CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL INSTITUTO POLITÉCNICO NACIONAL

UNIDAD ZACATENCO

DEPARTAMENTO DE CONTROL AUTOMÁTICO

Control del péndulo invertido rotativo por aprendizaje reforzado

T E S I S

Que presenta

Eduardo Yudho Montes de Oca

Para obtener el grado de

Maestro en Ciencias

En la especialidad de

Control Automático

Director de la tesis

Dr. Wen Yu Liu

Ciudad de México Agosto, 2023

Agradecimientos

Agradezco a mis padres, Hipólito y María, por creer en mí y apoyarme para alcanzar mis metas y objetivos. Con su ejemplo me han enseñado lo que significa la disciplina, la determinación y la perseverancia. También, agradezco a mis hermanos, María, Victor y Gustavo, por su apoyo y cariño, que han sido indispensables para lograr cada objetivo. Los amo.

Agradezco a mi amada esposa, Adilaí, por su infinita paciencia, apoyo y amor incondicional. Sin su constante motivación nada de esto sería posible. Tu ejemplo me ha enseñado de paciencia, amor y bondad. Te amo.

Agradezco de manera espacial al Dr. Wen Yu por compartir su consejo y experiencia para la realización de este trabajo. Su ejemplo me ha enseñado lo que representa ser un buen investigador y un gran ser humano.

Agradezo a los amigos que me han acompañado en cada proyecto y que han creido en mi y mis capacidades para siempre buscar alcanzar un objetivo más grande. Los llevo en el corazón.

Agradezco al Centro de Investigación y Estudios Avanzados del Instituto Politécnico Nacional y al departamento de Control Automático por brindarme los medios necesarios para mi formación como investigador. Así mismo, agradezco a mis compañeros y profesores con los que me crucé en el camino por su invaluable apoyo.

Agradezco al Consejo Nacional de Humanidades, Ciencia y Tecnología que a través de la beca de manutención de maestría que me fue otorgada me permitió aprovechar y dedicarme de tiempo completo a mi formación académica y mi trabajo de tesis.

Eduardo Yudho Montes de Oca

Dedicatorias

Dedicado a mis padres:

Hipólito y María

A mi esposa:

Adilaí

A mis hermanos:

María, Victor y Gustavo

Y a mis amados:

Hank y Halley

Resumen

En la actualidad, el uso de estrategias de control basadas en aprendizaje reforzado han resultado especialmente atractivas, debido a que no requieren de conocimiento de la dinámica del sistema. Se describe a los algoritmo de aprendizaje reforzado como un marco de referencia libre de modelo para resolver problemas de control óptimo, descritos como procesos de decisión de Markov. Por lo que, los algoritmo de aprendizaje reforzado comparten características teóricas con el control óptimo y el control adaptable.

En este trabajo de tesis, se presenta el uso de técnicas de aprendizaje reforzado para controlar al péndulo invertido rotativo sobre su equilibrio inestable, un sistema mecánico subactuado de dos grados de libertad. Se desarrollaron controladores basados en aprendizaje reforzado, por solución tabular y mediante aproximación por redes neuronales. Se obtuvo un compensador para controladores PD, en serie y en paralelo, basado en el modelo no lineal. Se realizó la comparación del compensador no lineal contra el uso de algoritmos de parendizaje reforzado como compensación para los controladores PD, ante una perturbación desconocida. Así mismo, se implementó un algoritmo, libre de modelo, basado en aprendizaje reforzado para sintonizar en línea a un regulador cuadrático lineal, cuando existe variación paramétrica.

De los resultados obtenidos se concluyó que el uso de redes neuronales como aproximación de la tabla Q permite reducir el tiempo de entrenamiento y evitar el problema de sobreestimación. Sin embargo, al combinar los algoritmo de aprendizaje reforzado con los controladores PD se identificó que las estrategias basadas en aprendizaje reforzado son suficientemente robustaz, al lidiar con perturbaciones desconocidas. Así mismo, al realizar la sintonización en línea del regulador cuadrático lineal se obtuvieron soluciones cercanas a los valores esperados en cada caso de variación paramétrica.

Abstract

Currently, the use of control strategies based on reinforcement learning have been specially attractive, since they do not require knowledge of the dynamics of the system. Reinforcement learning algorithms are described as a model-free frame of reference for solving optimal control problems, described as Markov decision processes. Therefore, reinforcement learning algorithms share theoretical characteristics with optimal control and adaptive control.

In this thesis, the use of reinforcement learning techniques is presented to control the rotary inverted pendulum over its unstable equilibrium, an underactuated mechanical system of two degrees of freedom. Control algorithms based on reinforcement learning, by tabular solution and by neural network approximation were developed. A compensator was obtained for PD controllers, serial and parallel, based on the nonlinear model. Comparison of the nonlinear compensator was performed against the use of reinforcement learning algorithms as compensation for PD controllers, for an unknown disturbance. Likewise, a model-free algorithm based on reinforcement learning was implemented to tune in line to a linear quadratic regulator, when there is parametric variation.

From the results obtained, it was concluded that the use of neural networks as an approximation of table Q allows reducing training time and avoiding the problem of overestimation. However, by combining reinforcement learning algorithms with PD controllers, it was identified that strategies based on reinforcement learning are robust enough to deal with unknown disturbances. Likewise, when performing the on-line tuning of the linear quadratic regulator, solutions close to the expected values were obtained in each case of parametric variation.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

Introducción

La teoría del control por retroalimentación es el estudio de los medios para desarrollar algoritmos de control para sistemas de ingeniería humana que garanticen criterios de rendimiento y seguridad [1]. Los controladores por retroalimentación funcionan bajo el principio de observar la salida de un sistema, comparandola contra la trayctoria deseada y calculando la señal de control necesaria para modificar el desempeño del sistema a partir del error [2]. Sin embargo, comunmente el uso de controladores clásicos requiere del conocimiento parcial o total de la dinámica de un sistema.

Cuando se conoce el modelo de un sistema dinámico es posible linealizarlo al rededor de un punto de operación y diseñar controladores lineales basados en el modelo. Uno de los controladores lineales más famosos es el regulador lineal cuadrático (LQR). Este controlador se diseña fuera de línea al resolver la ecuación algebráica de Ricatti, para mínimizar una función de costo preescrita. Sin embargo, los cambios en los parámetros del sistema comprometen la optimalidad del controlador y es necesario resintonizarlo.

Además, es posible usar técnicas de compensación que cancelen las dinámicas no lineales de la planta y se establezca un comportamiento más simple. Un esquema muy común es el control de robots manipuladores mediante el control PD con compensación de pares gravitacionales, el cual reduce el error en estado estacionario causado por los términos no lineales [3].

Por otro lado, cuando no se conoce un modelo exacto del sistema se recurre a controladores libres de modelo, entre los cuales destacan el control Proporcional-Integral-Derivativo (PID), el

control por modos deslizantes (SMC) y el control por redes neuronales. Aunque proporcionan un buen desempeño ante diferentes tareas y son relativamente fáciles de sintonizar, no presentan un desempeño óptimo y deben ser resintonizados ante perturbaciones e incertidumbre.

Recientemente, se han implementado técnicas de aprendizaje reforzado (RL) en problemas de control óptimo y control robusto, mediante el uso de la teoría de progrmación dinámica (DP) [3]. Resultando especialmente atractivos debido a que no requieren conocimiento de la dinámica del sistema y mediante el correcto diseño de las señales de recompensa es posible adaptarse a cambios en el sistema y posibles errores o perturbaciones [2].

El aprendizaje reforzado describe una familia de sistemas del aprendizaje automático que operan bajo los principios usados por animales, grupos sociales y sistemas naturales [2]. De acuerdo con Sutton [4], el aprendizaje reforzado se centra mucho más en el aprendizaje dirigido a objetivos desde la interacción que cualquier otro enfoque del aprendizaje automático. En pocas palabras, el aprendizaje reforzado consiste en aprender a relacionar situaciones con acciones para optimizar una recompensa numérica.

Los algoritmos de RL comparten características con el control óptimo y el control adaptable debido a que es un marco de referencia libre de modelo para resolver problemas de control óptimo, establecidos como procesos de decisión de Markov (MDPs) discretos. Muchos de los métodos de RL permiten el diseño de controladores adaptables que aprenden en tiempo real las soluciones a problemas de optimización preescritos por diseño [2]. Por lo tanto, el RL es una herramienta muy valiosa para hallar controladores óptimos, o casi óptimos, en casos donde la dinámica de un sistema es desconocida o se ve afectada por una incertidumbre significativa.

No obstante, los sistemas mecánicos subactuados son aquellos que poseen más grados de libertad que actuadores a controlar. Esta restricción implica que algunas de las configuraciones del sistema no pueden ser comandadas directamente, lo cual hace mucho más complicado el diseño de controladores. Por lo que se han convertido en un interesante caso de estudio en el control automático, y algunos son considerados como un punto de referencia, siendo utilizados para probar diferentes algoritmos de control [5].

Este tipo de sistemas son comunes en diferentes áreas y aplicaciones, tales como robótica móvil, sistemas flexibles, sistemas aeroespaciales y robots submarinos, entre otros [6]. Muchos de estos sistemas poseen la condición de subactuación por la naturaleza de su dinámica, como los aviones,

helicópteros, grúas y cohetes espaciales. Por otro lado, esta condición también puede aparecer en los sistemas cuando en su diseño se busca reducir costos o por fines prácticos, como los satélites con dos turbinas o los robots con eslabones flexibles. Sin embargo, también puede surgir ante fallas en los actuadores o imponerse artificialmente para crear sistemas no lineales complejos de bajo orden para adquirir información útil al controlar sistemas subactuados de orden superior [5].

Un ejemplo de sistema mecánico con la condición de subactuación impuesta artificialmente es el péndulo invertido rotativo, también conocido como péndulo Furuta en honor a su inventor [7–9], un sistema subactuado de dos grados de libertad. El sistema cuenta con un brazo actuado que rota sobre el plano horizontal y un péndulo que rota libremente sobre el plano vertical, el cual está sujeto al extremo del brazo mediante un pivote. Por lo tanto, se debe controlar indirectamente a la posición angular del péndulo mediante la posición del brazo, lo cual provoca dinámicas no lineales más pronunciadas. El motivo de su diseño proviene del sistema clásico de un péndulo montado sobre un carro, ya que la base rotativa elimina la limitación de movimiento y reduce las dinámicas no modeladas debidas al mecanismo de transmisión de potencia [7].

1.1 Programación Dinámica y Aprendizaje Reforzado

La programación dinámica (DP) y el aprendizaje reforzado (RL) son métodos algorítmicos para resolver problemas en donde las acciones son aplicadas a un sistema durante un periodo de tiempo extenso, para alcanzar un objetivo determinado. Usualmente se trata de problemas secuenciales de toma de decisiones, ya que son en tiempo discreto y se toman acciones en cada paso de tiempo. Las acciones se toman a partir de los resultados de las acciones anteriores y mediante una recompensa se evalua cada paso de la toma de decisiones, buscando optimizar a largo plazo el total de las recompensas acumuladas [10].

De acuerdo con Busoinu [10], en DP y RL un agente interactua con un entorno por medio de una señal de estado, la cual describe al estado del entorno, una señal de acción, la cual permite al agente influir sobre el entorno y una señal de recompensa, la cual evalua el desemepño inmediato. La dinámica del entorno, la función de recompensa, el conjunto de posibles estados y el conjunto de posibles acciones constituyen un proceso de decisión de Markov (MDP).

1.1.1 Procesos de decisión de Marcov (MDP)

De acuerdo con Sutton [4], los procesos de decisión de Markov representan un marco directo del problema de aprendizaje a partir de la interacción para lograr un objetivo. Un MDP está dado por la interacción entre un agente y el entorno. El agente es aquel que aprende y toma las decisiones. El entorno comprende todo lo que está fuera del agente.

Un MDP determinista está definido por un espacio de estados del entorno, $\mathcal{X}$, y el espacio de acciones del agente, $\mathcal{U}$. De forma concreta, el agente y el entorno interactuan en una secuencia de pasos de tiempo discreto, $k = 0, 1, 2, 3, \ldots$. En cada paso k, el agente recibe algunas representaciones del estado del entorno, $x_k \in \mathcal{X}$, y con base a éstas elige una acción $u_k \in \mathcal{U}$. Como consecuencia de esta acción, el agente recibe una recompensa numérica, $r_{k+1} \in \mathcal{R} \subset \mathbb{R}$, y se actualiza al estado x_{k+1} [4]. Generando una trayectoria:

$$x_0, u_0, r_1, x_1, u_1, r_2, x_2, u_2, r_3, \ldots$$

En un MDP finito, los conjuntos de estados, acciones y recompensas tienen un número finito de elementos. Por lo que, para valores particulares de las variables aleatorias, $x_{k+1} \in \mathcal{X}$ y $r_{k+1} \in \mathcal{R}$, existe una probabilidad de que esos valores ocurran en un tiempo k, según el estado y la acción precedentes:

$$p\left(x_{k+1}, r_{k+1} | x_k, u_k\right) \triangleq \Pr\left\{x_{k+1}, r_{k+1} | x_k, u_k\right\} \tag{1.1}$$

para toda $x_{k+1}, x_k \in \mathcal{X}$, $r_{k+1} \in \mathcal{R}$ y $u_k \in \mathcal{U}$. La función $p : \mathcal{X} \times \mathcal{R} \times \mathcal{X} \times \mathcal{U} \mapsto [0, 1]$ es una función determinística ordinaria de 4 argumentos, que describe la dinámica del MDP. Al tratarse de una distribución de probabilidad para cada estado x_k, y acción u_k, se tiene:

$$\sum_{x' \in \mathcal{X}} \sum_{r \in \mathcal{R}} p\left(x', r | x, u\right) = 1, \ \forall x \in \mathcal{X}, u \in \mathcal{U} \tag{1.2}$$

En un MDP, la propiedad de Markov señala que el estado contiene suficiente información sobre el pasado de la interacción agente-entorno, tal que la evolución del proceso de Markov depende sólo del estado actual y no de sus valores anteriores [4]. Por lo tanto, mediante un ligero abuso de notación, es posible reescribir a p como una función de 3 argumentos $p : \mathcal{X} \times \mathcal{X} \times \mathcal{U} \mapsto [0, 1]$:

$$p\left(x_{k+1} | x_k, u_k\right) \triangleq \Pr\left\{x_{k+1} | x_k, u_k\right\} = \sum_{r \in \mathcal{R}} p\left(x_{k+1}, r | x_k, u_k\right) \tag{1.3}$$

a la cual se conoce como probabilidad de transición de estado. Mientras que la recompensa esperada es una función de 2 argumentos que depende del par estado-acción:

$$r\left(x_k, u_k\right) \triangleq \mathbb{E}\left[r_{k+1} | x_k, u_k\right] = \sum_{r \in \mathcal{R}} r \sum_{x' \in \mathcal{X}} p\left(x', r | x_k, u_k\right) \tag{1.4}$$

Retornos

En el marco del aprendizaje reforzado, el objetivo del agente está formalizado en términos de la recompensa. Por lo que, es fundamental establecer una recompensa que realmente indique lo que se pretende lograr, es decir, la recompensa deberá comunicar al agente qué es lo que se desea alcanzar y no cómo se espera lograrlo [4].

Para una secuencia de recompensas recibidas después del paso k, $r_{k+1}, r_{k+2}, r_{k+3}, \ldots$, se define al retorno, $G_h\left(x_k\right)$, como alguna función específica de la secuencia de recompensas. Representa la recompensa obtenida por el agente a largo plazo. En DP y RL, el objetivo es optimizar el desempeño a largo plazo, utilizando sólo como retroalimentación el rendimiento inmediato. Esto se conoce formalmente como el reto de las recompensas retrasadas [4].

Existen diferentes tipos de retornos, dependiendo de la forma en que se acumule la recompensa [11,12]. En el caso más simple se tiene a la función de retorno como la suma de las recompensas:

$$G_h\left(x_k\right) \triangleq r_{k+1} + r_{k+2} + r_{k+3} + \cdots \tag{1.5}$$

Para tareas en donde la interacción agente-entorno se rompe de manera natural, surgen los conceptos de episodios y tiempo final, T. Cada episodio termina en estados especiales, denominados estados terminales, lo cual limita a la suma de recompensas hasta el paso T, que normalmente varía de episodio a episodio. Este tipo de teareas se conococen como episódicas, en donde suele ser necesario distinguir entre los estados terminales, $\mathcal{X}^+$, y no terminales, $\mathcal{X}$.

No obstante, en muchos casos la interacción agente-entorno no se rompe naturalmente en episodios identificables y continua sin límite. Este tipo de problemas se conocen como tareas continuas, en donde $T = \infty$ y el retorno facilmente se puede volver infinito. Por lo que se utiliza una definición de retorno que es ligeramente más compleja conceptualmente, pero más simple matemáticamente.

Para esto, se introduce el factor de descuento, γ, que da lugar al retorno descontado de horizonte infinito:

$$G_h\left(x_k\right) \triangleq \sum_{i=0}^{\infty} \gamma^i r_{k+i+1} = \sum_{i=0}^{\infty} \gamma^i r\left(x_{k+i}, u_{k+i}\right) \tag{1.6}$$

donde $\gamma \in [0, 1)$. En [10] se menciona que el factor de descuento puede ser interpretado intuitivamente como una medida de que tan "claro de visión" es el agente con respecto a las recompensas. Desde un punto de vista matemático, el descuento garantiza que el retorno siempre estará acotado si la recompensa está acotada.

Sutton [4], destaca que el factor de descuento determina el valor actual de las recompensas futuras. Si $\gamma < 1$, la suma infinita convergerá, siempre y cuando, la secuencia de recompensas esté acotada. Cuando $\gamma = 0$, se produce un fenómeno de miopía en el agente, debido a que sólo buscará optimizar la recompensa inmediata. Conforme γ tiende a 1, el agente toma en cuenta mayormente a las recompensas futuras, haciendo que sea hipermétrope.

En la práctica, es necesario elegir un valor adecuado de γ. Sin embargo, no hay un procedimiento general para elegir a γ. Por ejemplo, en un problema de clásico de estabilización en control automático, donde se requiere alcanzar un estado estacionario y mantenerse ahí, γ deberá ser suficientemente grande tal que las recompensas obtenidas al alcanzar el estado estacionario y permanecer ahí tengan una influencia significativa en cada estado inicial [10].

De igual forma, es posible aplicar el retorno descontado a problemas episódicos de horizonte finito:

$$G_h\left(x_k\right) = \sum_{i=0}^{T} \gamma^i r_{k+i+1} = \sum_{i=0}^{T} \gamma^i \rho\left(x_{k+i}, h\left(x_{k+i}\right)\right) \tag{1.7}$$

Una propiedad que resulta importante para los algoritmos de aprendizaje reforzado es que las recompensas sucesivas están relacionadas de forma recursiva:

$$\begin{aligned} G_h\left(x_k\right) &= r_{k+1} + \gamma r_{k+2} + \gamma^2 r_{k+3} + \gamma^3 r_{k+4} + \cdots \\ &= r_{k+1} + \gamma\left(r_{k+2} + \gamma r_{k+3} + \gamma^2 r_{k+4} + \cdots\right) \\ &= r_{k+1} + \gamma G_h\left(x_{k+1}\right) \end{aligned} \tag{1.8}$$

Lo cual se cumple en tareas episódicas para todo $k < T$ [4].

Políticas y Funciones de Valor

La mayoría de los algoritmos de RL involucran la estimación de las funciones de valor, las cuales son funciones que dependen de los estados o los pares estado-acción y estiman que tan bueno es para el agente estar un cierto estado o realizar una cierta acción en un estado determinado, todo en función del retorno esperado. Debido a que las recompensas dependen de las acciones que el agente tome, las funciones de valor se definen con respecto a las políticas [4].

Una política es un mapeo de los estados a las probabilidades de seleccionar cada posible acción [10]. Si el agente sigue una política h en el tiempo k, entonces $h(x)$ representa la probabilidad de que se tome la acción u a partir del estado x.

La función de valor de un estado x_k bajo la política h, se denota por $v_h(x_k)$, es el retorno esperado cuando se inicia en x_k y se sigue la política h. Para un MDP se define formalmente a la función de valor como:

$$v_h(x_k) \triangleq \mathbb{E}_h\left[G_h(x_k)\,|x_k\right] = \mathbb{E}_h\left[\sum_{i=0}^{\infty}\gamma^i r_{k+i+1}|x_k\right],\ \forall x_k \in \mathcal{X} \tag{1.9}$$

donde $\mathbb{E}_h[\cdot]$ denota el valor esperado de una variable aleatoria que sigue una política h.

De manera similar, se define al valor de tomar una acción u_k en un estado x_k bajo una política h, por $q_h(x_k, u_k)$ y es conocida como función de acción-valor:

$$q_h(x_k, u_k) \triangleq \mathbb{E}_h\left[G_h(x_k)\,|x_k, u_k\right] = \mathbb{E}_h\left[\sum_{i=0}^{\infty}\gamma^i r_{k+i+1}|x_k, u_k\right],\ \forall x_k \in \mathcal{X},\ u_k \in \mathcal{U} \tag{1.10}$$

Por lo tanto, se tienen dos tipos de funciones de valor, de estado-acción, $q_h(x, u)$, y de estado, $v_h(x)$. De acuerdo con [10], comunmente en la literatura se utiliza el nombre función de valor para referirse a $v_h(x)$. Sin embargo, propone utilizar los nombre funciones Q y funciones V para hacer una clara distinción entre ambas.

La función Q de una política h, $q_h : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$, proporciona el retorno obtenido cuando se aplica una determinada acción ante un estado específico, siguiendo la política h:

$$q_h(x_k, u_k) = r(x_k, u_k) + \gamma G_h(x_{k+1}) \tag{1.11}$$

Esta formula concisa puede ser obtenida al escribir $q_h(x_k, u_k)$ explícitamente como la suma descontada de las recompensas obtenidas al tomar u_k en x_k y seguir a h:

$$q_h(x_k, u_k) = \sum_{i=0}^{\infty} \gamma^i r(x_{k+i}, u_{k+i}) \tag{1.12}$$

Una propiedad fundamental de las funciones de valor usadas en DP y RL es que satisfacen relaciones recursivas. Es decir, para cualquier política h y cualquier estado x_k, se cumple la siguiente condición de consistencia entre el valor de x_k y el valor de su posible estado sucesor [4]:

$$\begin{aligned} q_h(x_k, u_k) &= r(x_k, u_k) + \sum_{i=1}^{\infty} \gamma^i r(x_{k+i}, u_{k+i}) \\ &= r(x_k, u_k) + \gamma \sum_{i=1}^{\infty} \gamma^{i-1} r(x_{k+i}, u_{k+i}) \\ &= r(x_k, u_k) + \gamma G_h(x_{k+1}) \end{aligned} \tag{1.13}$$

donde se usó la definición de $G_h(x_{k+1})$ para el última paso. Además, al analizar el segundo paso de la ecuación anterior, se tiene:

$$\begin{aligned} q_h(x_k, u_k) &= r(x_k, u_k) + \gamma \sum_{i=1}^{\infty} \gamma^{i-1} r(x_{k+i}, u_{k+i}) \\ &= r(x_k, u_k) + \gamma \left[r(x_{k+1}, u_{k+1}) + \gamma \sum_{i=2}^{\infty} \gamma^{i-2} r(x_{k+i}, u_{k+i}) \right] \\ &= r(x_k, u_k) + \gamma q_h(x_{k+1}, u_{k+1}) \end{aligned} \tag{1.14}$$

Se conoce a la ecuación anterior como ecuación de Bellman para q_h. Esta ecuación describe una relación entre el valor del par estado-acción actual y el valor de los posibles pares sucesivos [4]. La ecuación de Bellman para q_h establece que el valor de tomar una acción u_k en el estado x_k bajo la política h es igual a la suma de la recompensa inmediata y el valor en el siguiente estado:

$$q_h(x_k, u_k) = r(x_k, u_k) + \gamma q_h(x_{k+1}, u_{k+1}) \tag{1.15}$$

Resolver un problema de aprendizaje reforzado implica encontrar una política que alcance una gran recompensa a largo plazo. Una forma conveniente de carcaterizar a las políticas es usando sus funciones de valor. Para MDPs finitos, las funciones de valor establecen un orden parcial sobre las políticas. Una política h es mejor o igual que otra política h', $h \geq h'$, si y sólo si $v_h(x_k) \geq v_{h'}(x_k)$ para todo $x_k \in \mathcal{X}$ [4].

Además, siempre existe al menos una política que es mejor o igual que las demás y se conoce como política optima. Aunque exista más de una política optima, se denotada a todas ellas como h^*. Todas las políticas óptimas comparten la misma función V, llamada función de valor de estado óptima, denotada como v^*, y definida como:

$$v^*(x) \triangleq \min_h v_h(x) \tag{1.16}$$

Además, las políticas optimas también comparten la misma función Q, denotada por q^*, y definida como:

$$q^*(x,u) \triangleq \min_h q_h(x,u) \tag{1.17}$$

Cualquier política h^* que elija en cada estado a la acción con el valor óptimo q más pequeño es óptima, es decir:

$$h^*(x) \in \arg\min_u q^*(x,u) \tag{1.18}$$

En general, para una función Q dada, una política h que satisface:

$$h(x) \in \arg\min_u q(x,u) \tag{1.19}$$

se dice que es codiciosa en Q. Por lo que, para hallar una política óptima es posible hallar primero la función óptima q^* y usar la expresión (1.18) para calcular una política codiciosa en q^*.

Como q_* es la función de valor para una política, deberá satisfacer la condición de consistencia dada por la ecuación de Bellman para valores de estado-acción. Sin embargo, como se trata de la función Q óptima, la condición de consistencia puede escribirse sin hacer referencia a alguna política en específico. A esta ecuación se le conoce como ecuación de Bellman para q^*, o ecuación de optimalidad de Bellman [4].

La ecuaciónde optimalidad de Bellman caracteriza a q^* y establece que el valor óptimo de la acción u tomada en el estado x es igual a la suma de la recompensa inmediata y el valor óptimo de la mejor acción en el siguiente estado:

$$q^*(x_k,u_k) = r(x_k,u_k) + \gamma \min_u q_h(x_{k+1},u) \tag{1.20}$$

Cuando se tiene a v^*, es relativamente sencillo determinar la política óptima. Para cada estado x, existirán una o más acciones para las que se obtenga el mínimo en la ecuación de optimalidad de Bellman. Por lo que se requiere una busqueda de un paso adelante. Cuando se conoce a q^* es más fácil determinar las acciones óptimas. En este caso, el agente no requiere realizar una busqueda de un paso adelante, simplemente puede encontrar la acción que minimice a $q^*(s, a)$. Es más sencillo utilizar una política codiciosa en q^* debido a que incluye información sobre la calidad de las transiciones; mientras que una función V sólo describe la calidad de los estados [10]. Además, la función Q óptima permite seleccionar acciones óptimas sin tener que saber nada sobre los posibles estados sucesores y sus valores, es decir, sin tener que saber nada acerca de la dinámica del entorno [4].

1.1.2 Programación dinámica

El término programación diámica se refiere a una colección de algoritmos que pueden usarse para calcular políticas optimas a partir de un modelo perfecto del entorno, como un MDP. Los algoritmos clásicos de DP son de limitada utilidad en el aprendizaje reforzado debido a su suposición de un modelo perfecto y su gran costo computacional, pero siguen siendo importantes teoricamente [4].

La idea principal de la programación dinámica, y del aprendizaje reforzado en general, es el uso de funciones de valor para organizar y estructurar la busqueda de políticas adecuadas.

Iteración de la función de valor

Las técnicas de iteración de la función de valor usan la ecuación de optimalidad de Bellman para calcular una función de valor óptima de forma iterativa, a partir de la cual se deriva una política óptima [10].

Sea el conjunto de todas las funciones de valor Q denotado por $\mathcal{Q}$. Entonces, el mapeo de iteración $T : \mathcal{Q} \mapsto \mathcal{Q}$, calcula el lado derecho de la ecuación de optimalidad de Bellman para cualquier función Q. En el caso determinista este mapeo está dado como:

$$[T(q)](x_k, u_k) = r(x_k, u_k) + \gamma \min_u q(x_{k+1}, u) \tag{1.21}$$

El algoritmo de iteración inicia en una función arbitaria, q_0, y en cada paso k actualiza la función q usando:

$$q_{k+1} = T\left(q_k\right) \tag{1.22}$$

Es posible mostrar que T es una contracción con factor $\gamma < 1$ en la norma infinito, es decir, para cualquier par de funciones q y q', es cierto que:

$$\left\|T\left(q\right) - T\left(q'\right)\right\|_\infty \leq \gamma \left\|q - q'\right\|_\infty \tag{1.23}$$

Debido a que T es una contracción y tiene un único punto de equilibrio [10]. Al reescribir la ecuación de optimalidad de Bellman usando el mapeo de iteración, se establece que q^* es un punto de equilibrio de T [10]:

$$q^* = T\left(q^*\right) \tag{1.24}$$

Entonces, el único punto de equilibrio de T es en realidad q^* y la iteración de q converge asintóticamente a q^* conforme $k \to \infty$. Además, la iteración de q converge a q^* con una razón de γ, en el sentido de $\left\|q_{k+1} - q_*\right\|_\infty \leq \gamma \left\|q_k - q_*\right\|_\infty$. Con lo cual, una política óptima puede ser calculada de q^* como:

$$h^*\left(s\right) \in \arg\min_u q^*\left(x, u\right) \tag{1.25}$$

En el Algoritmo 1 se muestra el proceso de iteración de la función q.

Algoritmo 1 Iteración de la función de valor para un MDP

Entrada: Modelo MDP, función de recompensa r, factor de descuento γ
1: Inicializar a la función q, por ejemplo, $q_0 \leftarrow 0$
2: **repetir** {en cada iteración $k = 0, 1, 2, \ldots$}
3: **para** cada par (x, u) **hacer**
4: $q_{k+1}(x_k, u_k) \leftarrow r(x_k, u_k) + \gamma \operatorname{máx}_u q_k\left(x_{k+1}, u\right)$
5: **fin para**
6: **hasta que** $q_{k+1} = q_k$
Salida: $q^* = q_k$

El resultado anterior sólo garantiza la convergencia asintótica de la iteración de q, entonces el criterio para detener el Algoritmo 1 sólo se satisface asintóticamente. En la práctica es importante garantizar el desempeño del algoritmo cuando se detiene después de un número finito de itera-

ciones. Es posible detener la iteración de q cuando la dierencia entre dos funciones q consecutivas sea igual o menor que una cota determinada, $\varepsilon_{QI} > 0$, es decir, cuando $\|q_{k+1} - q_k\|_\infty \leq \varepsilon_{QI}$.

1.1.3 Aprendiaje reforzado

El aprendizaje reforzado implica aprender qué hacer, es decir, cómo mapear situaciones a acciones, para optimizar una señal de recompensa numérica. En RL no se indica al .ªprendiz"(agente) que acciones tomar, sino que debe descubir que acciones producen la mayor recompensa por prueba y error. En los casos más interesantes y complejos, las acciones no sólo afectan la recompensa inmediata, sino también a las situaciones futuras y sus recompensas [4].

En el aprendizaje reforzado la idea principal es capturar los aspectos más importantes del problema real para alcanzar un objetivo. El agente deberá ser capaz de medir, hasta cierto punto, los estados del entorno y tomar acciones que influyan en ellos.

Sutton y Barto [4], establecen que a diferencia del aprendizaje supervisado, en RL no se aprende de un conjunto de datos provistos por un supervisor externo con conocimiento de la tarea a resolver. En el aprendizaje supervisado, el objetivo es que el sistema generalice su respuesta para que actúe correctamente en situaciones no presentes en el conjunto de entrenamiento. Sin embargo, este método no es adecuado para aprender de la interacción. En problemas interactivos es comunmente impráctico obtener ejemplos del comportamiento deseado que sean correctos y representativos para todas las posibles situaciones. El objetivo en RL es que en un terreno desconocido, el agente sea capaz de aprender apartir de su propia experiencia.

Así mismo, el aprendizaje reforzado busca optimizar una señal de recompensa en lugar de tratar de encontrar una estructura oculta en colecciones de datos, provenientes de ejemplos de comportamientos concretos, como se hace generalmente en el aprendizaje no supervisado. Por esto, se considera al aprendizaje reforzado como un tercer paradigma del aprendizaje automático, junto al aprendizaje supervisado y el aprendizaje no supervisado [4].

Uno de los retos más grandes que se enfrentan en RL, y no en otros tipos deaprendizaje, es el equilirbio entre exploración y explotación. Para obtener una mejor recompensa, un agente de RL debe elegir las acciones que ha probado en el pasado y han resultado efectivas. Sin embargo, para descubrir esas acciones, tuvo que probar acciones que no había elegido antes.

1.2 Estado del arte

Las técnicas de control desarrolladas para robots completamente actuados no aplican direcatmente en el caso de sistemas mecánicos subactuados. Por lo que se han desarrollado algoritmos estabilizantes para sistemas subactuados, debido al interés de estabilizar sistemas como barcos, vehículos submarinos, helicópteros, aviones, satelites y robots móviles, los cuales son subactuados por diseño. Además, ante fallas en los actuadores es necesario implementar soluciones mediante software ya que son más económicas que el uso de actuadores redundantes [6].
El desarrollo de estos métodos está centrado en obtener técnicas de control generales para sistemas subactuados no lineales. Sin embargo, este objetivo es dificil de alcanzar y en su lugar se busca estabilizar ciertas clases de sistemas mecánicos. Para esto los investigadores se han enfocado en sistemas mecánicos simples, entre los cuales destacan como puntos de referencia en laboratorios de control automático el péndulo invertido, el péndulo invertido rotativo, el pendubot, el manipuldor planar con resortes en los eslabones, el péndulo dirigido por una rueda giratoria, el sistema barra esfera, entre otros. Aunque son ejemplos de sistemas mecánicos simples, representan un reto para el diseño de esquemas de control [5].

1.2.1 Control del péndulo invertido rotativo

Debido a que el péndulo invertido rotativo es un sistema altamente no lineal, de fase no mínima, de estructura simple, multivariable e inestable es comumente utilizado para validar la eficacia y el desempeño de múltiples algoritmos de control [5].
De acuerdo con Hamza [13], en la literatura se han abordado 4 objetivos de control del péndulo Furuta, los cuales son:

1. El balanceo del péndulo, el cual consiste en llevar al péndulo del equilibrio estable hasta el equilibrio inestable.

2. La estabilización del péndulo, es decir, regular la posición del péndulo para mantenerlo en el equilibrio inestable.

3. El diseño de un control por conmutación que permita la acción del control de balanceo y la estabilización del péndulo.

4. Seguimiento de trayectoria, en donde se busca que el brazo siga una trayectoria variante en el tiempo, mientras que el péndulo permanece en el equilibrio inestable.

La motivación inicial del estudio del péndulo invertido proviene de la necesidad de controladores para equilibrar a los cohetes durante su despegue. Los cohetes son altamente inestables durante el lanzamiento, por lo cual se requiere de estrategías que mantegan al sistema en la posición vertical [14].

Control lineal

Algunas de las estrategias que han demostrado un buen desempeño al controlar al péndulo Furuta, son los esquemas de control en cascada, debido a que permiten atenuar el efecto de las perturbaciones y mejoran la dinámica en lazo cerrado [15]. Los esquemas de control PID son extensamente utilizados en aplicaciones industriales debido a la simplicidad de su estructura y a que existen métodos de sintonización que no requieren conocimiento del modelo del sistema. Sin embargo, en sistemas altamente no lineales, sujetos a incertidumbre, es difícil obtener un buen desempeño.

Chawaphan, Aung y Maneetham [16], analizaron la robustez y capacidad del controlador PID para balancear al péndulo hasta una región cercana a la posición vertical. Propusieron un controlador para la parte actuada y otro para el péndulo, tal que sumaron las respuestas para obtener la señal de control. Realizaron simulaciones del péndulo rotativo y el péndulo montado sobre un carro y sintonizaron los controladores mediante prueba y error. Observaron que, aunque el controlador es capaz de balancear al péndulo bajo condiciones nominales, carece de robustez ante perturbaciones externas.

Adharsh et. al. [17], desarrollaron un modelo no lineal del péndulo Furuta. Mediante análisis determinaron que no es posible estabilizarlo con un sólo controlador y utilizaron dos PID en paralelo, debido a que el comportamiento del brazo y el péndulo son completamente diferentes. Establecieron como criterios de diseño un sobretiro máximo de 10 %, un tiempo de subida menor a 1 s y tiempo de asentamiento menor a 5 s. Concluyeron que el esquema propuesto fue suficientemente robusto para estabilizar al péndulo y al brazo en 0°.

Nath y Mitra [18], diseñaron un control por retroalimentación de estados mediante asignación de polos y añadieron un integrador para eliminar el error en estado estacionario, debido a que en

la retroalimentación sólo intervienen las acciones proporcional y derivativa. Al implementar el controlador en el sistema físico el error en estado estacionario se mantuvo dentro de una banda del 0.5 %.

En [19], propusieron un controlador PD basado en energía para invertir al péndulo Furuta y para estabilizarlo implementaron un LQR. Mediante simulaciones observaron que el controlador PD logró aproximar al péndulo al equilibrio inestable con 5 balanceos. Mientras que en el equipo físico el control PD requirió más balanceos para invertir el péndulo debido a que no consideraron a la fricción en el diseño. Con respecto a la estabilización destacaron que se produjeron oscilaciones sostenidas en el estado estacionario.

Sainzaya et. al. [20], utilizaron un controlador basado en energía para balancear al péndulo rotativo y un LQR para estabilizarlo. Propusieron implementar un controlador PID dependiente de la velocidad del péndulo para mejorar la respuesta del LQR. Destacaron que esta modificación proporcionó una mayor robustez ante perturbaciones; sin embargo, el LQR sigue dependiendo de una región lineal para garantizar la estabilización del péndulo.

Furuta, Yamakita y Kobayashi [7, 8], consideraron el problema de balanceo y estabilización del péndulo invertido rotativo mediante algoritmos basados en el espacio de estados. Para el balanceo del péndulo compararon un controlador por prealimentación con una secuencia de control *bang-bang* contra una retroalimentación de estado tipo *bang-bang*. Para la estabilización usaron un LQR. Asumieron que los parámetros del sistema se mantuvieron constantes y añadieron una masa de 15 g en el extremo final del péndulo, como incertidumbre paramétrica. Sin la variación paramétrica ambos controladores tuvieron un buen desempeño y el balanceo fue exitoso en ambos casos. Al añadir la masa de 15 g el controlador *feedforward* no logró realizar el balanceo y la variante por retroalimentación tuvo un desempeño satisfactorio.

Xu, Iwase y Furuta [21], abordaron el problema de balanceo del péndulo en tiempo óptimo. Destacaron que muchas de las soluciones de optimización en tiempo son difíciles de calcular en una implementación física, como el Principio del Máximo de Pontryagin. Destacaron el uso de técnicas de optimización no lineal para resolver el problema planteado. Mediante simulaciones compararon al método propuesto contra un controlador basado en energía y concluyeron que el control óptimo produjo una respuesta más rápida. Además, implementaron el método diseñado en el sistema físico con un LQR, para estabilizar al péndulo. Al añadir una masa al extremo

del péndulo el controlador mantuvo un buen desempeño Concluyeron que el diseño no sólo demostró un buen seguimiento de trayectoria, si no que volvió al sistema menos sensible a la incertidumbre paramétrica.

Paredes, et. al. [22], propusieron combinar la optimalidad del LQR con la robustez de un controlador de estructura variable (VSC), tal que se obtenga una respuesta estable y menos suceptible a perturbaciones externas e incertidumbres. Diseñaron un LQR y el VSC para sumar ambas señales de control. Para el VSC optaron por un control por modos deslizantes (SMC) que utilizó al vector de error y la ganancia de retroalimentación óptima del LQR como superficie de deslizamiento. Reemplazaron a la función signo por la función sigmoide para reducir el castañeo. Compararon al esquema propuesto contra un LQR, en donde el VSC-LQR produjo la respuesta más rápida y con menor sobre tiro. Simularon un caso de variación de parámetros en donde comprobaron que el SMC mejoró la robustes del LQR. Al implementar ambos controladores en el sistema físico llevaron manualmente al péndulo a la posición vertical superior y los controladores lo estabilizaron. Al aplicar una perturbación tipo escalón comprobaron que el LQR no logró estabilizar el péndulo, mientras que el VSC-LQR produjo menores sobretiros y mantuvo al péndulo cercano al equilibrio. Al añadir una masa al extremo del péndulo el LQR produjo mayores sobretiros. Por lo que el esquema VSC-LQR demostró mayor robustez.

Pandey y Adhyaru [23], diseñaron un controlador robusto para el péndulo invertido rotativo, considerando el objetivo de control de mejorar la robustez ante incertidumbre paramétrica por el enfoque de control óptimo. Plantearon un problema de optimización con restricciones y lo transformaron en un problema de control robusto al modificar el término no cuadrático de la ecuación de Hamilton-Jacobi-Bellman, junto con la incertidumbre del sistema. Validaron el controlador mediante simulaciones con parámetros constantes y ante variaciones de $\pm 30\,\%$ al coeficiente de amortiguamiento del brazo actuado. El LQR robusto mantuvo al péndulo equilibrado con parámetros constantes y distintas condiciones iniciales. Ante la incertidumbre el controlador mantuvo al péndulo cerca del equilibrio inestable, aunque se incrementó el sobretiro máximo.

Los controladores H_∞ y H_2 son una alternativa robusta a los controladores óptimos. El diseño de este tipo de controladores requiere de diferentes funciones de costo que hacen al sistema de control menos sensible a perturbaciones, incertidumbre y errores de modelación [24]. En [25] im-

plementaron un controlador basado en energía para balancear al péndulo Furuta y compararon un LQR, con un control por retroalimentación de estados y una combinación de controladores H_∞ y H_2. Durante los experimentos la retroalimentación de estados produjo la respuesta más rápida, la combinación H_2-H_∞ generó el menor sobretiro y las menores oscilaciones al rededor del equilibrio. Sin embargo, la magnitud de la señal de control superó el límite del motor de corriente de directa del sistema, por lo que requirieron saturar la señal. El LQR demandó la menor señal de control, debido a que se puede penalizar mediante diseño la demanda de energía del controlador. Concluyeron que el LQR produjo el mejor desempeño al comparar los resultados y el consumo de energía.

Yu, Wang y Lu [26], diseñaron un controlador robusto H_∞ para invertir y estabilizar al péndulo Furuta. Realizaron el diseño mediante la función de transferencia del sistema e implementaron técnicas de control robusto de orden fijo para simplificar la estructura del controlador. Plantearon un problema de optimización para un controlador de primer orden y lo desarrollaron mediante el toolbox de Matlab HIFOO (H-Infinity Fixed-Order Optimization). Compararon el controlador obtenido contra un esquema H_∞ estándar, de tercer orden. Ambos controladores cumplieron con el objetivo de control; sin embargo, comprobaron que el esquema propuesto fue más robusto que el controlador H_∞ estándar después de añadir un accesorio rotativo al extremo del péndulo, para introducir incertidumbre paramétrica y una perturbación aleatoria.

Ling et. al. [27], diseñaron un controlador basado en energía para invertir el péndulo Furuta y consideraron un modelo lineal de parámetros variables para desarrollar un controlador que estabilice al péndulo en el equilibrio inestable, mediante técnicas robustas de control predictivo. Mediante análisis determinaron la región en el espacio de estados en la cual el control era efectivo para la estabilización y definieron la condición de conmutación. Validaron el esquema propuesto mediante simulaciones. Concluyeron que durante la estabilización el controlador mostró no ser sensible al comportamiento de balanceo.

Así mismo, se ha abordado el problema del péndulo doble con la estructura del péndulo Furuta como en [28], donde propusieron un controlador H_∞, diseñado a través del problema de sensibilidad combinada, para estabilizar al péndulo doble. Buscaron asegurar la estabilidad y el desempeño robustos del sistema ante incertidumbres y perturbaciones. Validaron mediante simulaciones y experimentos en el equipo físico al esquema desarrollado contra un LQR. En la

primera simulación con parámetros constantes el LQR demostró un mejor desempeño. Al añadir una variación de $\pm 10\,\%$ al momento de inercia del brazo y el péndulo el control robusto redujo el tiempo de asentamiento y no presentó sobretiros. Ante una perturbación de 0.0349 *rad* a ambos péndulos después de 10 segundos en posición vertical, ambos controladores equilibraron los péndulos corectamente. Aunque, el control robusto redujo el sobretiro y el tiempo de asentamiento posterior a la perturbación. Finalmente, en el equipo físico sin perturbaciones ambos métodos mantuvieron al sistema con oscilaciones de ± 0.01 *rad* al rededor del equilibrio. Al aplicar una perturbación al péndulo interno, el LQR no logró estabilizar al sistema y el control robusto generó logró estabilizar el sistema en poco tiempo.

Zhang y Dixon [29], diseñaron un controlador por retroalimentación de estados no mínimos, también conocido como control Proporcional-Integral-Más (PIP), junto con un integrador anti *windup* para estabilizar al péndulo Furuta. Diseñaron el algoritmo en el dominio s y realizaron un proceso de optimización iterativo con una función de costo, que consideró la suma de los valores propios normalizados en lazo cerrado. De esta forma hallaron los valores propios de una región seleccionada y determinaron el control globalmente óptimo. Destacaron que una cualidad del control PIP es la robustez. Sin embargo, no es posible omitir el esquema anti *windup* debido a que el controlador posee un integrador y el error de posición del péndulo es grande ante la condición inicial en la posición vertical inferior.

Control no lineal

El control por modos deslizantes (SMC) es una estrategia comunmente utilizada debido a que su proveé la robustez necesaria al controlar sistemas no lineales ante la presencia de perturbaciones e incertidumbre [30]. En [31], analizaron el desempeño del sistema en las tareas de balanceo y estabilización utilizando únicamente el control por modos deslizantes. Diseñaron dos controladores, el primero para balancear el péndulo y el segundo para mantenerlo en el equilibrio inestable. Compararon los controladores propuestos contra un esquema PD-LQR, donde el controlador PD se utilizó para balancear al péndulo y el LQR para estabilizarlo. Los resultados mostraron que el SMC produjo una respuesta más rápida en el balanceo y mayor robustez ante perturbaciones, durante la estabilización. Sin embargo, reemplazaron a la función signo por una función sigmoide para reducir el efecto de castañeo.

Wadi, Lee y Romdhane [32], diseñaron un controlador no lineal basado en modos deslizantes para el péndulo Furuta ante perturbaciones externas. El algoritmo fue diseñado a partir de las ecuaciones de movimiento del sistema. Realizaron simulaciones en las cuales introdujeron perturbaciones e incertidumbre asociada al modelo y los instrumentos de medición. Compararon al controlador contra una retroalimentación de estados. Concluyeron que el SMC produjo un menor sobretiro y tiempo de asentamiento, así como una reducción en la señal de control. Sin embargo, reemplazaron a la función signo por la función tangente hiperbólica para reducir el castañeo.

Mientras que en [33], diseñaron un controlador basado en energía para el balanceo del péndulo Furuta y compararon a un SMC contra un LQR en la estabilización. Por simulaciones observaron que la señal de control más suave se produjo con el LQR, en donde la respuesta del sistema fue más rápida y con un menor sobretiro. Mientras que el control por modos deslizantes proporcionó mayor robustez ante la incertidumbre paramétrica y perturbaciones; sin embargo, se utilizó una regla de aproximación de orden fraccional para atenuar el castañeo.

Srivastava et. al. [34], presentaron el diseño de un controlador por modos deslizantes de segundo orden para el péndulo invertido rotativo. Utilizaron el algoritmo super-twisting para diseñar el controlador. Compararon mediante simulaciones al control propuesto contra un SMC de primer orden y un LQR. Añadieron una perturbación sobre el canal de la señal de control, con el objetivo de comparar la robustez de los tres métodos. El LQR fue incapaz de compensar la perturbación y produjo oscilaciones sostenidas en la posición del péndulo. Los métodos basados en modos deslizantes rechazaron la perturbación y mantuvieron al péndulo en la posición vertical. Aunque, el algoritmo super-twisting produjo un menor sobretiro y tiempo de asentamiento; además, la señal de control asociada al método de segundo orden presentó un menor castañeo que el SMC de primer orden aún con la función signo.

Park y Chwa [35], presentaron un controlador por modos deslizantes acoplados para el péndulo invertido sobre un carro y un péndulo Furuta. El esquema consiste en tomar una superficie de deslizamiento para el subsistema actuado y otra para el subsistema no actuado, para forzar su acoplamiento y así generar una dinámica cero. Diseñaron controladores para dos casos, en el primero se abordó la estabilización asintótica del péndulo y en el segundo se propuso un control de balanceo agresivo. Simularon el modelo del péndulo sobre un carro y al aplicar el

control de balanceo el péndulo se invirtió sin completar un balanceo y se mantuvo la estabilidad asintótica al equilibrarlo. Posteriormente, implementaron el controlador diseñado en el péndulo Furuta físico y el control de balanceo invirtió al péndulo en un sólo movimiento, a pesar de la incertidumbre, errores de modelación, ruido de medición y fricciones no lineales, por mencionar algunos factores. Además, ante una perturbación tipo impulso se mantuvo la estabilidad asintótica del sistema.

Aguilar y Moreno [36], abordaron el problema de seguimiento de trayectoria del péndulo Furuta. Propusieron un controlador basado en la técnica de linealización por retroalimentación, con el objetivo de mantener al error de seguimiento uniformemente acotado. Reescribieron al sistema en función de la dinámica del error, para hallar a la señal de control adecuada. Mediante análisis matemático determinaron que el controlador diseñado satisface las cualidades requeridas en el sistema de control. Implementaron el controlador en el sistema físico y concluyeron que el controlador cumplió con las características de estabilidad y robustez esperadas.

Aguilar, Gutierrez y Sossa [37], propusieron un controlador basado en energía, por el método de Lyapunov, para la estabilización del péndulo. Implementaron un controlador basado en linealización parcial por retroalimentación, con el cual se linealizó al subsistema actuado y posteriormente hallaron una función candidata de Lyapunov, con la cual diseñaron un controlador estabilizante. Los resultados mostraron que el sistema en lazo cerrado es local y asintóticamente estable al rededor del equilibrio inestable con un dominio de atracción computable por el Teorema de invarianza de LaSalle. Concluyeron que el dominio de estabilidad depende de los parámetros físicos, por lo que no es recomendable incrementar el dominio cuando el largo del brazo es pequeño en comparación con el péndulo.

Furuta, et. al. [9], propusieron un controlador adaptable de estructura variable basado en el modelo no lineal del péndulo rotativo para el control de posición. Combinaron el concepto de sistema de estructura variable (VSS), identificación adaptable y control no lineal. Compararon al controlador propuesto contra un LQR ante diferentes condiciones iniciales. El esquema adaptable garantizó la estabilidad en una región más amplia que el control óptimo, al no estar basado en un modelo linealizado. Introdujeron una modificación en la regla de adaptación para mejorar la robustez del controlador ante perturbaciones, ya que al implementar el controlador en el sistema físico detectaron que el sistema se vuelve inestable cuando se aplica una perturbación en los

parámetros y no se estiman.

Yamakita et. al. [38], consideraron el problema de balancear al péndulo Furuta doble. Para esto diseñaron un controlador basado en la energía y un algoritmo basado en un ciclo límite, el cual demostró ser más robusto ante incertidumbres. Mencionaron, que al ser un péndulo doble posee 5 estados de equilibrio. Por lo que, mediante los experimentos realizados concluyeron que al llevar al péndulo hasta la posición vertical superior mediante los equilibrios intermedios se obtiene una respuesta más robusta que al realizar el cambio de forma directa.

Mientras que en [39], propusieron un controlador a partir del problema del regulador por retroalimentación de estado (SFRP), para el cual hallaron una solución aproximada mediante el método de modos deslizantes. Validaron el controlador mediante experimentos en donde variaron los valores nominales de la longitud del brazo y el péndulo, así como de la masa del péndulo. Compararon la respuesta del esquema propuesto contra un regulador no lineal y no robusto. Destacaron que una de las ventajas de utilizar los modos deslizantes es que se mantiene acotado al error de seguimiento; además, el controlador proporcionó un buen desempeño al balancear al péndulo, a pesar de la incertidumbre, errores de modelación y perturbaciones externas.

Ordaz y Poznyak [40], abordaron el problema de estabilización robusta y adaptable del péndulo Furuta al rededor del equilibrio inestable. Consideraron como desconocido al modelo dinámico del sistema, así como a las perturbaciones externas. El análisis de estabilización se realizó a partir del método de elipsoides atractivas (AEM), con una modificación para utilizar información obtenida en línea. Implementaron el controlador en simulaciones computacionales y observaron que si se satisface la condición de excitación persistente regularizada, el método propuesto mantendría a las trayectorias del sistema dentro de la elipsoide de tamaño mínimo. Al comparar el desempeño contra el AEM tradicional determinaron que la mejoría en el desempeño fue sustancial.

Pujol, Mobayen y Acho [41], destacaron que la implementación en tiempo real de sistemas de control implica tomar en cuenta problemas como la incertidumbre, variación de parámetros, perturbaciones externas y la presencia de retardos en la retroalimentación. Por lo que abordaron un retardo aleatorio en la medición de la posición del péndulo Furuta y diseñaron un controlador robusto dependiente del retardo, basado en la teoría H_∞ y las desigualdades lineales matriciales (LMI). Implementaron una perturbación sobre la base en la que se encuentra el sistema y bus-

caron compensarla. Lo resultados mostraron que el controlador fue capaz de atenuar dinámicas aleatorias y perturbaciones externas, diferentes a las evaluadas en otros trabajos.

Control inteligente

Los controladores basados en lógica difusa son comunmente utilizados en sistemas no lineales, debido a que es fácil incorporar el conocimiento experto sobre el sistema en el control, tienen una menor dependencia del modelo del sistema, son robustos y es fácil modelar las reglas que requieren. Sin embargo, es díficil garantizar la estabilidad general de un controlador difuso [13]. Aún así, diferentes tipos de controladores han sido desarrollados para el péndulo Furuta.

Oltean [42], abordó los problemas de balanceo y estabilización del péndulo rotativo. Para el balanceo propuso un esquema PD en cascada, en donde el lazo interno realiza el control de posición del brazo; mientras que el lazo externo especifica la trayectoria deseada en el brazo para balancear el péndulo. Para la estabilización utilizaron un regulador PD difuso. Tomaron como entradas para el sistema difuso al error de posición del péndulo y la derivada del error. La salida fue la señal de control. Utilizaron como sistema de inferencia al método Min-Max. Simularon el controlador y requirió 4.5 s para invertir al péndulo, donde el esquema PD difuso fue capaz de estabilizarlo. Aplicaron una perturbación tipo impulso, de amplitud 10° y periodo de 4 s. El controlador propuesto fue suficientemente robusto para mantener al péndulo equilibrado en presencia de incertidumbre y perturbaciones.

En [43], propusieron un esquema de control no lineal, basado en la metodología Backstepping, combinado con un sistema difuso fuera de línea para equilibrar al péndulo invertido rotativo. Implementaron un sistema neuro-difuso para estimar la salida del controlador Backstepping, utilizando los estados como entradas. La señal de control constó de la suma del par estimado más la respuesta del controlador no lineal. Mencionaron que el bloque Backstepping se utilizó para recolectar pares de datos de entrada y de salida, tal que el sistema neuro-difuso se utilizó como un Sistema de Inferencia Neuro Difuso Adaptable con tres funciones de membresía de tipo campana para cada entrada y 81 reglas diseñadas. Al simular el sistema observaron que logró invertir al péndulo en 2.5 s con una señal de control muy grande. Al acotarla en ± 6 V notaron que el controlador sólo era efectivo en la región de $\pm 35.2°$ del equilibrio inestable. Sin embargo, la combinación propuesta mejoró el desempeño del controlador.

Ghorbani, Aliyari y Teshnehlab [44], desarrollaron un controlador tolerante a fallas para el péndulo rotativo con sincronización de caos, usando una señal caótica como referencia. El esquema constó de un LQR y un PID difuso. El LQR estabilizó al sistema cerca del punto de operación y el PID difuso realizó el seguimiento de la señal caótica. Las ganancias del controlador PID se sintonizaron con el método del centroide para la defuzzificación. Mediante simulaciones y la implementación en el sistema físico comprobaron que el esquema propuesto mejoró el desempeño del sistema.

Li [45], propuso implementar un controlador difuso compuesto (CFC), del tipo Mamdani, para lidiar con la tarea de estabilización del péndulo Furuta. El método desarrollado consta de 2 controladores PD difusos con dos entradas y una salida, en donde uno de los controladores está enfocado al brazo y el otro al péndulo. Un tercer controlador difuso realiza la composición de las respuestas de los controladores PD para generar la señal de control. En los experimentos llevaron al péndulo a una posición cercana al equilibrio de forma manual para que el CFC interviniera directamente. Concluyeron que el método propuesto es en esencia una retroalimentación de estados, por lo que el controlador garantiza la estabilidad del sistema.

Nguyen et. al. [46], consideraron el problema de estabilización y desarrollaron un controlador super-twisting continuo basado en lógica difusa. Propusieron una superficie de deslizamiento que consideró a la posición y velocidad del brazo y el péndulo. Las ganancias del algoritmo super-twisting se calcularon mediante lógica difusa. Utilizaron la metodología Min-Max para el sistema de inferencia y el método del centroide para la defuzzificación. Compararon al método propuesto contra un SMC y un SMC desacoplado. El esquema diseñado presentó una respuesta más rápida y el menor castañeo en señal de control. Además, evaluaron su robustez ante incertidumbre y perturbaciones, donde el algoritmo respondió rápidamente para mantener al péndulo equilibrado.

Diferentes algoritmos evolucionarios han sido propuestos para optimizar el proceso de sintonización de controladores tales como el PID, entre los cuales destacan los algoritmos genéticos (GA) y la optimización por enjambre de partículas (PSO), entre otros [47].

Hamza, Yap y Choudhury [13], desarrollaron un controlador PD difuso de tipo 2 en cascada para el péndulo Furuta. Utilizaron algoritmos genéticos (GA) y optimización por enjambre de partículas (PSO) para optimizar los parámetros del controlador. Como características de

diseño utilizaron al error en estado estacionario, tiempo de asentamiento, tiempo de subida, sobretiro máximo y señal de control. Simularon el sistema y realizaron pruebas en el sistema físico ante perturbaciones en la señal de control, variación de parámetros y ruido de medición. Los resultados mostraron que al usar GA se redujo el error en estado estacionario; mientras que con PSO se redujo el sobretiro, tiempo de asentamiento y señal de control. Concluyeron que es un esquema prometedor para otros sistemas no lineales e inestables.

Alarcón y Muñoz [48], diseñaron un esquema de control capaz de invertir al péndulo Furuta con una cantidad mínima de balanceos y de minimizar la tensión suministrada en la etapa de estabilización. El control del balanceo fue un conjunto de reglas que consideran a la posición y el signo de la velocidad del péndulo, así como al esfuerzo de control máximo y asignan una señal de control. Propusieron 5 regiones dentro del movimiento del péndulo, 4 corresponden al balanceo y 1 a la estabilización. Contemplaron 8 posibles valores de tensión. Utilizaron un algoritmo genético para definir los valores de la señal de control y plantearon un problema de optimización. Obtuvieron un conjunto de reglas que invertieron al péndulo con un balanceo. Al reducir el esfuerzo de control máximo se incrementó el número de balanceos para invertir el péndulo. Sin embargo, concluyeron que el algoritmo permite simplificar la tarea de estabilización.

Hassan y Mobayen [47], diseñaron controladores para equilibrar el péndulo Furuta a partir de algoritmos genéticos, optimización por enjambre de partículas y optimización por colonia de hormigas (ACO). Diseñaron una función de costo a minimizar, considerando al sobretiro, el tiempo de subida, el tiempo de asentamiento, el error en estado estacionario y la energía de control. Propusieron esquemas PID que serían sintonizados a través de los algoritmos evolucionarios. Realizaron simulaciones y experimentos en el sistema físico. Evaluaron la robustez y eficacia de los controladores ante variaciones de parámetros, ruido de medición y perturbaciones en la carga. El controlador ACO-PID produjo una respuesta más fina y robusta ante perturbaciones, indicando que el algoritmo ACO tiene una convergencia más rápida.

De igual forma, las redes neuronales resultan atractivas en el control de diferentes sistemas dinámicos, debido a sus propiedades de aproximación y robustez que han demostrado en simulaciones y aplicaciones en tiempo real [49]. En [49], diseñaron un controlador adaptable basado en redes neuronales para el péndulo Furuta. Lo diseñaron para controlar al péndulo en estado estacionario y ante la presencia de perturbaciones externas. Implementaron 2 redes neuronales,

usando el error de seguimiento filtrado en el lazo de control. La primera red se usó para aproximar el control equivalente en línea. La segunda red se usó para minimizar las oscilaciones y el tiempo de respuesta al encontrar la señal de control óptima. Mediante el método directo de Lyapunov hallaron las reglas de aprendizaje de ambas redes. Compararon mediante simulaciones al control propuesto contra dos esquemas basados redes neuronales. El esquema propuesto produjo un desempeño robusto y libre de fluctuaciones ante incertidumbre paramétrica y perturbaciones.

Moreno, Aguilar, Puga y Santibáñez [50], mostraron un esquema de control adaptable basado en una red neuronal. El objetivo fue garantizar que el brazo actuado puede seguir una señal de referencia mientras equilibra al péndulo en la posición vertical. Propusieron una estructura similar a la de una linealización por retroalimentación, usando a la red neuronal como compensador del modelo desconocido del sistema. La señal de control está compuesta por la salida de la red neuronal, una compensación por SMC y un controlador PD. Compararon al control diseñado contra un PID lineal, controladores PD y PID compensados por una red neuronal. Plantearon 3 experimentos, con el primero evaluaron la capacidad de cada método para el seguimiento de una trayectoria sinusoidal del brazo en presencia de una perturbación en el par aplicado. En el segundo utilizaron una trayectoria compleja y en el tercero retomaron la trayectoria sinusoidal con ruido Gaussiano en el par y las señales medidas. El esquema propuesto produjo una respuesta satisfactoria, capaz de rechazar perturbaciones externas y ruido de medición. Demostró desempeño en aplicaciones en tiempo real, además de contar con reglas de actualización calculadas mediante análisis riguroso.

En [51], presentaron un controlador por modelo de referencia, basado en redes neuronales paraconsistente recurrente (RPNN). El controlador constó de 3 RPNN, donde 2 de ellas modelaron al brazo y el péndulo; mientras que la tercera red se enfocó en controlar al sistema durante el seguimiento de una trayectoria. Destacaron que el tipo de funciones de activación que utilizan las redes neuronales paraconsistentes permiten compensar incertidumbres e información contradictoria, lo cual las convierte en una opción más robusta para su implementación en tareas de control. Compararon al controlador diseñado contra una asignación de polos, tal que los resultados mostraron mayor robustez, estabilidad y menor esfuerzo de control al utilizar las RPNN que el método clásico.

En [52], implementaron un regulador cuadrático lineal en combinación con una red neuronal

de base radial como compensador en línea. El LQR generó la señal de control necesaria para balancear y estabilizar al péndulo, mientras que la red neuronal abordaba la desviación en el sistema cuando se alejaba del punto de equilibrio sobre el cual se linealizó al modelo para diseñar el LQR. Propusieron una ley de aprendizaje por segmentos de amortiguamiento para activar la operación de la red. Por lo tanto, con hiperparámetros adecuados fue posible mejorar el desempeño del LQR cuando el sistema se alejó del punto de equilibrio. Algunas de las ventajas que destacaron fue la rápida reacción del controlador ante perturbaciones externas e incertidumbre paramétrica.

Mientras que en [53], propusieron un control basado en planeación de trayectoria y efecto de inercia para balancear el péndulo y estabilizarlo mediante un controlador adaptable no lineal basado en redes neuronales. El algoritmo para el balanceo del péndulo estuvo conformado por el seguimiento de una trayectoria tipo sinusoidal cuya amplitud incrementaba mientras que su periodo disminuía. Mediante una condición de balanceo, se produce la conmutación con el control por inercia que lleva al péndulo a la posición vertical y lo detiene inmediatamente. Para la estabilización usaron una retroalimentación de estados y una red neuronal de base radial que actúa cuando el sistema se aleja de la región lineal. Implementaron el diseño propuesto en el sistema físico y lo compararon contra un esquema bang-bang con un LQR. El balanceo por inercia produjo una respuesta más rápida que el método bang-bang, el cual produjo conmutaciones que pueden dañar al motor. La red neuronal añadió robustez al sistema en comparación con el LQR, debido a que el control no estaba limitado a una región lineal.

Algunas de las principales aplicaciones del RL se encuentran en los vehículos autónomos, ingeniería del control y robótica, en el control de sistemas mecánicos con dinámicas no lineales [54]. Allotta, Pugi y Bartolini [55], implementaron un controlador basado en energía para balancear al péndulo y un controlador basado en redes neuronales y aprendizaje reforzado para estabilizarlo. Realizaron el entrenamiento del controlador por aprendizaje reforzado mediante simulaciones con el modelo matemático en un máximo de 10,000 pruebas. En el sistema físico comprobaron que el controlador es estabilizó al péndulo en una región de $\pm 45°$ del equilibrio inestable. Además, diseñaron un LQR capaz de estabilizar al péndulo en la misma región. El control inteligente presentó un mejor desempeño al estabilizar el péndulo, aunque el LQR provocó oscilaciones de menor amplitud en el brazo actuado. Asociaron estos resultados al modelo dinámico

utilizado durante el entrenamiento.

Guida, Manrique, Camilo y Pappalardo [54], desarrollaron un controlador no lineal mediante el enfoque de aprendizaje reforzado, para el problema de balancear el péndulo Furuta en presencia de fricción seca como función de las fuerzas de reacción instantánea. La función de recompensa en el proceso de optmización presentó señales continuas y dispersas. Además, el aprendizaje se realizó mediante el algoritmo de gradiente profundo de política determinista (DDPG). Al realizar simulaciones computacionales con el esquema propuesto, observaron que el algoritmo demostró ser suficientemente general como para trabajar con un sistema con un modelo de fricción preciso.

Mientras que en [56], propusieron un sistema de control *Hardware-in-the-loop* (HIL) en tiempo real que permitió entrenar y probar el algoritmo de aprendizaje reforzado profundo mediante simulaciones y posteriormente implementarlo en un sistema físico. Utilizaron un esquema de doble red-Q profunda (DDQN) con un algoritmo de aprendizaje reforzado que prioriza la reproducción de experiencias, para implementar al agente. Definieron 21 acciones para balancear y equilibrar al péndulo de forma suave durante los experimentos en el péndulo físico. Realizaron una comparación del DDQN contra una red-Q profunda (DQN) y observaron que el algoritmo que prioriza la reproducción de experiencia eliminó la sobreestimación de Q y redujo el tiempo de entrenamiento. Validaron al método propuesto contra esquemas de control clásico y diferentes algoritmos de aprendizaje refozado. Los resultados mostraron que al usar el DDQN con el algoritmo propuesto, el péndulo puede ser más rápido que con los métodos clásicos.

1.2.2 Control de sistemas mecánicos subactuados

Así mismo, se tienen ejemplos notables del control de otros sistemas mecánicos subactuados como en [57], donde diseñaron un controlador PID en cascada mediante asignación de polos, para el problema de un péndulo invertido montado sobre un carro. El primer lazo corresponde al control del desplazamiento del carro y el segundo al control de posición del péndulo. Destacaron que los polos en lazo cerrado se asignaron mediante el diseño de un LQR. Para analizar el diseño propuesto implementaron en el sistema físico al PID propuesto y lo compararon contra el LQR diseñado en la asignación de polos. Invirtieron el péndulo de forma manual y observaron que el LQR tuvo un desempeño deficiente, en comparación con el esquema PID. Concluyeron que el

controlador cumplió con los requerimientos de robustez esperados.

Åström y Furuta [58], analizaron el problema de balanceo de un péndulo mediante estrategias de control basadas en la energía. Abordaron el problema del péndulo simple y dos péndulos montados sobre un carro. Destacaron que la aceleración máxima del pivote o anclaje es el parámetro más importante en esta tarea, debido a que se requiere que ésta sea al menos dos veces más grande que la aceleración de la gravedad para llevar al péndulo hasta la posición vertical en un sólo balanceo y con dos conmutaciones en la señal de control. Por otro lado, cuando la aceleración máxima está en el intervalo abierto de $(\frac{4}{3}g, 2g)$, en donde g representa a la aceleración de la gravedad, el objetivo se cumple en un sólo balanceo pero ahora con 3 conmutaciones de la señal de control. Concluyeron que cuando la aceleración máxima del pivote es menor que $\frac{4}{3}g$ el péndulo se deberá balancear varias veces.

Pueril, Yu y Sossa [59], propusieron un algoritmo de control para un péndulo invertido basado en aprendizaje reforzado . El controlador utilizó el método de Q-Learning en un esquema PD, tal que el sistema fuera capaz de mejorar su desempeño en línea y adaptarse a cambios en los parámetros. Usaron el método Q-Learning para balancear al péndulo. Además, combinaron técnicas híbridas de Q-Learning y controladores PD. Mediante simulaciones comprobaron que el control por RL no es capaz de invertir al péndulo en presencia de perturbaciones; mientras que el método combinado con el controlador PD mostró una mayor robustez bajo estas condiciones. Comprobaron que el controlador PD utilizado, tampoco fue capaz de invertir al péndulo sin el algoritmo de RL.

En [60], se abordó el análisis de controladores PD con modelos no lineales. Plantearon esquemas de controladores PD en serie y paralelo, con el objetivo de controlar el sistema barra y esfera. Incluyeron términos de compensación no lineal en ambas arquitecturas. En el esquema serial se añadió la compensación a la salida del lazo interno. En el esquema paralelo sumaron la respuesta de los controladores PD con los términos de compensación no lineal. Realizaron análisis de estabilidad de los controladores con el modelo no lineal completo, a través del método directo de Lyapunov. Concluyeron que la esfera se mantiene en cualquier posición de la barra ante un conjunto bien definido de condiciones iniciales. Los resultados experimentales comprobaron la estabilidad y desempeño de los controladores en problemas prácticos.

Galvan, Moreno, Yu y Ortíz [61], retomaron la compensación no lineal de controladores PD,

considerando al sistema plato y esfera. El objetivo de control es regular a los servomotores que mueven al plato para mantener a la bola en una posición. El esquema propuesto considera dos controladores PD, uno para el eje x y otro para el eje y. En ambos casos se agregan términos de compensación no lineal para generar la señal de control. Probaron el controlador propuesto mediante simulaciones para probar su efectividad. Posteriormente lo implementaron en el sistema físico y lo compararon contra el esquema sin compensación. Concluyeron que el controlador propuesto tuvo un mejor desempeño debido a la compensación no lineal.

Li y Yu [62], propusieron una compensación no lineal de controladores PD para el sistema barra y esfera mediante redes neuronales. Consideraron un controlador PD con compensación no lineal exacta y aproximaron al compensador mediante una red neuronal de base radial. Implementaron el esquema propuesto en dos sistemas barra y esfera físicos. Sincronizaron ambos sistemas mediante la compensación neuronal. Mostraron que los controladores PD no lograron sincronizar a los sistemas y llevar a las esferas a la posición deseada; mientras que al agregar la compensación neuronal se produjo una mejoría en la respuesta con un error de sincronización acotado.

Hendzel, Burghardt y Szuster [63], presentaron un controlador neuronal discreto para el sistema barra y esfera, en donde implementaron el método de aprendizaje reforzado para la actualización en línea de una red neuronal. El controlador contó con un esquema PD y un término supervisor, derivado del teorema de estabilidad de Lyapunov, para garantizar la estabilidad del sistema en lazo cerrado. Comprobaron la eficacia del controlador mediante simulaciones computacionales. Los resultados mostraron que el RL mejoró y aceleró el proceso de aprendizaje, previniendo el gasto de tiempo bajo prueba y error y eliminando la necesidad de entrenamiento previo del sistema. Plantearon un experimento con el peor escenario, donde los pesos sinápticos comenzaran todos con valor de 0 y aún así el controlador mantuvo al sistema estable y a los pesos acotados.

1.3 Objetivos

Diseñar estrategias de control para equilibrar al péndulo invertido rotativo sobre el equilibrio inestable, a partir de algoritmos de aprendizaje reforzado, para evaluar su capacidad de adaptarse a perturbaciones desconocidas o variaciones paramétricas, sin conocimiento de la dinámica del sistema.

El objetivo principal se divide en objetivos particulares para facilitar su cumplimiento. Los objetivos particulares a realizar son:

1. Investigar el estado del arte en el control lineal, no lineal e inteligente del péndulo invertido rotativo y otros sistemas mecánicos subactuados, para identificar problemas específicos en donde se puede implementar el aprendizaje reforzado.

2. Desarrollar el modelo no lineal del péndulo invertido rotativo y diseñar un compensador no lineal para esquemas de controladores PD en serie y paralelo.

3. Diseñar controladores basados en aprendizaje reforzado para equilibrar al péndulo invertido sobre el equilibrio inestable, mediante el algoritmo Q-Learning por solución tabular y aproximación con redes neuronales.

4. Diseñar compensadores basados en aprendizaje reforzado para los esquemas PD en serie y en paralelo, mediante el algoritmo Q-Learning por solución tabular y aproximación con redes neuronales.

5. Comparar el desempeño de los esquemas de control PD en serie y en paralelo sin compensación, con compensación no lineal y con los compensadores inteligentes ante una perturbación desconocida.

6. Implementar un algoritmo basado en aprendizaje reforzado para hallar en línea la solución de un LQR en tiempo discreto y comparar la respuesta del péndulo invertido rotativo con la solución aproximada contra la solución exacta ante variación paramétrica.

1.4 Estructura de la tesis

Esta tesis se divide en 5 capítulos. En este capítulo se presentó un panorama general de programación dinámica, aprendizaje reforzado y su planteamiento formal, desde el punto de vista de aprendizaje automático. Además, se presentó el estado del arte correspondiente al control del péndulo invertido rotativo y otros sistemas mecánicos subactuados, por estrategias de control lineal, no lineal e inteligente. En donde se incluye el uso de redes neuronales, control difuso, algoritmos genéticos y aprendizaje reforzado.
Los capitulos posteriores se dividen como:

- **Capítulo 2. Modelado y control del péndulo invertido rotativo.** En este capítulo se presenta el desarrollo del modelo no lineal del péndulo invertido rotativo, incluyendo el modelo electromecánico del actuador y finalmente se obtiene un modelo linealizado al rededor del equilibrio inestable. Posteriormente se presentan los esquemas de controladores PD en serie y paralelo. Se realiza el diseño de un compensador basado en el modelo no lineal del sistema, mediante análisis de estabilidad.

- **Capítulo 3. Control PD compensado por aprendizaje reforzado.** En este capítulo se introduce al aprendizaje reforzado desde el punto de vista del control automático. Se abordan los métodos de aprendizaje por diferencia temporal y el algoritmo Q-Learning. Además, se incluye la aproximación del aprendizaje reforzado mediante arquitecturas lineales y no lineales. Se presenta el algoritmo Q-Learning basado en la aproximación por redes neuronales. Posteriormente se realiza el diseño de controladores inteligentes basados en la solución tabular y por aproximación, para el control del péndulo Furuta. Así mismo, se diseñan compensadores inteligentes para los esquemas PD en serie y paralelo. Finalmente, se compara su desempeño contra el compensador basado en el modelo no lineal ante una perturbación desconocida.

- **Capítulo 4. Control óptimo libre de modelo por aprendizaje reforzado.** Se presenta el planteamiento de un problema de control óptimo y se extiende su solución a un marco de referencia del aprendizaje reforzado. Se presenta un algoritmo que determina en línea la solución de un regulador cuadrático lineal, sin conocimiento de la dinámica del

sistema. Posteriormente, se propone evaluar al algoritmo de ante variación paramétrica. Se compara la respuesta obtenida por la solución aproximada contra la solución real del controlador óptimo.

- **Capítulo 5. Conclusiones.** En este capítulo se brindan las conclusiones finales del trabajo realizado en esta tesis, resaltando aspectos del diseño y la validación de los controladores inteligentes. Así mismo, se plantean posibles áreas de oportunidad para trabajos a futuro.

CAPÍTULO 2

Modelado y control del péndulo invertido rotativo

El péndulo invertido rotativo, también conocido como péndulo Furuta, es un mecanismo conformado por un brazo que rota en el plano horizontal y un péndulo que rota sobre el plano vertical, tal como se muestra en la Figura 2-1. El sistema sólo cuenta con un actuador que aplica un par al brazo y controla indirectamente a la posición del péndulo. Por lo tanto, es un sistema mecánico subactuado de dos grados de libertad.

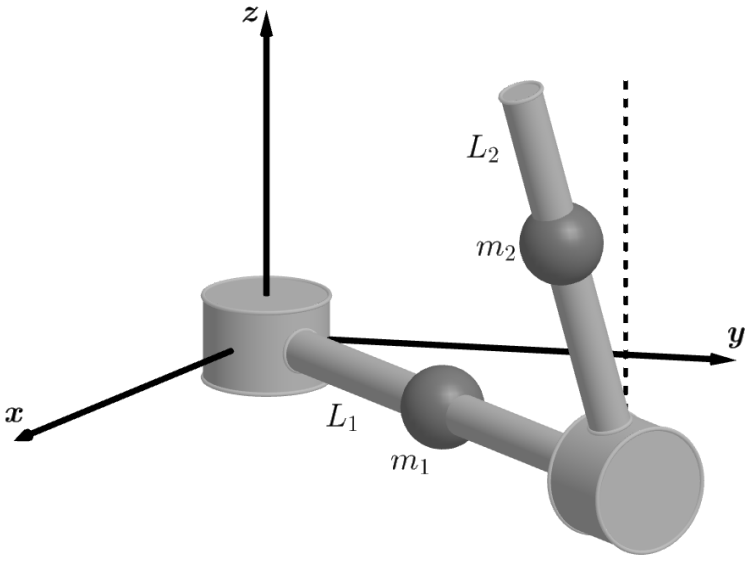

Figura 2-1: Péndulo Furuta

El péndulo Furuta requiere menos espacio que el clásico péndulo invertido montado sobre un car-

ro; además, se tienen menos dinamicas no modeladas a casusa de un mecanismo de transmisión de energía, ya que el eje sobre el que gira el brazo horizontal está directamente unido al eje del motor [7]. Sin embargo, este sistema es altamente no lineal debido a las fuerzas gravitacionales y al acoplamiento entre las fuerzas centripetas y de Coriolis [64].

La posición angular del brazo está dada por θ y la posición del péndulo por α, ambas se incrementa de forma positiva al rotar en el sentido de las manecillas del reloj. Además, α es cero cuando el péndulo se encuentra en la posición vertical superior. El brazo tiene longitud L_1, masa m_1, distancia al centro de masa l_1 y un momento de inercia J_1. Así mismo, el péndulo tiene longitud L_2, masa m_2, distancia al centro de masa l_2 y un momento de inercia J_2.

Cada junta rotacional está viscosamente amortiguada con factores de amortiguamiento b_1 y b_2, en donde b_1 está asociado a los rodamientos del motor y b_2 a la unión de ambos eslabones. De igual forma, se propone considerar una versión continua y diferenciable de la fricción de Coulomb, debido a que permite evitar problemas en las simulaciones numéricas que comunmente aparecen con los modelos discontinuos [5].

2.1 Modelado

Para el desarrollo del modelo dinámico del sistema se consideran las siguientes suposiciones [64]:

1. Se asume que el motor y el brazo horizontal están rigidamente unidos y son infinitamente rigidos.
2. El péndulo se asume infinitamente rigido.
3. Se asume despreciable a la inercia del rotor del motor.
4. Se tomará en cuenta al amortiguamiento viscoso y la fricción de Coulomb.

2.1.1 Cinemática

El análisis cinemático permite describir el movimiento de sistemas mecánicos sin considerar las fuerzas y pares que lo provocan, es decir, se trata de una descripción geométrica. La cinemática directa proporciona la posición y orientación del efector o el extremo final de un manipulador

a partir de las variables articulares del robot, las cuales corresponden a los ángulos para juntas rotacionales y las extensiones para juntas prismaticas [65].

Por lo que, se realizó el análisis geométrico de las juntas rotacionales que se muestra en la Figura 2-2. El lado izquierdo coresponde al plano horizontal y contiene el análisis del brazo y la proyección del péndulo sobre este plano; mientras que a la derecha se muestra al péndulo sobre el plano vertical.

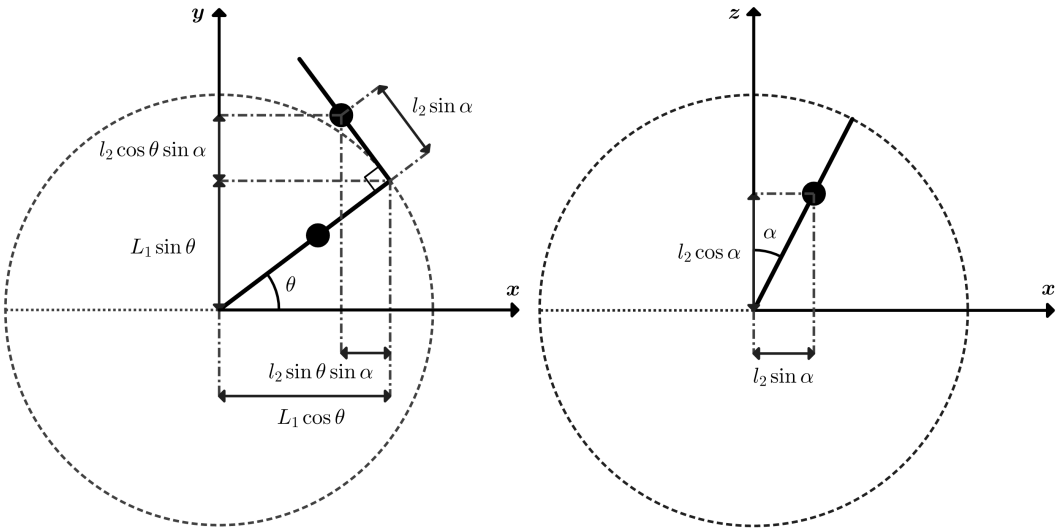

Figura 2-2: Planos horizontal y vertical

Se propone considerar al vector de coordenadas generalizadas $q(t) = [q_1(t),\ q_2(t)] = [\theta(t),\ \alpha(t)]$. Tal que la posición del centro de masa del brazo está dada por:

$$r_{c1} = [x_{c1}, y_{c1}, z_{c1}]^T \tag{2.1}$$

con $x_{c1} = l_1 \cos q_1$, $y_{c1} = l_1 \sin q_1$ y $z_{c1} = 0$. Mientras que para la posición del centro de masa del péndulo se tiene:

$$r_{c2} = [x_{c2}, y_{c2}, z_{c2}]^T \tag{2.2}$$

donde $x_{c2} = L_1 \cos q_1 - l_2 \sin q_1 \sin q_2$, $y_{c2} = L_1 \sin q_1 + l_2 \cos q_1 \sin q_2$ y $z_{c2} = l_2 \cos q_2$. Al derivar con respecto al tiempo a (2.1) se obtiene la expresión de la velocidad del centro de masa del brazo:

$$v_{c1} = [\dot{x}_{c1}, \dot{y}_{c1}, \dot{z}_{c1}]^T \tag{2.3}$$

con $\dot{x}_{c1} = -l_1 \sin q_1 \dot{q}_1$, $\dot{y}_{c1} = l_1 \cos q_1 \dot{q}_1$ y $\dot{z}_{c1} = 0$. Al calcular $v_{c1}^T v_{c1}$ se tiene:

$$v_{c1}^T v_{c1} = l_1^2 \dot{q}_1^2 \tag{2.4}$$

Mientras que para la velocidad del centro de masa del péndulo se calcula la derivada temporal de (2.5):

$$v_{c2} = [\dot{x}_{c2}, \dot{y}_{c2}, \dot{z}_{c2}]^T \tag{2.5}$$

con $\dot{x}_{c2} = -L_1 \sin q_1 \dot{q}_1 - l_2 \cos q_1 \sin q_2 \dot{q}_1 - l_2 \sin q_1 \cos q_2 \dot{q}_2$, $\dot{y}_{c2} = L_1 \cos q_1 \dot{q}_1 - l_2 \sin q_1 \sin q_2 \dot{q}_1 + l_2 \cos q_1 \cos q_2 \dot{q}_2$ y $\dot{z}_{c2} = -l_2 \sin q_2 \dot{q}_2$. Así mismo, de $v_{c2}^T v_{c2}$ se obtiene:

$$v_{c2}^T v_{c2} = \left(L_1^2 + l_2^2 \sin^2 q_2\right) \dot{q}_1^2 + 2 L_1 l_2 \cos q_2 \dot{q}_1 \dot{q}_2 + l_2^2 \dot{q}_2^2 \tag{2.6}$$

2.1.2 Energía del sistema

Es posible obtener las expresiones de energía cinética y potencial de un sistema mecánico en términos de las coordenadas generalizadas $\mathbf{q}(t)$. La energía cinética de un cuerpo rígido es la suma de dos términos, la energía cinética traslacional derivada por la concentración de la masa de un objeto en su centro de masa y la energía cinética rotacional del cuerpo sobre el centro de masa [65]. La energía cinética total del sistema está dada como:

$$K(q, \dot{q}) = K_1 + K_2$$

Donde las expresiones que describen a la energía cinética del brazo y el péndulo son:

$$K_1 = \frac{1}{2} m_1 v_{c1}^T v_{c1} + \frac{1}{2} J_1 \dot{q}_1 \tag{2.7}$$

$$K_2 = \frac{1}{2} m_2 v_{c2}^T v_{c2} + \frac{1}{2} J_2 \dot{q}_2 \tag{2.8}$$

Al sustituir a (2.4) en (2.7) y (2.6) en (2.8), se tiene a la energía cinética total como:

$$K(q, \dot{q}) = \frac{1}{2}\left(m_1 l_1^2 + J_1 + m_2 L_1^2 + m_2 l_2^2 \sin^2 q_2\right) \dot{q}_1^2 + m_2 L_1 l_2 \cos q_2 \dot{q}_1 \dot{q}_2 + \frac{1}{2}\left(m_2 l_2^2 + J_2\right) \dot{q}_2^2 \tag{2.9}$$

Por otro lado, del marco de referencia inercial propuesto para el análisis del sistema se tiene que la acción de la gravedad se produce en el eje z. Debido a que el brazo se mueve sobre el plano horizontal, su energía potencial es constante y se asume como $P_1 = 0$. Por lo que, sólo depende de la energía potencial del péndulo:

$$P(q) = g m_2 l_2 \cos q_2 \tag{2.10}$$

2.1.3 Modelo dinámico

La ecuaciones dinámicas de un sistema mecánico describen explicitamente la relación entre el movimiento y las fuerzas que lo provocan. Para esto se considera a las ecuaciones de movimiento de Euler-Lagrange, las cuales permiten describir la evolución de un sistema mecánico [65]. Para derivar las ecuaciones dinámicas del sistema es necesario construir su Lagrangiano, el cual está dado por la diferencia entre las energías cinética y potencial:

$$\mathcal{L}(\mathbf{q}, \dot{\mathbf{q}}) = K(\mathbf{q}, \dot{\mathbf{q}}) - P(\mathbf{q}) \tag{2.11}$$

Debido a que la energía cinética de un sistema mecánico es una función cuadrática del vector de velocidad $\dot{\mathbf{q}}$ [65], es posible reesribir a (2.9) como:

$$K(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2} \dot{\mathbf{q}}^T M(\mathbf{q}) \dot{\mathbf{q}} \tag{2.12}$$

donde $M(\mathbf{q}) \in \mathbb{R}^{2\times 2}$ es conocida como *Matriz de inercia* y deberá ser una matriz simétrica y definida positiva para todo $\mathbf{q} \in \mathbb{R}^n$, dada como:

$$M(\mathbf{q}) = \begin{bmatrix} J_1 + m_1 l_1^2 + m_2 L_1^2 + m_2 l_2^2 \sin^2 q_2 & m_2 L_1 l_2 \cos q_2 \\ m_2 L_1 l_2 \cos q_2 & J_2 + m_2 l_2^2 \end{bmatrix} \tag{2.13}$$

Se observa que la matriz de inercia es simétrica; además, determinar si es una matriz definida positiva se deberá comprobar que todos sus menores principales son positivos, es decir:

$$\det(m_1) = J_1 + m_1 l_1^2 + m_2 L_1^2 + m_2 l_2^2 \sin^2 q_2 \geq J_1 + m_1 l_1^2 + m_2 L_1^2 > 0 \tag{2.14}$$

$$\begin{aligned} \det(m_2) &= \det \begin{bmatrix} J_1 + m_1 l_1^2 + m_2 L_1^2 + m_2 l_2^2 \sin^2 q_2 & m_2 L_1 l_2 \cos q_2 \\ m_2 L_1 l_2 \cos q_2 & J_2 + m_2 l_2^2 \end{bmatrix} \\ &= \left(J_1 + m_1 l_1^2 + m_2 L_1^2\right)\left(J_2 + m_2 l_2^2\right) + \left(J_2 + m_2 l_2^2\right) m_2 l_2^2 \sin^2 q_2 - m_2^2 L_1^2 l_2^2 \cos^2 q_2 \end{aligned}$$

De donde es posible obtener:

$$\begin{aligned} \det\left(M(\mathbf{q})\right) &= \left(J_1 + m_1 l_1^2\right)\left(J_2 + m_2 l_2^2\right) + J_2 m_2 L_1^2 + m_2 l_2^2\left(m_2 L_1^2 + J_2 + m_2 l_2^2\right)\sin^2 q_2 & (2.15)\\ &\geq \left(J_1 + m_1 l_1^2\right)\left(J_2 + m_2 l_2^2\right) + J_2 m_2 L_1^2 > 0 & (2.16)\end{aligned}$$

Por lo tanto, $M\left(\mathbf{q}\right)$ es una matriz simétrica y positiva definida para toda $q \in \mathbb{R}^n$.

Por otro lado, se sabe que las ecuaciones de movimiento de Euler-Lagrange están dadas por la expresión:

$$\frac{d}{dt}\left(\frac{\partial}{\partial \dot{q}_k}\mathcal{L}\left(\mathbf{q}, \dot{\mathbf{q}}\right)\right) - \frac{\partial}{\partial q_k}\mathcal{L}\left(\mathbf{q}, \dot{\mathbf{q}}\right) = \tau_k \tag{2.17}$$

Tal que al considerar un sistema mecánico se tiene que:

$$\begin{aligned} \frac{d}{dt}\left(\frac{\partial}{\partial \dot{q}_k}\mathcal{L}\left(\mathbf{q}, \dot{\mathbf{q}}\right)\right) &= \sum_j m_{kj}\ddot{q}_j + \sum_{i,\,j}\frac{\partial m_{kj}}{\partial q_i}\dot{q}_i\dot{q}_j\\ \frac{\partial}{\partial q_k}\mathcal{L}\left(\mathbf{q}, \dot{\mathbf{q}}\right) &= \frac{1}{2}\sum_{i,\,j}\frac{\partial m_{ij}}{\partial q_k}\dot{q}_i\dot{q}_j - \frac{\partial P\left(\mathbf{q}\right)}{\partial q_k}\end{aligned}$$

De tal forma que se reescribe a (2.17) como:

$$\sum_j m_{kj}\ddot{q}_j + \sum_{i,\,j}\left(\frac{\partial m_{kj}}{\partial q_i} - \frac{1}{2}\frac{\partial m_{ij}}{\partial q_k}\right)\dot{q}_i\dot{q}_j + \frac{\partial P\left(\mathbf{q}\right)}{\partial q_k} = \tau_k \tag{2.18}$$

Al intercambiar el orden de la suma y aprovechar la simetría se obtiene [65]:

$$\sum_{i,\,j}\left(\frac{\partial m_{kj}}{\partial q_i} - \frac{1}{2}\frac{\partial m_{ij}}{\partial q_k}\right)\dot{q}_i\dot{q}_j = \sum_{i,\,j}\frac{1}{2}\left(\frac{\partial m_{kj}}{\partial q_i} + \frac{\partial m_{ki}}{\partial qj} - \frac{\partial m_{ij}}{\partial q_k}\right)\dot{q}_i\dot{q}_j \tag{2.19}$$

Donde los términos c_{ijk} son como los símbolos de Christoffel y están definidos como:

$$c_{ijk} := \frac{1}{2}\left[\frac{\partial m_{kj}}{\partial q_i} + \frac{\partial m_{ki}}{\partial q_j} - \frac{\partial m_{ij}}{\partial q_k}\right] \tag{2.20}$$

Los símbolos de Christoffel satisfacen que para una k fija se tiene $c_{ijk} = c_{jik}$, es decir, se tiene $c_{12k} = c_{21k}$ en:

$$\begin{aligned} c_{111} &= \tfrac{1}{2}\tfrac{\partial m_{11}}{\partial q_1} = 0 & c_{112} &= -\tfrac{1}{2}\tfrac{\partial m_{11}}{\partial q_2} = -\tfrac{1}{2}h_1\\ c_{121} &= c_{211} = \tfrac{1}{2}\tfrac{\partial m_{11}}{\partial q_2} = \tfrac{1}{2}h_1 & c_{122} &= c_{212} = 0\\ c_{221} &= \tfrac{\partial m_{12}}{\partial q_2} = h_2 & c_{222} &= \tfrac{1}{2}\tfrac{\partial m_{22}}{\partial q_2} = 0\end{aligned} \tag{2.21}$$

con $h_1 = m_2 l_2^2 \sin\left(2q_2\right)$ y $h_2 = -m_2 L_1 l_2 \sin\left(q_2\right)$.

A partir de 2.18 se observa que se tienen términos relacionados con la segunda derivada temporal de las coordenadas generalizadas, términos cuadraticos de la primera derivada temporal de $\mathbf{q}$, en donde los coeficientes dependen de $\mathbf{q}$ y términos que dependen sólo de $\mathbf{q}$ que corresponden a la derivada parcial de la energía potencial. Con respecto a los términos cuadraticos de la primera derivada temporal de $\mathbf{q}$ se sabe que para los productos del tipo $\dot{q}_i\dot{q}_j$ con $i = j$ son llamados términos centrífugos y para $i \neq j$ se denominan términos de Coriolis [65]. Al agrupar los términos se obtiene la siguiente expresión:

$$C\left(\mathbf{q}, \dot{\mathbf{q}}\right)\dot{\mathbf{q}} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}\dot{\mathbf{q}} \tag{2.22}$$

con $C\left(\mathbf{q}, \dot{\mathbf{q}}\right) \in \mathbb{R}^{2\times 2}$, cuyas entradas están dadas como:

$$\begin{aligned} c_{11} &= c_{111}\dot{q}_1 + c_{211}\dot{q}_2 = \frac{1}{2}m_2 l_2^2 \sin 2q_2\dot{q}_2 \\ c_{12} &= c_{121}\dot{q}_1 + c_{221}\dot{q}_2 = \frac{1}{2}m_2 l_2^2 \sin 2q_2\dot{q}_1 - m_2 L_1 l_2 \sin q_2\dot{q}_2 \\ c_{21} &= c_{112}\dot{q}_1 + c_{212}\dot{q}_2 = -\frac{1}{2}m_2 l_2^2 \sin 2q_2\dot{q}_1 \\ c_{22} &= c_{122}\dot{q}_1 + c_{222}\dot{q}_2 = 0 \end{aligned}$$

Por lo que la matriz de Coriolis será:

$$C\left(\mathbf{q}, \dot{\mathbf{q}}\right) = \begin{bmatrix} \frac{1}{2}m_2 l_2^2 \sin 2q_2\dot{q}_2 & \frac{1}{2}m_2 l_2^2 \sin 2q_2\dot{q}_1 - m_2 L_1 l_2 \sin q_2\dot{q}_2 \\ -\frac{1}{2}m_2 l_2^2 \sin 2q_2\dot{q}_1 & 0 \end{bmatrix} \tag{2.23}$$

Propiedad 1 (Propiedad de antisimetría) *Sea $M\left(\mathbf{q}\right)$ la matriz de inercia de un robot, con $C\left(\mathbf{q}, \dot{\mathbf{q}}\right)$ definida en términos de las entradas de $M\left(\mathbf{q}\right)$. Entonces la matriz $\dot{M}\left(\mathbf{q}\right) - 2C\left(\mathbf{q}, \dot{\mathbf{q}}\right)$ será una matriz antisimétrica, es decir, que satisface:*

$$\mathbf{z}^T\left(\dot{M}\left(\mathbf{q}\right) - 2C\left(\mathbf{q}, \dot{\mathbf{q}}\right)\right)\mathbf{z} = 0 \;\forall\; \mathbf{z} \in \mathbb{R}^n$$

Demostración. Al derivar con respecto al tiempo a la matriz de inercia:

$$\dot{M}\left(\mathbf{q}\right) = \begin{bmatrix} m_2 l_2^2 \sin 2q_2\dot{q}_2 & -m_2 L_1 l_2 \sin q_2\dot{q}_2 \\ -m_2 L_1 l_2 \sin q_2\dot{q}_2 & 0 \end{bmatrix}$$

Tal que de $\dot{M}\left(\mathbf{q}\right)$ y $C\left(\mathbf{q}, \dot{\mathbf{q}}\right)$ se tiene:

$$\dot{M}\left(\mathbf{q}\right) - 2C\left(\mathbf{q}, \dot{\mathbf{q}}\right) = \left(m_2 l_2^2 \sin 2q_2\dot{q}_1 - m_2 L_1 l_2 \sin q_2\dot{q}_2\right)\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

Sea el vector $\mathbf{z} = [z_1, z_2]^T$, con $z_1, z_2 \in \mathbb{R}$, tal que:

$$\mathbf{z}^T \left(\dot{M}(\mathbf{q}) - 2C(\mathbf{q}, \dot{\mathbf{q}}) \right) \mathbf{z} = \left(m_2 l_2^2 \sin 2q_2 \dot{q}_1 - m_2 L_1 l_2 \sin q_2 \dot{q}_2 \right) [z_1 z_2 - z_1 z_2] = 0$$

Por lo tanto, se concluye que $\dot{M}(\mathbf{q}) - 2C(\mathbf{q}, \dot{\mathbf{q}})$ es una matriz antisimétrica. ■

De igual forma, al derivar a la energía potencial con respecto al vector de coordenadas generalizadas se obtiene al vector de pares gravitacionales:

$$\mathbf{G}(\mathbf{q}) = \begin{bmatrix} 0 \\ -gm_2 l_2 \sin q_2 \end{bmatrix} \tag{2.24}$$

De tal forma que el modelo dinámico del robot será:

$$M(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \boldsymbol{\tau}$$

con $\boldsymbol{\tau} = [\tau_1,\ 0]$ el vector de fuerzas generalizadas en donde τ_1 corresponde al par aplicado con el motor de corriente directa al brazo horizontal. Además, se propone considerar a la fricción como $\mathbf{F}(\dot{\mathbf{q}}) = [b_1 \dot{q}_1, b_2 \dot{q}_2]^T$, donde b_1 y b_2 corresponden a los coeficientes de fricción viscosa del motor y el péndulo, respectivamente. Por lo tanto, se reescribe al modelo dinámico del sistema como:

$$M(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \boldsymbol{\tau} - \mathbf{F}(\dot{\mathbf{q}}) \tag{2.25}$$

2.1.4 Modelo electromecánico

Debido a que el péndulo Furuta utiliza un motor de corriente directa se considera el acoplamiento entre el sitema eléctrico y el sistema mecánico. El modelado de un motor de corritente directa se puede dividr en subsitemas electrico y mecánico. El subsistema eléctrico se obtiene a partir de la ley de tensión de Kirchhoff, tal que:

$$L_m \dot{i}_m + R_m i_m + K_b \dot{q}_1 = V_m \tag{2.26}$$

donde V_m es la tensión aplicada al servomotor, i_m la corriente de armadura, K_b la constante electromotriz, R_m y L_m la resistencia e inductancia de armadura, respectivamente. Se sabe que $L_m \dot{i}_m$ es mucho menor en magnitud en comparación con $R_m i$ y $K_m \dot{q}_1$, tal que se propone despreciar al término $L_m \dot{i}_m$ para simplificar el modelo y considerar a la corriente de armadura como:

$$i_m = \frac{V_m - K_b \dot{q}_1}{R_m} \tag{2.27}$$

La relación entre el par y la corriente de armadura está dada por $\tau_1 = K_m i_m$, donde K_m es la constante de par del motor. Además, se propone considerar a la eficiencia del motor, η_m, la relación de transmición de los engranes del servo, K_g, y la eficiencia de los engranes, η_g. Por lo que el par aplicado al brazo actuado estará dado por:

$$\tau_1 = \frac{\eta_g K_g \eta_m K_m}{R_m} \left(V_m - K_g K_b \dot{q}_1 \right) \tag{2.28}$$

Por lo que, al reescribir el modelo dinámico del sistema en función de la tensión aplicada al servomotor, se tiene:

$$M\left(\mathbf{q}\right) \ddot{\mathbf{q}} + C\left(\mathbf{q}, \dot{\mathbf{q}}\right) \dot{\mathbf{q}} + \mathbf{G}\left(\mathbf{q}\right) = \mathbf{U} - \mathbf{F}_m\left(\dot{\mathbf{q}}\right) \tag{2.29}$$

donde se agrupa a los términos disipativos en el vector $\mathbf{F}_m\left(\dot{\mathbf{q}}\right)$ con:

$$\mathbf{F}_m\left(\dot{\mathbf{q}}\right) = \begin{bmatrix} \left(b_1 + \beta_m K_g K_b\right) \dot{q}_1 \\ b_2 \dot{q}_2 \end{bmatrix} \tag{2.30}$$

con la tensión aplicada al motor como $u = V_m$ en el vector:

$$\mathbf{U} = \begin{bmatrix} \beta_m u & 0 \end{bmatrix}^T \tag{2.31}$$

donde:

$$\beta_m = \frac{\eta_g K_g \eta_m K_m}{R_m} \tag{2.32}$$

2.1.5 Linealización del modelo

El proceso de linealización resulta un método eficaz para aproximar el comportamiento de un sistema no lineal y consiste en describir su dinámica al rededor de un punto de equilibrio, considerando que esta aproximación sólo es válida en una pequeña vecindad del mismo. De acuerdo con Khalil [66], se dice que un punto $\mathbf{x}^*$ en el espacio de estados es un punto de equilibrio de un sistema $\mathbf{x} = f(\mathbf{x})$, si tiene la propiedad de que siempre que el estado inicie en el punto $\mathbf{x}^*$ se mantendrá en el mismo punto por el resto del tiempo, es decir $f(\mathbf{x}^*) = 0$.

Por lo tanto, para linealizar al modelo dinámico del péndulo Furuta 2.25 se propone considerar al vector de aceleración:

$$\ddot{\mathbf{q}} = M^{-1}(\mathbf{q})\left[\mathbf{U} - C(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} - \mathbf{G}(\mathbf{q}) - \mathbf{F}_m(\dot{\mathbf{q}})\right] \tag{2.33}$$

Con $M^{-1}(\mathbf{q})$ dada como:

$$M^{-1}(\mathbf{q}) = \frac{\operatorname{adj}(M(\mathbf{q}))}{\det(M(\mathbf{q}))} \tag{2.34}$$

donde:

$$\operatorname{adj}(M(\mathbf{q})) = \begin{bmatrix} J_2 + m_2 l_2^2 & -m_2 L_1 l_2 \cos q_2 \\ -m_2 L_1 l_2 \cos q_2 & J_1 + m_1 l_1^2 + m_2 L_1^2 + m_2 l_2^2 \sin^2 q_2 \end{bmatrix} \tag{2.35}$$

Con lo que se reescribe al vector de aceleración:

$$\ddot{\mathbf{q}} = \begin{bmatrix} f_{a_1} + g_{a_1} u \\ f_{a_2} + g_{a_2} u \end{bmatrix}$$

Con:

$$\begin{bmatrix} f_{a_1} \\ f_{a_2} \end{bmatrix} = \frac{1}{\det(M(\mathbf{q}))} \begin{bmatrix} M_{22} e_1 - M_{12} e_2 \\ -M_{21} e_1 + M_{11} e_2 \end{bmatrix}$$

$$\begin{bmatrix} g_{a_1} \\ g_{a_2} \end{bmatrix} = \frac{\beta_m}{\det(M(\mathbf{q}))} \begin{bmatrix} J_2 + m_2 l_2^2 \\ -m_2 L_1 l_2 \cos q_2 \end{bmatrix}$$

y:

$$\begin{aligned} e_1 &= -C_{11}\dot{q}_1 - C_{12}\dot{q}_2 - (b_1 + \beta_m K_g K_b)\dot{q}_1 \\ e_2 &= -C_{21}\dot{q}_1 + g m_2 l_2 \sin q_2 - b_2 \dot{q}_2 \end{aligned}$$

Se introduce al vector de estados $\mathbf{x} = [x_1,\ x_2,\ x_3,\ x_4]^T = [q_1, \dot{q}_1,\ q_2,\ \dot{q}_2]^T$ para reescribir al sistema como:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, u) \tag{2.36}$$

con:

$$\mathbf{f}(\mathbf{x}, u) = \begin{bmatrix} x_2 \\ f_{a_1} + g_{a_1} u \\ x_4 \\ f_{a_2} + g_{a_2} u \end{bmatrix} \tag{2.37}$$

Se busca la representación lineal del sistema en el espacio de estados, dada por:

$$\dot{\mathbf{x}} = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}^*} \Delta\mathbf{x} + \left.\frac{\partial \mathbf{f}}{\partial u}\right|_{\mathbf{x}^*} u = A\Delta\mathbf{x} + \mathbf{B}u \tag{2.38}$$

Donde $\mathbf{x}^*$ corresponde al punto de equilibrio sobre el cual se va a linealizar al sistema y con $\Delta\mathbf{x} := \mathbf{x} - \mathbf{x}^*$. Se propone hallar lo puntos de equilibrio al considerar $\dot{\mathbf{x}} = \mathbf{0}$ con $\mathbf{0} \in \mathbb{R}^4$y $u = 0$. De f_1 y f_3 en 2.37 se obtiene $x_2^* = 0$ y $x_4^* = 0$, con lo cual se reduce a f_2 y f_4:

$$0 = \frac{-gm_2^2 L_1 l_2^2 \cos x_2^* \sin x_2^*}{(J_1 + m_1 l_1^2)(J_2 + m_2 l_2^2) + J_2 m_2 L_1^2 + m_2 l_2^2 (m_2 L_1^2 + J_2 + m_2 l_2^2) \sin^2 q_2} \tag{2.39}$$

$$0 = \frac{\left(J_1 + m_1 l_1^2 + m_2 L_1^2 + m_2 l_2^2 \sin^2 x_2^*\right)(g m_2 l_2 \sin x_2^*)}{(J_1 + m_1 l_1^2)(J_2 + m_2 l_2^2) + J_2 m_2 L_1^2 + m_2 l_2^2 (m_2 L_1^2 + J_2 + m_2 l_2^2) \sin^2 q_2} \tag{2.40}$$

Como $f_3 = 0$ y $f_4 = 0$ sólo se satisfacen con $\sin x_2^* = 0$, se tiene a $x_2^* = n\pi$ con $n \in \mathbb{Z}$. Por otro lado, debido a que el equilibrio del sistema no depende de x_1, se asume $x_1^* = n\pi$, con $n \in \mathbb{Z}$. Por lo tanto, se propone considerar como puntos de equilibrio a $\mathbf{x}^* = [0,\ 0,\ 0,\ 0]^T$ y $\mathbf{x}^* = [0,\ \pi,\ 0,\ 0]^T$, que corresponden a las posiciones vertical superior e inferior, respectivamente.

Para la linealización del sistema se considera al punto de equilibrio $\mathbf{x}^* = [0,\ 0,\ 0,\ 0]^T$ en 2.38, tal que se tiene:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & 0 & 1 \\ 0 & a_{42} & a_{43} & a_{44} \end{bmatrix},\ \mathbf{B} = \begin{bmatrix} 0 \\ b_2 \\ 0 \\ b_4 \end{bmatrix} \tag{2.41}$$

con las entradas:

$$\begin{aligned}
a_{22} &= -\frac{(J_2 + m_2 l_2^2)(b_1 + \beta_m K_g K_b)}{(J_1 + m_1 l_1^2)(J_2 + m_2 l_2^2) + J_2 m_2 L_1^2} \\
a_{23} &= -\frac{g m_2^2 L_1 l_2^2}{(J_1 + m_1 l_1^2)(J_2 + m_2 l_2^2) + J_2 m_2 L_1^2} \\
a_{24} &= \frac{m_2 L_1 l_2 b_2}{(J_1 + m_1 l_1^2)(J_2 + m_2 l_2^2) + J_2 m_2 L_1^2} \\
a_{42} &= \frac{m_2 L_1 l_2 (b_1 + \beta_m K_g K_b)}{(J_1 + m_1 l_1^2)(J_2 + m_2 l_2^2) + J_2 m_2 L_1^2} \\
a_{43} &= \frac{g m_2 l_2 (J_1 + m_1 l_1^2 + m_2 L_1^2)}{(J_1 + m_1 l_1^2)(J_2 + m_2 l_2^2) + J_2 m_2 L_1^2}
\end{aligned}$$

$$
\begin{aligned}
a_{44} &= -\frac{b_2\left(J_1+m_1l_1^2+m_2L_1^2\right)}{\left(J_1+m_1l_1^2\right)\left(J_2+m_2l_2^2\right)+J_2m_2L_1^2} \\
b_2 &= \frac{\beta_m\left(J_2+m_2l_2^2\right)}{\left(J_1+m_1l_1^2\right)\left(J_2+m_2l_2^2\right)+J_2m_2L_1^2} \\
b_4 &= -\frac{\beta_m m_2L_1l_2}{\left(J_1+m_1l_1^2\right)\left(J_2+m_2l_2^2\right)+J_2m_2L_1^2}
\end{aligned}
$$

2.2 Control PD con compensación no lineal

Un sistema no lineal es aquel cuya dinámica de entrada-salida no satisface los dos principios de linealidad, superposición y homogeneidad. En ingeniería, es posible identificar dinámicas no lineales en circuitos eléctricos, electrónicos, sistemas de potencia, sistemas mecánicos, reacciones químicas, entre otros; y la forma de entender estos sistemas, en un primer acercamiento, fue a través de modelos lineales [5].

Si bien la linealización permite aprender más sobre el comportamiento de sistemas no lineales, presenta dos limitaciones básicas [5]:

1. Al tratarse de una aproximación en una vecindad al rededor de un punto de operación, sólo puede predecir el comportamiento local del sistema no lineal.

2. Existen fenómenos no lineales que no se pueden aproximar mediante la linealización, como escapes en tiempo finito, múltiples puntos de equilibrio, ciclos límite, por mencionar algunos.

Por otro lado, los controladores no lineales han tenido buenos resultados en la teoría; sin embargo, muchos de ellos son complejos y dificiles de implementar en los sistemas reales, ya que requieren de mayor conocimiento del sistema o presentan fenómenos como el castañeo [60]. Además, el diseño de algoritmos de control para sistemas mecánicos subactuados es una tarea complicada, debido a que son altamente no lineales por el acoplamiento entre juntas y las restricciones presentes que impiden comandar directamente algunas de las configuraciones del robot [5].

No obstante, los esquemas de control PID y sus variantes han sido unos de los métodos más utilizados a nivel industrial, debido a su estructura simple y su robustez ante condiciones de

operación nominales. Sin embargo, los esquemas de control PID convencionales presentan deficiencias en su desempeño ante sistemas no lineales. Por lo que, se propone diseñar y analizar la estabilidad del controlador PD basado en el modelo no lineal del péndulo invertido rotativo.

2.2.1 Esquemas de control PD

El problema de control es diseñar un algoritmo que compute la tensión u que se aplica al motor del brazo actuado. Por lo que se propone introducir terminos de compensación no lineal en un controlador PD tradicional, que garanticen la estabilidad asintótica del sistema en lazo cerrado. Se presentan dos controladores tipo PD para el péndulo invertido rotativo, tal como se presentan en la Figura 2-3, en donde se denota al compensador no lineal mediante π.

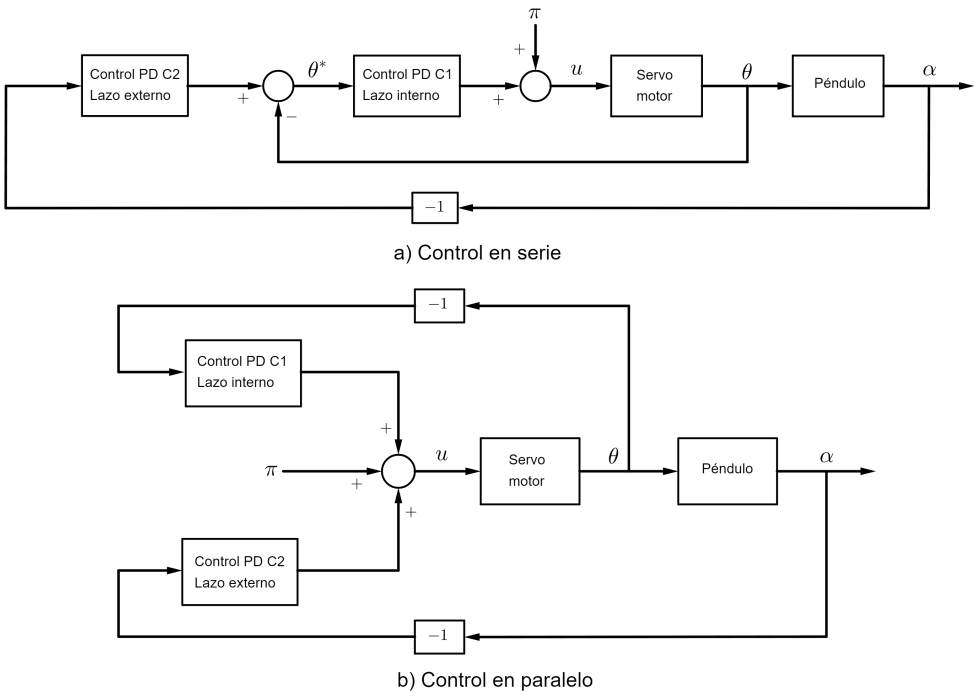

Figura 2-3: Esquemas de control PD

El primero se muestra en la Figura 2-3 (a) y es un controlador PD en serie, en donde el lazo interno regula la posición angular del brazo, dada por θ, mediante el controlador C_1. El lazo externo controla la posición angular del péndulo, dada por α, mediante el controlador C_2, para equilibrar al péndulo en la posición inestable $\alpha^* = 0$. Por lo tanto, el controlador PD en serie

tiene la forma:

$$\begin{aligned} u &= k_{p_1}\left(\theta^* - \theta\right) + k_{d_1}\left(\dot{\theta}^* - \dot{\theta}\right) + \pi \\ \theta^* &= -k_{p_2}\alpha - k_{d_2}\dot{\alpha} \end{aligned}$$

donde k_{p_1} y k_{d_1} son constantes positivas, que corresponden a los coeficientes proporcional y derivativo para el control del motor;k_{p_2} y k_{d_2} son constantes positivas, que corresponden a los coeficientes proporcional y derivativo para el control del péndulo. Además, se tiene:

$$\dot{\theta}^* = -k_{p_2}\dot{\alpha} - k_{d_2}\ddot{\alpha}$$

Por lo que, se puede reescribir al algoritmo en serie como:

$$u = -k_{p_1}\theta - k_{d_1}\dot{\theta} - k_{p_1}k_{p_2}\alpha - \left(k_{p_1}k_{d_2} + k_{d_1}k_{p_2}\right)\dot{\alpha} - k_{d_1}k_{d_2}\ddot{\alpha} + \pi \tag{2.42}$$

Por otro lado, el segundo controlador corresponde a un esquema PD en paralelo y se muestra en la Figura 2-3 (b). Como se aborda el problema de regulación, el objetivo de control es estabilizar al péndulo sobre el equilibrio inestable, $[\theta^*, \alpha^*] = [0, 0]$, de tal forma que $\left[\dot{\theta}^*, \dot{\alpha}^*\right] = [0, 0]$, con lo cual se define al algoritmo de control en paralelo como:

$$u = -k_{p_1}\theta - k_{d_1}\dot{\theta} - k_{p_2}\alpha - k_{d_2}\dot{\alpha} + \pi \tag{2.43}$$

Al observar a (2.42) y (2.43), es fácil observar que se puede reescribir a ambos esquemas de control como una representación única, dada por:

$$u = -a_1\theta - a_2\dot{\theta} - a_3\alpha - a_4\dot{\alpha} - a_5\ddot{\alpha} + \pi \tag{2.44}$$

Donde para el control PD en serie se tiene $a_1 = k_{p_1}$, $a_2 = k_{d_1}$, $a_3 = k_{p_1}k_{p_2}$, $a_4 = k_{p_1}k_{d_2} + k_{d_1}k_{p_2}$ y $a_5 = k_{d_1}k_{d_2}$. Mientras que para el control en paralelo se considera $a_1 = k_{p_1}$, $a_2 = k_{d_1}$, $a_3 = k_{p_2}$, $a_4 = k_{d_2}$ y $a_5 = 0$. En ambos casos se tiene $a_i > 0$ con $i = 1, \ldots, 5$.

2.2.2 Compensación basada en el modelo no lineal

Se retoma al vector de coordenadas generalizadas $q = [\theta, \alpha]^T$, tal que el error de regulación sea:

$$\tilde{q} = q^* - q$$

donde q^* es el vector de valores objetivo, $q^* = [\theta^*, \alpha^*]^T = [0, 0]^T$.

Es bien sabido que es posible demostrar la estabilidad de robots con controladores PD a través del método directo de Lyapunov. Por lo que, se sustituye a la ley de control (2.44) en el modelo completo del péndulo invertido rotativo con el servo motor, mostrado en (2.29), para obtener la expresión en lazo cerrado:

$$M_u(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = B\tilde{\mathbf{q}} + \mathbf{D} - \mathbf{F}_u(\dot{\mathbf{q}}) \tag{2.45}$$

con:

$$\begin{aligned}
M_u &= \begin{bmatrix} \theta_1 + \theta_2 \sin^2 q_2 & \theta_3 \cos q_2 + \beta_m a_5 \\ \theta_3 \cos q_2 - \beta_m a_5 & \theta_4 \end{bmatrix} \\
F_u &= \begin{bmatrix} (b_1 + \beta_m K_g K_b + \beta_m a_2)\dot{q}_1 + \beta_m a_4 \dot{q}_2 \\ b_2 \dot{q}_2 \end{bmatrix} \\
B &= \begin{bmatrix} \beta_m a_1 & \beta_m a_3 \\ \beta_m a_3 & 1 \end{bmatrix}, \; D = \begin{bmatrix} \beta_m \pi \\ -\beta_m a_5 \ddot{q}_1 + \beta_m a_3 q_1 + q_2 \end{bmatrix}
\end{aligned}$$

De donde es posible determinar si $M_u(x)$ es una matriz positiva definida a través de:

$$\begin{aligned}
\det(m_1) &= \det\left(\theta_1 + \theta_2 \sin^2 q_2\right) \\
&= J_1 + m_1 l_1^2 + m_2 L_1^2 + m_2 l_2^2 \sin^2 q_2 \\
&\geq J_1 + m_1 l_1^2 + m_2 L_1^2 > 0
\end{aligned}$$

$$\begin{aligned}
\det(m_2) &= \det \begin{bmatrix} \theta_1 + \theta_2 \sin^2 q_2 & \theta_3 \cos q_2 + \beta_m a_5 \\ \theta_3 \cos q_2 - \beta_m a_5 & \theta_4 \end{bmatrix} \\
&= \theta_1\theta_4 + \theta_2\theta_4 \sin^2 q_2 - (\theta_3 \cos q_2 + \beta_m a_5)(\theta_3 \cos q_2 - \beta_m a_5) \\
&= \theta_1\theta_4 + \theta_2\theta_4 \sin^2 q_2 - \theta_3^2 \cos^2 q_2 + \beta_m^2 a_5^2 \\
&= \left(J_1 + m_1 l_1^2\right)\theta_4 + J_2 m_2 L_1^2 + \theta_3^2 + \theta_2\theta_4 \sin^2 q_2 + \beta_m^2 a_5^2 \\
&\geq \left(J_1 + m_1 l_1^2\right)\theta_4 + J_2 m_2 L_1^2 + \theta_3^2 + \beta_m^2 a_5^2 > 0
\end{aligned}$$

De igual forma, de los menores principales de la matriz B se tiene:

$$\det(m_1) = \det(\beta_m a_1) = \beta_m a_1 > 0$$

$$\text{con } \beta_m > 0 \text{ y } a_1 > 0$$

$$\det(m_2) = \det\begin{bmatrix} \beta_m a_1 & \beta_m a_3 \\ \beta_m a_3 & 1 \end{bmatrix} = \beta_m a_1 - \beta_m^2 a_3^2$$

Por lo que, se deberá satisfacer la condición $a_1 > \beta_m a_3^2$ para garantizar que $\det(m_2) > 0$ y por consecuencia que B sea una matriz positiva definida.

Theorem 2 *El control PD en serie o en paralelo como en (2.44) con el término de compensación dado como:*

$$\begin{aligned} \pi &= \frac{1}{\beta_m}\left(b_1\dot{q}_1 + \beta_m K_g K_b \dot{q}_1 + \beta_m a_4 \dot{q}_2\right) \\ &\quad -\frac{\dot{q}_2}{\beta_m \dot{q}_1}\left(\beta_m a_3 q_1 + q_2 + g m_2 l_2 \sin q_2 - b_2\dot{q}_2 - \beta_m a_5 \ddot{q}_1\right) \end{aligned}$$

puede garantizar la estabilidad del del péndulo Furuta (2.29), si se satisface la condición:

$$a_1 > \beta_m a_3^2$$

Demostración. Como $M_u(x)$ y B son matrices positivas definidas, se propone la siguiente expresión cuadrática como función candidata de Lyapunov:

$$V(x, \dot{x}) = \frac{1}{2}\dot{q}^T M_u(q)\,\dot{q} + \frac{1}{2}\tilde{q}^T B \tilde{q}$$

Al derivarla con respecto al tiempo, se obtiene:

$$\dot{V} = \dot{q}^T M_u(q)\,\ddot{q} + \frac{1}{2}\dot{q}^T \dot{M}_u(q)\,\dot{q} - \dot{q}^T B\tilde{q}$$

Con:

$$M_u(q)\,\ddot{q} = B\tilde{q} + D - F(\dot{q}) - C(q, \dot{q})\,\dot{q} - G(q) \tag{2.46}$$

Entonces:

$$\begin{aligned} \dot{V} &= \dot{x}^T\left[B\tilde{q} + D - F_u(\dot{q}) - C(q,\dot{q})\,\dot{q} - G(q)\right] + \frac{1}{2}\dot{q}^T\dot{M}_u(q)\,\dot{q} - \dot{q}^T B\tilde{q} \\ &= \dot{q}^T\left[B\tilde{q} + D - F_u(\dot{q}) - G(q) - B\tilde{q}\right] + \frac{1}{2}\dot{q}^T\left[\dot{M}_u(q) - C(q,\dot{q})\right]\dot{q} \\ &= \frac{1}{2}\dot{q}^T\left[\dot{M}_u(q) - C(q,\dot{q})\right]\dot{q} + \dot{q}^T D - \dot{q}^T F_u(\dot{q}) - \dot{q}^T G(q) \end{aligned}$$

Como $M(q)$ y $C(q, \dot{q})$ satisfacen la propiedad de simetria y $\dot{M}_u(q) = \dot{M}(q)$, entonces:

$$\frac{1}{2}\dot{q}^T\left[\dot{M}_u(q) - C(q,\dot{q})\right]\dot{q} = 0 \tag{2.47}$$

Además:

$$\begin{aligned}\dot{q}^T D &= \beta_m \pi \dot{q}_1 - \beta_m a_5 \ddot{q}_1 \dot{q}_2 + \beta_m a_3 q_1 \dot{q}_2 + q_2 \dot{q}_2 \\ \dot{q}^T F_u(\dot{q}) &= (b_1 + \beta_m K_g K_b + \beta_m a_2)\dot{q}_1^2 + \beta_m a_4 \dot{q}_1 \dot{q}_2 + b_2 \dot{q}_2 \\ \dot{q}^T G(q) &= -g m_2 l_2 \sin q_2 \dot{q}_2\end{aligned}$$

Por lo tanto, se tiene:

$$\begin{aligned}\dot{V} &= \beta_m \pi \dot{q}_1 - \beta_m a_5 \ddot{q}_1 \dot{q}_2 + \beta_m a_3 q_1 \dot{q}_2 + q_2 \dot{q}_2 \\ &\quad - (b_1 + \beta_m K_g K_b + \beta_m a_2)\dot{q}_1^2 - \beta_m a_4 \dot{q}_1 \dot{q}_2 - b_2 \dot{q}_2^2 + g m_2 l_2 \sin q_2 \dot{q}_2 \\ &= -\beta_m a_2 \dot{q}_1^2 + \beta_m \pi \dot{q}_1 - \dot{q}_1 (b_1 \dot{q}_1 + \beta_m K_g K_b \dot{q}_1 + \beta_m a_4 \dot{q}_2) \\ &\quad + \dot{q}_2 (\beta_m a_3 q_1 + q_2 + g m_2 l_2 \sin q_2 - b_2 \dot{q}_2 - \beta_m a_5 \ddot{q}_1) \end{aligned} \tag{2.48}$$

Por lo que se propone a π como:

$$\begin{aligned}\pi &= \frac{1}{\beta_m}(b_1 \dot{q}_1 + \beta_m K_g K_b \dot{q}_1 + \beta_m a_4 \dot{q}_2) \\ &\quad - \frac{\dot{q}_2}{\beta_m \dot{q}_1}(\beta_m a_3 q_1 + q_2 + g m_2 l_2 \sin q_2 - b_2 \dot{q}_2 - \beta_m a_5 \ddot{q}_1)\end{aligned} \tag{2.49}$$

Con lo cual:

$$\dot{V} \leq -\beta_m a_2 \dot{q}_1^2 \tag{2.50}$$

donde $\beta_m a_2 > 0$ permite que $\dot{V}$ sea una función semidefinida negativa . Por lo que, por el método directo de Lyapunov se puede concluir que $[\theta, \alpha] = [0, 0]$ es un equilibrio estable. ■

Theorem 3 *El término de compensación no lineal:*

$$\pi = \begin{cases} \frac{1}{\beta_m}(b_1 \dot{q}_1 + \beta_m K_g K_b \dot{q}_1 + \beta_m a_4 \dot{q}_2) \\ -\frac{\dot{q}_2}{\beta_m \dot{q}_1}(\beta_m a_3 q_1 + q_2 + g m_2 l_2 \sin q_2 - b_2 \dot{q}_2 - \beta_m a_5 \ddot{q}_1) & si\ \dot{q}_1 \neq 0 \\ a_4 \dot{q}_2 & si\ \dot{q}_1 = 0 \end{cases}$$

garantiza la estabilidad asintótica del péndulo Furuta (2.29), si se satisface la condición:

$$a_1 > \beta_m a_3^2$$

Demostración. Para probar la estabilidad asintótica se acude al teorema de invarianza de LaSalle y se busca la región:

$$\Psi = \left\{ [\theta, \alpha] : \dot{V} = 0 \right\} \tag{2.51}$$

El conjunto invariante se obtiene del sistema en lazo cerrado cuando se asume $\dot{q}_1 = 0$, tal que $\ddot{q}_1 = 0$ y $\pi = a_4\dot{q}_2$, con lo que se tiene:

$$\begin{aligned}(\theta_3 \cos q_2 + \beta_m a_5)\,\ddot{q}_2 - m_2 L_1 l_2 \sin q_2 \dot{q}_2^2 + \beta_m a_1 q_1 + \beta_m a_3 q_2 &= 0 \\ \theta_4 \ddot{q}_2 - g m_2 l_2 \sin q_2 + \beta_m a_3 q_1 + q_2 + b_2 \dot{q}_2 &= 0\end{aligned}$$

Si $\dot{q}_2 \neq 0$ entonces el péndulo estaría en movimiento y q_2 no sería constante. Por lo que se reduce la expresión anterior:

$$\begin{aligned}\beta_m a_1 q_1 + \beta_m a_3 q_2 &= 0 \\ -g m_2 l_2 \sin q_2 + \beta_m a_3 q_1 + q_2 &= 0\end{aligned}$$

Para $q_2 = 0$, se tiene:

$$\begin{aligned}\beta_m a_1 q_1 &= 0 \\ \beta_m a_3 q_1 &= 0\end{aligned}$$

Siendo fácil identificar que esto sólo se satisface para $[q_1, q_2] = [0, 0]$. Por lo tanto, por el teorema de LaSalle se puede asegurar que $[q_1, q_2] = [0, 0]$ es asintóticamente estable, es decir, se garantiza que:

$$\lim_{t\to\infty} \tilde{q}\,(t) = 0 \tag{2.52}$$

Por lo que, para el péndulo Furuta con $\alpha^* = 0$ y $\theta^* = 0$, se tiene que:

$$\begin{aligned}\lim_{t\to\infty} \theta\,(t) &= 0 \\ \lim_{t\to\infty} \alpha\,(t) &= 0\end{aligned}$$

■

CAPÍTULO 3

Control PD compensado por aprendizaje reforzado

En este capítulo se presenta a los sistema de control como un problema formal de los procesos de decisión de Markov (MDP) con transiciones deterministas en tiempo discreto. Dichos conceptos se emplean en el diseño de los algoritmos de aprendizaje reforzado para el control del péndulo Furuta. Se aborda el problema de equilibrar la exploración y explotación en un espacio de estado-acción continuo, en donde resulta imposible visitar múltiples veces todos los pares en tiempo finito. Se estudia la eficacia de los controladores inteligentes con arquitecturas tabulares y por aproximación mediante redes neuronales. Para esto, se presentan los métodos de aproximación parametrica de las funciones de valor.

3.1 Sistemas de control

El marco de los MDP es abstracto y flexible, tal que puede ser aplicado a multiples problemas en diferentes formas. El límite entre el agente y el entorno está dado por todo aquello que no puede ser cambiado arbitrariamente por el agente, es decir, representa el límite del control absoluto del agente y no de su conocimiento sobre el entorno.

Por lo tanto, es posible establecer una analogía conceptual entre un sistema de control y un MDP, como se muestra en la Figura 3-1. Recordando que en un sistema de control, el controlador es aquel que interactua con un sistema dinámico. El controlador recibe mediciones de los estados del sistema y con ello determina la señal de control a aplicar.

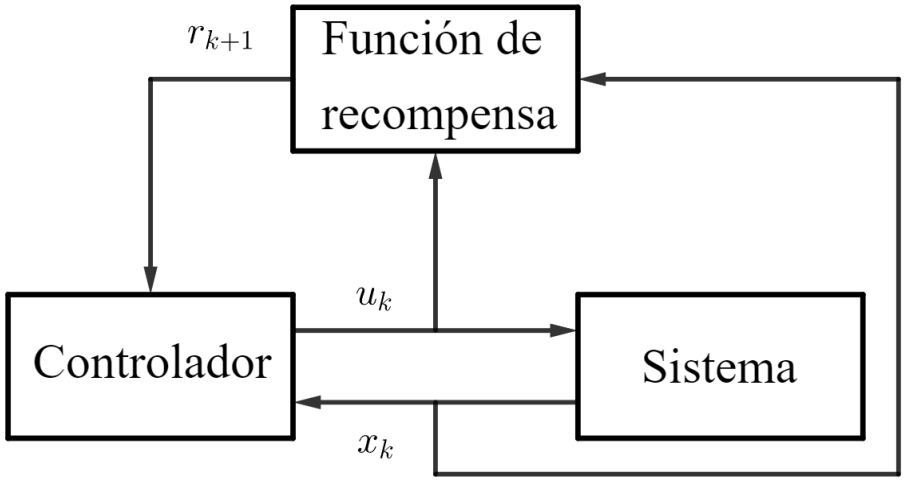

Figura 3-1: Sistema de control en la forma de un proceso de decisión de Markov

La dinámica del sistema, $f(\cdot)$, reemplaza a la función de transición, p, ya que describe el resultado de aplicar la acción u_k en el estado x_k en el paso de tiempo k, tal que el estado cambia a x_{k+1}, de acuerdo con la función $f : \mathcal{X} \times \mathcal{U} \rightarrow \mathcal{X}$:

$$x_{k+1} = f(x_k, u_k) \tag{3.1}$$

Además, se reescribe a la señal de recompensa, r_{k+1}, la cual evalua el desempeño inmediato de la acción de control de acuerdo con la función $\rho : \mathcal{X} \times \mathcal{U} \rightarrow \mathbb{R}$:

$$r_{k+1} = r(x_k, u_k) \tag{3.2}$$

donde se asume que $\|r\|_\infty = \sup_{x,u} |r(x,u)|$ es finito [10]. Recordando que la recompensa evalua el efecto inmediato de u_k en la transición de x_k a x_{k+1} y no sus efectos a largo plazo.
Como el controlador toma las acciones a partir de mediciones de los estados, se representa a la política $h : \mathcal{X} \rightarrow \mathcal{U}$, como:

$$u_k = h(x_k) \tag{3.3}$$

Es importante señalar que dados $f(\cdot)$ y $r(\cdot)$, basta con concer el estado actual x_k y la acción u_k para determinar el siguiente estado x_{k+1} y la recompensa r_{k+1}. Esta es la propiedad de Markov, la cual es esencial al proveer garantías teóricas en los algoritmos de DP y RL [10].

3.2 Aprendizaje por diferencia temporal

De acuerdo con [4], el aprendizaje por diferencia temporal (TD learning) es una idea central y novedosa en el aprendizaje reforzado. El TD learning es una combinación de ideas de los métodos Montecarlo y de DP. Los métodos de TD learning pueden aprender directamente de la experiencia, sin modelos del entorno, como los métodos de Monte Carlo. Además, los métodos TD actualizan sus estimaciones basadas en otras estimaciones aprendidas, sin esperar un resultado final, como en DP.

El aprendizaje por diferencia temporal es una forma de aprendizaje reorzado en línea, debido a que las acciones se mejoran en tiempo real en función de la estimación de sus funciones de valor mediante la observación de datos medidos a lo largo de las trayectorias del sistema [67].

En el paso de tiempo $k+1$, los métodos de TD forman inmediantamente un objetivo y realizan una actualización útil con la recompensa observada, r_{k+1}, y la estimación de la función de valor [4]. El método TD más simple realiza la actualización como:

$$V(x_k) \leftarrow V(x_k) + \alpha \left[r_{k+1} + \gamma V(x_{k+1}) - V(x_k) \right] \tag{3.4}$$

En donde $V(x_k)$ representa a la estimación de la función de valor $v(x_k)$. Es importante destacar que la operación dentro de los corchetes representa un tipo de error, el cual mide la dierencia entre el valor estimado en x_k y la estimación $r_{k+1} + \gamma V(x_{k+1})$. Se conoce a esta operación como error TD, δ_k, y se presenta en diferentes maneras dentro del RL [4].

3.3 Q-Learning

Uno de los primeros avances en el aprendizaje por refuerzo fue el desarrollo de un algoritmo de control por TD fuera de la política conocido como Q-learning [68]. Q-learning es un método libre de modelo para la iteración de la función Q y es el algoritmo más ampliamente utilizado en el aprendizaje reforzado [10]. La función Q aproxima directamente a la función de valor óptima q^*, independientemente de la política que se está siguiendo. Lo cual simplificó drásticamente su análisis y permitió las primeras pruebas de convergencia [4].

El algoritmo Q-learning inicia desde una función Q arbitraria, Q_0, y se actualiza sin requerir

un modelo, usando en su lugar la transición de estados observada y la recompensa, es decir, depende de $(x_k, u_k, x_{k+1}, r_{k+1})$ [68, 69]. Despúes de cada transición, la función Q se actualiza usando la siguiente regla:

$$Q_{k+1}(x_k, u_k) = Q_k(x_k, u_k) + \alpha_k \left[r_{k+1} + \gamma \min_u Q_k(x_{k+1}, u) - Q_k(x_k, u_k) \right] \tag{3.5}$$

donde $\alpha_k \in (0, 1]$, es la constante de aprendizaje. Dentro de los corchetes se muestra el error de diferencia temporal, δ_k, del algoritmo Q-learning. Es fácil observar que la política aún tiene efecto al determinar los pares estado-acción que se visitan y actualizan. Sin embargo, todo lo que se requiere para una correcta convergencia es que todos los pares se visiten infinitamente [4]. Conforme el número de transiciones k tiende a infitino, el algoritmo converge asintóticamente a q^*, si los espacio de estados y de acciones son discretos y finitos, y bajo las siguientes condiciones [69, 70]:

1. La suma $\sum_{k=0}^{\infty} a_k^2$ debe producir un valor finito, mientras que la suma $\sum_{k=0}^{\infty} a_k$ debe producir un valor infinito.

2. Todos los pares estado-acción son visitados infinitamente.

La primera condición es fácil de satisfacer, por ejemplo al seleccionar [12]:

$$\alpha_k = \frac{1}{(k+1)^{\omega}} \tag{3.6}$$

con $\omega \in (1/2, 1]$. De donde se puede distinguir a un factor de aprendizaje lineal, con $\omega = 1$, y un factor de aprendizaje polinomial, con $\omega \in (1/2, 1)$. De acuerdo con [10], el factor de aprendizaje requiere ser sintonizado en la práctica, debido a que influye en el número de transiciones requeridas por el algoritmo para obtener una buena solución.

Por otro lado, la segunda condición se puede satisfacer, entre otras formas, si el controlador tiene probabilidad no cero de elegir cualquier acción en cada estado, lo cual se conoce como exploración [4]. Además, el controlador debe explotar su conocimiento actual para obtener un mejor desempeño, por ejemplo, al elegir acciones codiciosas en la actual función de valor. Esto representa un problema de compensación exploración-explotación típico en RL [10].

Un método clásico para equilibrar la exploración y la explotación en Q-learning es aplicando la exploración ε-codiciosa (ε-greedy) que elige una acción como [4]:

$$u_k = \begin{cases} u \in \arg\min_{\bar{u}} Q_k(x_k, \bar{u}) & \text{Con probabilidad } 1 - \varepsilon_k \\ \text{Acción uniformemente aleatoria en } U & \text{Con probabilidad } \varepsilon_k \end{cases} \tag{3.7}$$

En donde $\varepsilon_k \in (0, 1)$ es la probabilidad de exploración en el paso k. Usualmente, la exploración se desvanece en el tiempo, tal que la política utilizada se convierte asintóticamente en codiciosa y por lo tanto en óptima. Esto se logra haciendo que ε_k tienda a 0 conforme k crece.

Es posible definir a la probabilidad de exploración de forma similar que al factor de aprendizaje, para satisfacer la segunda condición de convergencia [4]. En el Algoritmo 2 se presenta el algoritmo Q-learning con la exploración ε-greedy.

Algoritmo 2 Q-Learning con exploración ε-greedy

Entrada: Factor de descuento γ, probabilidad de exploración $\{\varepsilon_k\}_{k=0}^{\infty}$, factor de aprendizaje $\{\alpha_k\}_{k=0}^{\infty}$

1: Inicializar la función Q, por ejemplo, $Q_0 \leftarrow 0$
2: Medir el estado incial x_0
3: **para** cada paso de tiempo $k = 0, 1, 2, \ldots$ **hacer**
4: Elegir una acción, u_k, usando la política ε-codiciosa (3.7)
5: Aplicar u_k, medir el siguiente estado x_{k+1} y la recompensa r_{k+1}
6: $Q_{k+1}(x_k, u_k) \leftarrow Q_k(x_k, u_k) + \alpha_k [r_{k+1} + \gamma \min_u Q_k(x_{k+1}, u) - Q_k(x_k, u_k)]$
7: **fin para**

Cabe señalar que se considera una configuración idealizada, de tiempo infinito y no se especifíca ninguna salida de forma explícita. En su lugar, el resultado del algoritmo es la mejora del desempeño del controlador alcanzado a través de iteractuar con el proceso. Sin emabrgo, en la práctica, el algoritmo se dentendrá después de un número finito de pasos. Cuando se detiene el algoritmo Q-learning, la función de valor resultante y la correspondiente política ambiciosa se pueden interpretar como salidas y reutilizarlas [10].

3.4 Aproximación del aprendizaje reforzado

El aprendizaje reforzado es fruto de décadas de esfuerzo dentro de la inteligencia artificial y el aprendizaje automático hacia una mayor integración con la estadística, la optimización y otras áreas; entre ellas la teoría del control, en donde, por ejemplo, algunos métodos de RL abordan el clásico problema de "la maldición de la dimensionalidad", mediante aproximadores paramétricos [10].

Los algoritmos clásicos de DP y RL requieren representaciones exactas de la función de valor y las políticas. Sin embargo, una representación exacta sólo se alcanza al guardar las estimaciones del retorno para cada par estado-acción [10]. Muchos problemas en los que se busca implementar el RL tienen espacios de estados y acciones grandes o continuos. Por lo que, no es posible hallar una política óptima o la función de valor óptima, aún en el límite ínfinito de tiempo y de los datos. Por lo tanto, el objetivo cambia a encontrar una solución aproximada en su lugar, usando recursos computacionales limitados [4].

El problema con los espacios grandes y continuos no es sólo la memoria necesaria para las soluciones tabulares, sino el tiempo e información necesaria para llenarlas apropiadamente. En muchos de los casos, los estados visitados dificilmente serán medidos nuevamente. Por lo que, para tomar decisiones sensibles en dichos estados es necesario generalizar a partir de encuentros previos con diferentes estados, que en cierto modo sean similares [4].

Por lo que, para resolver este problema de generalización es necesario combinar al aprendizaje reforzado con métodos de aproximación de funciones, los cuales toman ejemplos de una función deseada e intentan construir una aproximación de la función completa [10]. La aproximación de funciones va de la mano con el aprendizaje supervisado, el tema principal en el estudio de aprendizaje automático, redes neuronales artificiales, reconocimiento de patrones y ajuste estadístico de curvas [4].

Los aproximadores se dividen en dos categrorías principales: paramétricos y no paramétricos. Los aproximadores paramétricos son mapeos desde el espacio de estados al espacio de funciones que pretenden representar [4]. El número de parámetros y la forma del mapeo se asignan desde el inicio, mientras que los parámetros se ajustan utilizando la información sobre la función objetivo. Por otro lado, la estructura de los aproximadores no paramétricos se deriva de la información. Contrario a su nombre, los aproximadores no paramétricos comunmente tienen parámetros pero el número de parámetros y sus valores se determinan de la información [10].

3.4.1 Aproximación paramétrica

Considere un aproximador de la función de valor Q, parametrizado por un vector n-dimensional, θ. El aproximador se denota por el mapeo de aproximación $F : \mathbb{R}^n \rightarrow \mathcal{Q}$, donde $\mathbb{R}^n$ es el espacio de parámetros y $\mathcal{Q}$ es el espacio de funciones de valor Q [10]. Cada vector de parámetros, θ,

provee una representación compacta de una función Q aproximada:

$$\widehat{Q}(x,u) = [F(\theta)](x,u) \tag{3.8}$$

donde $[F(\theta)](x,u)$ denota a la función de valor, $F(\theta)$, evaluada en el par estado-acción (x,u). Cabe señalar que cuando se trabaja con espacios de estados discretos, n es usualmente mucho menor que $|X| \cdot |U|$, donde la notación $|\cdot|$ hace referencia a la cardinalidad de un conjunto [10]. Sin embargo, el conjunto de funciones de valor Q que se pueden representar mediante F es un subconjunto de $\mathcal{Q}$, sólo se puede representar una función Q arbitraria con algún error de aproximación, el cual debe tomarse en cuenta [12].

En general, el mapeo F puede ser no lineal en los parámetros. Un ejemplo típico de un aproximador paramétrico no lineal es una red neuronal tipo *feed-forward* [12]. Sin embargo, un aproximador paramétrico lineal de la función de valor Q utiliza n funciones base (BFs, por sus siglas en inglés) $\phi_1, \ldots, \phi_n : X \times U \to \mathbb{R}$ y un vector de parámetros, θ. Las funciones de valor Q aproximadas se calculan con:

$$[F(\theta)](x,u) = \sum_{k=1}^{n} \phi_k(x,u)\,\theta_k = \phi^T(x,u)\,\theta \tag{3.9}$$

donde $\phi(x,u) = [\phi_1, \ldots, \phi_n]^T$ es el vector de BFs, a las cuales también se les conoce como carcaterísticas [12].

3.4.2 Iteración de la función Q

Existen métodos de RL para aproximar la función de valor que operan en línea o fuera de línea o por lotes. De los algoritmos en línea para la aproximación de funciones de valor, las versiones aproximadas del algoritmo Q-learning son ampliamente estudiadas y utilizadas desde inicios de la década de 1990 [10].

De acuerdo con [10], una forma simple de implementar la aproximación en el Q-learning es mediante el uso del descenso de gradiente. Por lo que es necesario que el mapeo F sea diferenciable en los parámetros [4].

Para simplificar la notación se denotará a la aproximación de la función Q en el paso k al retomar la expresión (3.8) con $\widehat{Q}(x,u)$, dejando la dependencia del vector de parámetros implícita. Para derivar el Q-learning basado en gradiente descendente, se asume que despúes de tomar una

acción u_k en el estado x_k, el algoritmo es provisto con la verdadera función de valor óptima del par estado-acción, $q^*(x_k, u_k)$, además del siguiente estado x_{k+1} y la recompensa r_{k+1} [10]. Bajo estas circunstancias, el algoritmo podría minimzar el error cuadrático entre el valor óptimo y el el valor del par estado acción actual:

$$\begin{aligned}\theta_{k+1} &= \theta_k - \frac{1}{2}\alpha_k \frac{\partial}{\partial \theta_k}\left[q^*(x_k, u_k) - \widehat{Q}(x, u)\right]^2 \\ &= \theta_k + \alpha_k \left[q^*(x_k, u_k) - \widehat{Q}(x_k, u_k)\right] \frac{\partial}{\partial \theta_k}\widehat{Q}(x_k, u_k)\end{aligned} \tag{3.10}$$

Evidentemente $q^*(x_k, u_k)$ no está disponible, pero se puede reemplazar por una estimación derivada del mapeo de la iteración de Q, tal como en Q-learning:

$$q^*(x_k, u_k) = r_{k+1} + \gamma \min_u \widehat{Q}(x_{k+1}, u) \tag{3.11}$$

Esta sustitución lleva a la actualización de la aproximación de la función Q:

$$\theta_{k+1} = \theta_k + \alpha_k \left[r_{k+1} + \gamma \min_u \widehat{Q}(x_{k+1}, u) - \widehat{Q}(x_k, u_k)\right] \frac{\partial}{\partial \theta_k}\widehat{Q}(x_k, u_k) \tag{3.12}$$

Cabe señalar que al igual que en el algoritmo clásico de Q-learning, la aproximación requiere de exploración. En el Algoritmo 3 se presenta al algoritmo Q-learning basado en el descenso del gradiente con la exploración ε-greedy.

Algoritmo 3 Q-Learning con aproximación lineal y exploración $\varepsilon - greedy$

Entrada: Factor de descuento γ, Funciones base $\phi_1, \ldots, \phi_n : X \times U \to \mathbb{R}$, probabilidad de exploración $\{\varepsilon_k\}_{k=0}^{\infty}$, factor de aprendizaje $\{\alpha_k\}_{k=0}^{\infty}$

1: Inicializar el vector de parámetros θ, por ejemplo, $\theta_0 \leftarrow 0$
2: Medir el estado incial x_0
3: **para** cada paso de tiempo $k = 0, 1, 2, \ldots$ **hacer**
4: Elegir una acción, u_k, usando la política ε-codiciosa (3.7)
5: Aplicar u_k, medir el siguiente estado x_{k+1} y la recompensa r_{k+1}
6: $\theta_{k+1} \leftarrow \theta_k + \alpha_k \left[r_{k+1} + \gamma \min_{u'}\left(\phi^T(x_{k+1}, u')\,\theta_k\right) - \phi^T(x_k, u_k)\,\theta_k\right] \phi(x_k, u_k)$
7: **fin para**

En la literatura se han utilizado diferentes aproximadores al trabajar con el algoritmo Q-learning [10], como aproximadores lineales, con codificación de mosaicos y funciones de base radial (RBF), reglas difusas, las cuales también pueden ser lineales en los parámetros. Y aproximadores no lineales basados en redes neuronales.

3.4.3 Aproximación por redes neuronales

Una arquitectura no lineal muy común es la del perceptron multicapa o red neuronal tipo feedforward con una capa oculta. Bajo esta arquitectura, el estado $x_k \in \mathbb{R}^n$ es transformado a través de una capa lineal, por la matriz de pesos $V \in \mathbb{R}^{n\times m}$:

$$V^T x_k = \sum_{i=1}^{n} v_{ji} x_i,\ j = 1, \ldots, m \tag{3.13}$$

Cada uno de estos escalares se convierte en la entrada a una función $\sigma(\cdot)$, llamada función sigmoide, la cuál es diferenciable, monotonamente creciente y con la propiedad [12]:

$$-\infty < \lim_{\xi\to-\infty} \sigma(\xi) < \lim_{\xi\to\infty} \sigma(\xi) < \infty \tag{3.14}$$

Algunas elecciones comunes son la función tangente hiperbólica y la función logística [71]:

$$\sigma(\xi) = \tanh(\xi) = \frac{e^{\xi} - e^{-\xi}}{e^{\xi} + e^{-\xi}},\quad \sigma(\xi) = \frac{1}{e^{\xi} + e^{-\xi}} \tag{3.15}$$

También es bastante común el uso de unidades lineales rectificadas o funciones ReLU [71]:

$$\sigma(\xi) = \text{máx}\{0, \xi\} = \ln\left(1 + e^{\xi}\right) \tag{3.16}$$

Como salida de las funciones de activación, se tiene a los escalares:

$$\sigma\left(V^T x_k\right) = \sigma\left(\sum_{i=1}^{n} v_{ji} x_i\right),\ j = 1, \ldots, m \tag{3.17}$$

Estos escalares son combinados lienalmente usando a la matriz $W \in \mathbb{R}^{m\times o}$ para producir la salida final:

$$\widehat{Q}(x_k, u_k) = W^T \sigma\left(V^T x_k\right) = \sum_{k=1}^{m} w_{kj} \sigma\left(\sum_{i=1}^{n} v_{ji} x_i\right),\ k = 1, \ldots, o \tag{3.18}$$

Se ha identificado que es una práctica común proveer la constante 1 como una entrada adicional a la capa lineal, esto provee un compensación constante ajustable para cada una de las entradas de las unidades no lineales, y expande el rango de mapeos que esta arquitectura puede aproximar de forma efectiva [10].

Además, existen generalizaciones de la arquitectura del perceptron con una capa oculta que involucran la alternancia de múltiples capas de funciones lineales y sigmoides. Los perceptrones multicapa puede ser representados de forma compacta al introducir ciertos mapeos para describir

las capas lineales y no lineales [71]. En particular, $L_1, \ldots, L_{m+1}$ denotan a las matrices de pesos de las capas lineales. Así mismo, $\Sigma_1, \ldots, \Sigma_m$ denotan a los mapeos que representan a las capas no lineales. Con lo cual, la salida del perceptron multicapa es:

$$F\left(L_1, \ldots, L_{m+1}, x\right) = L_{m+1}\Sigma_m L_m \cdots \Sigma_1 L_1 x_k \tag{3.19}$$

Entrenamiento de una red neuronal

Los problemas de entrenamiento de una red neuronal son problemas de optimización donde se busca un conjunto de parámetros o pesos de una arquitectura de aproximación que provean el mejor ajuste entre el mapeo realizado y un conjunto de pares de datos entrada-salida. Típicamente, estos problemas son de la forma de mínimos cuadrados:

$$\min_{W,V} \frac{1}{2} \sum_{i=1}^{m} e_j^2 \tag{3.20}$$

donde e_j es el vector de error:

$$e = q^*\left(x_k, u_k\right) - \widehat{Q}\left(x_k, u_k\right) \tag{3.21}$$

Por lo que es posible aplicar el método de gradiente descendiente estcoástico (SGD). Los métodos de SGD ajustan ligeramente los pesos de la red neuronal después de cada paso k, en la dirección que reduzca el error [4]. La actualización de pesos mediante SGD está dada por:

$$\begin{aligned} W_{k+1} &= W_k - \frac{1}{2}\alpha \nabla \left[q^*\left(x_k, u_k\right) - \widehat{Q}\left(x_k, u_k\right)\right]^2 \\ &= W_k + \alpha \left[q^*\left(x_k, u_k\right) - \widehat{Q}\left(x_k, u_k\right)\right] \nabla \widehat{Q}\left(x_k, u_k\right) \end{aligned} \tag{3.22}$$

donde α es una constante positiva y $\nabla \widehat{Q}\left(x_k, u_k\right)$ corresponde al gradiente de la red neuronal con respecto a las matrices de pesos. El gradiente $\nabla \widehat{Q}\left(x_k, u_k\right)$ puede ser calculado eficazmente usando un procedimiento especial conocido como backpropagation.

La naturaleza especial de un red neuronal tipo MLP tiene una importante consecuencia computacional en el gradiente del error cuadrático con respecto a los pesos:

$$E\left(L_1, \ldots, L_{m+1}\right) = \frac{1}{2}\left(\widehat{Q}\left(x_k, u_k\right) - q^*\left(x_k, u_k\right)\right)^2 \tag{3.23}$$

puede ser eficientemente calculado usando un procedimiento especial conocido como backpropagation [12]. En particular, la derivada parcial de la función de costo $E(L_1, \ldots, L_{m+1})$ con respecto al ij-ésimo componente de la matriz L_k, $L_k(i,j)$, está dado por:

$$\frac{\partial E(L_1, \ldots, L_{m+1})}{\partial L_k(i,j)} = -e^T L_{m+1} \bar{\Sigma}_m L_m \cdots L_{k+1} \bar{\Sigma}_k I_{ij} \Sigma_{k-1} \cdots \Sigma_1 L_1 x \tag{3.24}$$

donde Σ_n, $n = 1, \ldots, m$, es una matriz diagonal cuyas entradas son las derivadas de las funciones de activación de la n-ésima capa oculta, I_{ij} es la matriz obtenida al establecer todos los valores de L_k iguales a 0, excepto al ij-ésimo componente, el cual se establece como 1.

Existe un resultado importante relacionado con la capacidad de aproximación de un perceptron multicapa. Es posible mostrar que esta red, con una buena selección de pesos, puede aproximar arbitrariamente cerca cualquier función $J : S \to \mathbb{R}$ que sea continua sobre un conjunto cerrado y acotado S, con una cantidad suficientemente grande de funciones de activación en la capa oculta [71]. Para esto, una sola capa oculta es suficiente. En la práctica, el número de capas ocultas es usualmente 1 o 2, y casi nunca mayor a 3 [12].

3.4.4 Q-learning con redes neuronales

Al aproximar a la función Q mediante redes neuronales es posible implementar el algoritmo de Q-Learning. Para esto se propone implementar una red neuronal que tome como entradas a los estados y cuyas salidas correspondan al valor de la función Q de cada una de las acciones.

Sin embargo, en las arquitecturas no lineales el aprendizaje tiende a ser inestable; por lo que, en la literatura se han desarrollado alternativas para mejorar el desempeño de las redes neuronales en la aproximación de funciones de valor en RL.

Una de estas alternativas es el uso de una segunda red conocida como red objetivo [72], la cual mejora la estabilidad del aprendizaje. Esta segunda red genera el valor objetivo en el error TD, δ_k. La principal característica de la red objetivo es que sólo actualiza los valores de sus matrices de pesos cada N pasos de tiempo discreto. Al actualizar sus pesos sinápticos toma los valores actuales de la red neruonal principal, es decir, $W^- = W$. Con esta modificación, la actualización del algoritmo Q-Learning está dada por:

$$\theta_{k+1} = \theta_k + \alpha \left[r_{k+1} + \gamma \min_u Q\left(x_k, u_k; \theta^-\right) - Q\left(x_k, u_k; \theta_k\right) \right] \nabla_\theta Q\left(x_k, u_k; \theta_k\right) \tag{3.25}$$

en donde $Q\left(x_k, u_k; \theta^-\right)$ corresponde a la red neuronal objetivo y $Q\left(x_k, u_k; \theta\right)$ a la red neuronal principal.

La segunda alternativa se conoce como repetición de la experiencia. Lin [73, 74], propuso almacenar las muestras $(x_k, u_k, r_{k+1}, x_{k+1})$ en cada paso de tiempo. De tal forma que duarnte el entrenamiento el agente pueda tomar muestras de forma aleatoria del conjunto de datos almacenados, para actualizar a la función Q. Tal que la repetición de experiencias suaviza los cambios en la distribución de los datos, lo cual estabiliza el aprendizaje [75].

En el Algoritmo 4, se presenta el uso de redes neuronales para aproximar la función Q y utilizar el algoritmo Q-learning basado en el gradiente descendiente estocástico, con red neuronal objetivo y repetición de experiencia para mejorar la estabilidad del aprendizaje:

Algoritmo 4 Q-Learning con aproximación por redes neuronales

Entrada: Factor de descuento γ, probabilidad de exploración $\{\varepsilon_k\}_{k=0}^{\infty}$, constante de aprendizaje $\{\alpha_k\}_{k=0}^{\infty}$

1: Inicializar los pesos sinápticos de la red neuronal con valores aleatorios en el intervalo $[0, 1]$.
2: Medir el estado incial x_0
3: **para** episodio $e = 1, \ldots, M$ **hacer**
4: **para** cada paso de tiempo $k = 0, 1, \ldots, T$ **hacer**
5: Elegir una acción, u_k, usando la política ε-codiciosa (3.7)
6: Aplicar u_k, medir el siguiente estado x_{k+1} y la recompensa r_{k+1}
7: Almacenar las muestras $(x_k, u_k, r_{k+1}, x_{k+1})$ en el conjunto $\mathcal{D}$
8: Tomar un subconjutno $(x_j, u_j, r_{j+1}, x_{j+1})$ aleatorio de $\mathcal{D}$
9: Calcular la aproximación $Q\left(x_j, u_j; \theta\right)$ y el objetivo $r_{j+1} + \gamma \min_u Q\left(x_j, u_j; \theta^-\right)$
10: Aplicar el método de gradiente descendiente estocástico $\left(q^*\left(x_j, u_j\right) - \widehat{Q}\left(x_j, u_j\right)\right)^2$ con respecto a θ.
11: Cada N pasos actualizar $\theta^- = \theta_k$
12: **fin para**
13: **fin para**

3.5 Simulaciones para el aprendizaje reforzado

La implementación de los controladores basados en aprendizaje reforzado se realizó mediante modelos de Simulink, como el que se muestra en la Figura 3-2. Se propuso utilizar el tiempo de muestreo de 2 *ms* que indica el fabricante Quanser [76].

Se incluye un subsistema que contiene a la *política* ε-greedy, que depende de la probabilidad de

exploración ε, el espacio de acciones, la tabla Q_k o las matrices de pesos V y W, así como el estado actual x_k. El bloque identificado como *entorno*, consta del modelo no lineal del péndulo Furuta y el cálculo del índice de los vectores de estado x_k y x_{k+1}, para la tabla Q.

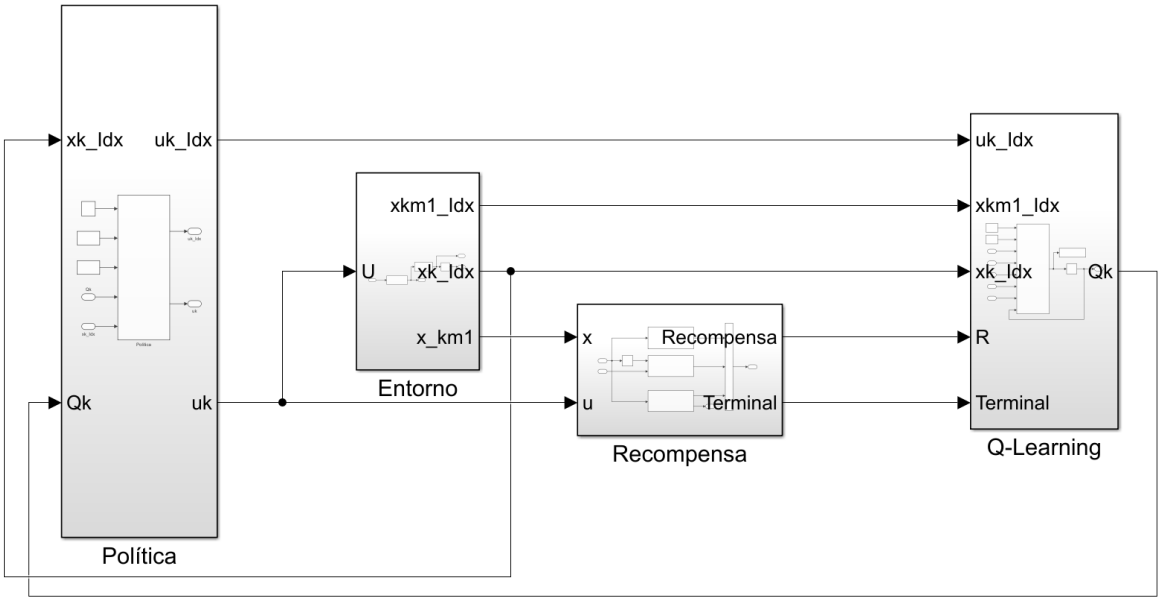

Figura 3-2: Modelo de Simulink para el control del péndulo Furuta por RL

Así mismo, se incluye un subsistema que realiza el cálculo de la *recompensa* y envía una señal para detener la simulación cuando la respuesta del sistema ha excedido las condiciones establecidas en cada controlador.

Debido a que el objetivo de control es estabilizar al péndulo Furuta sobre el equilibrio inestable, $x = [0, 0, 0, 0]^T$, se propone a la función de recompensa como una función de costo cuadrática:

$$r\left(x_k, u_k\right) = x_k^T Q x_k + R u_k^2 \tag{3.26}$$

donde $Q \in \mathbb{R}^{4\times4}$ y $R \in \mathbb{R}$, con $Q = \text{diag}\left(50, 0.5, 50, 0.5\right)$, para penalizar a las posiciones más lejanas al equilibrio inestable y reducir la influencia de las velocidades en la recompensa. Con $R = 0.1$ se busca evitar que el controlador priorice reducir la señal de control antes de llevar al equilibrio al péndulo Furuta.

Finalmente, en el subsistema identificado como *Q-Learning* se implementa el algoritmo de aprendizaje, ya sea por solución tabular o mediante la aproximación por redes neuronales.

3.5.1 Solución por tabla

Para realizar el control del péndulo Furuta por aprendizaje reforzado mediante la solución tabular del algoritmo Q-learning, es necesario discretizar el espacio de estados continuo. El vector de estados del péndulo invertido rotativo, x, está compuesto por la posición y velocidad angular del brazo y el péndulo, $x = \left[\theta, \dot{\theta}, \alpha, \dot{\alpha}\right]^T$. La señal de control para el motor del brazo se establecio como $u \in \{-V_m, 0, V_m\}\ V$.

Para realizar la discretización se tomó en cuenta un rango de movimiento para el brazo de $\pm 45°$ y de $\pm 25°$ para el péndulo. Por lo tanto, se establecieron los rangos $\theta \in [-\pi/4, \pi/4]$ *rad,* $\alpha \in [-0.4363, 0.4363]$ *rad* y $\dot{\theta}, \dot{\alpha} \in [-\pi, \pi]$ *rad/s.*

La discretización del espacio de estados continuo se realizó para 19 divisiones en las posiciones y 11 para las velocidades. La posición del brazo tuvo una resolución de 0.0873 *rad* y el péndulo de 0.0485 *rad.* Las velocidades tuvieron una resolución de 0.6283 *rad/s.* Se realizaron todas las posibles combinaciones con los valores discretos de posición y velocidad del brazo y el péndulo, tal que, se generó un espacio de estados discreto con $19^2 \times 11^2 = 43,681$ elementos. Finalmente, la tabla Q del algoritmo Q-Learning se determinó como $Q \in \mathbb{R}^{43,681 \times 3}$.

Además, se estableció una señal de bonificación que premia a las condiciones cercanas al equilibrio y penaliza cuando superan los rangos establecidos en la discretización. La señal de bonificación tiene la forma:

$$b(x_k, u_k) = \begin{cases} 100 & \text{si } |\theta| > 45° \text{ y } |\alpha| > 25° \\ -10 & \text{si } |\theta| \leq 5° \text{ y } |\alpha| \leq 3° \\ -50 & \text{si } \theta = 0° \text{ y } \alpha = 0° \\ 0 & \text{de cualquier otra forma} \end{cases} \tag{3.27}$$

Por lo tanto, la señal de recompensa real depende de la función cuadrática (3.26) y la bonificación (3.27), tal que:

$$r_{k+1} = r\left(x_k, u_k\right) + b(x_k, u_k) \tag{3.28}$$

La señal para detener la simulación estuvo sujeta a los rangos establecidos en la discretización. La implementación del Algoritmo 2 se realizó con la función de Matlab que se muestra en la Figura 3-3. Dicha función depende de los índices de u_k, x_k y x_{k+1} con respecto a los espacios discretos de acciones y estados para actualizar la tabla Q de forma correcta.

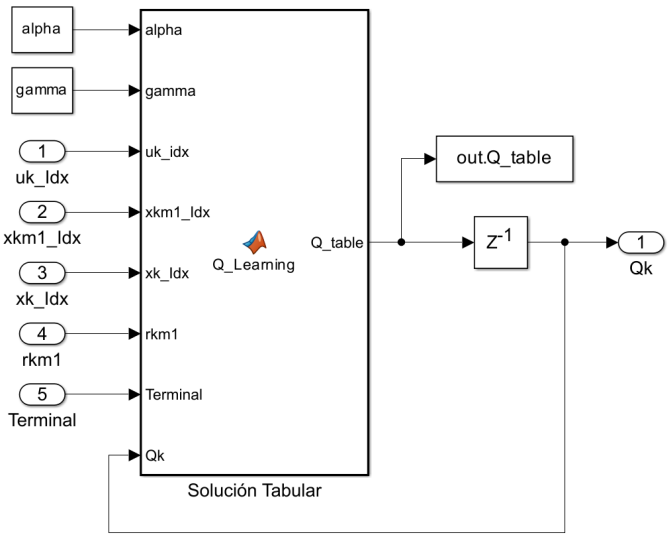

Figura 3-3: Algoritmo Q-Learning basado en tabla

Así mismo, se tomó en cuenta a la recompensa r_{k+1} y a la condición para detener la simulación para realizar la actualización de la tabla Q, debido a que:

$$\delta_k = \begin{cases} r_{k+1} + \gamma \min_u Q_k\left(x_{k+1}, u\right) & \text{si } \mathit{Terminal} = 0 \\ r_{k+1} & \text{si } \mathit{Terminal} = 1 \end{cases} \tag{3.29}$$

Mediante el uso de bloques de retardos se elimina el problema de lazos algebráicos en la simulación al tomar valores previos en algunas variables, como la tabla Q_k. Por otro lado, se alamcenó a la tabla Q_{k+1} a través de un bloque *To Workspace*, para posteriormente utilizarla durante la validación del controlador.

3.5.2 Aproximación por redes neuronales

Para el Algoritmo 4 se utilizó el espacio de estados continuos, de tal forma que la red neuronal tomó como entrada al vector $x = \left[\theta, \dot{\theta}, \alpha, \dot{\alpha}\right]^T$. Se conservó a la recompensa (3.28), al espacio de acciones discreto $u \in \{-V_m, 0, V_m\}\ V$ y a los rangos $\theta \in [-\pi/4, \pi/4]$ *rad,* $\alpha \in [-0.4363, 0.4363]$ *rad* y $\dot{\theta}, \dot{\alpha} \in [-\pi, \pi]$ *rad/s* como condiciones para detener la simulación. El espacio de acciones discreto estuvo dado como $u \in \{-V_m, 0, V_m\}\ V$.

Se propuso una red neuronal tipo perceptron multicapa con 400 nodos en la capa oculta. Por lo tanto, la red neuronal tuvo una estructura 4-400-3 con $V \in \mathbb{R}^{4\times400}$ y $W \in \mathbb{R}^{400\times3}$. Como

funciones de activación se utilizaron funciones ReLu en la capa oculta y una función lineal en la capa de salida. En la Figura 3-4 se muestra un esquema representativo de la red neuronal de una capa oculta que se utilizó en la aproximación.

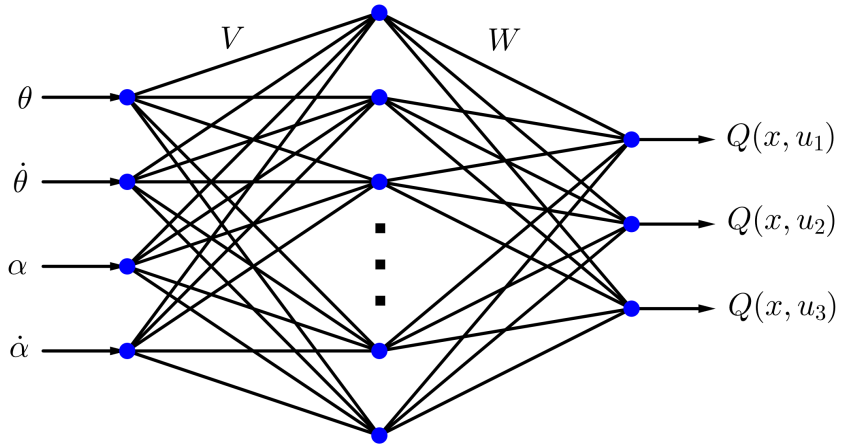

Figura 3-4: Aproximación de la función Q por redes neuronales

En la Figura 3-5 se muestra la función de Matlab utilizada para implementar el Algoritmo 4, con la aproximación por redes neuronales. La función depende del espacio de acciones, de los estados x_k, x_{k+1} y el conjunto de datos para la repetición de experiencia $\mathcal{D}$. Además, necesita de las matrices de pesos sinápticos de la red neuronal principal y de la red objetivo.

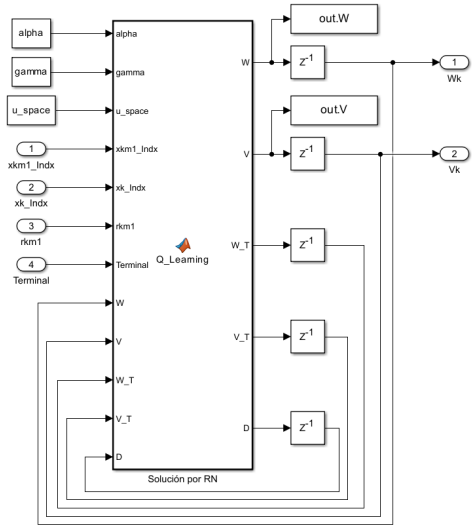

Figura 3-5: Algoritmo Q-Learning basado en redes neuronales

La función *Q_ Learning* también depende de la recompensa r_{k+1} y de la señal para detener la simulación debido a que el error de diferencia temporal está dado como:

$$\delta_k = \begin{cases} r_{k+1} + \gamma \min_u Q_k\left(x_{k+1}, u; \theta^-\right) & \text{si } \textit{Terminal} = 0 \\ r_{k+1} & \text{si } \textit{Terminal} = 1 \end{cases} \tag{3.30}$$

Para este algoritmo se almacenó mediante un bloque *To Workspace* a las matrices de pesos sinápticos W y V, para utilizarlas durante la validación del controlador.

3.6 Control por aprendizaje reforzado

A continuación se presentan los resultados obtenidos al implementar controladores basados en aprendizaje reforzado para equilibrar al péndulo Furuta en el equilibrio inestable. Se realizaron simulaciones para comparar a la solución Tabular del algoritmo Q-Learning contra el aproximador paramétrico no lineal basado en redes neuronales.

3.6.1 Resultados

Se implementaron los controladores basados en aprendizaje reforzado para el control del péndulo Furuta sobre el equilibrio inestable, $x = [0, 0, 0, 0]^T$. Para la simulación del modelo no lineal del péndulo Furuta se utilizaron los parámetros que se muestran en la Tabla 3.1, los cuales fueron obtenidos del fabricante Quanser [76, 77].

Tabla 3.1: Parámetros del péndulo Furuta

Parámetro	**Valor**	**Parámetro**	**Valor**
m_1	0.2570 Kg	b_1	2.4×10^{-3} $N{\cdot}s/m$
m_2	0.0970 Kg	b_2	2.4×10^{-3} $N{\cdot}s/m$
L_1	0.2159 m	R_m	2.6 Ω
l_1	0.0619 m	K_m	7.68×10^{-3} $V{\cdot}s/rad$
L_2	0.20 m	K_b	7.68×10^{-3} $N{\cdot}m/A$
l_2	0.1635 m	K_g	70
J_1	9.9829×10^{-4} $Kg{\cdot}m^2$	η_m	0.69
J_2	3.2341×10^{-4} $Kg{\cdot}m^2$	η_g	0.9

Resultados con solución tabular

El entrenamiento del algoritmo Q-Learning, basado en la tabla Q, se realizó mediante 10,000 episodios, con un máximo de 2,000 pasos de tiempo discreto. Se propuso a la constante de aprendizaje $\alpha = 0.35$, debido a que es posible obtener un buen desempeño con un valor constante que no sea muy pequeño [78]. Con el factor de descuento $\gamma = 0.98$, se busca que las recompensas obtenidas en el estado estacionario tengan mayor influencia sobre cualquier condición inicial [10]. Para la probabilidad de exploración se propuso una modificación para cada episodio:

$$\varepsilon_{e+1} = \varepsilon_{\text{mín}} + \Delta\varepsilon \exp\left(-\gamma_\varepsilon e\right) \tag{3.31}$$

donde e representa al episodio actual, $\varepsilon_{\text{mín}}$ es el valor mínimo de exploración, $\Delta\varepsilon = \varepsilon_{\text{máx}} - \varepsilon_{\text{mín}}$ y γ_ε es la constante de decaimiento. Durante el entrenamiento se propuso a $\varepsilon_{\text{máx}} = 1$, $\varepsilon_{\text{mín}} = 0$ y $\gamma_\varepsilon = 1 \times 10^{-3}$.

En cada episodio se generaron condiciones iniciales aletorias para las posiciones. Con la posición del brazo en $\theta_0 \in [-30°, 30°]$ y la del péndulo en $\alpha_0 \in [-15°, 15°]$. Por lo tanto, el vector de estado inicial fue $x_0 = [\theta_0, 0, \alpha_0, 0]^T$. Se propuso al espacio de acciones como $u \in \{-5, 0, 5\}\ V$.

La validación se realizó con 20 experimentos de máximo 10 s con el vector de estado inicial $x_0 = [\theta_0, 0, \alpha_0, 0]^T$. En la Figura 3-6 se muestra la respuesta de la posición angular del brazo, θ, y el péndulo, α, durante los experimentos realizados. Aunque, la mayoría de experimentos duraron menos de 1 s, se resalta que el controlador mostró indicios de poder mantener al péndulo Furuta al rededor del equilibrio inestable, como en el experimento de mayor duración, aproximadamente 1.6 s.

Así mismo, se observó que la decisión de incluir una señal de penalización por fallar en la tarea de control influyó en el comportamiento del brazo actuado, al no superar el límite de $\pm 45°$. Es fácil observar que la posición del péndulo superó en repetidas ocasiones el rango de $\pm 25°$. Esto implica que la mayoría de los experimentos de validación se detuvieron a causa del péndulo y sólo en dos ocasiones la posición del brazo superó los límites establecidos.

Por lo tanto, bajo el entrenamiento propuesto el controlador inteligente no fue capaz controlar al sistema sobre el equilibrio inestable. Considerando la mejor respuesta obtenida, se asocia el resultado de la validación al tiempo de entrenamiento.

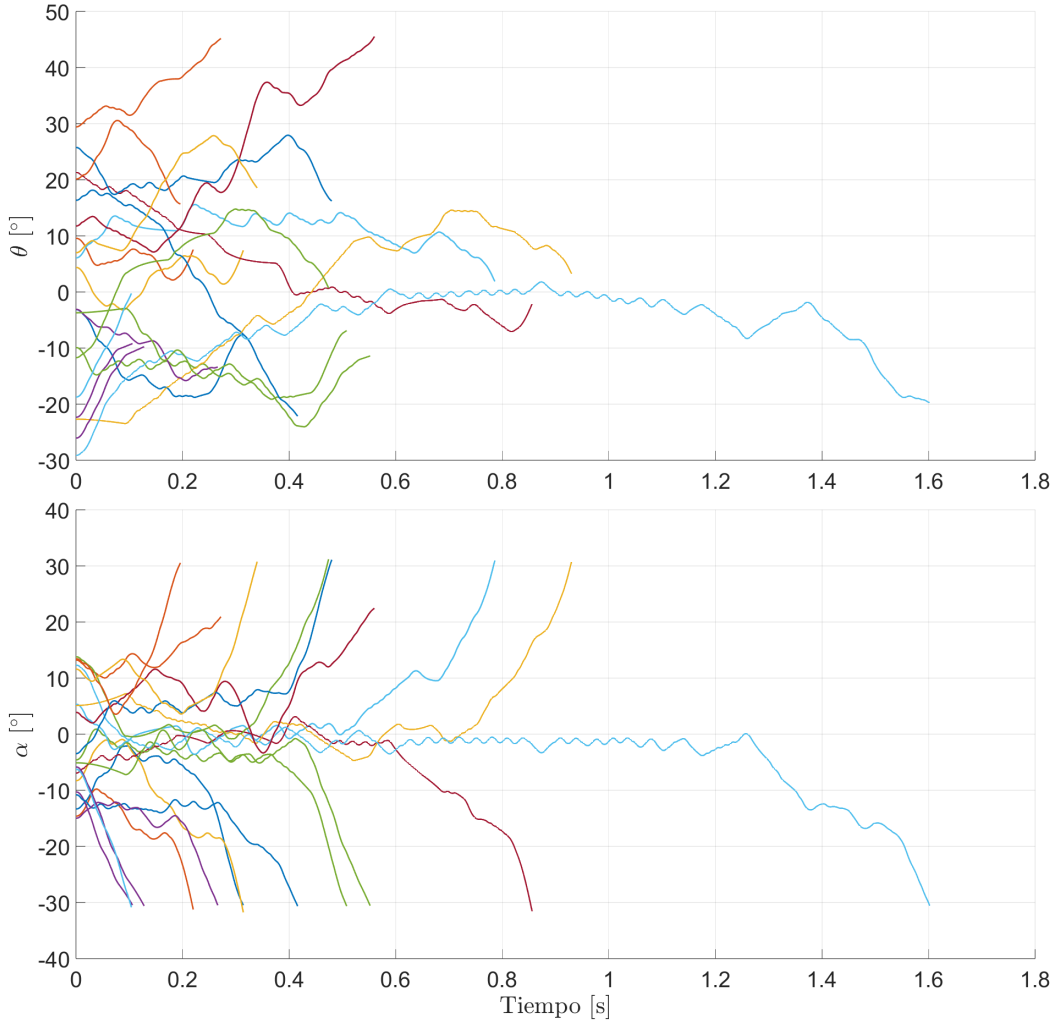

Figura 3-6: Control por Q-Learning con tabla

Resultados con la solución por redes neuronales

Para el entrenamiento del algoritmo Q-Learning, con en el aproximador no lineal basado en redes neuronales, se realizaron 500 episodios, con un máximo de $2,000$ pasos de tiempo discreto. Se usó al factor de descuento $\gamma = 0.98$ y la constante de aprendizaje $\alpha = 0.001$.

Se conservó a la expresión (3.31) para implementar una política ε-greedy. Se propuso a $\varepsilon_0 = 1$, $\varepsilon_{\text{máx}} = 1$, $\varepsilon_{\text{mín}} = 0$ y $\gamma_\varepsilon = 1 \times 10^{-3}$. Así mismo, se conservó al vector de condiciones iniciales $x_0 = [\theta_0, 0, \alpha_0, 0]^T$ con $\theta_0 \in [-30°, 30°]$ y $\alpha_0 \in [-15°, 15°]$. Se propuso almacenar 250 muestras de experiencia. Con el espacio de acciones como $u \in \{-5, 0, 5\}$ V.

La validación del entrenamiento se realizó con 20 experimentos, de máximo 10 s, con condiciones iniciales aleatorias, dadas por el vector $x_0 = [\theta_0, 0, \alpha_0, 0]^T$. En la Figura 3-7 se presenta el comportamiento del brazo y el péndulo durante los primeros 3 s debio a que se alcanzó el estado estacionario.

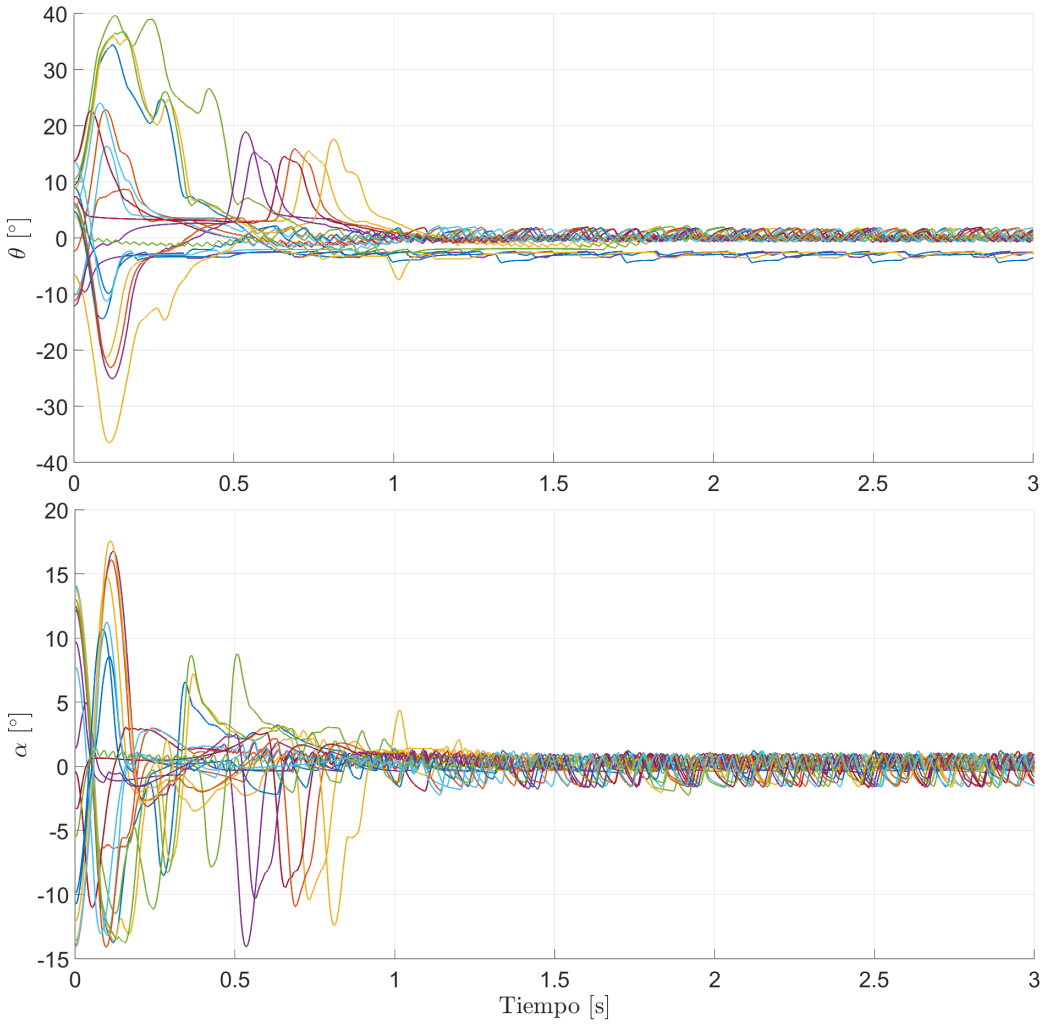

Figura 3-7: Control por Q-Learning con redes neuronales

El aproximador no lineal produjo un tiempo de asentamiento máximo cercano a 1 s. Aunque el sistema presentó oscilaciones sostenidas en el estado estacionario, se mantuvo cercano al equilibrio inestable. El brazo presentó oscilaciones sostenidas de $\pm 2°$ y el péndulo se mantuvo entre $\pm 1.5°$. Sin embargo, en algunos experimentos la posición del brazo presentó un error en estado estacionario de aproximadamente $-3.5°$.

La respuesta transitoria del péndulo se mantuvo acotada en aproximadamente $\pm 15°$ para las condiciones iniciales más alejadas del equilibrio. En el caso del brazo actuado se destaca que las restricciones impuestas en la simulación y penalización de la recompensa mantuvieron la respuesta dentro de un rango de $\pm 40°$.

En la Figura 3-8 se muestra el comportamiento de la señal de control que produjo el controlador inteligente durante los 20 experimentos durante los primeros 3 s. Resulta evidente que las oscilaciones sostenidas en el estado estacionario se deben al comportamiento de la señal u_k durante los experimentos.

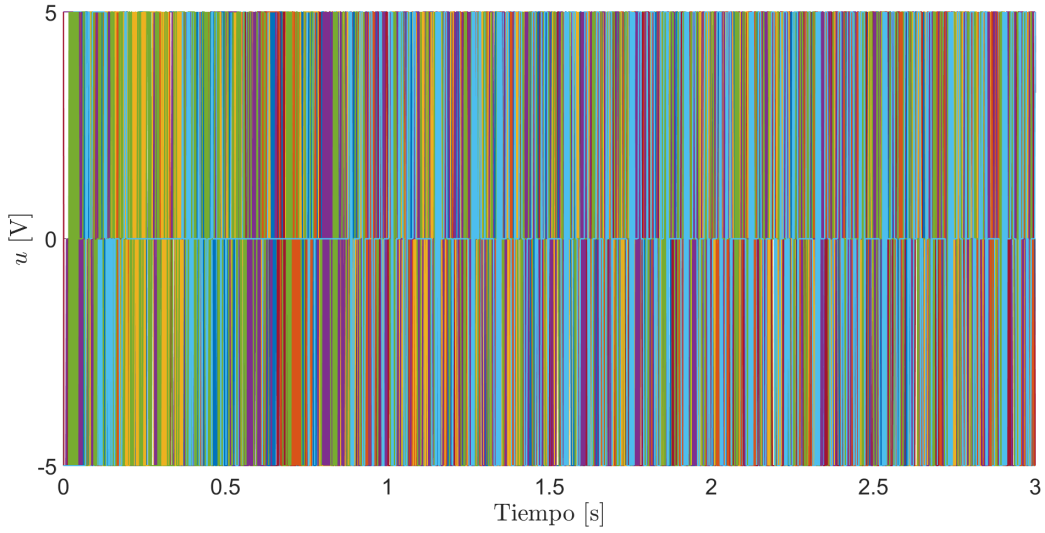

Figura 3-8: Señal de control: Control por Q-Learning con redes neuronales

Por lo tanto, el controlador basado en el aproximador no lineal demostró una mejor capacidad para controlar del péndulo Furuta al rededor del equilibrio inestable. Además, requirió de menos tiempo de entrenamiento que la solución tabular, gracias a las estrategias utilizadas.

3.7 Compensación con aprendizaje reforzado

Ahora se busca evaluar la capacidad del aprendizaje reforzado para compensar a los esquemas de control PD en serie y paralelo vistos previamente. Se realizó el entrenamiento para ambos esquemas de control PD sin considerar ningún tipo de perturbación, con el objetivo de analizar la capacidad del aprendizaje reforzado de compensar perturbaciones desconocidas.

A través de simulaciones, se realizó la comparación de la solución tabular contra la aproximación por redes neuronales al compensar esquemas de control PD. El objetivo de control es establizar al péndulo al rededor del equilibrio inestable, $x = [0, 0, 0, 0]^T$. Para esto se conservó una estructura similar a la mostrada en la Figura 3-2 con un tiempo de muestreo de 2 *ms*.

Así mismo, se retomó a la función de recompensa (3.28), la discretización realizada para la solución tabular, con $Q \in \mathbb{R}^{43,681\times3}$, y la estructura propuesta para la red neuronal, con $V \in \mathbb{R}^{4\times400}$ y $W \in \mathbb{R}^{400\times3}$.

Por otro lado, los esquemas de control PD en serie y paralelo están dados por la expresión:

$$u = -a_1\theta - a_2\dot{\theta} - a_3\alpha - a_4\dot{\alpha} - a_5\ddot{\alpha} + \pi + d \tag{3.32}$$

en donde π estará dada por la compensación no lineal calculada previamente o por la señal de control obtenida mediante aprendizaje reforzado, según corresponda. Así mismo, se propone considerar una perturbación, d, para evaluar la robustez de los controladores durante la validación. En la Figura 3-9 se muestra la implementación de la ley de control (3.32) en Simulink.

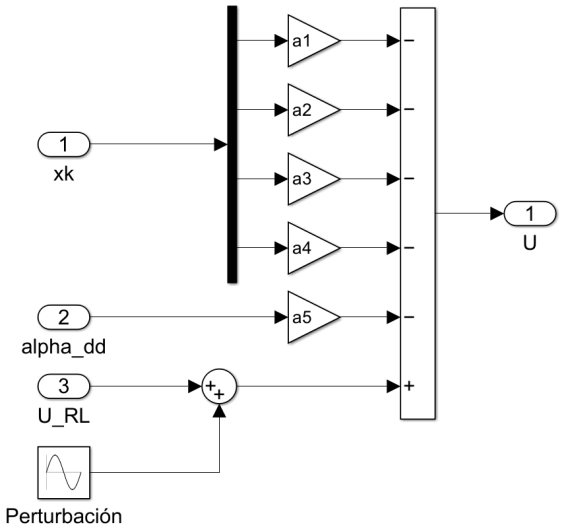

Figura 3-9: Modelo de Simulink para los esquemas de control PD

Cabe remarcar que durante el entrenamiento no se utilizó ninguna perturbación y sólo durante la validación se aplicó una señal sinusoidal de amplitud 1 y frecuencia de 0.5 *rad/s*. Se optó por esta señal debido a que afectó el desempeño de los esquemas PD con la compensación diseñada con el modelo no lineal.

3.7.1 Resultados PD en paralelo

A continuación se presentan los resultados obtenidos al entrenar controladores basados en aprendizaje reforzado para realizar la compensación de controladores PD en paralelo, para equilibrar al péndulo invertido rotativo. Para el esquema de control con compensación (3.32), donde $a_1 = k_{p_1}$, $a_2 = k_{d_1}$, $a_3 = k_{p_2}$, $a_4 = k_{d_2}$ y $a_5 = 0$, se utilizó a $k_{p_1} = 2.1754$, $k_{d_1} = 2.0342$, $k_{p_2} = 2.8483$, $k_{d_2} = 1$. Mientras que para el caso sin compensación y compensado por RL se utilizó a las ganancias $k_{p_1} = 1$, $k_{d_1} = 2$, $k_{p_2} = 22$, $k_{d_2} = 2$.

Se propuso al espacio de acciones discreto como $u \in \{-1, 0, 1\}$ V para implementar la compensación por aprendizaje reforzado mediante el algoritmo Q-Learning basado en la tabla Q y en redes neuronales. En ambos casos se retomaron las restricciones $\theta \in [-45°, 45°]$ y $\alpha \in [-25°, 25°]$ para detener las simulaciones.

Para la solución tabular se realizaron $1,500$ episodios, con un máximo de $2,000$ pasos de tiempo discreto. Con la constante de aprendizaje $\alpha = 0.3$ y el factor de descuento $\gamma = 0.98$. Para la probabilidad de exploración (3.31), se tomó a $\varepsilon_{\text{máx}} = 1$, $\varepsilon_{\text{mín}} = 0.01$ y $\gamma_\varepsilon = 5 \times 10^{-3}$.

En la aproximación por redes neuronales se realizaron 250 episodios, con un máximo de $2,000$ pasos de tiempo discreto. Con el factor de de descuento $\gamma = 0.98$ y la constante de aprendizaje $\alpha = 0.001$. Para la exploración (3.31), se conservó a $\varepsilon_{\text{máx}} = 1$, $\varepsilon_{\text{mín}} = 0.01$ y $\gamma_\varepsilon = 5 \times 10^{-3}$.

Al inicio de cada episodio se utilizó al vector de estado $x_0 = [\theta_0, 0, \alpha_0, 0]^T$, con valores aleatorios para la posición del brazo en $\theta_0 \in [-30°, 30°]$ y la del péndulo en $\alpha_0 \in [-15°, 15°]$.

Para validar y comparar a las diferentes estrategias de control se realizaron 2 experimentos con una duración máxima de 30 s y condiciones iniciales $x_0 = [\theta_0, 0, \alpha_0, 0]^T$. Para el primer experimento se propuso a $\theta_0 = 20°$ y $\alpha_0 = -1°$. En el segundo experimento se tomó a $\theta_0 = -2°$ y $\alpha_0 = 10°$. Se comparó al control PD sin compensación contra la compensación no lineal (PD+π), la compensación con la tabla Q (PD+Q) y la compensación con el aproximador basado en redes neuronales (PD+NN).

En la Figura 3-10 se presentan los primeros 15 s de la respuesta del brazo con todos los esquema PD en paralelo, debido a que se alcanzó el estado estacionario. Es fácil distinguir que el controlador PD mantuvo oscilando al brazo actuado entre $\pm 50°$, demostrando que el controlador tradicional es incapaz de compensar la perturbación elegida.

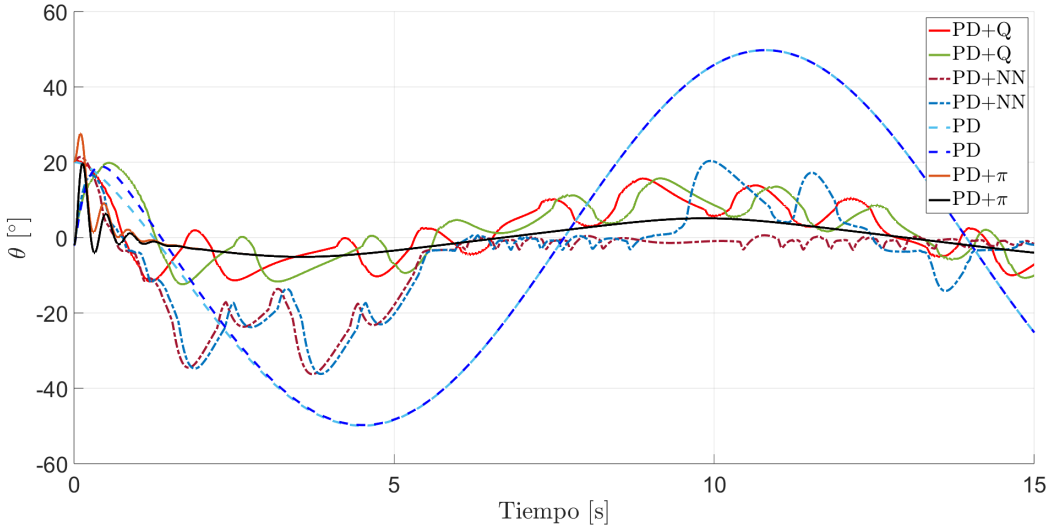

Figura 3-10: Brazo: Controladores PD en paralelo

Por otro lado, destaca el comportamiento de la compensación basada en el modelo no lineal (PD+π), que presenta un tiempo de asentamiento de aproximadamente 1 s. Durante el estado estacionario mantuvo al brazo en un rango de $\pm 5°$. Demostrando que el esquema PD+π es suficientemente robusto como para atenuar el efecto de la perturbación propuesta sobre la respuesta del brazo actuado.

En el caso de la compensación por aprendizaje reforzado (PD+Q), utilizando la tabla Q, se observó que el controlador mostró un desempeño similar al obtenido con el controlador PD+π, con un tiempo de asentamiento de aproximadamente 1.5 s. Mantuvo acotada a la respuesta del brazo en una banda de $\pm 15°$. Además, se relacionó a las oscilaciones presentadas en el estado estacionario con el espacio de acciones discreto que se utilizó en el aprendizaje reforzado.

Mientras que el controlador compensado con la aproximación por redes neuronales (PD+NN) presentó respuestas con mayores diferencias. Durante los primeros 6 s de las simulaciones, el esquema PD+NN llevó al brazo a un posición cercana a los $-40°$ para compensar la perturbación. En el primer experimento, con $\theta_0 = 20°$ y $\alpha_0 = -1°$, mantuvo al brazo en una posición cercana a $0°$ durante el estado estacionario. Sin embargo, para el segundo experimento, con $\theta_0 = -2°$ y $\alpha_0 = 10°$, presentó oscilaciones entre $\pm 20°$. Al igual que en el compensador basado en table, se relacionó a las oscilaciones en la respuesta al espacio de acciones discreto.

De igual forma, en la Figura 3-11 se presentan los primeros 15 s de la respuesta del péndulo con todos los esquema PD en paralelo. Se destaca el hecho de que el controlador PD produjo una respuesta transitoria suave para equilibrar al péndulo. Durante el estado estacionario mantuvo a la respuesta acotada entre $\pm 0.3°$.

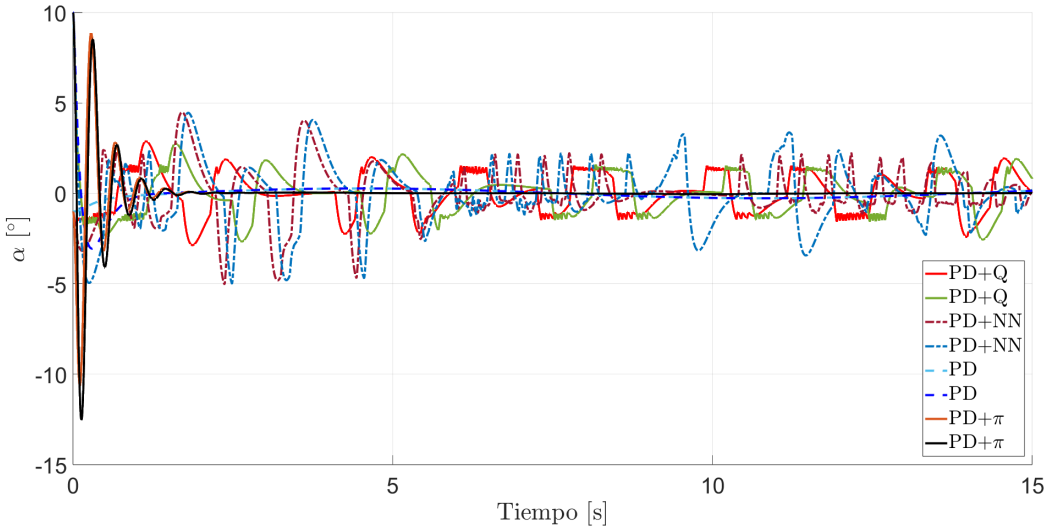

Figura 3-11: Péndulo: Controladores PD en paralelo

Mientras que al implementar el compensador π se produjo una respuesta transitoria que no superó un rango de $\pm 15°$. Durante el estado estacionario se mantuvo a la posición del péndulo dentro de una banda de $\pm 0.05°$, aproximadamente. Proporcionando el menor error en estado estacionario de los 4 esquemas analizados.

En el caso de la compensación por aprendizaje reforzado (PD+Q), con tabla Q, se obtuvo un comportamiento similar en ambos experimentos. En ambos experimentos, se mantuvo a la posición del péndulo en un rango de $\pm 2.5°$, durante el estado estacionario. Las oscilaciones se deben al comportamiento del brazo en el estado estacionario.

Por otro lado, con el esquema PD+NN se presentaron mayores variaciones en la posición del péndulo. En los primeros 6 s, el péndulo se mantuvo entre $\pm 5°$. Posteriormente, se redujo la magnitud de las oscilaciones al mantener al brazo en una posición cercana a $0°$. Sin emabrgo, en el segundo experimento, con $\theta_0 = -2°$ y $\alpha_0 = 10°$, las variaciones en el péndulo fueron de hasta $\pm 3.5°$. El comportamiento registrado con el esquema PD+NN mostró se ve particularmente afectado por los valores pico de la señal de perturbación, durante los 30 s de los experimentos.

En la Figura 3-12 se presenta la comparación de las señales de control obtenidas con cada método. La señal de control corresponde a la tensión aplicada al motor de brazo actuado. De tal forma que es resultado de la señal calculada por el controlador PD, más la perturbación y la señal del compensador, según sea el caso.

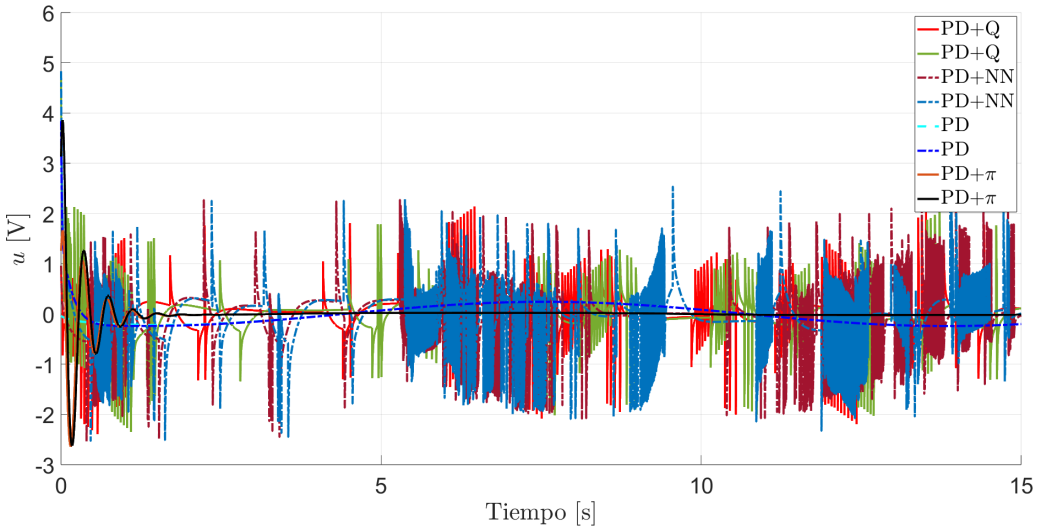

Figura 3-12: Señal de control: Controladores PD en paralelo

Con el controlador PD se mantuvo a la señal de control entre $\pm 0.25\ V$ en el estado estacionario. Con el controlador PD+π se redujo a la señal de control hasta $\pm 0.025\ V$, en el estado estacionario. Caso contrario a lo observado con los compensadores basados en aprendizaje reforzado que mantuvieron a la señal de control con oscilaciones de $\pm 2\ V$, en los valores pico de la señal sinusoidal. Cabe señalar que existen intervalos en donde la señal de control fue cercana a $0\ V$, en donde se mantuvo al péndulo Furuta cerca del equilibrio inestable.

3.7.2 Resultados PD en serie

A continuación se presentan los resultados obtenidos con los controladores PD en serie al equilibrar al péndulo Furuta. Para el esquema de control con compensación (3.32), donde $a_1 = k_{p_1}$, $a_2 = k_{d_1}$, $a_3 = k_{p_1}k_{p_2}$, $a_4 = k_{p_1}k_{d_2} + k_{d_1}k_{p_2}$ y $a_5 = k_{d_1}k_{d_2}$, se utilizó a las ganancias $k_{p_1} = 2.823$, $k_{d_1} = 0.0342$, $k_{p_2} = 1.56$, $k_{d_2} = 0.658$. Mientras que para el caso sin compensación y compensado por RL se utilizó a las ganancias $k_{p_1} = -4.85$, $k_{d_1} = 3.8$, $k_{p_2} = -10$, $k_{d_2} = -1.5$.

Se propuso al espacio de acciones discreto como $u \in \{-1, 0, 1\}$ V para implementar la compensación por aprendizaje reforzado mediante el algoritmo Q-Learning basado en la tabla Q y en redes neuronales. Se conservaron las restricciones $\theta \in [-45°, 45°]$ y $\alpha \in [-25°, 25°]$.

Para la solución tabular se realizaron $2,000$ episodios, con un máximo de $2,000$ pasos de tiempo discreto. Se consideró a $\alpha = 0.25$ y a $\gamma = 0.98$. Para la probabilidad de exploración (3.31), se tomó a $\varepsilon_{\text{máx}} = 1$, $\varepsilon_{\text{mín}} = 0.01$ y $\gamma_{\varepsilon} = 3 \times 10^{-3}$.

Para la aproximación por redes neuronales se realizaron $1,000$ episodios, con un máximo de $2,000$ pasos de tiempo discreto. Usando a $\gamma = 0.98$ y $\alpha = 0.001$. Con $\varepsilon_{\text{máx}} = 1$, $\varepsilon_{\text{mín}} = 0.01$ y $\gamma_{\varepsilon} = 5 \times 10^{-3}$ para la probabilidad de exploración (3.31). Al inicio de cada episodio se utilizó al vector de estado $x_0 = [\theta_0, 0, \alpha_0, 0]^T$, con valores aleatorios para la posición del brazo en $\theta_0 \in [-30°, 30°]$ y la del péndulo en $\alpha_0 \in [-15°, 15°]$.

Se realizaron 2 experimentos con una duración máxima de 30 s y condiciones iniciales dadas por $x_0 = [\theta_0, 0, \alpha_0, 0]^T$. Retomando para el primer experimento a $\theta_0 = 20°$ y $\alpha_0 = -1°$ y para el segundo experimento a $\theta_0 = -2°$ y $\alpha_0 = 10°$. Se comparó al control PD sin compensación contra la compensación no lineal (PD+π), la compensación por tabla Q (PD+Q) y la compensación con el aproximador basado en redes neuronales (PD+NN).

En la Figura 3-13 se muestra la comparación de los controladores PD en serie, durante los primeros 15 s de simulación.

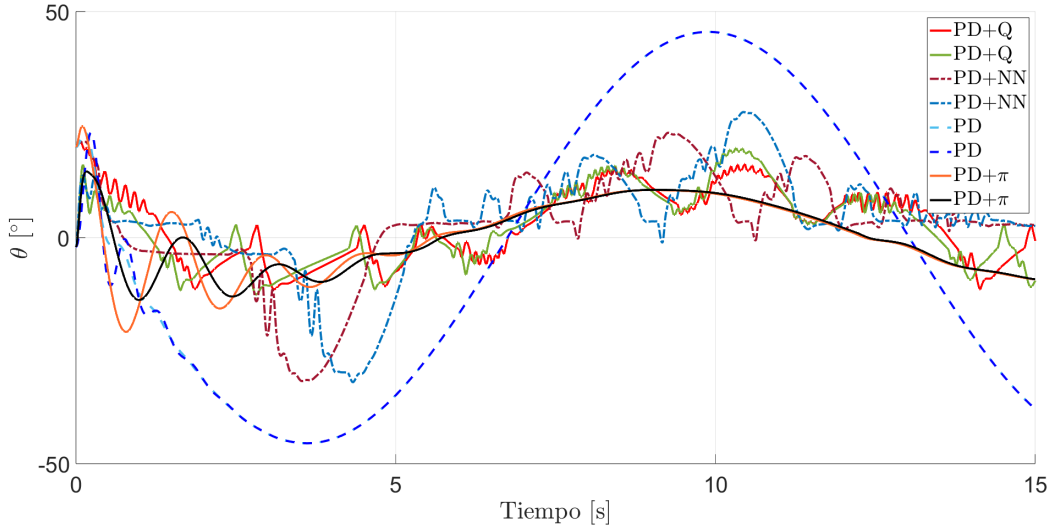

Figura 3-13: Brazo: Controladores PD en serie

Aunque, se presentó una respuesta transitoria con más oscilaciones, en el estado estacionario se mantuvo acotada a la posición entre ±45°. Lo cual representó una mejoría con respecto al esquema en paralelo. Cabe señalar que la sintonización de los controladores PD en serie resultó más complicada que para el esquema en paralelo.

Por otro lado, al implementar al controlador PD+π se observó una evidente mejoría con respecto al caso anterior, al mantener al brazo dentro de un rango de ±10° en el estado estacionario. Sin embargo, presentó un tiempo de asentamiento de aproximadamente 6 s y una respuesta transitoria con oscilaciones más evidentes.

Con respecto a la compensación por aprendizaje reforzado, PD+Q, se observó que la posición del brazo presentó una tendencia similar a la obtenida con el compensador π. Sin embargo, el copensador basado en la tabla Q mantuvo acotada a la respuesta entre ±15°, aproximadamente. Mientras que el compensador basado en la aproximación por redes neuronales (PD+NN), se destacó por haber presentado las mayores oscilaciones durante la compensación, manteniendo a la respuesta dentro del rango de ±30°. Además, ambos compensadores presentaron oscilaciones relacionadas con el espacio de acciones discreto.

Así mismo, en la Figura 3-14 se presenta la comparación del desempeño de los esquemas PD en serie en la posición del péndulo. Se resalta la presencia de sobretiros significativos con los controladores PD y PD+π.

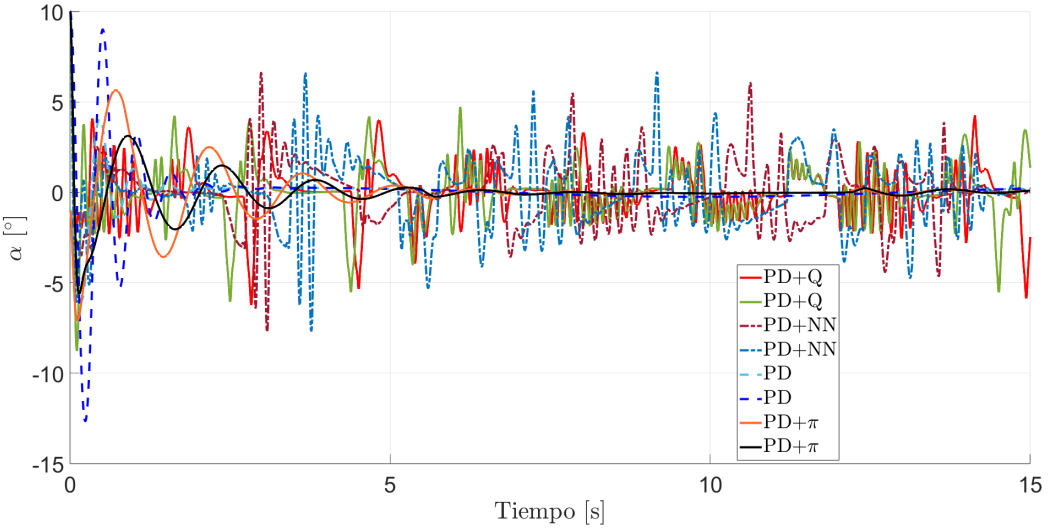

Figura 3-14: Péndulo: Controladores PD en serie

La magnitud de los sobretiros en el péndulo durante la respuesta transitoria, con el esquema PD sin compensación, alcanzó valores de 9° y $-12°$ durante el segundo experimento, con $\theta_0 = -2°$ y $\alpha_0 = 10°$. Una vez alcanzado el estado estacionario, se mantuvo al péndulo en el rango de $\pm 0.25°$. Por su parte, el controlador PD+π presentó menores sobretiros en la respuesta transitoria y mantuvo al péndulo en una banda de $\pm 0.03°$, aproximadamente.

Por otro lado, el controlador PD+Q mostró ser capaz de mantener al péndulo en un rango cercano a 0°. Mantuvo al péndulo oscilando en una rango de $\pm 2.5°$, con puntos específicos en donde se alcanzaron valores de $\pm 5°$. Mientras que la proximación por redes neuronales (PD+NN) presentó sobretiros de 6.5° y 7.5° con la posición del péndulo dentro del rango de $\pm 4°$. En ambos casos los compensadores basados en aprendizaje reforzado eliminaron las oscilaciones observadas con los controladores PD y PD+π.

Así mismo, en la Figura 3-15 se compara a la señal de control obtenida con cada método. Con el controlador PD se obtuvo una señal de control de $\pm 0.2\ V$ en el estado estacionario. Con el controlador PD+π se observó que esta señal se redujo a $\pm 0.06\ V$.

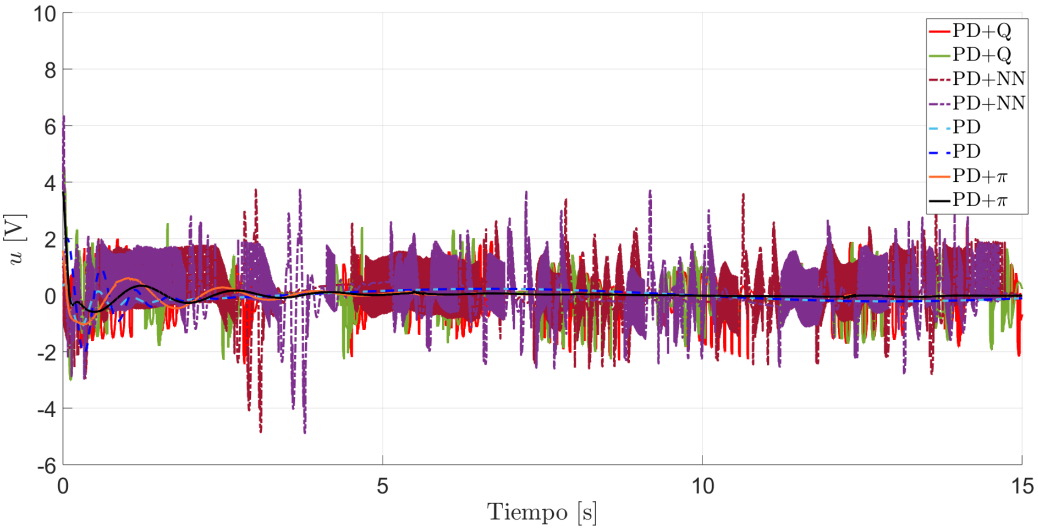

Figura 3-15: Señal de control: Controladores PD en serie

Sin embargo, en el caso de los compensadores basados en aprendizaje reforzado se observó que el espacio de acciones discreto provocó que la señal de control se mantuviera en entre $\pm 2\ V$. Con regiones en donde se redujo la frecuencia de las conmutaciones del compensador o la magnitud de la tensión hasta valores cercanos a $0\ V$.

3.8 Discusión de resultados

Se realizaron simulaciones en Matlab Simulink para evaluar la capacidad del aprendizaje reforzado para equilibrar al péndulo Furuta al rededor del punto de equilibrio inestable. Se utilizó al algoritmo Q-Learning y se comparó a la solución basada en la tabla Q contra el uso de la aproximación de la función Q por redes neuronales.

Los resultados obtenidos mostraron que el entrenamiento realizado para la solución tabular no proporcionó el resultado esperado, debido a que durante la validación el péndulo siempre caía antes de alcanzar los 2 s de simulación. Además, la solución basada en redes neuronales requirió de mucho menos tiempo de entrenamiento para alcanzar una respuesta satisfactoria.

Se asocia el fallo de la solución tabular al tiempo de entrenamiento, el rango decondiciones iniciales sobre el cual se pedía que el controlador fuera efectivo o a la señal de recompensa. Recordando que en el aprendizaje reforzado la señal de recompensa es primordial.

Así mismo, se realizaron simulaciones en Matlab Simulink para evaluar la capacidad del aprendizaje reforzado al implementarlo como compensación de esquemas de control PD en serie y paralelo, en el control del péndulo Furuta sobre el equilibrio inestable. Nuevamente se comparó a la solución tabular contra la aproximación por redes neuronales.

Además, se comparó a los compensadores basados en aprendizaje reforzado contra el compensador π, diseñado con el conocimiento del modelo no lineal del péndulo invertido rotativo.

Con el objetivo de comparar los resultados obtenidos durante las simulaciones se optó por implementar un índice de desempeño que tome en cuenta al error de posición en el péndulo y el brazo, así como a la magnitud de la señal de control aplicada al motor del brazo actuado. Por lo que se propuso al índice de desempeño como una función cuadrática que penalice a los errores más grandes y considere en menor medida a la señal de control. Como el objetivo de control es llevar al brazo y al péndulo a 0° se formuló a la función de costo como:

$$J\left(x_k, u_k\right) = \sum_{k}^{T} \theta_k^2 + \alpha_k^2 + 0.1 u_k^2 \tag{3.33}$$

donde T representa el tiempo terminal de cada episodio. Al aplicar el índice de desempeño en cada experimento para cada uno de los esquemas de control se obtuvieron las valores que se muestran a continuación.

La Tabla 4.1 contiene los resultados obtenidos con los esquemas de control PD en paralelo. Se observa que el controlador PD obtuvo valores alto en ambos experimentos debido a que presentó un error significativo en la posición del brazo. Por otro lado, el buen desempeño del compensador no lineal, π, produjo los valores mas pequeños de todos los experimentos.

Tabla 3.2: Índices de desempeño de los controladores PD en paralelo

Controlador	**Experimento 1**	**Experimento 2**
PD	$5,558.2$	$5,554.3$
PD+π	124.43	149.13
PD+Q	484.2	506.47
PD+NN	681.5	767.9

En el caso de los compensadores por aprendizaje reforzado se observa que el mejor desempeño se obtuvo con la solución tabular, debido a que la aproximación por redes neuronales provocó sobretiros de mayor magnitud en la respuesta del sistema, en ambos experimentos.
Mientras que en la Tabla 3.3 se muestran los resultados correspondientes a los esquemas de control PD en serie. Debido a que la sintonización propuesta, para el esquema PD en serie, redujo la magnitud del error en estado estacionario se obtuvieron valores más pequeños con el índice de desempeño que en el caso del controlador PD en paralelo sin compensación. Así mismo, el esquema PD+π presento el mejor desempeño de los 4 controladores implementados.

Tabla 3.3: Índices de desempeño de los controladores PD en serie

Controlador	**Experimento 1**	**Experimento 2**
PD	$4,873.7$	$4,926.6$
PD+π	611.89	555.84
PD+Q	$1,095.7$	$1,277.7$
PD+NN	$1,880.4$	$2,178.4$

Con respecto a los controladores compensados por aprendizaje reforzado se observó que el mejor desempeño se obtuvo al implementar la solución tabular. Nuevamente, se asocia la diferencia en el desempeño a los sobretiros provocados por la aproximación mediante redes neuronales.
Finalmente, se resalta que mediante la solución tabular esposible obtener un desempeño cercano al obtenido con el compensador no lineal, π. Sin embargo, mediante la aproximación por redes neuronales se reduce el costo computacional de una tabla Q de gran dimensión.

CAPÍTULO 4

Control óptimo libre de modelo por aprendizaje reforzado

De acuerdo con Lewis [1,2,67], el control adaptable y el control óptimo representan dos filosofías diferentes del diseño de controladores por retroalimentación. Normalmente, el cálculo de controladores óptimos para sistemas lineales implica un diseño fuera de línea al resolver las ecuaciones de Hamilton-Jacobi-Bellman (HJB), por ejemplo, la ecuación de Riccati, usando el conocimiento completo del sistema dinámico.

Mientras que los controladores adaptables aprenden en línea a controlar sistemas desconocidos a partir de información medida en tiempo real a lo largo de las trayectorías del sistema. Usualmente, los controladores adaptables no se diseñan para ser óptimos en el sentido de minimizar funciones de desempeño preestablecidas por el usuario, pero deben satisfacer ciertas condiciones de optimalidad inversa.

Por otro lado, el aprendizaje reforzado implica una relación de causa y efecto entre las acciones y las recompensas o los castigos [1, 67]. Para esto, los algoritmos de aprendizaje reforzado se construyen a partir de la idea de que las decisiones de control efectivas deben recordarse, mediante la señal de refuerzo, de tal forma que sea más probable volver a usarlas. Además, el aprendizaje reforzado está basado en información del entorno que se evalua en tiempo real y puede considerarse como un aprendizaje basado en las acciones [2]. Por lo que, desde un punto de vista teórico, el aprendizaje reforzado está relacionado con métodos de control adaptable y control óptimo. De esta forma es posible utilizar al aprendizaje reforzado para resolver problemas de control óptimo.

A continuación se presenta el uso del aprendizaje reforzado para diseñar un controlador óptimo adaptable que aprenda la solución de control óptima al resolver la ecuación algebraica de Riccati en línea, sin resolver específicamente la ecuación de Riccati y sin conocer la dinámica del sistema.

4.1 Regulador Cuadrático Lineal (LQR)

Una gran variedad de sistemas en tiempo discreto pueden ser descritos en la forma de espacio de estado lineal e invariante en el tiempo:

$$x_{k+1} = Ax_k + Bu_k \tag{4.1}$$

donde k es el índice de tiempo discreto, $x_k \in \mathbb{R}^n$ es el vector de estados, $u_k \in \mathbb{R}^m$ el vector de entradas de control, $A \in \mathbb{R}^{n\times n}$ y $B \in \mathbb{R}^{n\times m}$. Además, la ecuación de transición de estado corresponden con un MDP determinista [2]. Las políticas de interés en este tipo de sistemas son las retroalimentaciones de estados de la forma:

$$u_k = h\left(x_k\right) = -Kx_k \tag{4.2}$$

con la política de control $K \in \mathbb{R}^{m\times n}$, una matriz de ganancia de retroalimentación constante. Se debe elegir a K de tal forma que la matriz del sistema en lazo cerrado, $A_{lc} = A - BK$, tenga todos sus valores propios estríctamente dentro del círculo unitario [2].

Así mismo, es posible asignar un costo por paso o utilidad, que representa una medición del costo del control en cada paso. Una forma estándar es la función cuadrática:

$$r\left(x_k, u_k\right) = x_k^T Q x_k + u_k^T R u_k \tag{4.3}$$

donde $Q \in \mathbb{R}^{n\times n}$ y $R \in \mathbb{R}^{m\times m}$, con $Q = Q^T \geq 0$ y $R = R^T > 0$ para que la función de costo está bien definida. Se asume que el par (A, B) es estabilizable, es decir, que existe una ganancia de retroalimentación K que haga al sistema en lazo cerrado asintóticamente estable.

Por otro lado, el costo total de un estado x_k bajo la política de control $h\left(x_k\right)$, $V_h\left(x_k\right)$, está definido como la suma de la utilidad en cad paso donde la política $h\left(x_k\right)$ ha incurrido, desde el paso k en adelante, tal que [2]:

$$V_h\left(x_k\right) = \frac{1}{2}\sum_{i=k}^{\infty} r\left(x_i, u_i\right) = \frac{1}{2}\sum_{i=k}^{\infty}\left(x_i^T Q x_i + u_i^T R u_i\right) \tag{4.4}$$

Esta definición implica la siguiente relación por recurrencia:

$$\begin{aligned} V_h(x_k) &= \frac{1}{2}\left(x_k^T Q x_k + u_k^T R u_k\right) + \frac{1}{2}\sum_{i=k+1}^{\infty}\left(x_i^T Q x_i + u_i^T R u_i\right) \\ &= \frac{1}{2} r(x_k, u_k) + V_h(x_{k+1}) \end{aligned} \tag{4.5}$$

El resultado anterior corresponde con la ecuación de Bellman para un LQR [2]. Además, al retomar la definición del control (4.2), la utilidad (4.3) y la función de costo total (4.4), se tiene:

$$\begin{aligned} V_h(x_k) &= \frac{1}{2}\sum_{i=k}^{\infty}\left(x_k^T Q x_k + u_k^T R u_k\right)\Big|_{u_k=-Kx_k} \\ &= \frac{1}{2}\sum_{i=k}^{\infty} x_k^T\left(Q + K^T R K\right) x_k \end{aligned}$$

Por lo tanto, V_h es una función cuadrática en el estado y puede ser expresada como:

$$V_h(x_k) = \frac{1}{2} x_k^T P x_k \tag{4.6}$$

donde $P \in \mathbb{R}^{n\times n}$ es la matriz de costo para la política K. De esta forma, se expresa a la ecuaión de Bellman como:

$$2V_h(x_k) = x_k^T P x_k = x_k^T Q x_k + u_k^T R u_k + x_{k+1}^T P x_{k+1} \tag{4.7}$$

Al sustituir a la dinámica del sistema (4.1) y la ganancia de retoralimentación (4.2), asumiendo una política por retroalimentación de estado constante, $h(x_k)$, que sea estacionaria para alguna ganancia estabilizante K, se tiene:

$$2V_h(x_k) = x_k^T P x_k = x_k^T\left(Q + K^T R K + (A - BK)^T P (A - BK)\right) x_k \tag{4.8}$$

Como esto se debe cumplir para todos los estados x_k se considera:

$$(A - BK)^T P (A - BK) - P + Q + K^T R K = 0 \tag{4.9}$$

Esta ecuación matricial es lineal en P y se le conoce como la ecuación de Lyapunov cuando K es fija [1]. Con lo cual se observa que la ecuación de Bellman para un LQR en tiempo discreto es equivalente a una ecuación de Lyapunov [2]. Resolviendo esta ecuación con una ganancia K prestablecida se obtiene que $P = P^T > 0$.

Así mismo, se tiene a la función Hamiltoniana para un LQR en tiempo discreto como:

$$H\left(x_k, u_k\right) = x_k^T Q x_k + u_k^T R u_k + \left(A x_k + B u_k\right)^T P \left(A x_k + B u_k\right) - x_k^T P x_k \tag{4.10}$$

De acuerdo con Lewis [2], la función Hamiltoniana es equivalente al error de diferencia temporal en un MDP. Una condición necesaria para la optimalidad es la condición de valor estacionario, dada por:

$$\frac{\partial H\left(x_k, u_k\right)}{\partial u_k} = 0 \tag{4.11}$$

Por lo tanto, al derivar la función Hamiltoniana (4.10) con respecto de u_k e igualar a 0, se tiene:

$$R u_k + B^T P \left(A x_k + B u_k\right) = 0 \tag{4.12}$$

de donde se despeja al controlador:

$$u_k = -K x_k = -\left(R + B^T P B\right)^{-1} B^T P A x_k \tag{4.13}$$

Entonces, la política será:

$$K = \left(R + B^T P B\right)^{-1} B^T P A \tag{4.14}$$

Al sustituir a la política de control en la ecuación de Bellman (4.7) y simplificar el resultado se obtiene a la ecuación de Hamilton-Jacobi–Bellman en tiempo discreto:

$$A^T P A - P + Q - A^T P B \left(R + B^T P B\right)^{-1} B^T P A = 0 \tag{4.15}$$

Esta ecuación es cuadrática en P y se conoce como la ecuación algebráica de Riccati y es equivalente a la ecuación de optimalidad de Bellman para un LQR [2]. De acuerdo con Bradtke [79], esta es una forma simple pero computacionalmente costosa de obtener a K^*, sí se conocen modelo precisos del sistema y la función de costo.

Existen diferentes métodos fuera de línea para resolver la ecuación algebraica de Riccati como el algoritmo de Hewer y la recursión de Lyapunov [2]. Sin embargo, es necesario el completo conocimiento de la dinámica de la planta, (A, B), para su implementación.

Por otro lado, es posible implementar en línea un algoritmo basado en la función de acción y valor Q, sin conocer la dinámica del sistema, sólo midiendo información a lo largo de las trayectorias del sistema. Esto produce algoritmos de control óptimo adaptable que convergen en línea a soluciones de control óptimas.

4.1.1 Función de acción y valor Q

La función de acción y valor Q para un LQR en tiempo discreto, siguiendo la política K, se deriva de la función de valor como [1]:

$$Q(x_k, u_k) = \frac{1}{2} r(x_k, u_k) + V_h(x_{k+1}) \tag{4.16}$$

$$= \frac{1}{2}\left(x_k^T Q x_k + u_k^T R u_k\right) + V_h(x_{k+1}) \tag{4.17}$$

Donde la señal de control u_k es arbitraria y la política $u_k = h(x_k)$ se sigue para $k+1$ y los subsecuentes pasos. Escribiendo a la función Q como:

$$Q(x_k, u_k) = \frac{1}{2}\left(x_k^T Q x_k + u_k^T R u_k\right) + \frac{1}{2}\left(Ax_k + Bu_k\right)^T P\left(Ax_k + Bu_k\right) \tag{4.18}$$

con la solución de la ecuación algebráica de Riccati, P, se tiene a la función Q para el LQR en tiempo discreto:

$$Q(x_k, u_k) = \frac{1}{2}\begin{bmatrix} x_k \\ u_k \end{bmatrix}^T \begin{bmatrix} A^T P A + Q & A^T P B \\ B^T P A & B^T P B + R \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} \tag{4.19}$$

Se define:

$$Q(x_k, u_k) = \frac{1}{2} z_k^T S z_k = \frac{1}{2}\begin{bmatrix} x_k \\ u_k \end{bmatrix}^T \begin{bmatrix} S_{xx} & S_{xu} \\ S_{ux} & S_{uu} \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} \tag{4.20}$$

Con $z_k \triangleq [x_k,\ u_k]^T$ y la matriz $S = S^T > 0$ de la forma:

$$\begin{bmatrix} A^T P A + Q & A^T P B \\ B^T P A & B^T P B + R \end{bmatrix} = \begin{bmatrix} S_{xx} & S_{xu} \\ S_{ux} & S_{uu} \end{bmatrix} = \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1l} \\ s_{21} & s_{22} & \cdots & s_{2l} \\ \vdots & \vdots & \ddots & \vdots \\ s_{l1} & s_{l2} & \cdots & s_{ll} \end{bmatrix}$$

con $S \in \mathbb{R}^{l \times l}$, $S_{xx} \in \mathbb{R}^{n \times n}$, $S_{xu} = S_{ux}^T \in \mathbb{R}^{n \times m}$, $S_{uu} \in \mathbb{R}^{m \times m}$.

Por otro lado, para la actualización de la política se considera:

$$\frac{\partial}{\partial u} Q(x_k, u_k) = 0 \tag{4.21}$$

De la representación (4.19) se obtiene:

$$0 = B^T P A x_k + \left(B^T P B + R\right) u_k \tag{4.22}$$

Con lo cual, es posible expresar a la señal de control óptima como una función que depende del vector de estado x_k:

$$u_k = -(B^T P B + R)^{-1} B^T P A x_k \tag{4.23}$$

Sinembargo, al considerar a la derivada parcial de la expresión (4.20) con respecto a la señal de control, se tiene:

$$u_k = -S_{uu}^{-1} S_{ux} x_k \tag{4.24}$$

La expresión (4.23) requiere conocimiento de la dinámica del sistema (A, B) para realizar la mejora de la política de control. mientras que (4.24) sólo requiere de conocimiento sobre la matriz S de la función Q [79].

4.2 Control óptimo adaptable por Q-learning

Ahora se presenta un algoritmo de control adaptable basado en Q-Learning que converge en línea a la solución de un LQR en tiempo discreto para sistemas completamente desconocidos. Para esto se resuelve la ecuación algebráica de Riccati en tiempo real, sólo usando la información medida a lo largo de las trayectorias del sistema [2].

De acuerdo con [2], la matriz S de la función Q (4.20) se puede estimar en línea, sin conocer la dinámica del sistema, a través de técnicas de identificación. Para esto, se escribe a la función Q (4.20) en forma paramétrica:

$$Q(x, u) = Q(z) = W^T (z \otimes z) = W^T \phi(z) \tag{4.25}$$

con W el vector formado por los elementos de S y $\otimes$ como el producto de Kronecker. La función $\phi(z) = z \otimes z$ es la base polinomial cuadrática en términos de los elementos de los z. Al eliminar las entradas redundantes de $\phi(z)$ se garantiza que W sólo contendrá los $(n+m)(n+m+1)/2$ elementos de la mitad superior de S, teniendo la forma:

$$W = [s_{11}, 2s_{12}, \cdots, 2s_{1l}, s_{22}, \cdots, 2s_{2l}, s_{33}, \cdots, 2s_{3l}, , \cdots, s_{ll},]^T \tag{4.26}$$

Mientras que el vector de funciones cuadráticas tiene la forma:

$$\phi(z) = z \otimes z = \left[z_1^2, z_1 z_2, \cdots, z_1 z_l, z_2^2, z_2 z_3, \cdots, z_2 z_l, \cdots, z_l^2\right]^T \tag{4.27}$$

Ahora, se define al error de diferencia temporal:

$$e_k = -Q\left(x_k, u_k\right) + r\left(x_k, u_k\right) + Q\left(x_{k+1}, h\left(x_{k+1}\right)\right) \tag{4.28}$$

Sustituyendo la aproximación de la función Q en el error de diferencia temporal:

$$e_k = -W^T \phi\left(z_k\right) + r\left(x_k, u_k\right) + W^T \phi\left(z_{k+1}\right) \tag{4.29}$$

El paso de evaluación de la función Q de un algoritmo de iteración de la política está dado como:

$$W_{j+1}^T \left(\phi\left(z_k\right) - \phi\left(z_{k+1}\right)\right) = \frac{1}{2}\left(x_k^T Q x_k + u_k^T R u_k\right) \tag{4.30}$$

con el paso de mejora de la política:

$$h_{j+1}\left(x_k\right) = \arg\min_u \left(W_{j+1}^T \phi\left(x_k, u\right)\right), \ \forall x \in X \tag{4.31}$$

Cabe señalar que en (4.30) se tienen $\left(n+m\right)\left(n+m+1\right)/2$ incognitas, las cuales conforman al vector W, lo cual representa perfectamente el tipo de ecuaciones utilizadas en la identificación de sistemas. Por lo tanto, para su implementación en línea se recurrió al uso de Mínimos Cuadrados Recursivos (RLS) para el vector de parámetros W_{j+1} con el vector de regresión [79]:

$$\Phi_k = \phi\left(z_k\right) - \phi\left(z_{k+1}\right) \tag{4.32}$$

Los datos que se deben medir en cada paso son $\left(x_k, u_k, r\left(x_k, u_k\right), x_{k+1}, u_{k+1}\right)$, donde u_{k+1} se calcula como $h_j\left(x_{k+1}\right)$, con $h_j\left(\cdot\right)$ como la política actual. Es necesario añadir ruido de prueba a la señal de control, u_k, para garantizar la excitación persistente. Las relaciones de recurrencia del algoritmo RLS están dadas por [79]:

$$\begin{aligned} W_{j+1} &= W_j + \frac{P_{c_j}\Phi_k\left(r\left(x_k, u_k\right) - W_j^T \Phi_k\right)}{1 + \Phi_k^T P_{c_j} \Phi_k} \\ P_{c_{j+1}} &= P_{c_j} - \frac{P_{c_j}\Phi_k \Phi_k^T P_{c_j}}{1 + \Phi_k^T P_{c_j} \Phi_k} \end{aligned} \tag{4.33}$$

Cabe señalar que este algoritmo requiere que la ganancia inicial de retroalimentación deberá ser establizante. Dicha ganancia es fácil de encontrar mediante pruebas iniciales en la planta [79]. A continuación, se presenta el algoritmo basado en iteración de la política mediante Q-learning para hallar la solución en línea de un LQR en tiempo discreto.

Algoritmo 5 Q-Learning en línea para LQR discreto

1: Inicializar la política de retroalimentación $u_k = -K_0 x_k$ para $j = 0$, la cual deberá ser estabilizante.
2: Inicializar el vector de parámetros, $W_0 \leftarrow 0$
3: Medir el estado incial x_0
4: **para** cada paso de tiempo $k = 0, 1, 2, \ldots$ **hacer**
5: Inicializar la matriz de covarianza, $P_{c_0} \leftarrow 0$
6: **para** $i = 1$ hasta N **hacer**
7: Deterimine la señal de control, $u_k = -K_j x_k + e_k$, donde K_j es el controlador actual y e_k la componente de exploración de la señal de control.
8: Aplicar u_k, medir la transición de estado x_{k+1}, la recompensa r_{k+1} y calcular $u_{k+1} = -K_j x_{k+1}$.
9: Calcular los conjuntos de bases cuadráticas $\phi(z_k)$ y $\phi(z_{k+1})$
10: Actualizar la estimación de los parámetros de las función Q, W_j, usando el algoritmo de RLS.
11: $k = k + 1$
12: **fin para**
13: Determinar la matriz simétrica S con el vector de parámetros W_j.
14: Realizar la mejora de la política con $K_{j+1} = S_{uu}^{-1} S_{ux}$
15: Inicializar el vector de parámetros $W_{j+1} = W_j$
16: $j = j + 1$
17: **fin para**

La actualización del vector de parámetros, W, se realiza hasta que el algoritmo RLS converja, para esto se utiliza como parámetro a la norma Euclidiana de la variación existente entre cada actualización, $\|W_{j+1} - W_j\| \leq \varepsilon_W$, para ε_W una constante pequeña. De acuerdo con Bradtke [79], considerar un número máximo de actualizaciones en W antes de actualizar K permite mejorar la convergencia y garantizar la condición de excitación persistente:

$$\varepsilon_0 I \leq \frac{1}{N} \sum_{i=1}^{N} \phi_{k-i} \phi_{k-i}^T \leq \bar{\varepsilon}_0 I \tag{4.34}$$

donde N_0, $\varepsilon_0 \leq \bar{\varepsilon}_0$ son número positivos.

De igual forma, se utiliza a la norma Euclidiana de la variación en la estimación $\|K_{j+1} - K_j\| \leq \varepsilon_K$, para ε_K una constante pequeña. De esta forma, se determina la convergencia del algoritmo y se detiene la estimación.

4.3 Simulaciones

Se implementó al Algoritmo 5 a través del modelo de Matlab Simulink que se muestra en la Figura 4-1. Constó de dos subsistemas, el primero se utilizó para implementar el algoritmo de aprendizaje reforzado, identificado como *LQR por RL*. Mientras que en el segundo subsistema se realizó la simulación del modelo no lineal del péndulo Furuta, se aplicó al controlador por retroalimentación de estados y se añadió al ruido de prueba con un bloque de ruido blanco de banda limitada, con potencia de ruido de 0.5 y un tiempo de muestreo de 0.13 s.

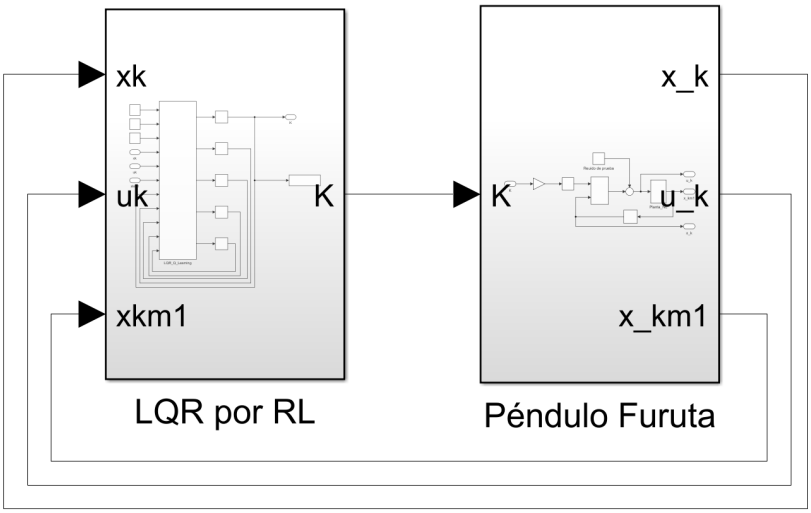

Figura 4-1: Modelo de Simulink para un LQR por aprendizaje reforzado

El interior del subsistema *LQR por RL* se muestra en la Figura 4-2. Se utilizó una función de Matlab que depende de las matrices Q y R de la función de utilidad (4.3), de la señal de control u_k y de los estados x_k y x_{k+1}. Además es necesario utilizar bloques de retardo para actualizar a K, W y P_c. Así mismo, se establecieron contadores para conocer el número de actualizaciones realizadas al vector K y el número de pasos de tiempo discreto que han transcurrido desde la última actualización, que una vez superado el límite establecido se realizaba la actualización del algoritmo de aprendizaje reforzado.

Así mismo, se utilizaron bloques *To Workspace* para almacenar a los valores del vector de ganancias K, la señal de control aplicada, u_k, y el comportamiento de la posición del brazo, θ, y el péndulo, α.

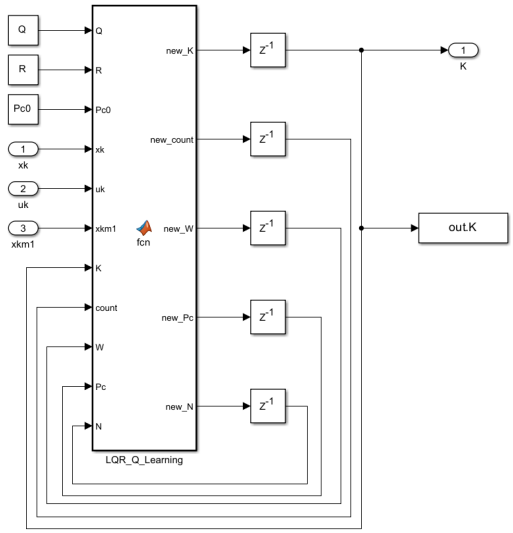

Figura 4-2: Función de Matlab para un LQR por aprendizaje reforzado

4.3.1 Resultados

Se propuso evaluar la capacidad del algoritmo para hallar la solucón en línea de un LQR en tiempo discreto ante incertidumbre paramétrica. Se retomaron los valores mostrados en la Tabla 3.1, con un tiempo de muestreo de 2 *ms*. Se propuso comenzar los experimentos con el péndulo *mediano*, de longitud $L_2 = 0.3365\ m$, $l_2 = 0.1556\ m$ y masa $m_2 = 0.127\ Kg$, para posteriormente reemplazarlo por el péndulo *corto* cuyos valores de longitud y masa se presentan en Tabla 3.1. Se realizó la discretización del modelo lineal del péndulo Furuta mediante Matlab con el método de retenedor de orden cero (zoh). Al considerar el péndulo mediano se obtuvo:

$$A_m = \begin{bmatrix} 1 & 0.0020 & -0.0001 & 0 \\ 0 & 0.9616 & -0.1041 & 0.0012 \\ 0 & 0 & 1.002 & 0.0020 \\ 0 & 0.0383 & 0.1945 & 0.9978 \end{bmatrix}, \ \mathbf{B}_m = \begin{bmatrix} 0.0001 \\ 0.0691 \\ -0.0001 \\ -0.0689 \end{bmatrix} \tag{4.35}$$

Mientras que para el péndulo corto, se tiene:

$$A_c = \begin{bmatrix} 1 & 0.0019 & -0.0001 & 0 \\ 0 & 0.9442 & -0.1426 & 0.0021 \\ 0 & 0.0001 & 1.0003 & 0.0020 \\ 0 & 0.0655 & 0.2739 & 0.9961 \end{bmatrix}, \ \mathbf{B}_c = \begin{bmatrix} 0.0001 \\ 0.1003 \\ -0.0001 \\ -0.1177 \end{bmatrix} \tag{4.36}$$

Posteriormente se realizó el diseño de un LQR para cada sistema, considerando a la función de costo cuadrática planteada para el algoritmo de aprendizaje reforzado. Se propuso a las matrices, Q y R, como $Q = 20I$ y $R = 1$. De tal forma que los vectores de ganancias fueron:

$$K_m = \begin{bmatrix} -3.6148 & -5.4437 & -69.5539 & -10.3393 \end{bmatrix}^T \tag{4.37}$$

$$K_c = \begin{bmatrix} -3.2097 & -4.8446 & -58.4409 & -8.0511 \end{bmatrix}^T \tag{4.38}$$

donde K_c corresponde a la solución del LQR para los valores nominales y K_m a la solución con los cambios en el péndulo.

En la simulación del algoritmo basado en aprendizaje reforzado se conservó a $Q = 20I$ y $R = 1$ para la función de utilidad. Se estableció al valor inicial de la matriz de covarianza como $P_{c_0} = 90I$. Con respecto al vector de parámetros W se observó un mejor desempeño cuando se estableciá como $W_0 = \mathbf{0}$.

Para determinar un vector de ganancias iniciales se realizaron pruebas con el sistema en lazo cerrado y se optó por el vector $K_0 = [-1, -1, -10, -1]^T$. Por otro lado, para determinar la convergencia del algoritmo se propuso a ε_W yε_K como $\varepsilon = 1 \times 10^{-9}$. Con lo cual, se actualizó al vector K cuando $\|W_{j+1} - W_j\| \leq \varepsilon$ o cuando los pasos de tiempo discreto k desde la última actualización superaron un máximo de $150,000$. Así mismo, el algoritmo se detiene cuando $\|K_{j+1} - K_j\| \leq \varepsilon$.

Para implementar el experimento propuesto se estableció que al detenerse el algoritmo de aprendizaje reforzado se realizaría una prueba ante un punto de referencia dado. Después se realizaría el cambio en los parámetros del péndulo para iniciar nuevamente al algoritmo. Al obtener la solución se repetirá la prueba ante el punto de referencia propuesto.

En la Figura 4-3 se muestra el comportamiento del péndulo brazo y el péndulo durante la sintonización. Se observó que el brazo mantuvo su rango de movimiento entre $\pm 250°$ al inicio de cada sintonización, con $K_0 = [-1, -1, -10, -1]^T$. Se tomó esta decisión al observar que mejoraba el desempeño en la segunda etapa con el vector de ganancias iniciales K_0 en comparación con el vector hallado durante la primer sintonización. Así mismo, el péndulo mostró una respuesta oscilatoria que se mantuvo acotada entre $\pm 80°$, durante el inicio de ambas etapas. Sin embargo, durante la mayor parte del experimento se mantuvo a la respuesta del brazo y el péndulo en rangos de $\pm 30°$ y $\pm 6°$, respectivamente.

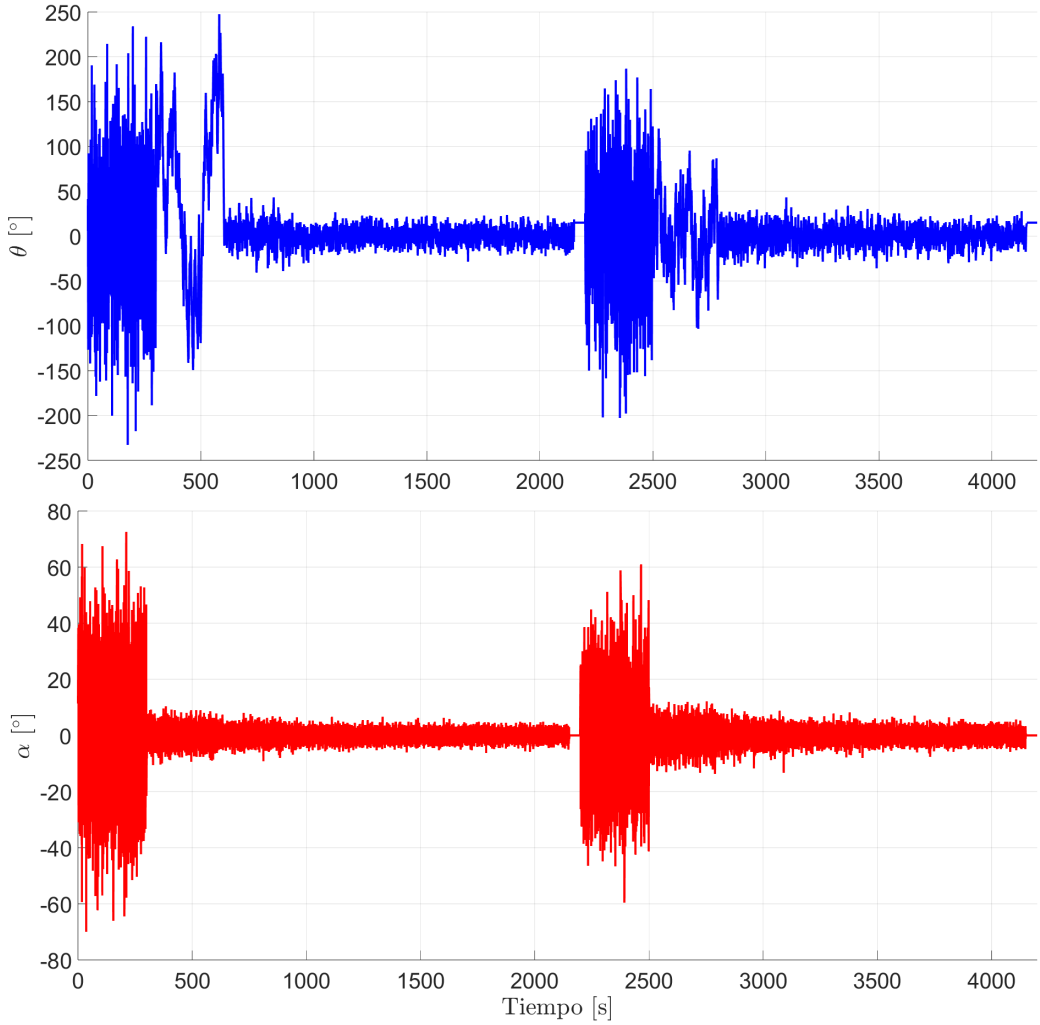

Figura 4-3: Comportamiento del sistema durante la sintonización

En la Figura 4-4 se muestra la sintonización del vector de ganancias, K, para el LQR del péndulo furuta con los péndulos mediano y corto. En la parte superior se presenta a las 3 componentes de menor magnitud, para facilitar su visualización. Mientras que en la parte inferior aparece la componente que alcanza valores más grandes y está relacionada con la posición del péndulo. Los vectores obtenidos en cada caso fueron:

$$\hat{K}_m = \begin{bmatrix} -4.2965 & -5.2935 & -68.2348 & -9.6118 \end{bmatrix}^T \tag{4.39}$$

$$\hat{K}_c = \begin{bmatrix} -3.5859 & -4.7077 & -56.9293 & -7.9692 \end{bmatrix}^T \tag{4.40}$$

De la primera sintonización se resalta que el error entre los valores deseados y los estimados es mayor en las componentes K_1 y K_4, en comparación con los resultados obtenidos para K_2 y K_3. Sin embargo, en la segunda sintonización se redujo el error en K_1 y K_4.

Así mismo, se identificó que en la gran mayoría de actualizaciones se supero el límite de $150,000$ pasos de tiempo discreto, debido a que se estableció a $\varepsilon_W = 1 \times 10^{-9}$. No obstante, se obtuvo un buen desempeño durante la sintonización al haber obtenido valores muy cercanos a los calculados previamente.

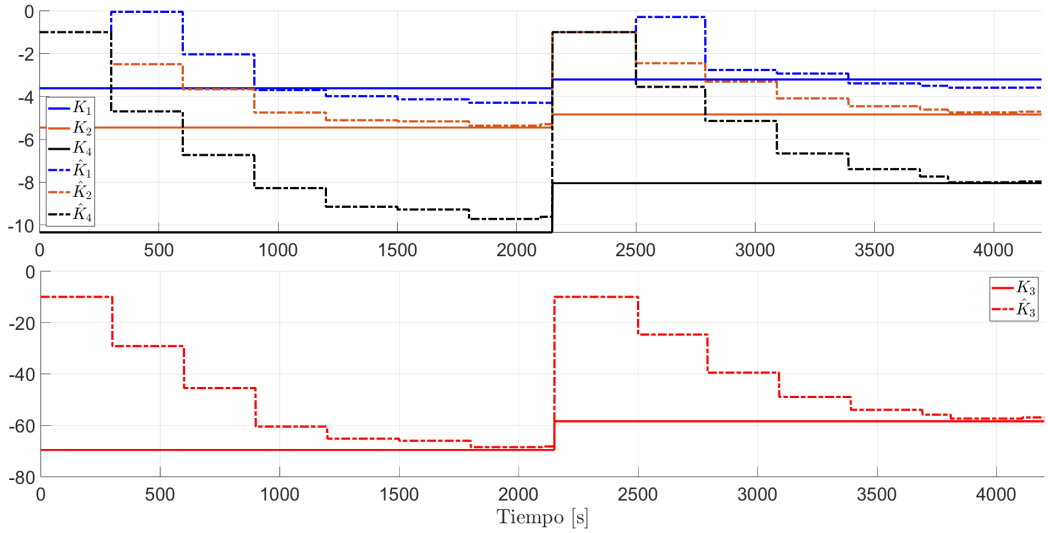

Figura 4-4: Sintonización de ganancias de un LQR

Posteriormente se propuso un experimento para comparar la respuesta del sistema con los diferentes vectores K. Se propuso al vector de condiciones iniciales $x_0 = [\theta_0, 0, \alpha_0, 0]^T$, con $\theta_0 = -15°$ y $\alpha_0 = 10°$. Se realizaron simulaciones, con los péndulos corto y mediano, en donde se comparó a los vector K_m (4.37) y K_c (4.38) contra los obtenidos por aprendizaje reforzado, $\hat{K}_m$ (4.39) y $\hat{K}_c$ (4.40), respectivamente.

En la Figura 4-5 se presenta la comparación de los 4 controladores, en donde las etiquetas LQR_i indican el uso de los vectores calculados analíticamente y las etiquetas RL_i el uso de los vectores obtenidos por aprendizaje reforzado.

Al analizar el comportamiento del brazo se observó que los controladores obtenidos por RL presentaron comportamientos muy similares. Además, produjeron una respuesta más rápida, lo cual puede indicar una señal de control ligeramente mayor que la de los controladores óptimos.

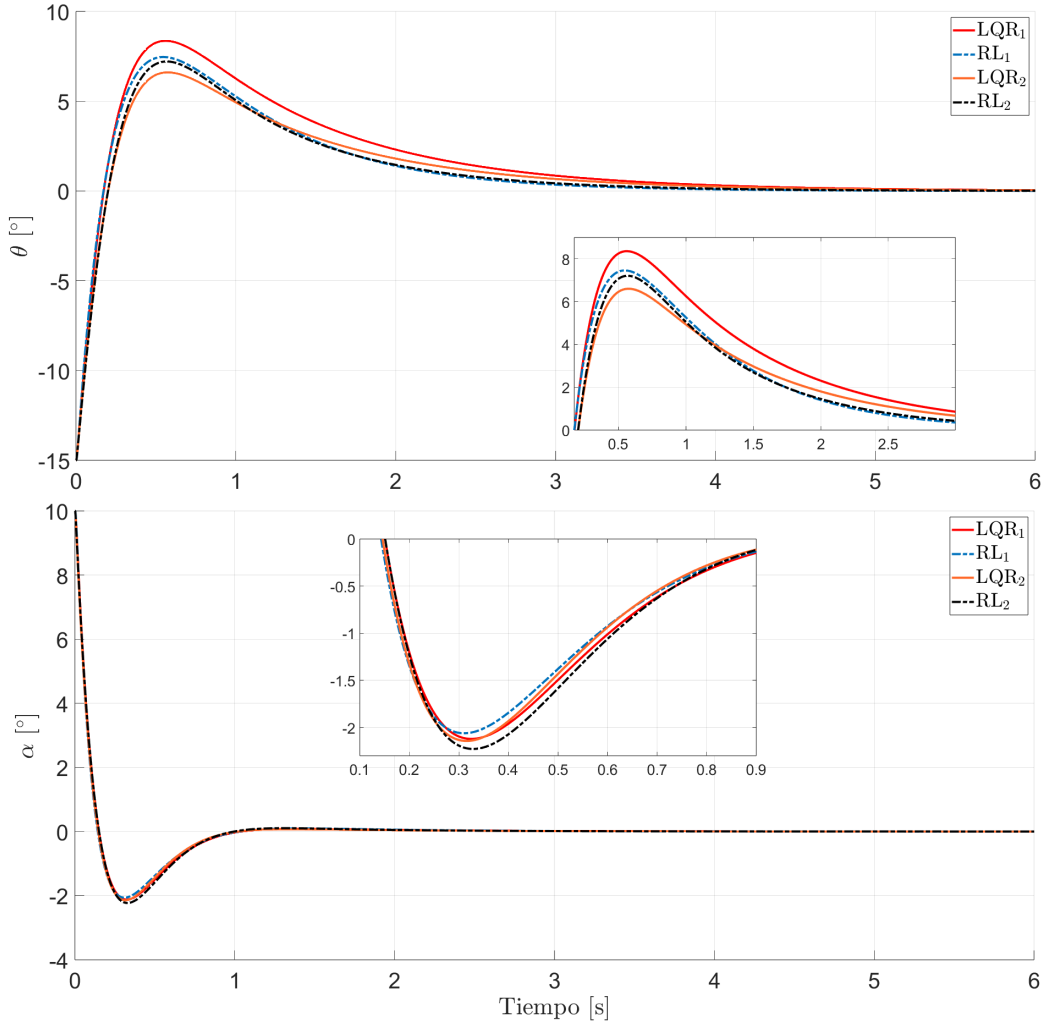

Figura 4-5: Comparación de respuestas con LQR

Por otro lado, la respuesta del péndulo mostró un comportamiento similar con los 4 controladores. Se observó que los controladores óptimos mantuvieron un sobretiro muy cercano. Sin embargo, los controladores sintonizados mediante aprendizaje reforzado mostraron una diferencia mayor entre las magnitudes obtenidas de sobretiro máximo.

En la Figura 4-6 se muestra la comparación de la señal de control obtenida con cada vector K. Al realizar un acercamiento sobre los primero 50 ms se observa que el vector $\hat{K}_m$ (4.39) produjo una señal ligeramente mayor que el controlador óptimo, provocando un menor tiempo de asentamiento y un menor sobretiro en el péndulo. Mientras que el vector $\hat{K}_c$ (4.40) produjo una señal de control muy parecida a la obtenida con el vector K_c (4.38).

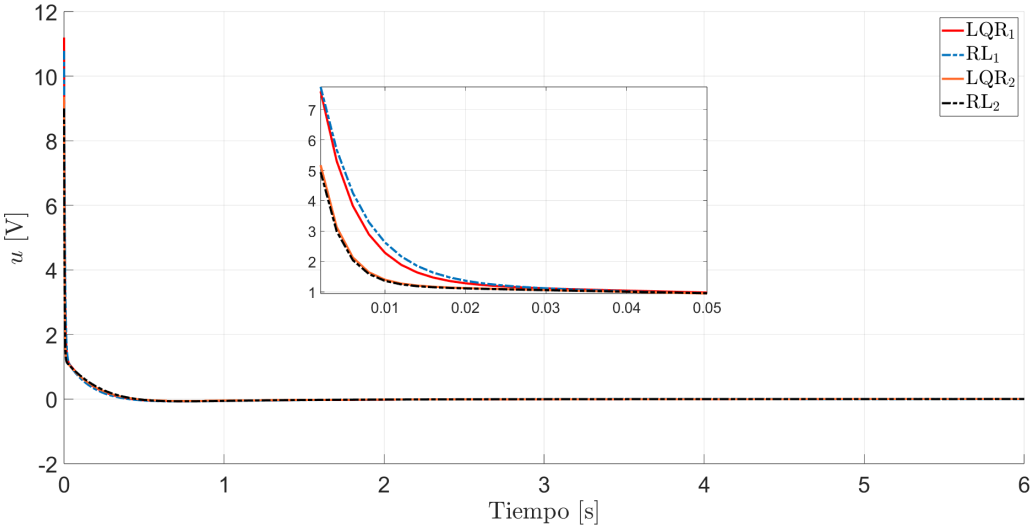

Figura 4-6: Señal de control: Comparación de respuestas con LQR

4.4 Discusión de resultados

Se realizaron simulaciones en Matlab Simulink para implementar un algoritmo basado en aprendizaje reforzado que permite hallar en línea la solución de un LQR en tiempo discreto, para controlar al péndulo Furuta sobre el equilibrió inestable. Se propuso considerar a dos péndulo con longitud y masa diferentes, identificados como péndulo *mediano* y péndulo *chico*.

Se discretizó al sistema con un tiempo de muestreo de 2 *ms* y se realizó el diseño de un regulador cuadrático lineal para cada péndulo. De tal forma que fuera posible realizar una comparación de los resultados obtenidos de forma analítica contra la respuesta del algoritmo inteligente.

Los resultados obtenidos durante la sintonización mostraron que al implementar el algoritmo basado en aprendizaje reforzado es posible hallar soluciones cercanas a los valores calculados. Sin embargo, con ambos péndulos se obtuvo al menos una componente del vector K más lejana del controlador óptimo.

Con el objetivo de comparar el desempeño de los controladores obtenidos se realizó un experimento con el vector de condiciones iniciales $x_0 = [\theta_0, 0, \alpha_0, 0]^T$, con $\theta_0 = -15°$ y $\alpha_0 = 10°$. Los resultados mostraron respuestas muy parecidas con los 4 controladores, con respecto a su capacidad de llevar al péndulo Furuta al equilibrio inestable con el menor esfuerzo de control

posible.

Con el objetivo de realizar una comparación del desempeño de cada controlador se recurre a la función de costo del regulador cuadrático lineal:

$$J\left(x_k, u_k\right) = \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k \tag{4.41}$$

con las matrices $Q = 20I$ y $R = 1$. Los resultados obtenidos se presentan en la siguiente tabla:

Tabla 4.1: Índices de desempeño de LQR

Controlador	**Costo total**
LQR_1	$8,114.9$
RL_1	$8,151.2$
LQR_2	$7,121.1$
RL_2	$7,134.8$

Las etiquetas LQR_1 y RL_1 corresponden a los experimentos con el péndulo mediano. Mientras que LQR_2 y RL_2 corresponden al uso del péndulo corto. Se destaca que los controladores obtenidos por aprendizaje reforzado proporcionaron un desempeño muy cercano al obtenido con los controladores óptimos. En ambos casos, el incremento registrado por la función de costo es muy pequeño, con un incremento del $0.45\,\%$ con el péndulo mediano y del $0.19\,\%$ con el péndulo corto.

De tal forma que el algoritmo basado en aprendizaje reforzado representa una alternativa que no requiere de conocimiento sobre la dinámica del sistema para hallar soluciones cercanas a los controladores óptimos.

CAPÍTULO 5

Conclusiones

En los últimos años el aprendizaje reforzado se ha convertido en un tema de gran interés y utilidad en el control automático. Esto se debe a que representa un marco libre de modelo para resolver problemas de control óptimo, planteados como procesos de decisión de Markov. En donde el agente recibe información del sistema en forma de estados y a partir de llos toma una acción. Por lo que es posible compararlo con un control por retroalimentación de estados.

En este trabajo se diseñaron controladores basados en aprendizaje reforzado para controlar al péndulo invertido rotativo sobre su equilibrio inestable. Se propuso evaluar al aprendizaje reforzado en tres tareas diferentes. El control del péndulo Furuta mediante el algoritmo Q-Learning, basado en la tabla Q y en la aproximación de la función Q por redes neuronales. En la compensación de arquitecturas PD en serie y paralelo, por Q-Learning, ante perturbaciones desconocidas. Y en la sintonización en línea de un regulador cuadrático lineal cuando existe variación paramétrica. En los tres casos se plantearon algoritmos basados en Q-Learning, lo cual permite describir una solución libre de modelo.

Los resultados obtenidos en el control por aprendizaje reforzado mostraron que el uso de una tabla Q requierió de un mayor de tiempo de entrenamiento y no se cumplió el objetivo de equilibrar al péndulo invertido. Por otro lado, el uso de la aproximación de la función Q, por redes neuronales, requirió de menos tiempo de entrenamiento y se obtuvo un controlador capaz de mantener al sistema cerca del equilibrio inestable una vez que se realizó la validación. Además, se destaca el uso de estrategias pensadas para mejorar la estabilidad del aproximador no lineal,

como lo es la red objetivo y la repetición de experiencia.

Al abordar el uso del aprendizaje reforzado en la compensación de esquemas de controladores PD en serie y paralelo se observó que el entrenamiento de la solución tabular se redujo. Además, el uso de la tabla Q produjo un compensador útil para lidiar con una perturbación desconocida, duarnte la validación. Se resalta la similitud en la respuesta obtenida con la tabla Q y el compensador diseñado con el modelo no lineal del sistema. Así mismo, al implementar la compensación mediante el aproximador no lineal de la función Q se obtuvo un desempeño satisfactorio, con un menor costo computacional que al utilizar una tabla Q de grandes dimensiones. Se resalta el hecho de no haber utilizado a la perturbación durante el entrenamiento de los compensadores inteligentes. Esto debido a que se busco evaluar la eficacia de la política ε-greedy en el entrenamiento, al facilitar la exploración del espacio de estados y acciones.

Posteriormente, se implementó un algoritmo basado en aprendizaje reforzado que permite hallar en línea la solución de un LQR en tiempo discreto. Se propuso considerar dos péndulos, de longitud y masa diferentes, para evaluar la capacidad del algoritmo para hallar vectores de ganancias, $\hat{K}$, cercanos a los valores óptimos. Los resultados obtenidos mostraron que las soluciones halladas por aprendizaje reforzado proporcionan un desempeño similar al obtenido con los controladores óptimos, sin requerir conocimiento del modelo linealizado del sistema.

Por lo tanto, se concluye que los algoritmos implementados en este trabajo permitieron comprobar que el aprendizaje reforzado es una solución viable en problemas de control donde se desconoce el modelo. Así mismo, se destaca la importancia de la correcta selección de una señal de recompensa, para mejorar el entrenamiento y obtener un controlador funcional.

5.1 Trabajo a futuro

El trabajo realizado en la presente tesis permitió identificar diversas áreas de oportunidad. Algunos de los problemas que se pueden abordar en un futuro, son:

- Explorar el uso de espacios de acciones continuos, para resolver el problema de una señal de control con comutaciones de alta frecuencia y oscilaciones en la respuesta del sistema, que estuvieron presentes en el control y la compensación de los esquemas PD.

- Abordar un modificación del algoritmo de aprendizaje reforzado utilizado para hallar la solución de un LQR, que permita reducir el tiempo que tarda en converger. De tal forma, que sea posible utilizarlo en línea para porblemas reales en donde las perturbaciones y la incertidumbre perjudican el desempeño de los controladores óptimos.

BIBLIOGRAFÍA

[1] F. L. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE Circuits and Systems Magazine*, vol. 9, no. 3, pp. 32–50, 2009.

[2] D. Vrabie, K. G. Vamvoudakis, and F. L. Lewis, *Optimal Adaptive Control and Differential Games by Reinforcement Learning Principles*. Control, Robotics and Sensors, Institution of Engineering and Technology, 2012.

[3] W. Yu and A. PerrusquÃa, *Human-Robot Interaction Control Using Reinforcement Learning*. Systems Science and Engineering, Wiley, 2021.

[4] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, 2nd ed., 2018.

[5] J. Moreno-Valenzuela and C. Aguilar-Avelar, *Motion Control of Underactuated Mechanical Systems*. Cham, Switzerland: Springer, 2018.

[6] I. Fantoni and R. Lozano, *Non-linear Control for Underactuated Mechanical Systems*. London: Springer, 2002.

[7] K. Furuta, M. Yamakita, and S. Kobayashi, "Swing up control of inverted pendulum," in *Proceedings IECON '91: 1991 International Conference on Industrial Electronics, Control and Instrumentation*, vol. 3, pp. 2193–2198, 1991.

[8] K. Furuta, M. Yamakita, and S. Kobayashi, "Swing-up control of inverted pendulum using pseudo-state feedback," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 206, no. 4, pp. 263–269, 1992.

[9] M. Yamakita, K. Furuta, K. Konohara, J. Hamada, and H. Kusano, "Vss adaptive control based on nonlinear model for titech pendulum," in *Proceedings of the 1992 International Conference on Industrial Electronics, Control, Instrumentation, and Automation*, vol. 3, pp. 1488–1493, 1992.

[10] L. Busoniu, R. Babuska, B. D. Schutter, and D. Ernst, *Reinforcement Learning and Dynamic Programming Using Function Approximators*. USA: CRC Press, Inc., 1st ed., 2010.

[11] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.

[12] D. P. Bertsekas and J.Ñ. Tsitsiklis, *Neuro-dynamic programming.*, vol. 3 of *Optimization and neural computation series*. Athena Scientific, 1996.

[13] M. F. Hamza, H. J. Yap, and I. A. Choudhury, "Genetic algorithm and particle swarm optimization based cascade interval type 2 fuzzy pd controller for rotary inverted pendulum system," *Mathematical Problems in Engineering*, vol. 2015, 2015.

[14] M. F. Hamza, H. J. Yap, and I. A. Choudhury, "Current development on using rotary inverted pendulum as a benchmark for testing linear and nonlinear control algorithms," *Mathematical Systems and Signal Processing*, vol. 116, pp. 347–369, 2019.

[15] S.-K. Oh, H.-J. Jang, and W. Pedrycz, "A comparative experimental study of type-1/type-2 fuzzy cascade controller based on genetic algorithms and particle swarm optimization," *Expert Systems with Applications*, vol. 38, no. 9, pp. 11217–11229, 2011.

[16] T. Chawaphan, M. M. Aung, and D. Maneetham, "Modelling of rotary inverted pendulum based on pid controller," *International Journal of Latest Engineering Research and Applications*, vol. 5, 2020.

[17] L. M. Adharsh, A. Kunjumuhammed, J. Tomy, U. G, M. Sivadas, and A. Mohan, "Stabilization of rotary inverted pendulum using pid controller," in *2021 8th International Conference on Smart Computing and Communications (ICSCC)*, pp. 376–380, 2021.

[18] V.Ñath and R. Mitra, "Swing-up and control of rotary inverted pendulum using pole placement with integrator," in *2014 Recent Advances in Engineering and Computational Sciences (RAECS)*, pp. 1–5, 2014.

[19] V. Sukontanakarn and M. Parnichkun, "Real-time optimal control for rotary inverted pendulum," *American Journal of Applied Sciences*, vol. 6, pp. 1106–1115, 2009.

[20] G. Sainzaya, F.-N. Yu, T.-L. Hsieh, and C.-Y. Yang, "Lqr control with refined pid to balance rotary inverted pendulum with time-varying uncertainty," in *2017 International Conference on Fuzzy Theory and Its Applications (iFUZZY)*, pp. 1–6, 2017.

[21] Y. Xu, M. Iwase, and K. Furuta, "Time optimal swing-up control of single pendulum," *Journal of Dynamic Systems, Measurement and Control*, vol. 123, pp. 518–527, 2001.

[22] I. Paredes, M. Sarzosa, M. Herrera, P. Leica, and O. Camacho, "Optimal-robust controller for furuta pendulum based on linear model," in *2017 IEEE Second Ecuador Technical Chapters Meeting (ETCM)*, pp. 1–6, 2017.

[23] A. Pandey and D. Adhyaru, "Robust control design for rotary inverted pendulum with unmatched uncertainty," *International Journal of Dynamics and Control*, 2022.

[24] B. Hassibi, A. H. Sayed, and T. Kailath, *Indefinite-Quadratic estimation and control: a unified approach to H 2 and Hâ ̂ž theories*. SIAM, 1999.

[25] A. Al-Jodah, H. Zargarzadeh, and M. K. Abbas, "Experimental verification and comparison of different stabilizing controllers for a rotary inverted pendulum," in *2013 IEEE International Conference on Control System, Computing and Engineering*, pp. 417–423, 2013.

[26] C.-H. Yu, F.-C. Wang, and Y.-J. Lu, "Robust control of a furuta pendulum," in *Proceedings of SICE Annual Conference 2010*, pp. 2559–2563, 2010.

[27] K. Ling, P. Falugi, J. Maciejowski, and L. Chisci, "Robust predictive control of the furuta pendulum," *IFAC Proceedings Volumes*, vol. 35, no. 1, pp. 37–42, 2002. 15th IFAC World Congress.

[28] S. D. Sanjeewa and M. Parnichkun, "Control of rotary double inverted pendulum system using mixed sensitivity h_∞ controller," *International Journal of Advanced Robotic Systems*, vol. 16, no. 2, p. 1729881419833273, 2019.

[29] L. Zhang and R. Dixon, "Robust nonminimal state feedback control for a furuta pendulum with parametric modeling errors," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 8, pp. 7341–7349, 2021.

[30] P. Faradja, G. Qi, and M. Tatchum, "Sliding mode control of a rotary inverted pendulum using higher order differential observer," in *2014 14th International Conference on Control, Automation and Systems (ICCAS 2014)*, pp. 1123–1127, 2014.

[31] S. Kurode, A. Chalanga, and B. Bandyopadhyay, "Swing-up and stabilization of rotary inverted pendulum using sliding modes," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 10685–10690, 2011. 18th IFAC World Congress.

[32] A. Wadi, J.-H. Lee, and L. Romdhane, "Nonlinear sliding mode control of the furuta pendulum," in *2018 11th International Symposium on Mechatronics and its Applications (ISMA)*, pp. 1–5, 2018.

[33] N. J. Mathew, K. K. Rao, and N. Sivakumaran, "Swing up and stabilization control of a rotary inverted pendulum," *IFAC Proceedings Volumes*, vol. 46, no. 32, pp. 654–659, 2013. 10th IFAC International Symposium on Dynamics and Control of Process Systems.

[34] D. Srivastava, S. Sharma, K. Patel, and A. Mehta, "Second order smc controller design for rotary inverted pendulum: An underactuated system," in *2018 2nd International Conference on Power, Energy and Environment: Towards Smart Technology (ICEPE)*, pp. 1–6, 2018.

[35] M.-S. Park and D. Chwa, "Swing-up and stabilization control of inverted-pendulum systems via coupled sliding-mode control method," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 9, pp. 3541–3555, 2009.

[36] C. Aguilar-Avelar and J. Moreno-Valenzuela, "A feedback linearization controller for trajectory tracking of the furuta pendulum," in *2014 American Control Conference*, pp. 4543–4548, 2014.

[37] C. Aguilar-Ibanez, O. F. Gutierrez, and H. Sossa-Azuela, "Control of the furuta pendulum by using a lyapunov function," in *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 6128–6132, 2006.

[38] M. Yamakita, M. Iwashiro, Y. Sugahara, and K. Furuta, "Robust swing up control of double pendulum," in *Proceedings of 1995 American Control Conference - ACC'95*, vol. 1, pp. 290–295, 1995.

[39] L. Ramos, J. Ruiz-León, and S. Čelikovský, "Robust regulation via sliding modes of a rotary inverted pendulum," *IFAC Proceedings Volumes*, vol. 33, no. 14, pp. 179–183, 2000. 3rd IFAC Symposium on Robust Control Design (ROCOND 2000), Prague, Czech Republic, 21-23 June 2000.

[40] P. Ordaz and A. S. Poznyak, "Adaptive-robust stabilization of the furuta's pendulum via attractive ellipsoid method," *Journal of Dynamic Systems, Measurement and Control*, vol. 138, 2016.

[41] G. Pujol-Vazquez, S. Mobayen, and L. Acho, "Robust control design to the furuta system under time delay measurement feedback and exogenous-based perturbation," *Mathematics*, vol. 8, no. 12, 2020.

[42] S.-E. Oltean, "Swing-up and stabilization of the rotational inverted pendulum using pd and fuzzy-pd controllers," *Procedia Technology*, vol. 12, pp. 57–64, 2014. The 7th International Conference Interdisciplinarity in Engineering, INTER-ENG 2013, 10-11 October 2013, Petru Maior University of Tirgu Mures, Romania.

[43] M. Rahmanian, M. Teshnehlab, and M. Aliyari Shoorehdeli, "An off-line fuzzy backstepping controller for rotary inverted pendulum system," in *2007 International Conference on Intelligent and Advanced Systems*, pp. 109–113, 2007.

[44] F. Ghorbani, M. Aliyari Shooredeli, and M. Teshnehlab, "Fault tolerant improvement with chaos synchronization using fuzzy-pid control," in *2013 13th Iranian Conference on Fuzzy Systems (IFSC)*, pp. 1–5, 2013.

[45] J.-H. Li, "Composite fuzzy control of a rotary inverted pendulum," in *2013 IEEE International Symposium on Industrial Electronics*, pp. 1–5, 2013.

[46] N. P. Nguyen, H. Oh, Y. Kim, J. Moon, J. Yang, and W.-H. Chen, "Fuzzy-based super-twisting sliding mode stabilization control for under-actuated rotary inverted pendulum systems," *IEEE Access*, vol. 8, pp. 185079–185092, 2020.

[47] I. Hassanzadeh and S. Mobayen, "Controller design for rotary inverted pendulum system using evolutionary algorithms," *Mathematical Problems in Engineering*, 2011.

[48] C. Alarcón and C. Muñoz, "Minimum time swing-up controller applied to a rotary inverted pendulum," in *2017 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, pp. 1–6, 2017.

[49] S. Zabihifar, A. Yushchenko, and H.Ñavvabi, "Robust control based on adaptive neural network for rotary inverted pendulum with oscillation compensation," *Neural Computing and Applications*, vol. 32, 2020.

[50] J. Moreno-Valenzuela, C. Aguilar-Avelar, S. A. Puga-Guzmán, and V. Santibáñez, "Adaptive neural network control for the trajectory tracking of the furuta pendulum," *IEEE Transactions on Cybernetics*, vol. 46, no. 12, pp. 3439–3452, 2016.

[51] A. de Carvalho, B. A. Angelico, J. F. Justo, A. M. de Oliveira, and J. I. da Silva Filho, "Model reference control by recurrent neural network built with paraconsistent neurons for trajectory tracking of a rotary inverted pendulum," *Applied Soft Computing*, vol. 133, p. 109927, 2023.

[52] H. V. Nghi, D. Phuoc-Nhien, N. T. Minh-Nguyet, N. Tu-Duc, N. P. Luu, P. Son-Thanh, L. T. Hong-Lam, and D. Xuan-Ba, "A lqr-based neural-network controller for fast stabilizing rotary inverted pendulum," in *2021 International Conference on System Science and Engineering (ICSSE)*, pp. 19–22, 2021.

[53] X. Yang and X. Zheng, "Swing-up and stabilization control design for an underactuated rotary inverted pendulum system: Theory and experiments," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 9, pp. 7229–7238, 2018.

[54] D. Guida, C. A. Manrique-Escobar, and C. M. Pappalardo, "A reinforcement learning controller for the swing-up of the furuta pendulum," in *New Technologies, Development and Application III. NT 2020. Lecture Notes in Networks and Systems*, vol. 128, pp. 31–37, 2020.

[55] B. Allotta, L. Pugi, and F. Bartolini, "Reinforcement neural network for the stabilization of a furuta pendulum," in *Proceedings of EUCOMES 08* (M. Ceccarelli, ed.), (Dordrecht), pp. 287–294, Springer Netherlands, 2009.

[56] Y. Dai, K. Lee, and S. Lee, "A real-time hil control system on rotary inverted pendulum hardware platform based on double deep q-network," *Measurement and Control*, vol. 54, no. 3–4, pp. 417–428, 2021.

[57] A. Ghosh, T. Krishnan, and B. Subudhi, "Robust proportional−integral−derivative compensation of an inverted cart−pendulum system: an experimental study," *IET Control Theory and Applications*, vol. 6, pp. 1145–1152, 2012.

[58] K. Åström and K. Furuta, "Swinging up a pendulum by energy control," *IFAC Proceedings Volumes*, vol. 29, no. 1, pp. 1919–1924, 1996.

[59] G. Puriel-Gil, W. Yu, and H. Sossa, "Reinforcement learning compensation based pd control for inverted pendulum," in *2018 15th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, pp. 1–6, 2018.

[60] W. Yu, "Nonlinear pd regulation for ball and beam system," *The International Journal of Electrical Engineering and Education*, vol. 46, no. 1, pp. 59–73, 2009.

[61] S. Galvan-Colmenares, M. A. Moreno-Armendáriz, W. Yu, and F. Ortiz-Rodriguez, "Modeling and nonlinear pd regulation for ball and plate system," in *World Automation Congress 2012*, pp. 1–6, 2012.

[62] X. Li and W. Yu, "Synchronization of ball and beam systems with neural compensation," *International Journal of Control, Automation and Systems*, vol. 8, pp. 491–496, 2010.

[63] Z. Hendzel, A. Burghardt, and M. Szuster, "Reinforcement learning in discrete neural control of the underactuated system," in *Artificial Intelligence and Soft Computing*, vol. 7894, pp. 64–75, Springer Berlin Heidelberg, 06 2013.

[64] B. Cazzolato and Z. Prime, "On the dynamics of the furuta pendulum," *Journal of Control Science and Engineering*, vol. 2011, pp. 1–8, 2011.

[65] M. W. Spong and M. Vidyasagar, *Robot dynamics and control*. USA: Wiley, 1989.

[66] H. K. Khalil, *Nonlinear systems*. USA: Prentice Hall, 3 ed., 2002.

[67] F. L. Lewis, D. Vrabie, and K. G. Vamvoudakis, "Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers," *IEEE Control Systems Magazine*, vol. 32, no. 6, pp. 76–105, 2012.

[68] C. J. C. H. Watkins, *Learning from Delayed Rewards*. PhD thesis, King's College, Oxford, 1989.

[69] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, 1992.

[70] J.Ñ. Tsitsiklis, "Asynchronous stochastic approximation and Q-learning," *Machine Learning*, vol. 16, no. 3, pp. 185–202, 1994.

[71] D. P. Bertsekas, *Reinforcement Learning and Optimal Control*. Optimization and computation series, Athena Scientific, 2019.

[72] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik,

I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[73] L.-J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Machine Learning*, vol. 8, pp. 293–321, 1992.

[74] L.-J. Lin, *Reinforcement Learning for Robots Using Neural Networks*. PhD thesis, Carnegie Mellon University, USA, 1992. UMI Order No. GAX93-22750.

[75] C. F. Touzet, "Neural reinforcement learning for behaviour synthesis," *Robotics and Autonomous Systems*, vol. 22, no. 3, pp. 251–281, 1997. Robot Learning: The New Wave.

[76] Quanser Inc., *SRV02 Rotary Pendulum User Manual*, 2010.

[77] Quanser Inc., *SRV02 User Manual*, 2009.

[78] E. Even-Dar and Y. Mansour, "Learning rates for q-learning," *Journal of Machine Learning Research*, vol. 5, pp. 1–25, 2003.

[79] S. J. Bradtke, *Incremental Dynamic Programming for On-Line Adaptive Optimal Control*. PhD thesis, University of Massachusetts, USA, 1995. UMI Order No. GAX95-10446.