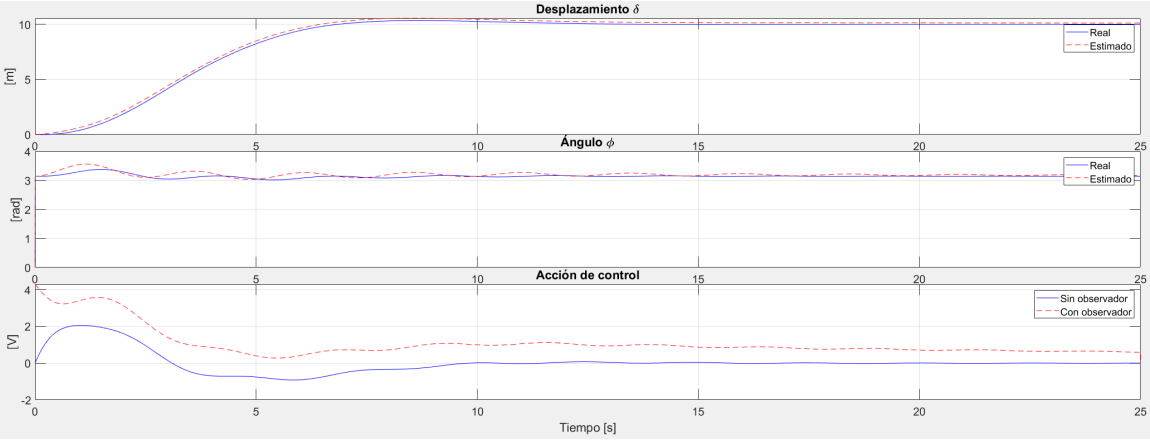

Figura 15: Evolución temporal del sistema para desplazamiento de 10m.

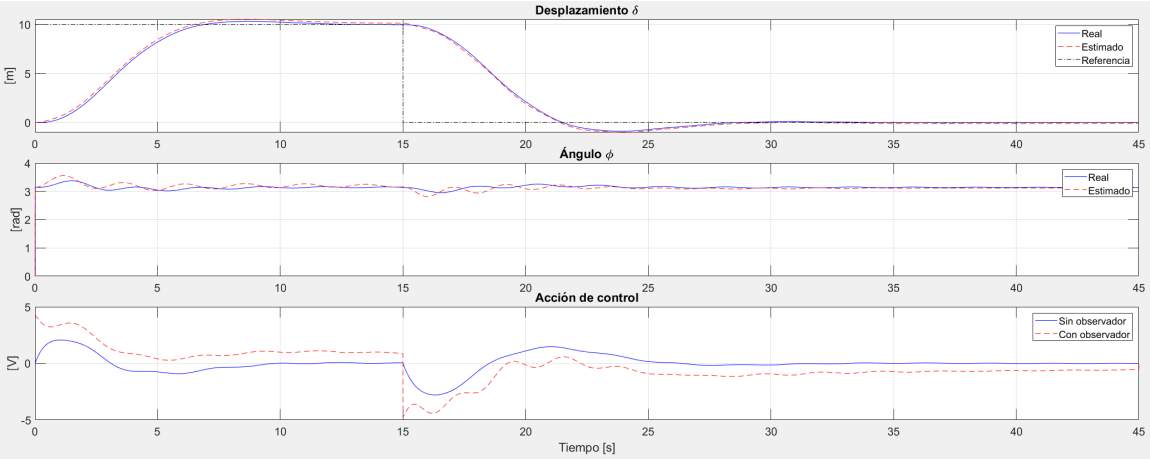

Figura 16: Evolución temporal del sistema para desplazamiento completo con recogida de carga.

En ambos casos, se observa que el sistema logra estabilizar el péndulo en $\phi = \pi$ mientras sigue correctamente la referencia de desplazamiento. El observador presenta un buen desempeño, con pequeñas oscilaciones, las cuales se atenúan con el tiempo. Se destaca una mayor acción de control en el sistema con observador al iniciar el transitorio debido al error de estimación inicial.

4.4. Efecto de la zona muerta

Finalmente, se introduce una zona muerta en la acción de control de $\pm 1\,\mathrm{V}$ para simular las limitaciones del actuador.

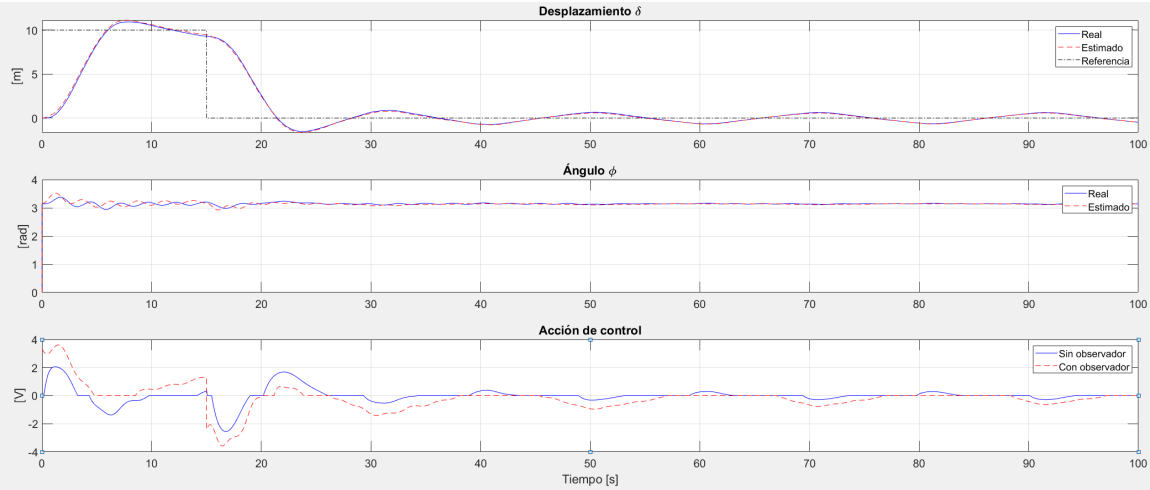

Figura 17: Evolución temporal del sistema para desplazamiento completo con zona muerta en acción de control.

Se destaca la presencia de una oscilación que no permite estabilizar el carro en la ubicación deseada, efecto similar al observado en el caso de estudio anterior. Graficando el plano de fase se observa cómo se genera el ciclo límite correspondiente cuando el carro intenta volver al punto de partida.

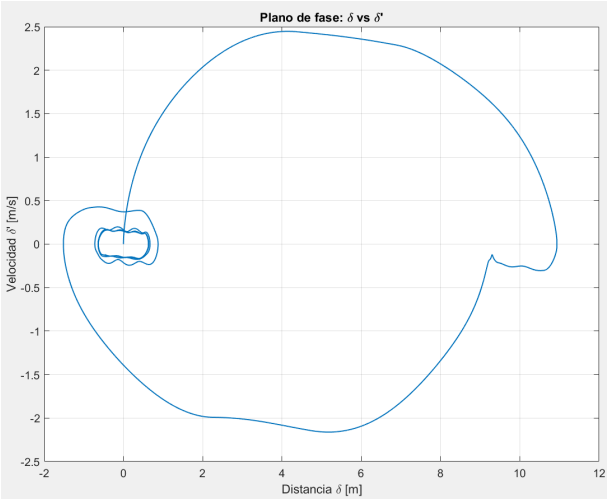

Figura 18: Plano de fase evidenciando presencia de ciclo límite como consecuencia de la zona muerta en la acción de control

5. Conclusiones

A partir del desarrollo de esta actividad práctica se lograron consolidar conocimientos centrales vinculados al diseño y análisis de sistemas de control en diferentes niveles de complejidad. Los principales aspectos alcanzados pueden resumirse a continuación:

1. Se diseñaron controladores PID y LQR, evaluando su desempeño ante diferentes condiciones iniciales, perturbaciones y restricciones físicas. Esto permitió adquirir criterio para seleccionar controladores adecuados en función del modelo y los requerimientos del sistema.

2. Se expresó el modelo dinámico de distintos procesos reales, incluyendo un motor de corriente continua y un sistema carro-péndulo. En ambos casos, se obtuvo la representación linealizada en espacio de estados y se logró simular al sistema a través del método de integración por Euler.

3. Se diseñaron observadores de estado para estimar variables no medibles, como la corriente de armadura en el motor. Las simulaciones mostraron que el desempeño del sistema con observador fue comparable al caso ideal con estados accesibles, evidenciando la aplicabilidad de esta técnica en escenarios reales.

4. Se incorporó la acción integral mediante ampliación del sistema, lo que permitió garantizar el seguimiento de referencias incluso frente a perturbaciones externas o cambios abruptos en la dinámica (como el aumento de la carga en el carro-péndulo).

5. Se analizó la influencia de parámetros de diseño del LQR (como la matriz R) en la magnitud y agresividad de la acción de control, fortaleciendo el entendimiento del compromiso entre esfuerzo de control y desempeño deseado.

6. Se modelaron y simularon sistemas no lineales en tiempo continuo, abordando conceptos fundamentales como el equilibrio, la linealización, la aparición de ciclos límite y la estabilidad del sistema. Esto se evidenció, por ejemplo, al observar el impacto de la zona muerta en la acción de control, la cual impide alcanzar el equilibrio y genera oscilaciones persistentes.

7. Finalmente, se fortalecieron habilidades transversales como la implementación de algoritmos de simulación en MATLAB y Python, el análisis crítico del comportamiento dinámico de sistemas físicos, la adaptación del diseño ante nuevas condiciones y la capacidad de interpretar, ajustar e implementar estrategias de control en entornos simulados teniendo en cuenta una posible realización en un sistema físico real y su factibilidad.

6. Correcciones

6.1. Ajuste en la acción de control

Corrección en la implementación de la zona muerta

Inicialmente, la zona muerta en la acción de control fue implementada mediante una simple condición tipo `if`, anulando la señal de control si se encontraba dentro del umbral:

```
if −0.25 < u_k < 0.25:
    u_k = 0.0
```

Sin embargo, la zona muerta debe anular únicamente la parte central de la entrada, y desplazar la respuesta de forma continua fuera de ese umbral.

Por recomendación del profesor, se corrigió el modelo utilizando una estructura más adecuada, donde se define la salida del actuador u_o como:

$$u_o = \begin{cases} u - \text{ZM} \cdot \text{sign}(u), & \text{si } |u| > \text{ZM} \\ 0, & \text{si } |u| \leq \text{ZM} \end{cases}$$

En Python, se implementó de forma vectorial como:

```
def zona_muerta(u, zm=0.25):
    return np.where(np.abs(u) > zm, u − zm * np.sign(u), 0.0)
```

Esta corrección garantiza una correcta representación del actuador. Observando la resupuesta del actuador presente en el sistema del motor en ambos casos para compararla:

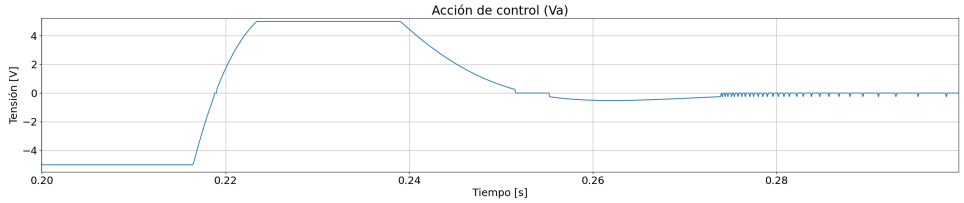

Figura 19: Respuesta temporal de la acción de control antes de las correcciones

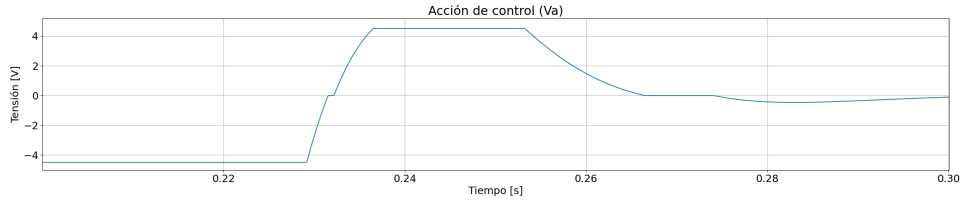

Figura 20: Respuesta temporal de la acción de control luego de las correcciones

Eliminación del recorte numérico en la acción de control

En las primeras etapas del desarrollo, la acción de control fue limitada numéricamente mediante la función `np.clip()`, con el fin de evitar "a la fuerza" valores excesivos en la señal de salida:

```
u_k = np.clip(u_k, −5, 5)
```

En realidad, el objetivo del diseño en espacio de estados con simulaciones de Euler es verificar si las variables de estado y la acción de control permanecen dentro de los límites físicos del sistema. Si esto no se cumple, lo correcto no es recortar artificialmente la acción de control, sino reformular el controlador LQR, ajustando adecuadamente las matrices de ponderación Q y R para reducir el esfuerzo de control requerido.

Por lo tanto, este "clipeo" fue eliminado, a continuación se pueden resultados del sistema del motor con los ajustes realizados insertando correcta definición de la zona muerta del actuador y eliminando el "clipeo" de la acción de control.

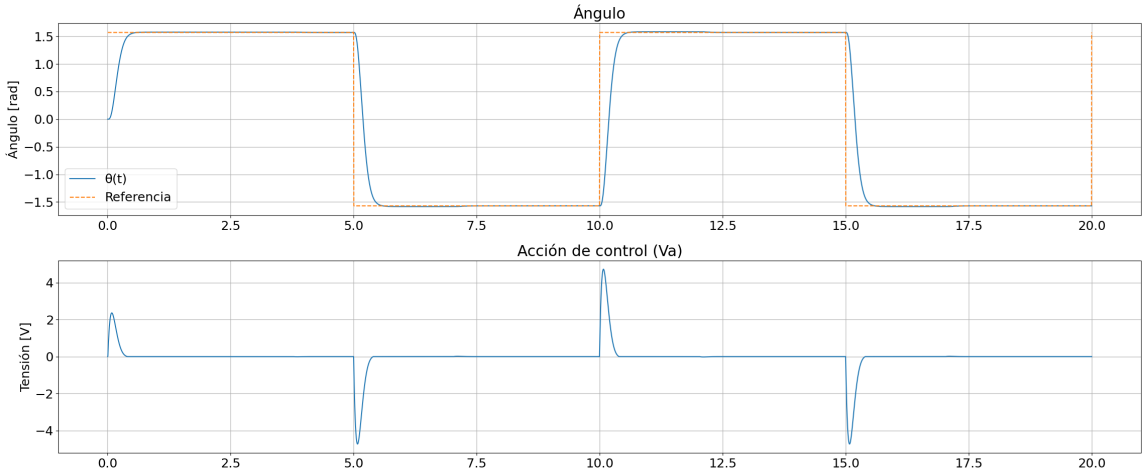

Figura 21: Respuesta temporal luego de las correcciones

7. Enlaces y recursos

- Repositorio GitHub del estudiante: `https://github.com/BrunoUNC/Sistemas-De-Control-II-`
  `git`