

Instituto Tecnológico de Costa Rica

Área Académica de Ingeniería Mecatrónica



Diseño de un sistema de navegación avanzado para el nanosatélite GWSat

**Informe de Proyecto de Graduación para optar por el título de Ingeniero en
Mecatrónica con el grado académico de Licenciatura**

Gabriel Francisco Alba Romero

Cartago, agosto de 2020

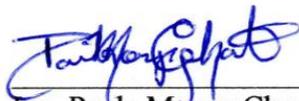
This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc/4.0/> or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

INSTITUTO TECNOLÓGICO DE COSTA RICA
CARRERA DE INGENIERÍA MECATRÓNICA
PROYECTO DE GRADUACIÓN
ACTA DE APROBACIÓN

El Profesor Asesor, da fe de que el presente Proyecto de Graduación ha sido aprobado y cumple con las normas establecidas por la Carrera de Ingeniería Mecatrónica como requisito para optar por el título de Ingeniero en Mecatrónica, con el grado académico de Licenciatura, del Instituto Tecnológico de Costa Rica.

Estudiante: Gabriel Francisco Alba Romero

Nombre del Proyecto: Diseño de un sistema de navegación avanzado para el nanosatélite GWSat



Ing. Paula Monge Chanto

Profesor Asesor

Cartago, lunes 10 de agosto del 2020.

Declaratoria de Autenticidad

Declaro que el presente Proyecto de Graduación ha sido realizado enteramente por mi persona, utilizando y aplicando literatura referente al tema e introduciendo conocimientos propios.

En los casos en que he utilizado bibliografía, he procedido a indicar las fuentes mediante las respectivas citas bibliográficas.

En consecuencia, asumo la responsabilidad total por el trabajo de graduación realizado y por el contenido del correspondiente informe final.

A handwritten signature in dark ink, appearing to read 'Gabriel Alba Romero', with a large, sweeping flourish at the end.

Cartago, Julio 2020

Gabriel Alba Romero

Céd: 1-1684-0897

Resumen

Este documento es un informe para optar por el título de licenciado en ingeniería en mecatrónica. En este se presenta la metodología para el diseño de un controlador clásico y un controlador inteligente para el nanosatélite GWSat, esto con el objetivo de en un futuro diseñar un sistema híbrido que integre ambas tecnologías. El primer controlador que se diseñó es un controlador LQR. El segundo se diseñó por medio del algoritmo de entrenamiento TD3, el cual se basa en aprendizaje por refuerzo profundo para su optimización. En el caso del controlador LQR se logró un tiempo de estabilización de 950s, un ángulo de error de 0.15° y un error en la velocidad angular de 3×10^{-5} rad/s. Para el caso del TD3 se obtuvo un tiempo de estabilización de 700 s, un ángulo de error de 1.4° y un error en la velocidad angular de 7.3×10^{-6} rad/s. Los aportes principales de este proyecto son dos controladores implementados en MATLAB los cuales son fáciles de utilizar y modificar para cualquier caso que se quiera probar, además de ambientes programados en los que se pueden ajustar los diferentes parámetros de los controladores en caso de que se desee un desempeño distinto.

Palabras clave: sistema de control de orientación, controlador LQR, aprendizaje por refuerzo profundo, TD3, nanosatélite

Abstract

This document is a report to opt for a bachelor's degree in mechatronics engineering. This presents the methodology for the design of a classic controller and an intelligent controller for the GWSat nanosatellite, with the aim of designing a hybrid system that integrates both technologies, in the foreseeable future. The first controller that was designed is an LQR controller. The second was designed using the TD3 training algorithm, which is based on deep reinforcement learning for optimization. The LQR controller achieved a stabilization time of 950s, an error angle of 0.15° and an error in the angular velocity of 3×10^{-5} rad/s. For the TD3 case, it achieved a 700 s stabilization time, an 1.4° error angle, and an 7.3×10^{-6} rad/s angular velocity error. The main contributions of this project are two controllers implemented in MATLAB which are easy to use and modify for any case you want to test, also a series of programmed environments where you can modify the controllers' different parameters in order to obtain a different performance.

Keywords: attitude control system, LQR controller, deep reinforcement learning, TD3, nanosatellite

*A mis padres, y a todos
aquellos apasionados por
el espacio...*

Agradecimientos

En primer lugar, le agradezco a mis padres por su apoyo incondicional y su cariño a lo largo de toda mi vida. A mi hermana por siempre hacerme reír y olvidar de los malos momentos, y a mi tío por siempre haber estado ahí cuando lo necesité.

Le agradezco al profesor Adolfo Chávez por la oportunidad que me brindó ofreciéndome este proyecto y por permitirme desarrollarlo en el SETECLab. Gracias por el apoyo y la motivación brindada.

A la profesora Paula Monge le agradezco haber sido mi guía a la hora de presentar este proyecto. También, gracias por la disposición a ayudarme y orientarme en cualquier momento.

Le agradezco al Área Académica de Ingeniería en Mecatrónica por todos los conocimientos que me impartieron y me permitieron llegar a este momento.

Le agradezco profundamente a mis amigos y colegas, porque sin su compañía no habría llegado hasta este punto. Rubén, Isaac, Heiner, Ana, Carlos, Luis, Sebas y Liche gracias por todas las experiencias vividas, les deseo lo mejor en esta siguiente etapa.

A Erika le agradezco todo su cariño y apoyo en todo momento, por siempre hacer mejores mis días y por siempre motivarme a dar lo mejor de mí.

Finalmente, a Caramelo, porque sin importar que pase siempre me hacés feliz.

Sinceramente,

Gabriel Alba R.

Cartago, 15 de julio 2020

ÍNDICE GENERAL

1	Introducción	1
1.1	Entorno del proyecto.....	1
1.2	Definición del problema	2
1.2.1	Generalidades	2
1.2.2	Justificación.....	3
1.2.3	Síntesis del problema.....	4
1.3	Objetivos.....	4
2	GW-Sat.....	5
2.1	Subsistemas.....	5
2.1.1	Sistema de potencia eléctrica.....	5
2.1.2	Sistema de control térmico	6
2.1.3	Sistema de comunicaciones	7
2.1.4	Sistema de propulsión y carga útil.....	7
2.2	Sistema de control y determinación de orientación (ADCS).....	7
3	Marco teórico	9
3.1	CubeSats	9
3.1.1	Contexto Histórico.....	9
3.1.2	Especificaciones técnicas	9
3.2	Fundamentos astronómicos.....	11
3.2.1	Marcos de referencia	11
3.2.2	Terminología Orbital	13
3.2.3	Cinemática Rotacional.....	15
3.2.4	Dinámica rotacional.....	21
3.2.5	Perturbaciones	22
3.2.6	Importancia del modelado correcto	25
3.3	Sensores	26
3.3.1	Sensores de Sol.....	26

3.3.2	GPS.....	26
3.3.3	Star Tracker	27
3.3.4	Giroscopio de 3 ejes	27
3.3.5	Magnetómetro de 3 ejes.....	29
3.4	Actuadores	29
3.4.1	Magnetorquers.....	29
3.4.2	Propulsores μ CAT	30
3.5	Algoritmos de inteligencia artificial	30
3.5.1	Redes neuronales artificiales	31
3.5.2	Aprendizaje de la neurona artificial.....	33
3.6	Aprendizaje por refuerzo	35
3.6.1	Métodos de gradiente de política.....	38
3.6.2	Redes actor-crítico.....	38
3.6.3	Exploración vs explotación	40
3.6.4	Aprendizaje profundo por refuerzo	40
3.6.5	DDPG	40
3.6.6	TD3.....	43
3.7	Sistemas de control automático	46
3.7.1	Controladores Clásicos.....	46
3.7.2	PID.....	46
3.7.3	Realimentación de Estados.....	46
3.7.4	Controladores por medio de IA	47
3.7.5	Redes Neuronales	47
3.8	Medidas de desempeño y eficiencia	48
3.9	Estado del arte de ADCS´	49
3.9.1	Controladores Clásicos.....	49
3.9.2	Controladores con IA.....	49
4	Metodología	51
4.1	Determinación de los métodos de control.....	52
4.2	Diseño de los algoritmos de control.....	53
4.3	Comparación de los métodos de control.....	55

5	Resultados y análisis	57
5.1	Antecedentes	58
5.2	Determinación de los métodos de control.....	64
5.2.1	Propuestas generadas para el control clásico.....	64
5.2.2	Parámetros de evaluación de los controladores clásicos	65
5.2.3	Selección de candidatos del controlador clásico	65
5.2.4	Propuestas generadas para el control inteligente.....	68
5.2.5	Parámetros de evaluación de los controladores inteligentes	68
5.2.6	Selección de candidatos del controlador inteligente.....	70
5.3	Controlador LQR.....	72
5.3.1	Estudio de las características del satélite.....	73
5.3.2	Definición de las características de desempeño	74
5.3.3	Desarrollo del algoritmo de control.....	76
5.3.4	Validación del algoritmo	81
5.4	Controlador por medio de inteligencia artificial.....	87
5.4.1	Estudio de las características del satélite.....	87
5.4.2	Definición de los objetivos de desempeño	88
5.4.3	Determinación de los parámetros y arquitectura inicial de la red neuronal ...	88
5.4.4	Entrenamiento de la red neuronal.....	90
5.4.5	Validación del algoritmo	110
5.5	Comparación entre controladores	117
5.5.1	Parámetros de comparación.....	117
5.5.2	Comparación desempeño con respecto a referencia de GWU	118
5.5.3	Comparación de la utilización del torque de los controladores.....	119
6	Conclusiones	125
7	Recomendaciones.....	126
8	Bibliografía.....	127
	Apéndices	131
A.1	Tiempos de ejecución de los algoritmos de control.....	131
A.2	Iteraciones de diseño LQR.....	132
A.3	Iteraciones del diseño del controlador inteligente	134

A.4	Mediciones tiempo de ejecución por modelo	136
A.5	Mediciones tiempo de ejecución por arquitectura	137
A.6	Resultados prueba sin aleatoriedad.....	138
A.7	Diagramas de bloques del sistema con el controlador LQR	142
A.8	Código del Regulador LQR	143
A.9	Código error de los cuaternios y velocidad angular.....	144
A.10	Código para la determinación de la matriz de ganancias	145
A.11	Diagramas de bloques del sistema del controlador inteligente	147
A.12	Código para el entrenamiento de la red neuronal por medio del algoritmo TD3 150	
A.13	Código para la función de reset aleatorio del ambiente	153
Anexos	154
A.1	Teorema del Transporte	154

ÍNDICE DE FIGURAS

Figura 2-1: Bloques del subsistema de potencia eléctrica.....	6
Figura 2-2: Sistema ADCS con los sensores y actuadores.....	8
Figura 3-1: Dimensiones estándar de CubeSats.....	10
Figura 3-2: Marco de referencia ECI.....	12
Figura 3-3: Marco de Referencia B.....	12
Figura 3-4: Parámetros orbitales.....	14
Figura 3-5: Torque de arrastre atmosférico.....	23
Figura 3-6: Ilustración sobre el efecto atmosférico sobre la trayectoria de un satélite sin propulsión en una LEO.....	24
Figura 3-7: Torque de gradiente gravitacional.....	25
Figura 3-8: Giroscopio flotante de un eje.....	28
Figura 3-9: Diseños esquemáticos de los propulsores μ CAT.....	30
Figura 3-10: Esquema de una neurona biológica.....	31
Figura 3-11: Modelo de una neurona artificial.....	32
Figura 3-12: Representación de una red neuronal artificial típica.....	33
Figura 3-13: Mínimo local vs mínimo global.....	34
Figura 3-14: Olvido Catastrófico en la recompensa de aprendizaje por refuerzo.....	35
Figura 3-15: Problema de aprendizaje por refuerzo.....	36
Figura 3-16: Arquitectura actor-crítico.....	39
Figura 3-17: Algoritmo actor-crítico de DDPG.....	42
Figura 3-18: Parámetros de desempeño de un sistema.....	48
Figura 4-1: Etapas del desarrollo de un ADCS para el GWSat.....	51
Figura 4-2: Relación entre los objetivos específicos.....	52
Figura 4-3: Diagrama de flujo del objetivo específico 1.....	53
Figura 4-4: Diagrama de flujo del objetivo específico 2 y 3.....	55
Figura 4-5: Diagrama de flujo del objetivo específico 4.....	56
Figura 5-1: Error de los cuaternios del controlador PD. Matlab 2020a.....	58
Figura 5-2: Error de la velocidad angular del controlador PD. Matlab 2020a.....	60
Figura 5-3: Ángulo de error del controlador PD. Matlab 2020a.....	61
Figura 5-4: Torque generado por el controlador PD. Matlab 2020a.....	62
Figura 5-5: Torque acumulado del controlador PD. Matlab 2020a.....	63
Figura 5-6: Error del cuaternio del controlador LQR.....	82
Figura 5-7: Error de la velocidad angular del controlador LQR.....	83
Figura 5-8: Ángulo de error del controlador LQR.....	84
Figura 5-9: Torque generado por el controlador LQR.....	85
Figura 5-10: Torque acumulado del control LQR.....	86

Figura 5-11: Forma general de la primera función de recompensa	90
Figura 5-12: Gráfica de Recompensa de la Iteración 1	91
Figura 5-13: Forma general de la segunda función de recompensa	93
Figura 5-14: Recompensa de la tercera iteración	94
Figura 5-15: Comportamiento divergente del error de los cuaternios	95
Figura 5-16: Comportamiento divergente del error de la velocidad angular	95
Figura 5-17: Gráfica de Recompensa de la Iteración 4	98
Figura 5-18: Forma general de la tercera iteración de la recompensa	100
Figura 5-19: Comportamiento convergente de los cuaternios (arriba) y velocidad angular (abajo)	101
Figura 5-20: Error de los cuaternios de la séptima iteración	105
Figura 5-21: Error de la velocidad angular de la séptima iteración	105
Figura 5-22: Olvido Catastrófico en la recompensa de la séptima iteración	106
Figura 5-23: Recompensa del algoritmo TD3 escogido	108
Figura 5-24: Error de los cuaternios del controlador TD3.	110
Figura 5-25: Error de la velocidad angular del controlador TD3.	111
Figura 5-26: Ángulo de error del controlador TD3.	113
Figura 5-27: Torque generado por el controlador TD3.	114
Figura 5-28: Torque acumulado del control TD3	115
Figura 5-29: Torque utilizado por los propulsores con el controlador PD	120
Figura 5-30: Torque utilizado por los propulsores con el controlador LQR	121
Figura 5-31: Torque utilizado por los propulsores con el controlador TD3	122

ÍNDICE DE TABLAS

Tabla 5-1: Tabla resumen del error de los cuaternios para el control PD.	59
Tabla 5-2: Tabla resumen del error de la velocidad angular para el control PD.	61
Tabla 5-3: Resumen del error del ángulo de puntería para el control PD.	62
Tabla 5-4: Tabla resumen del torque ejercido para el control PD.	64
Tabla 5-5: Tabla morfológica de los candidatos de controladores clásicos.	66
Tabla 5-6: Tiempos de ejecución controladores.	67
Tabla 5-7: Tabla morfológica de los candidatos de controladores por IA.	70
Tabla 5-8: Parámetros de desempeño para diferentes modos de operación.	75
Tabla 5-9: Parámetros de desempeño control PD.	75
Tabla 5-10: Resultados primera iteración LQR.	77
Tabla 5-11: Resultados quinta iteración LQR.	78
Tabla 5-12: Características candidato ideal.	79
Tabla 5-13: Resultados controlador LQR optimizado.	81
Tabla 5-14: Tabla resumen del error de los cuaternios para el control LQR.	82
Tabla 5-15: Tabla resumen del error de la velocidad angular para el control LQR.	84
Tabla 5-16: Tabla resumen del error del ángulo de puntería para el control LQR.	85
Tabla 5-17: Tabla resumen del torque ejercido para el control LQR.	87
Tabla 5-18: Híper-Parámetros de la primera iteración del controlador inteligente Fuente: Elaboración propia.	89
Tabla 5-19: Híper-Parámetros de la cuarta iteración del controlador inteligente Fuente: Elaboración propia.	96
Tabla 5-20: Tiempos de ejecución para los diferentes modelos de simulación	102
Tabla 5-21: Tiempos de ejecución para las diferentes arquitecturas.	103
Tabla 5-22: Ámbitos de los cuaternios y torque para los distintos modelos	103
Tabla 5-23: Puntos de cambio de signo entre la recompensa anterior y actual.	104
Tabla 5-24: Parámetros de la red neuronal elegida	107
Tabla 5-25: Tabla resumen del desempeño de la iteración no aleatorizada.	109
Tabla 5-26: Tabla resumen del error de los cuaternios para el control TD3.	111
Tabla 5-27: Tabla resumen del error de la velocidad angular para el control TD3.	112
Tabla 5-28: Tabla resumen del error del ángulo de puntería para el control TD3.	114
Tabla 5-29: Tabla resumen del torque ejercido para el control TD3.	116
Tabla 5-30: Parámetros de desempeño del controlador TD3	116
Tabla 5-31: Parámetros del índice de rendimiento.	117
Tabla 5-32: Índice de rendimiento de los distintos controladores.	118
Tabla 5-33: Cumplimiento de los requisitos por parte de los controladores diseñados	119
Tabla 5-34: Tabla resumen del torque ejercido por los propulsores para el control PD. ...	120
Tabla 5-34: Tabla resumen del torque ejercido por los propulsores para el control LQR. ...	121
Tabla 5-35: Tabla resumen del torque ejercido por los propulsores para el control TD3. ...	122

Tabla 5-37: Tabla resumen del torque ejercido por los propulsores para los distintos controladores.	123
Tabla 5-36: Controlador elegido por modo de operación.....	124

LISTA DE ACRÓNIMOS

μCAT	Propulsores de Arco de Micro-Cátodo
ACS	Sistema de Control de Orientación
ADCS	Sistema de Determinación y Control de Orientación
AN	Neurona Artificial
ANN	Red Neuronal Artificial
BN	Neurona Biológica
CRAC	Cluster Aeroespacial de Costa Rica
DCM	Matriz de Coseno Directo
DDPG	Política de Gradiente Determinístico-Profunda
DPG	Política de Gradiente Determinístico
DQN	Red Profunda Q
ECEF	Fijado en la Tierra, Centrado en la Tierra
ECI	Inercial Centrado en la Tierra
GPS	Sistema de Posicionamiento Global
GWU	Universidad George Washington
IA	Inteligencia Artificial
LEO	Órbita Baja Terrestre
LQG	Regulador Cuadrático Gaussiano
LQR	Regulador Cuadrático Lineal
MTQ	Magnetorquer
OBC	Computadora a Bordo
PD	Proporcional, Derivativo
PID	Proporcional, Integral, Derivativo
PWM	Modulación de Ancho de Pulso
PWPF	Frecuencia de Pulso, Ancho de Pulso
RAAN	Ascenso Derecho del Nodo de Ascenso
RE	Realimentación de Estados
RL	Aprendizaje por Refuerzo

SSO	Órbita Semi Sincrónica
TD	Diferencia Temporal
TD3	Política Gemela de Gradiente Determinístico-Profunda con Atraso
USNA	United States Naval Academy
UV	Ultravioleta

1 Introducción

1.1 Entorno del proyecto

El 8 de marzo del 2016 se anunció la creación de un consorcio con el cual se pretendía posicionar a Costa Rica en la industria del sector aeroespacial. Este consorcio se llama el *Costa Rica Aerospace Cluster* (CRAC) el cual en el momento de su formación contaba con 25 empresas tanto del área electrónica y metalmecánica, así como de servicios especializados. De acuerdo con datos del 2015 de Procomer, en ese año se exportaron \$1557 millones solamente en el área de la industria aeroespacial. Además de las empresas asociadas, CINDE, INA, UCER, ITCR, Inteco, DGAAC, ACAE y CICR apoyaron la iniciativa. [1]

Por lo anterior en el 2017 nace el Laboratorio de Sistemas Espaciales (SETEC Lab) del ITCR en el contexto del desarrollo de la primera misión espacial de Costa Rica, de la creación del CRAC y de las capacidades que se generan en el ITCR, de manera que se pueda apoyar al campo de ingeniería espacial, y que de esta manera se convierta en una herramienta de desarrollo en el país. [2]

En mayo del 2018 se cumplió un hito histórico en el contexto de la historia aeroespacial de Costa Rica. El primer satélite construido en el país fue puesto en órbita. A este proyecto se le llamó el proyecto Irazú el cual se basó en lanzar al espacio un satélite de tipo CubeSat (un cubo de 10 cm de lado y 1 kilogramo) para monitorizar la fijación de carbono en árboles costarricenses en el área de San Carlos. [3]

De esta forma, en noviembre del 2018 el TEC anunció su asociación a la Universidad George Washington (GWU) para participar en una nueva misión espacial, el satélite GWSat. En este caso, el desarrollo de un satélite de tipo CubeSat 3U, parecido al del proyecto Irazú, pero con tres módulos en vez de uno solo. En este caso el ITCR aportará la misión científica y el sistema de control de navegación del dispositivo. [4]

Esta misión abre las puertas de la colaboración internacional entre Costa Rica y el mundo en materia aeroespacial. Con el potencial de participar cada vez en misiones más

complejas e importantes. Además de demostrar el talento costarricense en esta área tan importante para el futuro.

Además, paralelamente a estos eventos, la Industria 4.0 también ha comenzado a tener un impacto en cómo se desarrollan nuevas tecnologías, creando nuevos paradigmas, así como encontrando soluciones innovadoras a problema con estos nuevos paradigmas. La Industria 4.0 se compone de las siguientes tecnologías principalmente: robótica, inteligencia artificial (IA), nanotecnología, computación cuántica, biotecnología, internet de las cosas, 5G, e impresión 3D [5].

Con estas nuevas tecnologías la industria espacial cambiará para siempre, ofreciendo soluciones más eficientes y de mejor calidad a un costo menor. En el caso de controladores para satélites esto significa controladores más robustos y eficientes.

Esta revolución industrial además de los cambios que trae tecnológicamente, también creará una necesidad cada vez mayor por profesionales que en vez de ser altamente especializados en una sola área, tengan conocimiento técnico en muchas áreas. En este sentido, los ingenieros mecatrónicos se perfilan como profesionales esenciales para la cuarta revolución industrial.

1.2 Definición del problema

1.2.1 Generalidades

El TEC en asociación con la GWU está diseñando la misión científica y el sistema de control de navegación de GWSat, un satélite que se utilizará para monitorizar las condiciones ambientales en los humedales del país, puntualmente en el de Palo Verde en la provincia de Guanacaste. La órbita de este satélite estará a una altitud aproximada de 400 km.

Los satélites necesitan de sistemas de control avanzado para poder realizar las diferentes tareas que tienen que realizar. Una de estas tareas es mantenerse en órbita. Este es un problema que es inevitable en satélites en la zona llamada “órbita baja terrestre” (LEO, en inglés), una zona entre 200 y 2000 km de altura. A estas altitudes la atmósfera todavía es suficientemente densa para frenar constantemente los satélites. Por esta razón, los satélites

en estas órbitas tienen que ser impulsados periódicamente de manera que se puedan mantener en la órbita deseada.

También, una de las tareas importantes que tiene que realizar un satélite es orientarse de manera correcta de modo que reciba la información transmitida por diferentes equipos en la Tierra.

Otro sistema importante en un satélite es la parte de estimación del vector de velocidad y orientación del sistema. En este caso esto se hace por medio de datos recolectados con respecto a la posición del sol, posición GPS del satélite y posición del polo norte magnético de la Tierra.

1.2.2 *Justificación*

El GWSat hace uso de propulsores para realizar las maniobras de orientación relacionadas con la misión científica. Este propelente es limitado e independiente en cada uno de los propulsores, por lo que, para maximizar el uso de este, es importante asegurarse que se agoten de manera uniforme entre todos.

La importancia del ajuste de la orientación del satélite se centra en la misión científica del propio satélite. Éste se tiene que orientar correctamente de manera que reciba la información transmitida por los sensores instalados en Palo Verde. En caso de que la orientación no sea la deseada se puede perder información valiosa.

En el campo de la tecnología espacial también se ha visto como se ha utilizado en diversas ocasiones algoritmos de inteligencia artificial para los sistemas de control de orientación. Estos se desarrollan pues los satélites presentan no-linealidades inherentes dadas las maniobras, incertidumbres y perturbaciones desconocidas a las que están expuestos. Aunque sistemas de control clásicos se han implementado en satélites, las redes neuronales son una herramienta prometedora para este tipo de aplicaciones pues son capaces de aproximar cualquier función no lineal a cualquier nivel de precisión. [6]

De esta manera, el diseño de sistemas de control tanto robustos como eficientes es de alta importancia para estos sistemas de manera que puedan cumplir de la mejor manera con

la misión, y al mismo tiempo maximizar la vida útil del satélite. Esto importa, pues a mayor vida útil, se podrá obtener una cantidad mayor de información, y por lo tanto la relación costo/beneficio de la misión será mejor.

1.2.3 *Síntesis del problema*

Se requiere maximizar la eficiencia y robustez del sistema de control de orientación del nanosatélite GW-Sat con el fin de alargar la vida útil de este.

1.3 Objetivos

El objetivo general que se plantea para este proyecto es diseñar un sistema de orientación avanzado y eficiente para el nanosatélite GW-Sat. Los objetivos específicos para lograr se detallan a continuación:

- Determinar al menos dos métodos de control avanzado para controlar la orientación del satélite, un control clásico, y otro que utilice algoritmos de inteligencia artificial
- Diseñar un método de control realimentado, para ajustar la orientación del satélite, por medio de métodos de control clásico
- Diseñar un método de control realimentado, para ajustar la orientación del satélite, por medio del uso de algoritmos de inteligencia artificial.
- Comparar los dos métodos de control, en búsqueda del algoritmo más robusto y eficiente en el uso de propelente.

2 GW-Sat

En este capítulo se describirán las principales características del nanosatélite GW-Sat. Primeramente, se describirán los principales subsistemas, seguidamente se describirá el ADCS. Esto se hace para proveer el contexto necesario para familiarizarse con el dispositivo que se está trabajando.

2.1 Subsistemas

En esta sección se presenta un resumen de los subsistemas que componen el nanosatélite GW-Sat

2.1.1 Sistema de potencia eléctrica

Este subsistema es el que se encarga de distribuir la potencia eléctrica para todos los componentes del satélite de manera que se pueda asegurar su desempeño correcto. Está compuesto de los componentes [7] :

- Arreglo de paneles solares DHV: los paneles solares que se utilizan tienen también características notables, como lo son sensores de temperatura y UV, así como un *magnetorquer* dentro del substrato del panel.
- NanoPower P60 Dock de GOMSpace: Fuente de poder modular que permite múltiples configuraciones. Acepta hasta 4 módulos, de los cuales en este diseño se utilizan un ACU, PDU y una computadora a bordo.
- NanoPower P60 ACU-200 de GOMSpace: funciona como el convertidor de potencia de los paneles solares hacia las baterías y la fuente de poder.
- NanoPower P60 PDU-200 de GOMSpace: provee nueve canales de salida con salidas configurables en tensiones de 3.3 V, 5 V, 8 V, 12 V, 18 V, y 24 V.
- NanoPower BPX de GOMSpace: batería de ion de litio de alta capacidad que proveen 29.6 V nominal y una capacidad de 77 Wh con una masa de ~0.5 kg

Esto se resume en la Figura 2-1.

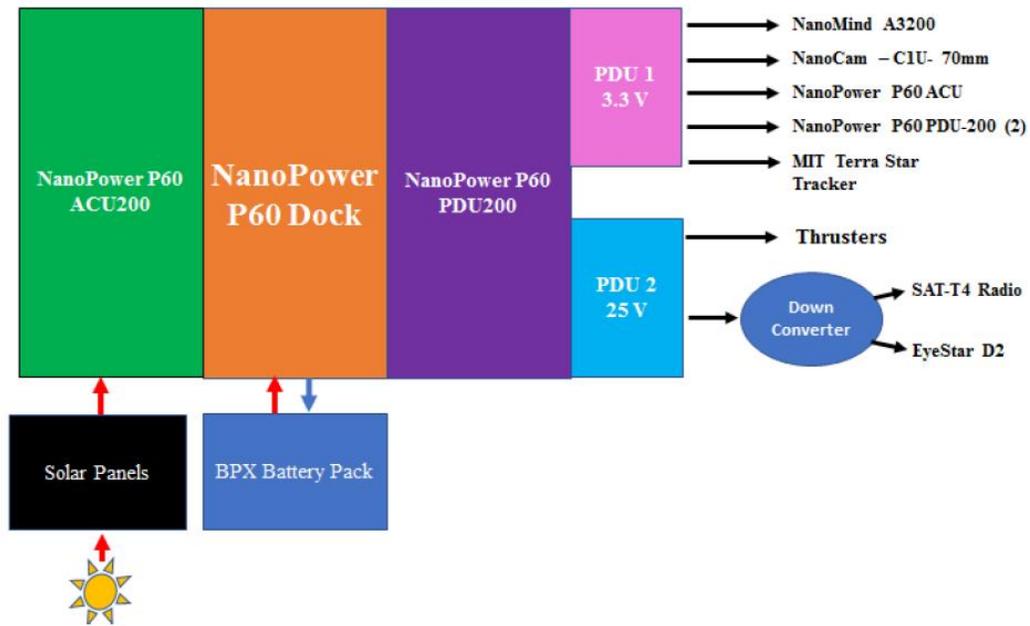


Figura 2-1: Bloques del subsistema de potencia eléctrica

Fuente: [7]

2.1.2 Sistema de control térmico

Los sistemas de control térmico son fundamentales para asegurar el funcionamiento y supervivencia de los CubeSats. Dado que en el espacio tanto las temperaturas como la radiación son extremas. Los problemas que resuelve este sistema se pueden categorizar en 4 diferentes aspectos, los cuales se listan a continuación. [7]

1. Interacciones con el ambiente espacial
2. Recolección de calor: identificación de las áreas donde el calor se está acumulando
3. Rechazo de calor: remoción de calor excesivo dentro del sistema
4. Almacenamiento y suministro de calor: mantener la temperatura dentro del CubeSat a un nivel apropiado, previniendo que los componentes se sobrecalienten o se congelen, al fin y al cabo, protegiéndolos de una posible falla.

2.1.3 Sistema de comunicaciones

Este subsistema consiste en tres componentes principales. [7]

1. SATT4 VHF radio de USNA: Este se utiliza para la comunicación directa entre el satélite y la estación de tierra de la USNA, la estación de tierra de Costa Rica y la antena de VHF en el campus de GWU.
2. GlobalStar Duplex Radio de EyeStar-D2: esta se utiliza como vínculo global para tener acceso las 24 horas del día al satélite. Además, se utiliza como la mayor fuente de transferencia de imágenes dada su mayor velocidad en comparación a la antena VHF.
3. NanoCom TR-600 *Software Defined Radio* de GOMSpace: esta se eligió como respaldo para el EyeStar-D2, de manera que, si fallara, esta es capaz de transmitir imágenes.

2.1.4 Sistema de propulsión y carga útil

La carga útil de este satélite se compone de una cámara a bordo, esto para aprovechar el sistema de estabilización en 3 ejes y también para verificar la precisión de los propulsores. La cámara que se eligió es la NanoCam de GOMSpace.

Para el sistema de propulsión se utilizan los propulsores μ CAT desarrollados por la propia GWU. [7]

2.2 Sistema de control y determinación de orientación (ADCS)

Este sistema utiliza la A3200 OBC de GOMSpace como la computadora de vuelo que se encarga de los cálculos y de la ejecución de los algoritmos de control.

Esta computadora incluye capacidades básicas para un ADCS por medio de los siguientes sistemas:

- Magnetómetros de 3 ejes

- Giroscopios de 3 ejes

Sin embargo, estos sensores no proveen suficiente información para la determinación de la orientación del satélite, pues solamente permiten medir rotaciones relativas y no absolutas. Es importante recordar que esta es una medición fundamental para cumplir con el objetivo principal de la misión, la estabilización en 3 ejes.

Para poder determinar la orientación y poder mejorar la medición de la velocidad de rotación se incorporaron los siguientes sensores:

- Sensores UV
- GPS
- *Star Tracker*

Además, dos tipos de actuadores se utilizan para cumplir con los requerimientos de apuntar. [7]

- *Magnetorquers*
- Propulsores

Los sensores y actuadores se pueden ver resumidos en la Figura 2-2. Estos se describirán con mayor detalle en la sección 3.3 y 3.4.

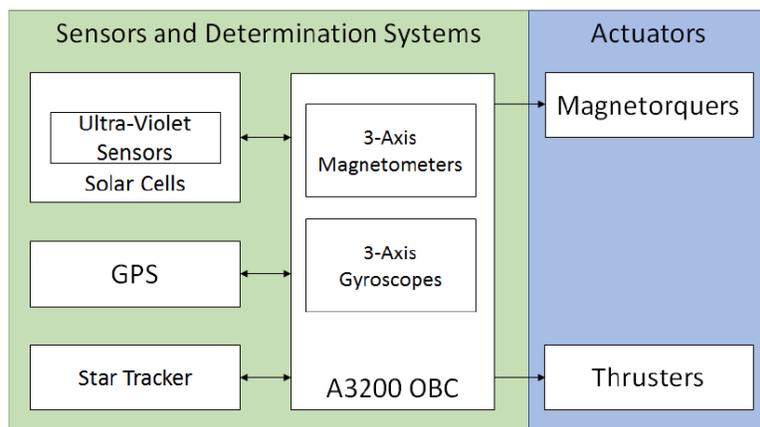


Figura 2-2: Sistema ADCS con los sensores y actuadores

Fuente: [7]

3 Marco teórico

3.1 CubeSats

En esta sección del capítulo se presenta el contexto histórico de los CubeSats, así como sus especificaciones técnicas y sus sistemas dispensadores. Esto se hace para poder entender el dispositivo con el que se estará trabajando de la mejor manera durante el desarrollo del proyecto.

3.1.1 *Contexto Histórico*

Cuando se conoce el contexto histórico del sistema con el que se está trabajando, se puede entender de mejor manera este. En especial, se puede entender el propósito con el que fueron creados y que cambios han tenido en el proceso.

Los CubeSats comenzaron como un esfuerzo colaborativo entre Jordi Puig-Suari, un profesor de la Universidad Estatal Politécnica de California (Cal Poly), y Bob Twiggs, un profesor en el Laboratorio de Desarrollo de Sistemas Espaciales (SSDL) de la Universidad de Stanford. La intención original del proyecto era el de proveer acceso al espacio que fuera asequible para la comunidad científica universitaria, cumpliendo con creces este objetivo. Gracias a los CubeSats muchas universidades, colegios e incluso escuelas han iniciado sus programas de CubeSats. [8]

3.1.2 *Especificaciones técnicas*

Un CubeSat es parte de la clasificación de satélites conocida como satélites pequeños. Estos son todo tipo de satélites que tengan una masa menor a 300 kg. Sin embargo, un CubeSat tiene que cumplir con criterios específicos que controlan tanto la forma, el tamaño, y el peso.

Estos estándares tan específicos son el mayor factor por el cual se puede reducir el costo de estos de manera que sean accesibles. Dado que están estandarizados, los componentes se pueden producir en masa, haciendo que sean menos costosos que el desarrollo de un satélite pequeño hecho a la medida. Además del costo de desarrollo, la forma y tamaño estandarizado permite reducir costos de transporte al, y despliegue en el espacio.

Los CubeSats se pueden encontrar en distintos tamaños, todos los cuales están basados en la “Unidad” estándar, expresada como 1U. Un CubeSat de 1U es un cubo de 10 cm x 10 cm x 11 cm, con una masa aproximada de entre 1 a 1.33 kg. Actualmente, otros tamaños se han vuelto populares, como lo son 1.5U, 2U, 3U, y 6U. En la Figura 3-1 se puede ver los tamaños de 1U y 3U.



Figura 3-1: Dimensiones estándar de CubeSats

Fuente: [8]

Además de las dimensiones, estos satélites tienen que cumplir con ciertos requerimientos mecánicos. Para estos requerimientos se utiliza el sistema de coordenadas que se muestra en la Figura 3-3, de manera que sigan el marco de referencia del cuerpo.

A continuación, se detallan los requerimientos mecánicos de la masa y el centro de gravedad para un CubeSat 3U. [9]

- La masa máxima de un CubeSat 3U debe ser de 4.00 kg
- El centro de gravedad de un CubeSat debe estar localizado dentro de 2 cm de su centro geométrico en las direcciones X y Y
- El centro de gravedad de CubeSats 3U y 3U+ debe estar localizado dentro de 7 cm de su centro geométrico en la dirección Z

3.2 Fundamentos astronómicos

De manera que se pueda tener el contexto entero del proyecto, es vital definir los fundamentos que describen el movimiento en el espacio.

3.2.1 Marcos de referencia

Para el diseño de un sistema de control de orientación es importante definir los marcos de referencia que se utilizan.

Marco de referencia inercial: El marco de referencia Inercial Centrado en la Tierra (ECI), también llamado marco I, tiene su origen en el centro de la Tierra. Sus ejes se ubican de manera que el eje Z está en la dirección del polo norte geográfico, el eje X en dirección del Equinoccio de Primavera, también visto como el nodo de ascenso entre el plano ecuatorial de la Tierra y la eclíptica de sol, en donde los rayos del sol son perpendiculares al Ecuador. Por último, el eje Y completa el sistema de coordenadas. [10] Este se puede ver en la Figura 3-2.

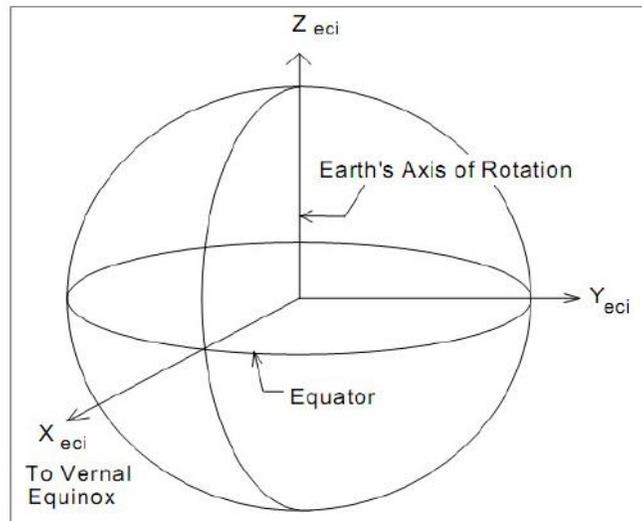


Figura 3-2: Marco de referencia ECI

Fuente: [11]

Marco de referencia del cuerpo: El marco de referencia centrado en el cuerpo del objeto, también llamado marco B tiene su origen en el centro de la masa. Además, los ejes están alineados con los ejes de inercia principales del satélite. Es importante mencionar que todos los sensores y actuadores también se encuentran alineados con este marco de referencia.

[10] Este marco de referencia se puede ver en la Figura 3-3.

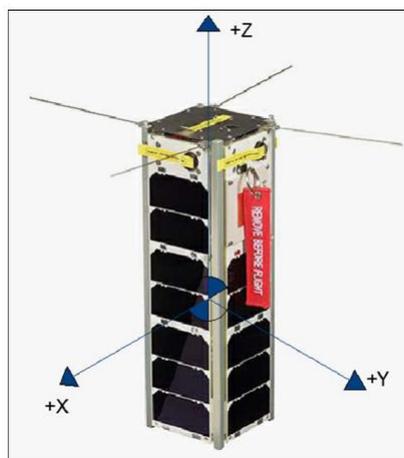


Figura 3-3: Marco de Referencia B

Fuente: [11]

Estos marcos de referencia son de alta importancia para la misión, pues por medio de una transformación de estos es que se logra determinar la orientación que se desea para el satélite. Haciendo que se pueda tomar las acciones de control necesarias para lograr esta orientación. También es importante mencionar que estos marcos de referencia se utilizan para encontrar la ubicación en el espacio de un satélite, un dato muy importante para sistemas ADCS.

3.2.2 Terminología Orbital

Para entender de la mejor manera el área en la cual se está desarrollando el proyecto es necesario dar contexto de la terminología adecuada cuando un satélite está en órbita, de esta manera, se define los siguientes contextos: [10]

Periastro: Se define como el punto más cercano de la órbita al cuerpo que está orbitando. En el caso de que esté orbitando la Tierra se le llama Perigeo.

Apoastro: Se define como el punto más lejano de la órbita con respecto al cuerpo que está orbitando. En el caso de una órbita terrestre se le llama Apogeo.

Inclinación(i): Este se define como el ángulo que existe entre el plano orbital del objeto y el plano de referencia del cuerpo que se está orbitando.

Línea de Nodos: La línea de nodos corresponde a la intersección que existe entre el plano orbital y el plano de referencia. Para un satélite terrestre, el nodo de ascenso corresponde al punto donde el satélite cruza el plano ecuatorial de Sur a Norte. El nodo de descenso, por lo tanto, corresponde al punto donde el satélite cruza el plano ecuatorial de norte a sur.

Ascenso Derecho del Nodo de Ascenso (RAAN): Representa el ángulo sobre el plano ecuatorial medido desde el equinoccio de primavera hasta el nodo de ascenso de la órbita en dirección Este.

Argumento del Perigeo (ω): Representa el ángulo medido sobre el plano orbital, en la dirección del movimiento, entre el nodo de ascenso y el perigeo

Estos términos se pueden ver resumidos en la Figura 3-4.

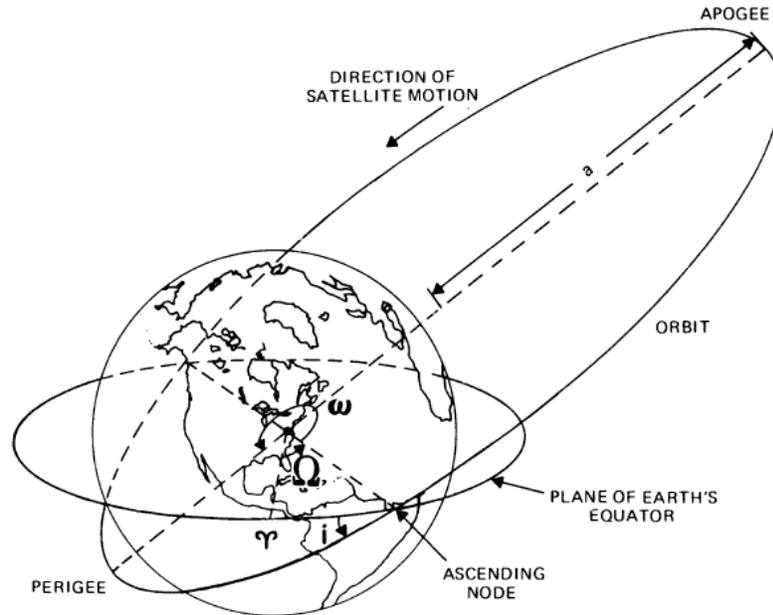


Figura 3-4: Parámetros orbitales.

Fuente: [10]

Excentricidad (e): Esta característica define el nivel de “circularidad” de una órbita. Siempre que la excentricidad sea igual a 0 se estará viendo una órbita totalmente circular, si se encuentra entre 0 y 1 se trata de una órbita elíptica, si es igual a 1 es una órbita parabólica, y si es mayor a 1 se trata de una órbita hiperbólica.

$$e = \frac{r_a - r_p}{r_a + r_p} \quad (3-1)$$

donde r_a : altitud apoastro

r_p : altitud periaastro

Semieje Mayor (a): Para órbitas elípticas, esta se define como la mitad del diámetro mayor y se define por la ecuación (3-2). Además, se puede ver representado en la Figura 3-4

$$a = \frac{r_a - r_p}{2} \quad (3-2)$$

Anomalía Media (M): Este se define como el ángulo sobre el plano orbital que existe entre el perigeo y la posición del satélite. Se define con la letra M. La forma más simple de conocer la anomalía media se especifica en la ecuación (3-3).

$$M = 360 \cdot \left(\frac{\Delta t}{P} \right) \quad (3-3)$$

Donde Δt es el tiempo que ha pasado desde la última pasada por el perigeo y P es el periodo orbital.

3.2.3 Cinemática Rotacional

La cinemática rotacional es un área que es fundamental conocerla cuando se quiere diseñar un ADCS, pues todos los movimientos se basarán tanto en la cinemática como en la dinámica rotacional.

Matriz de coseno de dirección: Cuando se considera el marco de referencia ECI, llamado I, y el marco de referencia fijado al cuerpo, llamado B, podemos expresar los vectores unidad del marco I como una función de los vectores del marco B. Como se muestra en la ecuación (3-4):

$$\begin{bmatrix} \mathbf{i}_1 \\ \mathbf{i}_2 \\ \mathbf{i}_3 \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{bmatrix} = C^{I/B} \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{bmatrix} \quad (3-4)$$

Donde $C_{jk} = \mathbf{i}_j \cdot \mathbf{b}_k$ es el coseno del ángulo entre los dos vectores unidad y $C^{I/B}$ es conocida como la matriz de coseno de dirección (DCM), la cual describe la orientación de I relativa a B. Esta matriz tiene las propiedades indicadas en la ecuación (3-5).

$$[C^{I/B}]^{-1} = [C^{I/B}]^T = C^{B/I} \quad (3-5)$$

Dado un vector arbitrario \mathbf{I} , este puede ser descrito como una función de los vectores unidad de los marcos B y I, como se indica en la ecuación (3-6):

$$\mathbf{I} = l_1 \mathbf{b}_1 + l_2 \mathbf{b}_2 + l_3 \mathbf{b}_3 = l'_1 \mathbf{i}_1 + l'_2 \mathbf{i}_2 + l'_3 \mathbf{i}_3 \quad (3-6)$$

Por lo tanto

$$\begin{bmatrix} l'_1 \\ l'_2 \\ l'_3 \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} \begin{bmatrix} l_1 \\ l_2 \\ l_3 \end{bmatrix} = C^{I/B} \begin{bmatrix} l_1 \\ l_2 \\ l_3 \end{bmatrix} \quad (3-7)$$

De esta manera, el DCM puede ser usado para realizar un cambio en el marco de coordenadas. Para construir el DCM se tienen las matrices (3-8), (3-9) y (3-10) para las rotaciones de cada eje. [12]

$$C_1(\theta_1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_1 & \sin\theta_1 \\ 0 & -\sin\theta_1 & \cos\theta_1 \end{bmatrix} \quad (3-8)$$

$$C_2(\theta_2) = \begin{bmatrix} \cos\theta_2 & 0 & -\sin\theta_2 \\ 0 & 1 & 0 \\ \sin\theta_2 & 0 & \cos\theta_2 \end{bmatrix} \quad (3-9)$$

$$C_3(\theta_3) = \begin{bmatrix} \cos\theta_3 & \sin\theta_3 & 0 \\ -\sin\theta_3 & \cos\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3-10)$$

Ángulos de Euler: El DCM que representa la orientación de un satélite puede ser construido por medio de tres rotaciones distintas alrededor de los ejes del satélite. A estos ángulos de rotación se les llama *roll* (ϕ), *pitch* (θ), y *yaw* (ψ) para los ejes X, Y, y Z respectivamente. El orden en el cual se realizan dichas rotaciones también es importante [13].

Para la primera rotación cualquiera de los ejes puede ser elegido. Para la segunda rotación uno de los otros, ahora rotados, tiene que ser utilizado. Por último, para la última rotación, se tiene que utilizar alguno de los dos ejes que no se utilizaron para la segunda rotación, lo cual nos da un total de 12 secuencias de rotación distintas.

En este caso, por ejemplo, si se desea que la secuencia de rotación sea 3-2-1, la primera rotación alrededor del tercer eje tendría un ángulo θ_3 , la segunda rotación es alrededor del segundo eje rotado con un ángulo θ_2 , y la última rotación se da alrededor del primer eje rotado con un ángulo θ_1 . El resultado es la multiplicación de las DCM's de cada una de las rotaciones, como se puede ver en las ecuaciones (3-11) y (3-12). Dado que la multiplicación de matrices no es conmutativa, un cambio en el orden de las rotaciones cambiará totalmente el resultado final de la DCM de las tres rotaciones. [12]

$$C^{I/B} = C_1(\theta_1)C_2(\theta_2)C_3(\theta_3) \quad (3-11)$$

$$= \begin{bmatrix} c(\theta_1) c(\theta_3) & c(\theta_2) s(\theta_3) & -s(\theta_2) \\ s(\theta_1) s(\theta_2) c(\theta_3) - c(\theta_1) s(\theta_3) & s(\theta_1) s(\theta_2) s(\theta_3) + c(\theta_1) c(\theta_3) & s(\theta_1) c(\theta_2) \\ c(\theta_1) s(\theta_2) c(\theta_3) + s(\theta_1) s(\theta_3) & s(\theta_1) s(\theta_2) c(\theta_3) - s(\theta_1) c(\theta_3) & c(\theta_1) c(\theta_2) \end{bmatrix} \quad (3-12)$$

Donde $c = \cos$ y $s = \text{sen}$

Cuaternios: Un cuaternio se puede ver como un vector de cuatro componentes con algunas operaciones adicionales definidas sobre éste. A cualquier cuaternio \mathbf{q} se le puede definir en dos partes, una parte con tres vectores $\mathbf{q}_{1:3}$ y una parte escalar q_4 . [14]

$$\mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} \quad (3-13)$$

La orientación del marco I con respecto al marco B por lo tanto puede ser descrito por medio de un ángulo de rotación θ y un eje de rotación, el *eigenaxis* de Euler \mathbf{e} . Este *eigenaxis* tiene la misma dirección tanto en el marco I como en el marco B, como se indica en la ecuación (3-14). [7]

$$\begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \mathbf{e}_3 \end{bmatrix} = \mathbf{C}^{I/B} \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \mathbf{e}_3 \end{bmatrix} \quad (3-14)$$

La rotación se define por los cuaternios, los cuales se definen como

$$\begin{aligned} q_1 &= e_1 \text{sen} \left(\frac{\theta}{2} \right) \\ q_2 &= e_2 \text{sen} \left(\frac{\theta}{2} \right) \\ q_3 &= e_3 \text{sen} \left(\frac{\theta}{2} \right) \\ q_4 &= \text{cos} \left(\frac{\theta}{2} \right) \end{aligned} \quad (3-15)$$

Los primeros tres componentes del cuaternio se conocen como la parte vectorial, mientras que el cuarto componente se conoce como la parte escalar. Así como un círculo unitario, la magnitud del cuaternio es la unidad.

Además, el DCM en este caso se puede expresar de la como se ve en la ecuación (3-16).

$$C^{B/I}(\mathbf{q}^{B/I}) = \begin{bmatrix} 1 - 2(q_2^2 + q_3^2) & 2(q_1q_2 + q_3q_4) & 2(q_1q_3 - q_2q_4) \\ 2(q_1q_2 - q_3q_4) & 1 - 2(q_1^2 + q_3^2) & 2(q_2q_3 + q_1q_4) \\ 2(q_1q_3 + q_2q_4) & 2(q_2q_3 - q_1q_4) & 1 - 2(q_1^2 + q_2^2) \end{bmatrix} \quad (3-16)$$

De esta ecuación se puede concluir que $\mathbf{q}^{B/I}$ describe la misma rotación que $-\mathbf{q}^{B/I}$. También, es importante mencionar que la representación en forma de cuaternio de transformaciones sucesivas es solamente el producto de los cuaternios de sus transformaciones constituyentes. Por ejemplo, si se toma dos rotaciones sucesivas hacia C desde A de la siguiente manera.

$$C(\mathbf{q}^B): A \rightarrow B \quad (3-17)$$

$$C(\mathbf{q}^C): B \rightarrow C \quad (3-18)$$

Ambas rotaciones se pueden expresar en un solo cálculo como el producto de $C(\mathbf{q}^B)$ y $C(\mathbf{q}^C)$ de donde obtenemos $C(\mathbf{q}^{C/A})$, el cual se puede calcular como se muestra en la ecuación (3-19). [7]:

$$\mathbf{q}^{C/A} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} q_4^C & q_3^C & -q_2^C & q_1^C \\ -q_3^C & q_4^C & q_1^C & q_2^C \\ q_2^C & -q_1^C & q_4^C & q_3^C \\ -q_1^C & -q_2^C & -q_3^C & q_4^C \end{bmatrix} \begin{bmatrix} q_1^B \\ q_2^B \\ q_3^B \\ q_4^B \end{bmatrix} \quad (3-19)$$

En el caso que, por ejemplo, se quiera determinar el cuaternio de orientación del satélite con respecto a un marco de referencia S, denotado como $\mathbf{q}^{B/S}$. Dada la orientación actual del satélite con respecto al marco I como $\mathbf{q}^{B/I}$, y la orientación del marco S con respecto al marco I como $\mathbf{q}^{S/I}$. El cuaternio de interés se puede calcular como se muestra en la ecuación (3-20).

$$\mathbf{q}^{B/S} = \begin{bmatrix} q_4^{S/I} & q_3^{S/I} & -q_2^{S/I} & q_1^{S/I} \\ -q_3^{S/I} & q_4^{S/I} & q_1^{S/I} & q_2^{S/I} \\ q_2^{S/I} & -q_1^{S/I} & q_4^{S/I} & q_3^{S/I} \\ -q_1^{S/I} & -q_2^{S/I} & -q_3^{S/I} & q_4^{S/I} \end{bmatrix} \mathbf{q}^{B/I} \quad (3-20)$$

Cuando la parte vectorial de $\mathbf{q}^{B/S}$ sea cero, quiere decir que el marco de referencia B está alineado con el marco de referencia S.

Ecuaciones Diferenciales Cinemáticas: En las secciones anteriores se trató la descripción de la orientación de un marco de referencia. En esta sección se describirá de manera resumida, la cinemática, lo que quiere decir que la orientación relativa entre dos marcos de referencia es dependiente del tiempo. Esta relación se describe por medio de ecuaciones diferenciales [15]. Dado que la parametrización por medio de cuaternios ha probado ser la más útil para el análisis de la cinemática de aeronaves, solamente se explicará este caso. [11]

El teorema rotacional del *eigenaxis* de Euler establece que al rotar un cuerpo rígido alrededor de un eje que esté fijado al cuerpo y estacionario en un marco de referencia inercial, la orientación del cuerpo puede ser cambiada de cualquier orientación a cualquier otra. Este eje de rotación, que durante el movimiento se mantiene en la misma posición relativo tanto al cuerpo como al marco de referencia inercial se le llama eje de Euler o *eigenaxis*. En la ecuación (3-21) se plantea la ecuación diferencial cinemática de cuaternios. [15]

$$\dot{\mathbf{q}} = \frac{1}{2} (\mathbf{q}_4 \boldsymbol{\omega} - \boldsymbol{\omega} \times \mathbf{q}) \quad (3-21)$$

$$\dot{q}_4 = -\frac{1}{2} \boldsymbol{\omega}^T \mathbf{q}$$

Donde

$$\boldsymbol{\omega} \times \mathbf{q} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \quad (3-22)$$

A la hora de modelar el sistema se hace uso de estas ecuaciones diferenciales. Por esta razón, es importante que estas se determinen de manera correcta, pues si no, el modelo del satélite será incorrecto. Esto afectaría gravemente el desarrollo de un controlador, pues se estaría diseñando para un sistema distinto.

3.2.4 Dinámica rotacional

Cuando la orientación actual y la tasa de rotación del marco de referencia fijado en el satélite, marco B, es conocida, su tasa de cambio en el tiempo debe ser descrita. Primeramente, se tiene que el momento angular de un cuerpo rígido se puede calcular como se describe en la ecuación (3-23).

$$\mathbf{H} = \mathbf{J}\boldsymbol{\omega} \quad (3-23)$$

Donde J es la matriz de inercia del satélite y $\boldsymbol{\omega} = \boldsymbol{\omega}^{B/I}$ es la tasa rotacional del cuerpo con respecto al marco I, expresado en el marco B. Además, la tasa de cambio del vector de momento angular con respecto al tiempo se define en la ecuación (3-24)

$$\dot{\mathbf{H}} = \mathbf{T} \quad (3-24)$$

Donde T es el torque externo actuando sobre el centro de masa del cuerpo. Además, por el teorema del transporte explicado en el anexo A.1.

$$\dot{\mathbf{H}} = \left\{ \frac{d\mathbf{H}}{dt} \right\}_I = \left\{ \frac{d\mathbf{H}}{dt} \right\}_B + \boldsymbol{\omega}^{B/I} \times \mathbf{H} \quad (3-25)$$

La cual, dado que $\dot{\mathbf{J}} = 0$ y denotando $\{d\omega/dt\}_B = \{d\omega/dt\}_I = \dot{\omega}$, resulta en las ecuaciones rotacionales del movimiento de Euler definidas como:

$$\mathbf{J}\dot{\omega} + \omega \times \mathbf{J}\omega = \mathbf{T} \quad (3-26)$$

Donde \mathbf{T} es el vector del torque en el marco B. Si se utiliza esta ecuación junto con la ecuación (3-21), se pueden obtener las ecuaciones dinámicas que describen al satélite. Además, el torque externo total sería la combinación entre los torques de perturbación y los torques de control.

3.2.5 Perturbaciones

Las perturbaciones en este tipo de sistemas son de importancia para el modelado completo del sistema, pues pueden afectar la tasa rotacional, así como la dinámica orbital del satélite. Las perturbaciones más importantes que se identificaron para GW-Sat son el arrastre atmosférico, el gradiente gravitacional y la perturbación J2.

Arrastre Atmosférico: Este tipo de torque es causado por la fuerza de arrastre aerodinámica no actuando sobre el centro de masa del satélite. La ecuación (3-27) es la que se utiliza para calcular este torque.

$$T_{aa} = \frac{1}{2} \rho C_d A V^2 (r_{cp} - r_{COM}) \quad (3-27)$$

Donde ρ es la densidad de la atmósfera a la altitud del satélite, C_d es el coeficiente de arrastre del satélite en cuestión, A es el área de superficie efectiva, V es la velocidad del satélite con respecto a la atmósfera, y $(r_{cp} - r_{COM})$ es la distancia entre el centro de presión aerodinámica y el centro de masa del satélite [12]. En la Figura 3-5 se puede ver un diagrama de como el torque actúa sobre el satélite.

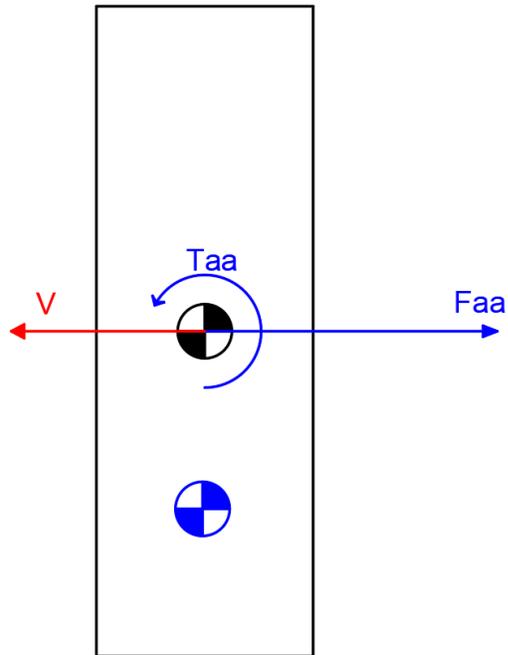


Figura 3-5: Torque de arrastre atmosférico

Fuente: Elaboración Propia

Aparte del torque que ejerce esta perturbación, también se tiene que, al ejercer una fuerza en contra del movimiento del satélite este desacelerará, por lo que la órbita de este cambiará continuamente, este es un problema especialmente importante en las LEO's dado que la densidad atmosférica es considerable a estas altitudes. En la Figura 3-6 se puede ver de una manera simplificada como se vería el efecto de arrastre atmosférico sobre la trayectoria de una órbita. Este diagrama es altamente simplificado y no a escala.

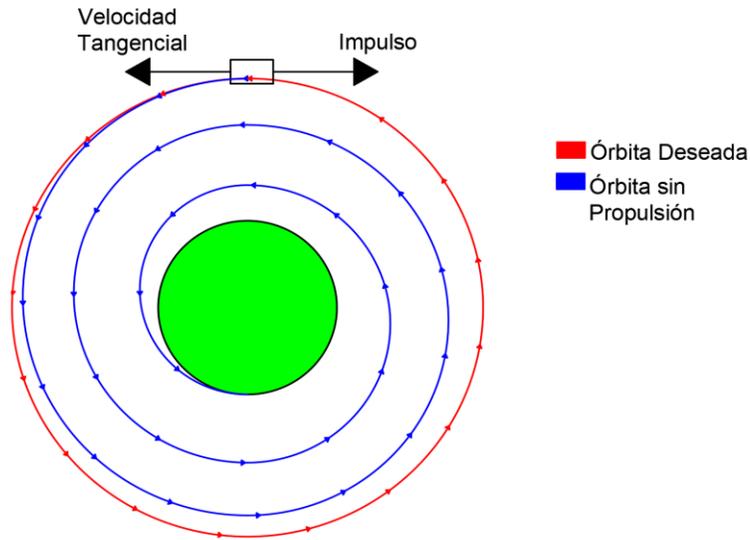


Figura 3-6: Ilustración sobre el efecto atmosférico sobre la trayectoria de un satélite sin propulsión en una LEO

Fuente: Elaboración Propia

Gradiente Gravitacional: Dado que el campo gravitacional sobre un cuerpo no es uniforme, un torque de gradiente gravitacional existe y se describe como lo especifica la ecuación (3-28): [12]

$$T_{gg} = 3n^2 \vec{\sigma}_3 \times J \vec{\sigma}_3 \quad (3-28)$$

Donde n es la velocidad angular de órbita promedio y $\vec{\sigma}_3$ está en la dirección del centro de la Tierra. En la Figura 3-7 se puede ver de manera simplificada esta perturbación y el torque que genera.

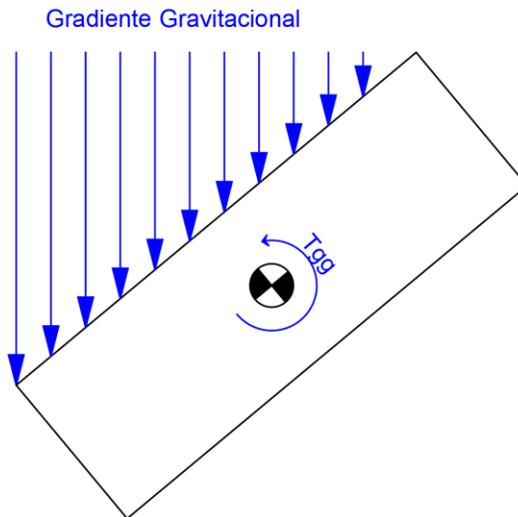


Figura 3-7: Torque de gradiente gravitacional

Fuente: Elaboración Propia

Perturbación J2: Dado que la Tierra no es una esfera perfecta, sino más bien es un esferoide oblató, el radio es mayor cerca del Ecuador que en los polos. Debido a esta condición se tiene una serie matemática que describe el efecto de esta forma sobre la gravedad del planeta. Dado que es una serie matemática, el término J2 representa el más significativo, con los consiguientes siendo despreciables pues su efecto es más de mil veces menor. [16]

Con este fenómeno el principal cambio que se ve en la órbita es el cambio en el RAAN y el Argumento del perigeo, los cuales variarán a un ritmo constante [16]. El modelado de esta perturbación es importante pues la órbita cambiará de manera importante a lo largo del tiempo, por lo que las maniobras de ajuste de orientación no serán las mismas para el mismo punto de la órbita en instantes distintos de tiempo.

3.2.6 Importancia del modelado correcto

El modelado correcto de las tres secciones anteriores (cinemática, dinámica y perturbaciones) permite crear un modelo matemático más preciso de lo que realmente sucede una vez que el satélite se encuentra en la órbita terrestre y por lo tanto hace que el diseño de algoritmos de control sea más fiable, pues se pueden simular en ambientes más realistas.

3.3 Sensores

Los sensores corresponden a una parte esencial de un ADCS, pues son los que nos permiten cumplir con el primer objetivo de un sistema de estos, el cual es la determinación de la orientación. Además, para el sistema de control nos permiten realimentar los valores necesarios para controlar la planta como lo son el cuaternio actual, así como la velocidad angular actual. También, otros como el GPS nos permite conocer la ubicación del satélite con respecto al marco de referencia inercial, un dato importante para determinar la orientación deseada.

3.3.1 *Sensores de Sol*

Este tipo de sensores son los más utilizados, de tal forma que la mayoría satélites han estado equipados con estos. Estos son preferidos pues ofrecen una alta versatilidad dado que el radio angular del sol es prácticamente constante en toda la órbita terrestre. [10]

El funcionamiento de estos generalmente se da por medio de la diferenciación entre la medición de varios de estos de manera que se pueda obtener un vector de dirección del Sol. Como se logra esto es utilizando por lo menos dos sensores por eje y se diferencia cada eje para obtener sus componentes y, consecuentemente obtener el vector unitario correspondiente. [14]

Otra característica importante de estos sensores es su bajo costo en especial cuando se comparan con los *star trackers*. Además, muchos paneles solares tienen este tipo de sensores, por lo que ni siquiera es necesario tener sensores por aparte.

3.3.2 *GPS*

Al utilizar esta tecnología es posible determinar la posición exacta del satélite en todo momento. De hecho, es el único sensor que es capaz de determinar la posición del satélite. Además, si se utilizan dos sensores GPS que estén separados entre sí, también se puede determinar la orientación de este. [17]

Para poder determinar la orientación del satélite se utilizan por lo menos dos sensores GPS los cuales se encargan de medir la diferencia de fase que existe entre ellos, y sabiendo

la distancia entre estos, además de la posición de los satélites GPS de los cuales se está recibiendo la señal se puede determinar con un alto grado de precisión la orientación del satélite. [14]

Por lo tanto, lo más importante de este sensor es su alta versatilidad dado que se puede determinar tanto la ubicación en todo momento, así como su orientación siempre y cuando se tengan por lo menos dos de estos sensores.

3.3.3 *Star Tracker*

Los sensores de estrellas miden coordenadas de estrellas en el marco de referencia del satélite y proveen información sobre la orientación de este cuando estas coordenadas se comparan con direcciones conocidas de estrellas obtenidas de un catálogo estelar. En general se trata de los sensores más precisos cuando se trata de medir la orientación. [10]

Entre sus desventajas se encuentran que son extremadamente sensibles a la luz, por lo que no sirven si hay algún objeto emitiendo o reflejando luz en su dirección; además, su frecuencia de operación es baja.

Entre sus ventajas se encuentra que los datos que devuelven no tienen que ser procesados; además, un solo sensor de estos es capaz de proveer la información de orientación, velocidad angular y tasa de aceleración con una alta precisión. [17]

En resumen, se puede ver como la mayor ventaja de este tipo de sensor es su alta precisión. Por esto, para misiones en las que esta tenga que ser extremadamente alta se recomienda uno de estos sensores.

3.3.4 *Giroscopio de 3 ejes*

Un giroscopio es un instrumento que utiliza masas girando rápidamente para detectar y responder a cambios en la orientación inercial del eje de giro [10]. Estos se clasifican principalmente en dos tipos: giroscopios de tasa y giroscopios de tasa integradora.

La arquitectura más precisa de giroscopios para aeronaves, se conocen como giroscopios flotantes de un eje, los cuales tienen el rotor y su motor impulsor contenido dentro de un cilindro como se muestra en la Figura 3-8.

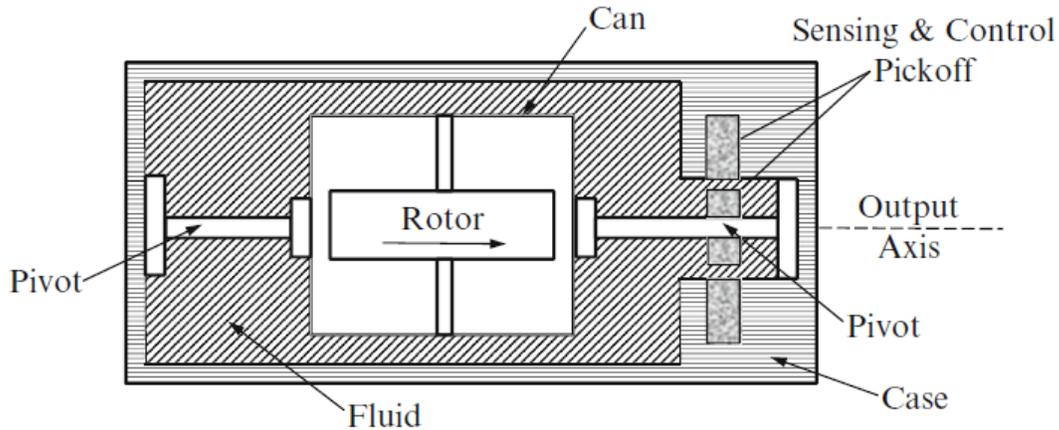


Figura 3-8: Giroscopio flotante de un eje

Fuente: [14]

Un modelo matemático utilizado ampliamente para un giroscopio de tasa integradora se da en la ecuación (3-29).

$$\omega(t) = \omega^t(t) + \beta^t(t) + \eta_v(t)$$

$$\dot{\beta}^t(t) = \eta_u(t) \quad (3-29)$$

Donde ω^t es la verdadera tasa de rotación, ω es la tasa medida, β^t es el verdadero sesgo, y η_v y η_u representan procesos independientes de ruido blanco Gaussiano. [14]

Los giroscopios se utilizan para detectar rotaciones relativas, no absolutas, por lo que nos permite determinar la velocidad angular. Además, una vez que se tiene la orientación actual se puede utilizar para medir el error entre la orientación deseada y la actual.

3.3.5 Magnetómetro de 3 ejes

Los magnetómetros miden la suma del campo magnético que hay en el ambiente que son de interés, además del campo magnético del propio satélite. Si estos se quieren utilizar como sensores de orientación es importante que se tenga un campo magnético bien modelado, lo que limita su uso básicamente a LEO's.

Los campos magnéticos locales pueden ser producidos por una variedad de factores como los son: materiales ferromagnéticos, motores eléctricos, instrumentación, o especialmente *magnetorques*. Esto no presenta un problema siempre y cuando se conozcan los campos locales; pero, en caso de que no se conozcan, estos sensores se deberán colocar lejos de estas fuentes de contaminación magnética. [14]

3.4 Actuadores

Los actuadores son la parte que completa el ADCS, pues son los responsables de que se genere la acción que se desea de manera que se pueda alcanzar la orientación deseada.

3.4.1 Magnetorquers

Los *magnetorquers* son actuadores que, como su nombre lo implica, funcionan a base del principio del electromagnetismo. Estos generan un momento magnético, m , el cual puede generar un torque cuando se toma en cuenta el campo magnético, B . Como se expresa en la ecuación (3-30).

$$T_{MT} = m \times B \quad (3-30)$$

Si se toma una corriente, I , fluyendo por una bobina, de área, A , y con una cantidad de vueltas, N . El momento que se genera se puede modelar por medio de la ecuación (3-31). [11]

$$m = NIA \quad (3-31)$$

3.4.2 Propulsores μ CAT

Las descargas de arco en vacío que ablate y consumen material del cátodo, produce *jets* de plasma totalmente ionizados a alta velocidad. Para este tipo de sistemas el cátodo funciona tanto como electrodo para la descarga, así como propelente.

A continuación, se presentan dos diseños esquemáticos de este tipo de propulsor

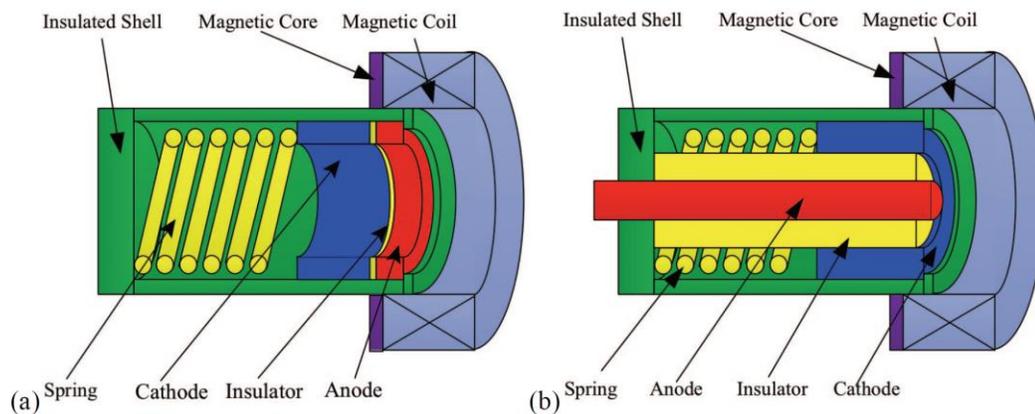


Figura 3-9: Diseños esquemáticos de los propulsores μ CAT

Fuente: [18]

3.5 Algoritmos de inteligencia artificial

Una corriente importante en el desarrollo de algoritmos es el diseño de modelos algorítmicos que solucionen cada vez problemas más complejos. Buenos resultados se han obtenido al modelar la inteligencia biológica y natural, resultando en lo que se conoce como “sistemas inteligentes”. Entre estos sistemas inteligentes se incluyen: [19]

- Redes neuronales artificiales
- Computación evolutiva
- Inteligencia de enjambre
- Sistemas inmunes artificiales
- Sistemas difusos

Además, cuando se juntan con lógica, razonamiento deductivo, sistemas expertos, razonamiento basado en casos, y sistemas de aprendizaje de maquina simbólico; todos estos algoritmos inteligentes forman parte del campo de la IA. [19]

Por medio de inteligencia artificial se pueden desarrollar “sistemas inteligentes”. Estos son capaces de encontrar soluciones que serían impensables y/o imposibles con métodos clásicos de ingeniería, siendo así una herramienta muy poderosa para la resolución de problemas. En el caso específico de teoría de control, estos son muy útiles para controlar sistemas físicos de alta complejidad, y no linealidades.

Dado que la naturaleza de este proyecto es el de un algoritmo de control, esta sección se centrará en uno de los sistemas inteligentes mencionados anteriormente: redes neuronales artificiales; las cuales se ha probado que se utilizan como sistemas de control para sistemas espaciales. [6, 20]

3.5.1 *Redes neuronales artificiales*

Los pilares sobre los cuales se basan los sistemas neuronales biológicos son las células del sistema nervioso, llamadas neuronas. Como se muestra en la Figura 3-10, una neurona consiste en dendritas, soma, axones y sinapsis. [19]

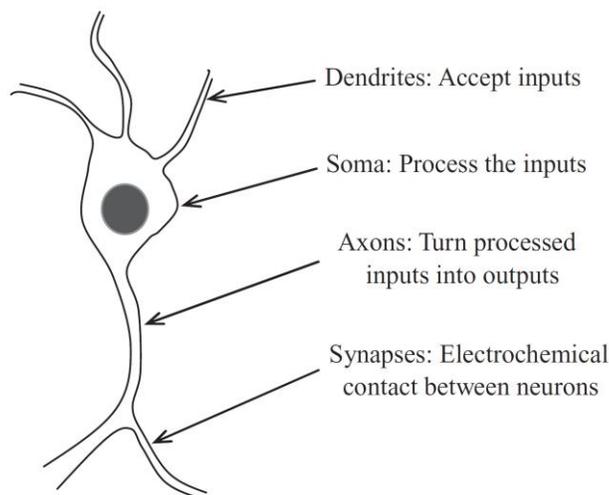


Figura 3-10: Esquema de una neurona biológica

Fuente: [21]

Una neurona artificial (AN) es un modelo de una neurona biológica (BN). Cada AN recibe señales, ya sea del ambiente, o de otras ANs, recolecta estas señales, y cuando se activa, transmite esta señal a todas la ANs conectadas. En la Figura 3-11 se puede ver una representación gráfica de una neurona artificial. Las señales de entrada son inhibidas o excitadas por medio de pesos numéricos asociados con cada conexión hacia la AN. Además, la activación de una AN y la fuerza de la señal de salida se controlan por medio de una función f , llamada función de activación. Por lo tanto, la AN recolecta todas las señales entrantes, y determina una señal neta de entrada como una función de sus respectivos pesos. Esta señal neta de entrada funciona como entrada para la función de activación, la cual calcula la señal de salida de la AN. [19]

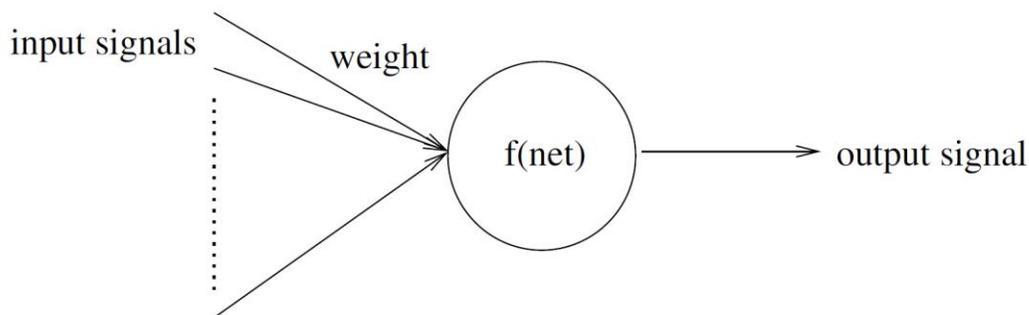


Figura 3-11: Modelo de una neurona artificial

Fuente: [19]

Una red neuronal artificial (ANN) es una red con capas de ANs. Generalmente consiste en una capa de entrada, capas ocultas y una capa de salida. Las ANs en una capa están conectadas parcial o completamente a las ANs de la capa siguiente. Además, es posible que haya conexiones de realimentación a capas pasadas. En la Figura 3-12 se puede ver una representación típica de una ANN. [19]

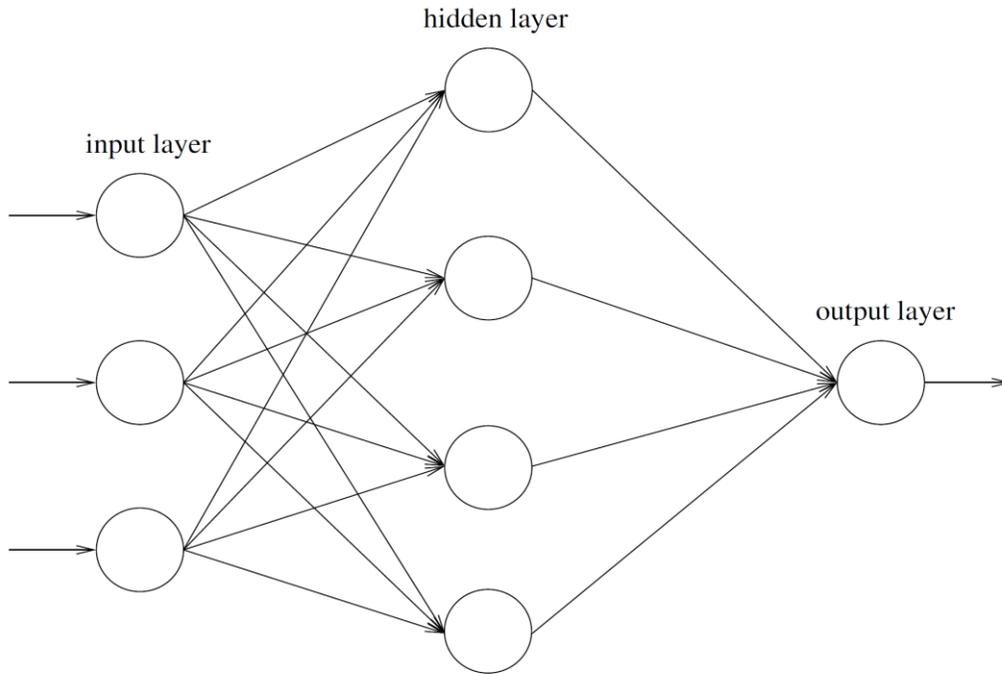


Figura 3-12: Representación de una red neuronal artificial típica

Fuente: [19]

En los casos en los que no se tiene un modelo matemático, o que el modelado de un sistema es muy complejo, se fomenta el uso de este tipo de redes. Pues éstas por medio de un entrenamiento son capaces de modelar dentro de sí sistemas de alta complejidad y dimensionalidad con un alto grado de precisión y sin necesidad de simplificaciones como en los modelos tradicionales. De esta manera, también se puede modelar un control no lineal de una planta con perturbaciones e incluso fallos desconocidos.

3.5.2 *Aprendizaje de la neurona artificial*

Dado que el valor de los pesos y del umbral θ es lo que permite que la red neuronal tenga como salidas los valores deseados, se necesitan encontrar los valores que nos permiten llegar a ese resultado. Esto se puede realizar calculándolos en problemas simples, pero cuando los problemas son más complejos esto no solo se vuelve tedioso, sino también impráctico. Además, que, si no se tiene conocimiento previo de la función aparte de los datos, estos valores no se podrían calcular. Es por esta razón que se utiliza aprendizaje, de manera

que cada AN aprenda los mejores valores posibles de los datos que se tienen. El aprendizaje consiste en ajustar los pesos y los valores de umbral hasta que se satisfaga un criterio.

Hay tres tipos principales de aprendizaje: [19]

- Aprendizaje supervisado: Es el tipo de aprendizaje por medio del cual se le provee con un conjunto de datos, a la red neuronal, que consiste en vectores de entrada y una salida deseada asociada con cada vector de entrada. El propósito de este tipo de aprendizaje consiste en minimizar el error entre la salida real y la salida deseada.
- Aprendizaje no supervisado: se utiliza cuando el objetivo es descubrir patrones o características en los datos de entrada sin asistencia de una fuente externa.
- Aprendizaje por refuerzo: Se utiliza cuando el objetivo es recompensar a la red neuronal por un buen rendimiento, y penalizarla por mal rendimiento.

La existencia de mínimos locales, en el valor del error, puede hacer que una red neuronal no pueda seguir aprendiendo. Estos hacen que la red asuma que su aprendizaje ya no va a seguir mejorando pues está en un error mínimo, cuando en realidad puede ser que haya un mínimo absoluto más bajo. Esto se puede ver representado en la Figura 3-13.

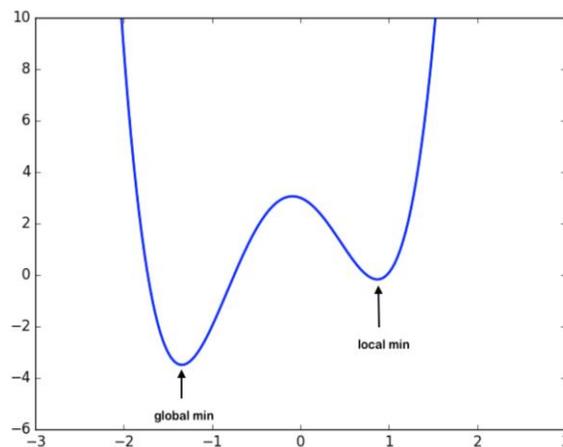


Figura 3-13: Mínimo local vs mínimo global

Fuente: [22]

Otra condición que pueden presentar las redes neuronales durante su entrenamiento es el fenómeno del olvido catastrófico. Esto sucede cuando esencialmente la red “olvida” lo que había aprendido y tiene que volver a comenzar el entrenamiento. Se puede ver gráficamente en la Figura 3-14, donde la línea roja representa la recompensa promedio de un sistema de aprendizaje por refuerzo.

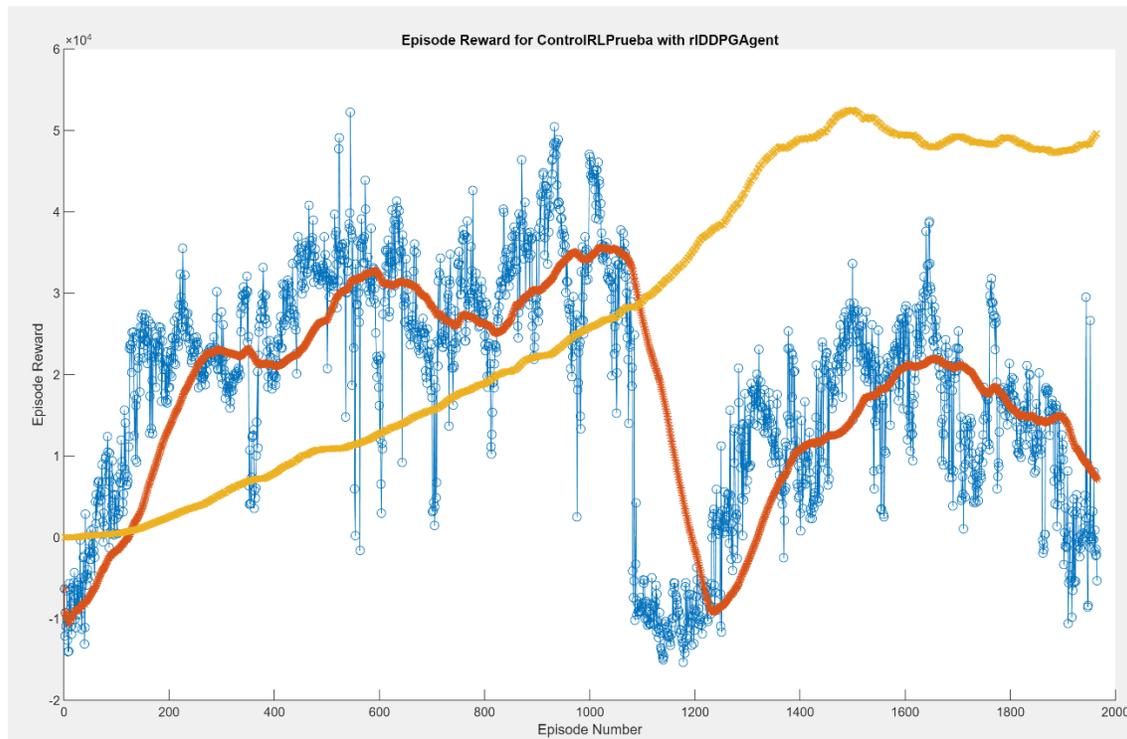


Figura 3-14: Olvido Catastrófico en la recompensa de aprendizaje por refuerzo

Fuente: Elaboración propia

3.6 Aprendizaje por refuerzo

Si se define de manera formal, el aprendizaje por refuerzo (RL, por sus siglas en inglés) es el aprendizaje de un mapeo de situaciones a acciones con el objetivo de maximizar la recompensa escalar. En síntesis, se trata de un aprendizaje por medio de prueba y error, donde el agente no tiene conocimiento previo de que acción tomar y tiene que descubrir que acciones son mejores. En la Figura 3-15 se puede ver ilustrado un problema típico de aprendizaje por refuerzo.

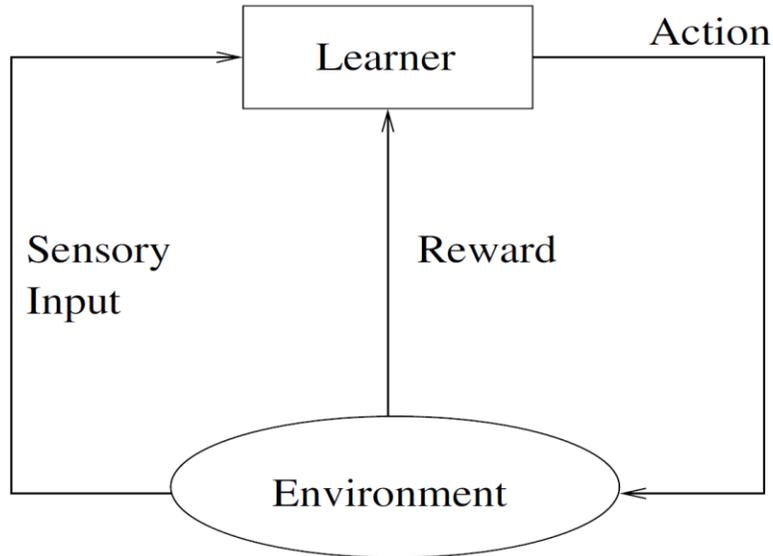


Figura 3-15: Problema de aprendizaje por refuerzo

Fuente: [19]

En esencia, lo que sucede es que el agente recibe entradas sensoriales de su entorno. Una acción se ejecuta, de donde el agente recibe la señal de refuerzo o recompensa. Como se explicó antes, esta recompensa puede ser tanto positiva como negativa, dependiendo en que tan correcta es la acción.

Un agente de aprendizaje por refuerzo debe tener los siguientes componentes:

- Una política: Constituye la función de toma de decisiones del agente.
- Una función de recompensa: Define la meta del agente, define que es “bueno” y que es “malo”
- Una función valor: Especifica la meta a largo plazo. Se utiliza para predecir recompensas futuras, y para definir que es “bueno” a largo plazo

La función valor presenta varios modelos por medio de los cuales se puede tomar en cuenta el futuro. Entre estos modelos se encuentran:

- El modelo de horizonte finito: Optimiza la recompensa esperada por una cantidad de pasos.

- El modelo descontado de horizonte infinito: Toma en cuenta la recompensa entera de largo plazo, pero cada recompensa recibida en el futuro es descontada geoméricamente.
- El modelo de la recompensa media: Prefiere las acciones que optimizan la recompensa media a largo plazo del agente.

Hay dos tipos principales de modelos de aprendizaje por refuerzo: el aprendizaje de Diferencia Temporal (TD) y el aprendizaje-Q. [19]

Aprendizaje de Diferencia Temporal: Este tipo de aprendizaje aprende la póliza de valor utilizando la regla de actualización descrita en la ecuación (3-32):

$$V(s) = V(s) + \eta(r + \gamma V(s') - V(s)) \quad (3-32)$$

Donde η es la tasa de aprendizaje, r es la recompensa inmediata, γ es el factor de descuento, s es el estado actual, y s' es un estado futuro. De manera que, si un estado s es visitado, su valor estimado se actualiza para ser más cercano a $r + \gamma V(s')$.

Aprendizaje-Q: Este tipo de aprendizaje busca aprender los valores descontados esperados de refuerzo de tomar una acción a en un estado s , luego continúa escogiendo las acciones de manera óptima. La manera de relacionar los valores Q a la función de valor, se tiene que tomar en cuenta la ecuación (3-33):

$$V^*(s) = \max Q^*(s, a) \quad (3-33)$$

Donde $V^*(s)$ es el valor de s asumiendo que la mejor acción se toma desde un inicio. Por lo tanto, la regla de actualización de aprendizaje-Q se describe en la ecuación (3-34):

$$Q(s, a) = Q(s, a) + \eta(r + \gamma \cdot \max Q^*(s', a') - Q(s, a)) \quad (3-34)$$

De esta manera, el agente toma la acción con el valor más alto de Q . [19]

3.6.1 *Métodos de gradiente de política*

Como se explicó antes, normalmente se intenta estimar una función de valor y luego de esta se deriva una política. Mientras tanto, los métodos de gradiente de política funcionan de una manera distinta. En este caso la política se controla por medio de un vector de parámetros $\vec{\theta}$. $J(\theta)$ es una medida del desempeño para la política actual, llamada π_{θ} . Esto hace que estos métodos sean capaces de manejar espacios de acción y estado estable mejor que los métodos tradicionales de aproximación de funciones.

Esto se ve especialmente en el caso de espacios de alta dimensionalidad, donde la garantía de que un algoritmo de aproximación de funciones converja es baja; mientras que, con los algoritmos de gradiente de política se puede ver que son más estables y logran converger incluso en condiciones difíciles. Esto funciona pues basándose en los pesos actuales, la red sigue actualizando los pesos directamente. [23]

3.6.2 *Redes actor-crítico*

Una arquitectura que se presenta como prometedora para aplicaciones de control es la arquitectura de red actor-crítico. Este consiste en dos redes neuronales distintas. Una se utiliza para implementar el controlador (el actor) y la otra se usa para implementar el aprendizaje por refuerzo (el crítico).

La red del actor se puede ver como el controlador dado que es el que implementa la política. Mientras que la red del crítico es el aprendiz pues se encarga de evaluar la política y se puede utilizar para realizar la optimización de la política. [24]

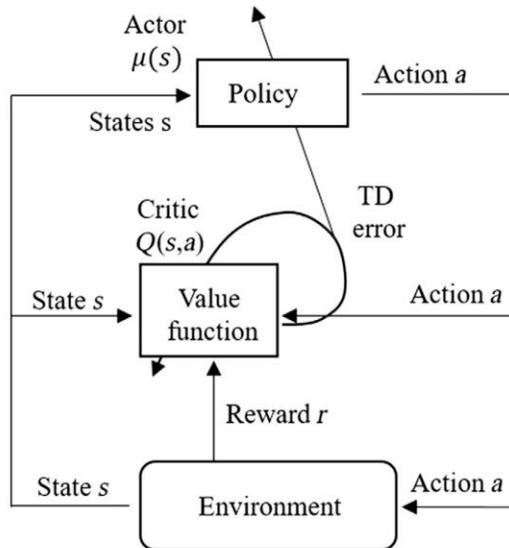


Figura 3-16: Arquitectura actor-crítico

Fuente: [23]

Como se puede ver en la Figura 3-16, la red del actor recibe el estado actual (s) y produce una señal de control (a). La red del crítico recibe el estado actual (s), la acción (a) y la recompensa por el estado resultante (r) de manera que produce un valor Q que evalúa el par estado-acción. Luego, el crítico se utiliza para estimar la acción óptima para poder actualizar los pesos en la red del actor. [24, 23]

Otra característica importante de este tipo de red es que puede trabajar con espacios de acción continuos. [23]

Por lo tanto, para este proyecto, las redes de tipo actor-crítico representan una buena opción para el desarrollo del controlador dado que se tiene un mejor aprendizaje dada la existencia de dos redes en vez de solamente una. Además, el hecho de que puedan trabajar en espacios de acción continuos es un punto clave, dado que el sistema a controlar tiene un espacio de acción continuo.

3.6.3 *Exploración vs explotación*

En las etapas tempranas del entrenamiento de una red neuronal es importante explorar una diversidad de acciones. Esto es porque, no necesariamente las acciones que son mejores al inicio son las mejores en el espectro absoluto de posibles acciones para el sistema. El problema que se encuentra con esto es que es necesario decidir en cierto momento que ya se ha probado suficiente y se tiene que comenzar a explotar el conocimiento generado. Este es un dilema muy conocido en el área de aprendizaje por refuerzo, al que se le llama el Dilema de la Exploración-Explotación y es un proceso altamente empírico.

3.6.4 *Aprendizaje profundo por refuerzo*

Recientemente, se ha visto progreso para resolver tareas complejas de entradas sensoriales, de alta dimensionalidad y sin procesar por medio de la combinación del aprendizaje profundo para el procesamiento sensorial, junto con aprendizaje por refuerzo, lo que dio lugar a la Red Profunda Q (DQN, por sus siglas en inglés).

Esta red es capaz de alcanzar un desempeño nivel humano en muchos videojuegos de Atari. Sin embargo, aunque DQN resuelve problemas con un espacio de observaciones de alta dimensionalidad, solo puede manejar un espacio de acción discreto y de baja dimensionalidad, una gran desventaja cuando se considera que muchas tareas de control tienen espacios de acción continuos y altamente dimensionados. Además, DQN no puede ser aplicado a espacios continuos pues este algoritmo con valores continuos requeriría de un proceso de optimización iterativo en cada paso. [25]

Dadas las desventajas que presenta en algoritmo de DQN, se tuvo que crear una alternativa que mantuviera las buenas propiedades de observaciones de alta dimensionalidad e interpretación de entradas sensoriales, pero además resolviera las debilidades del manejo de un espacio de acción solamente discreto y de baja dimensionalidad. A este se le llamó DDPG, del cual se discutirá en la próxima sección.

3.6.5 *DDPG*

Al modificar un algoritmo del tipo actor-crítico, específicamente la Política de Gradiente Determinístico (DPG, por sus siglas en inglés) con los aspectos positivos de DQN,

nos permite utilizar aproximadores de función por medio de redes neuronales para aprender en espacios grandes de estados y acciones continuas. Este algoritmo se llama Política de Gradiente Determinístico-Profunda (DDPG, por sus siglas en inglés). [25]

Uno de los mayores problemas que presenta el DPG es que tiene un desempeño pobre y a menudo no converge. Por lo tanto, se hicieron cambios claves para implementar el DDPG. La innovación más importante es el *replay buffer*. Este es básicamente un buffer en el cual se recolecta información del agente, como el estado, la acción y la recompensa, y la almacena. Una vez que este buffer contiene suficiente información, este es mezclado de manera que se pueda obtener una muestra aleatoria de este. Este tipo de entrenamiento resulta en una mejor convergencia, pues la ANN no se entrena con datos sucesivos, sino con muestras aleatorias de cierto tamaño. [23]

Otra adición importante que ayuda para asegurar una mayor convergencia de la red neuronal durante el entrenamiento es tener una red objetivo. Estas redes son una copia de las redes del actor y del crítico. Esto se realiza pues una red no debería actualizar sus pesos directamente de sus salidas. En la Figura 3-17 se puede ver el caso en el que la red objetivo sigue a la red actual con una cierta “distancia”. De esta manera, la red objetivo retorna el valor Q para el cálculo del objetivo TD. Luego de lo cual se calcula la función de pérdida y se actualiza con la red actual. Esta modificación previene amplificación o efectos de realimentación. [23]

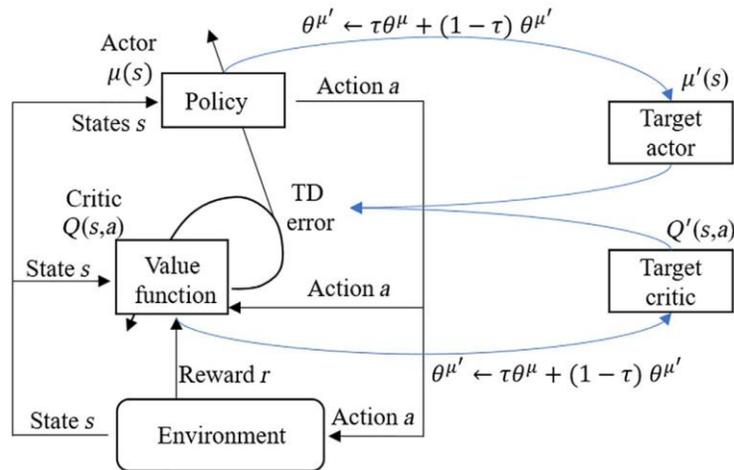


Figura 3-17: Algoritmo actor-crítico de DDPG

Fuente: [23]

Una parte fundamental, que define y diferencia, de cada algoritmo de aprendizaje de una red neuronal es el conjunto de reglas que siguen para la actualización de sus pesos. Su importancia se centra en el hecho de que dependiendo del grupo de reglas que se utilice este puede aprender de mejor manera el sistema que se esté tratando de controlar. Para el caso de DDPG, estas reglas se pueden dividir en dos pasos. [23]

1. Primeramente, se actualizan los pesos de la red del crítico. Esto se hace por medio del y_i de objetivo TD, como se ve en la ecuación (3-35), y de la función de pérdida, descrita en la ecuación (3-36). Como se puede ver de la Figura 3-17, para el cálculo del objetivo TD se utiliza la salida de la red objetivo del crítico.

Un mini lote es tomado del *replay buffer* y de esta manera se calcula según la ecuación (3-35) el objetivo TD. Luego, se calcula el error TD, con la ecuación (3-36), utilizando el valor Q de la red actual del crítico.

$$y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1} | \theta^{\mu'})) | \theta^{Q'}) \quad (3-35)$$

$$L = \frac{1}{N} \sum_i (y_i - Q(s_i, a_i | \theta^Q))^2 \quad (3-36)$$

Donde i es índice dentro del mini lote y N es el tamaño de dicho lote.

2. En este paso se actualiza la red del actor. Para esto DDPG/DPG tienen una regla especial de actualización. A esta regla se le llama el “teorema del gradiente de política determinística”, la cual se describe en la ecuación (3-37), y tiene como resultado el gradiente del desempeño de la política. La ecuación (3-38) muestra el gradiente de política muestreado.

$$\nabla_{\theta^{\mu}} J \approx E_{\mu'} [\nabla_a Q(s, a | \theta^Q) |_{s=s_t, a=\mu(s_t)} \nabla_{\theta^{\mu}} \mu(s | \theta^{\mu}) |_{s=s_t}] \quad (3-37)$$

$$\nabla_{\theta^{\mu}} J \approx \frac{1}{N} \sum_i [\nabla_a Q(s, a | \theta^Q) |_{s=s_i, a=\mu(s_i)} \nabla_{\theta^{\mu}} \mu(s | \theta^{\mu}) |_{s=s_i}] \quad (3-38)$$

En resumen, el algoritmo DDPG toma las mejores características de DPG y DQN y las combina en un solo algoritmo. Además, agrega el concepto de utilizar una red objetivo y del *replay buffer* para terminar de optimizar el proceso de aprendizaje de la red. De esta manera, se consigue un algoritmo de aprendizaje novedoso y poderoso capaz de actuar en un campo continuo y de alta complejidad.

3.6.6 TD3

Un problema que se ha visto recurrentemente tanto en aprendizaje Q como en aprendizajes de gradientes de política es que estos tienden a tener un sesgo de sobreestimación. Esta es una propiedad en donde la maximización, que se da en tanto en aprendizaje Q como en los algoritmos DPG, induce una sobreestimación constante del valor Q.

Para solucionar este problema, se ha generado un tipo de algoritmo llamada Doble DQN. La solución consiste en estimar el valor de la política actual con una función de valor objetivo separada, permitiendo que las acciones sean evaluadas sin sesgo de maximización. Sin embargo, dado que la política cambia lentamente en sistemas actor-crítico, el valor actual y objetivo se mantienen muy cerca como para eliminar el sesgo.

Este problema se puede tratar adaptando el aprendizaje Q Doble a un formato actor-crítico al utilizar un par de críticos independientemente entrenados. Mientras que esto permite una estimación menos sesgada, inclusive un estimado sin sesgo, pero con alta varianza puede generar sobreestimaciones. Por esta razón se creó el algoritmo TD3, el cual aprovecha la noción de que un estimado sufriendo de sesgo de sobreestimación se puede utilizar como un límite superior del estimado del valor real, de esta manera favoreciendo subestimaciones, las cuales no se propagan durante el aprendizaje, ya que acciones con valores bajos son evitadas por la política. [26]

Este algoritmo es, en esencia el sucesor del DDPG. Hasta hace poco, DDPG era de los algoritmos más utilizados para problemas de control continuo como robótica y manejo automático. Los problemas que se expresaron en los párrafos anteriores pueden causar que el agente caiga en un mínimo local o que sufra de olvido catastrófico. La Política Gemela de Gradiente Determinístico-Profunda con Atraso (TDDDPG, por sus siglas en inglés), o mejor conocido como TD3, se encarga de reducir el sesgo como se explica previamente. Esto lo hace por medio de 3 características principales. [27]

1. Utilizando dos redes de críticos
2. Actualizaciones con atraso del actor
3. Regularización del ruido de la acción

El primer punto ya se trató en párrafos anteriores, por lo que se tocan los otros dos puntos ahora.

Las redes objetivo se consideran una buena manera de introducir estabilidad al entrenamiento de los agentes; sin embargo, en el caso de los métodos actor-crítico se presentan algunos inconvenientes. Esto sucede por la interacción que existe entre el actor y

el crítico. El entrenamiento del agente comienza a divergir cuando una mala política se sobreestima, por lo que, en ese caso la política continuaría empeorando dado que está actualizando sobre estados con errores altos.

La solución que propone TD3 a este problema es que las actualizaciones del actor se deben dar menos frecuentemente que las del crítico. Esto lo que permite es que el crítico se vuelva más estable antes de que se utilice para actualizar el actor, y por lo tanto que tenga una menor varianza, resultando en una mejor política. [27]

Por último, la regularización del ruido. Los métodos de DPG tienden a producir valores objetivos con una alta varianza al actualizar el crítico. Esto es causado por sobreajustes a picos en la estimación del valor. Para reducir esta varianza, TD3 utiliza una técnica llamada suavizado de la política objetivo. En esencia, lo que hace es que reduce la varianza agregando pequeñas cantidades de ruido al objetivo y promediando sobre mini lotes. El rango del ruido es limitado para que los valores objetivo se mantengan cerca de la acción original. [27]

En resumen, el TD3 es el sucesor del DDPG, tomando este algoritmo y adicionándole una red de crítico más, actualizaciones con atraso al actor y regularización del ruido del actor. Además, la actualización del crítico se hace tomando el peor de ambos de manera que se pueda evitar que el algoritmo asuma que es mejor de lo que realmente es.

3.7 Sistemas de control automático

Los sistemas de control automático han sido fundamentales para el avance de la ingeniería y la ciencia. Estos se han convertido en una parte vital en los sistemas de vehículos espaciales [28]. En esta sección se presenta la base teórica de los métodos de control de interés para el desarrollo de este proyecto. En primera instancia se explica la teoría de controladores clásicos y luego los controladores por medio de IA.

3.7.1 Controladores Clásicos

Para el desarrollo de este proyecto es importante describir la utilidad tanto de un control PID, como el antecedente; así como describir un control óptimo, en este caso el Regulador Cuadrático Lineal (LQR).

3.7.2 PID

La utilidad de los controles PID se encuentra en que se aplican en la mayoría de los sistemas de control. En especial cuando no se conoce el modelo matemático de la planta y, por lo tanto, no se puede utilizar un método de control analítico [28]. Este se compone de tres ganancias principales: [29]

- K_p : Escala la señal de error que viene de la comparación entre la salida del sistema y la entrada de referencia
- K_i : Se encarga de que se elimine el error de estado estacionario.
- K_d : Controla las salidas de alta frecuencia del sistema, haciendo que no haya cambios bruscos en la planta

En general, por su característica de que no necesita de un modelo, el control PID se considera un control bastante robusto. Aun así, generalmente, no representa la mejor opción desde el punto de vista de desempeño para sistemas de alto orden. Pues la complejidad del diseño es exponencial con respecto al orden del sistema.

3.7.3 Realimentación de Estados

El diseño de sistemas de control en el espacio de estados se puede dividir principalmente en dos métodos: la asignación de polos y el regulador cuadrático lineal.

Cuando se diseña por medio de asignación de polos, este nos permite asignar los polos según las características dinámicas del sistema de manera totalmente libre. Mientras que cuando se diseña por medio de un LQR nos permite minimizar el costo de la señal de control, logrando de esta manera cumplir con la ley de control óptimo. [28]

Este tipo de controladores, dentro de la teoría del control moderno, se consideran ideales para el control de plantas de orden alto, pues al utilizar la forma de espacio de estados, todo se vuelven operaciones matriciales. Estas operaciones son sencillas de realizar en software especializado como MATLAB; lo que nos permite diseñar estos controladores de manera eficiente. Además, el control LQR presenta mejor rendimiento en general a la hora del diseño de controladores.

3.7.4 Controladores por medio de IA

En esta sección se describen diferentes algoritmos de inteligencia artificial que se utilizan en la actualidad. Este tipo de controladores representan un nuevo paradigma en el área de la teoría de control.

3.7.5 Redes Neuronales

Las metodologías tradicionales de control se basan principalmente en la teoría de sistemas lineales, mientras que los sistemas reales son no-lineales por naturaleza y tienen dinámicas no modeladas, ruido ineliminable, incertidumbre, entre otros; lo cual crea problema para los ingenieros tratando de diseñar algoritmos de control. Por lo tanto, diseñar un algoritmo de control eficiente se vuelve retador.

Una de las razones por las cuales las redes neuronales se han popularizado en aplicaciones de control es, precisamente, que satisfacen algunos de estos criterios. Además, desde el punto de vista práctico, el paralelismo y adaptabilidad de las redes neuronales son ventajas adicionales. El uso de redes neuronales en aplicaciones de control, como lo son las aplicaciones aeroespaciales, ha experimentado un rápido crecimiento recientemente. Típicamente, hay dos etapas involucradas cuando se utilizan redes neuronales para control: [21]

- Identificación del sistema

- Diseño del control

3.8 Medidas de desempeño y eficiencia

En los sistemas de control hay una diversidad de aspectos por medio de los cuales se puede diseñar, de manera que se cumplan con los criterios de desempeño que se desean. Entre estos se encuentran las siguientes características: [28]

- Sobreimpulso: Es el máximo valor pico de la curva de respuesta, medido a partir de la unidad. Se expresa en porcentaje.
- Tiempo de estabilización: Es el tiempo que se necesita para que la curva de respuesta alcance un rango alrededor del valor final, del tamaño especificado.
- Error en estado estacionario: Este es el valor en porcentaje de la diferencia que existe entre el valor final y el valor de referencia que se quería alcanzar.

En la Figura 3-18 se pueden observar estos parámetros.

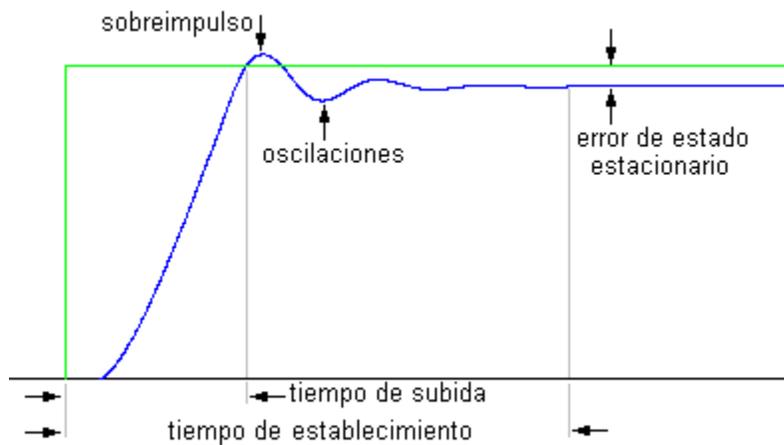


Figura 3-18: Parámetros de desempeño de un sistema

Fuente: [30]

Además del desempeño del sistema de control, también hay casos en los que se quiere medir que tan eficiente es el control con respecto al consumo energético. En estos casos se utiliza el área bajo la curva de la señal de control para encontrar el gasto energético del

sistema. Es importante recordar que este parámetro es totalmente independiente de los parámetros de desempeño.

3.9 Estado del arte de ADCS´

En esta sección se detallan las diferentes técnicas de control que se utilizan en la actualidad para el diseño de controladores de orientación para satélites, para poder proveer el contexto necesario para el desarrollo adecuado del proyecto. En primera instancia se tocará el tema de los controladores clásicos utilizados y en la segunda parte se tratará el tema de controladores por medio de IA.

3.9.1 *Controladores Clásicos*

El control PID es uno de los controles más utilizados en los sistemas de control de orientación de satélites. Las ventajas de los controles PID es su simple estructura, la robustez de su matriz de inercia, entre otras. Un uso que se le ha dado es en el de la creación de un controlador robusto considerando la velocidad angular y una limitación en el torque de control [31]. De igual forma, se han desarrollado controles PD para casos en los que no es imperativo que el error sea cero, sino que pueda tener una cierta tolerancia [32].

También, hay casos en los cuales se han utilizado técnicas de control modernas, como lo son el regulador cuadrático lineal (LQR). Esto dado que, siempre y cuando el sistema sea controlable, este tipo de controlador presenta un mejor desempeño en estabilidad, así como en que el tiempo de maniobra en satélites es menor; además de ser capaz de alcanzar requerimientos en sobreimpulso, tiempo de subida y minimizar el error de estado estacionario. [33]

3.9.2 *Controladores con IA*

Cuando se diseñan sistemas de control para satélites, muchas veces se presentan problemas por la dinámica tan compleja y, en especial, no lineal que se da por las no-linealidades inherentes del modelo dado las maniobras con ángulos tan amplios, incertidumbres y perturbaciones del ambiente desconocidas.

Aunque se han propuesto muchos controladores clásicos, las redes neuronales son una herramienta prometedora para aplicaciones de control, dado que son capaces de aproximar cualquier función no lineal a cualquier nivel de precisión. [6]

Dados los problemas que se mencionaron anteriormente, también se han utilizado sistemas de lógica difusa, los cuales tienen la desventaja de la falta de la habilidad de auto aprendizaje y de que reduce considerablemente la precisión de control. Por lo tanto, se ha decidido de manera general que la combinación de lógica difusa con redes neuronales no es solamente recomendable para lidiar con las incertidumbres y los sistemas no lineales. Sino también cumplir con los requerimientos de precisión y adaptación en línea. Además, se han detectado problemas con la configurabilidad y aprendizaje de este tipo de sistemas; por lo que se han propuesto soluciones por medio de aprendizaje por refuerzo, para ser capaz de realizar adaptación en línea del controlador. [34]

Por último, en los últimos años se ha visto un uso cada vez mayor de algoritmos de aprendizaje por refuerzo para el desarrollo de controladores inteligentes. Estos aprovechan el concepto de redes neuronales para poder actuar en ambientes con alta incertidumbre, además del aprendizaje por refuerzo para entender que acciones son mejores dado un escenario aleatorio. Esto lo que permite es una alta versatilidad en sistemas muy complejos, así como robustez.

4 Metodología

En este capítulo se explica la metodología, la cual consiste en resolver un problema dividiéndolo en subtarefas más pequeñas. Para esto se planteó el objetivo general y se dividió en cuatro objetivos específicos, los cuales a su vez se dividieron en tareas más pequeñas de manera que permita su consecución.

El objetivo general consiste en que el sistema actual de control no se encuentra optimizado para maximizar el uso del propelente metálico. La propuesta de solución es desarrollar dos sistemas de control avanzados y novedosos, uno por medio de algoritmos clásicos de teoría de control y otro por medio de algoritmos de IA aplicados al control automático. El desarrollo de este tipo de sistema se divide en tres etapas, como se muestra en la Figura 4-1. El presente proyecto se limitará al desarrollo del sistema de control y la optimización, mientras que, en paralelo, en la Escuela de Ingeniería Electrónica se estará trabajando en el área del desarrollo del sistema de estimación de la orientación. Posteriormente, en otro proyecto, se realizará la programación en el microcontrolador de manera que finalmente se pueda implementar en el satélite para su lanzamiento.

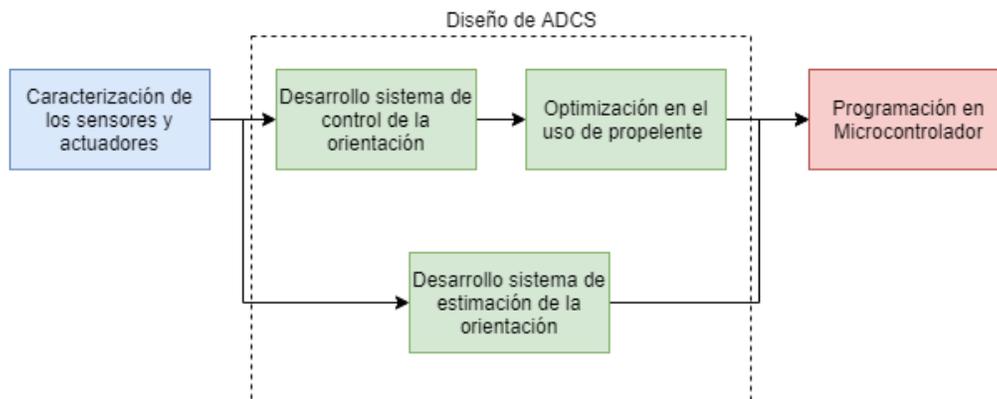


Figura 4-1: Etapas del desarrollo de un ADCS para el GWSat

Fuente: Elaboración propia

En la Figura 4-2, se muestra la relación que existirá entre los objetivos específicos presentados en la sección 1.3, para la consecución del objetivo específico.

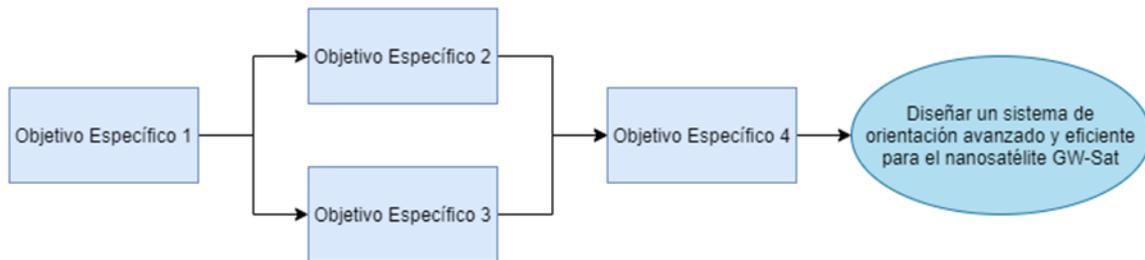


Figura 4-2: Relación entre los objetivos específicos

Fuente: Elaboración propia

4.1 Determinación de los métodos de control

Este primer objetivo representa un paso crucial para poder continuar con el desarrollo de los futuros objetivos. En este caso se utilizará como entregable una serie de tablas morfológicas en las cuales se tomarán diferentes aspectos para determinar el mejor candidato para ambos casos. En la Figura 4-3 se muestra el diagrama de flujo del primer objetivo específico.

La primera actividad se basa en el estudio de las técnicas de control que se utilizan en la actualidad tanto en casos generales como para casos de control de satélites de manera que se pueda tener un amplio panorama de las opciones que existen y con las que se podría trabajar.

La segunda actividad consiste en la generación de propuestas en base a los conocimientos obtenidos de la investigación bibliográfica. Se pretende generar por lo menos tres candidatos de cada control. Seguidamente, la tercera consiste en tomar los candidatos que se generaron en el paso anterior y compararlos entre sí por medio de tablas morfológicas, de manera que se pueda tener una evaluación objetiva de los candidatos.

Por último, la cuarta actividad consiste en tomar la comparación de candidatos que se hizo en el paso anterior y elegir de esta el candidato con el mejor puntaje para cada uno de los casos. Una vez que se concluya este paso se habrá cumplido con el primer objetivo específico del proyecto.

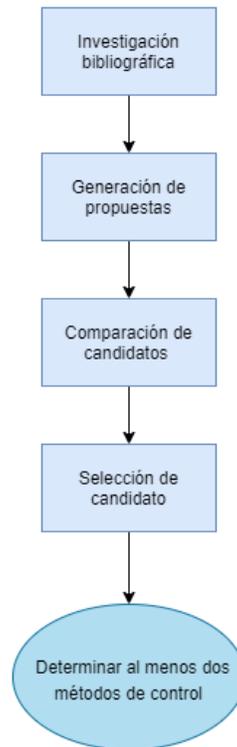


Figura 4-3: Diagrama de flujo del objetivo específico 1

Fuente: Elaboración propia

4.2 Diseño de los algoritmos de control

Los objetivos 2 y 3 se tratarán en una sola sección, pues ambos consisten en el diseño de métodos de control para la orientación del satélite. También, se desarrollarán en paralelo para aprovechar la similitud entre ambos.

La primera actividad que se muestra en la Figura 4-4, se refiere al estudio previo que se realiza en el Capítulo 2, donde se describen las características más importantes del satélite.

Esto es importante pues describe el contexto en el cual se estará trabajando para el desarrollo del método de control.

La segunda actividad se centra en los parámetros de desempeño tanto mínimos como ideales que se quieren alcanzar con respecto al sistema de control. Para este caso, se utilizó el error en estado estacionario y el tiempo de estabilización que fueron definidos por la GWU, a la hora de plantear el proyecto, como requisitos mínimos. De manera ideal, se planteó una mejora porcentual sobre el control previamente desarrollado.

La tercera actividad consta del desarrollo del algoritmo. Una vez definidos los parámetros de desempeño deseados se inició con el proceso iterativo del desarrollo del algoritmo, luego la validación de este algoritmo y la optimización de este, iterando cuantas veces sea necesario para que se alcance un valor que se considera satisfactorio.

La validación del algoritmo se hizo por medio de simulaciones de manera que luego de cada iteración se pudiera verificar que el sistema seguía funcionando luego de los cambios realizados en la iteración actual.

La optimización del algoritmo se dio en base a la eficiencia en el uso del propelente metálico de los propulsores μ CAT. Para esto, se utilizó como medida del uso de propelente el área bajo la curva de la señal de control. Después de cada iteración, se buscará reducir el área bajo la curva de las señales de control. Esta reducción se seguirá dando hasta que se logre llegar a un nivel de eficiencia que se considera satisfactorio para los objetivos del proyecto. Este estará entre el valor máximo y el valor ideal; o, incluso, si es posible habrá sobrepasado el nivel ideal.

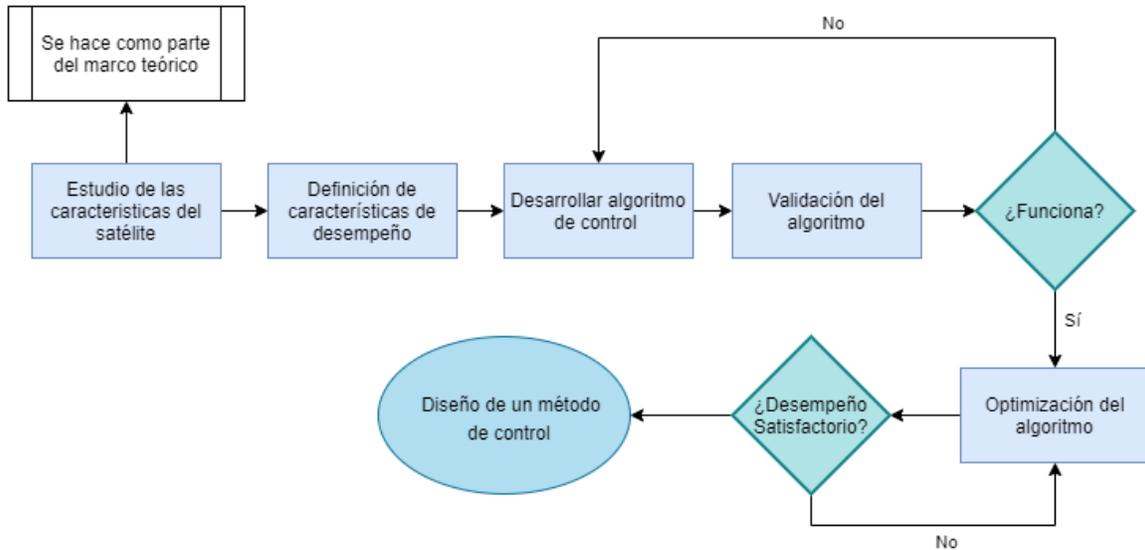


Figura 4-4: Diagrama de flujo del objetivo específico 2 y 3

Fuente: Elaboración propia

4.3 Comparación de los métodos de control

El cuarto objetivo se centra en el objetivo principal del proyecto; encontrar el método de control que cumpla con los requisitos especificados por GWU, pero además logre la mayor eficiencia en el uso de propelente, de manera que se pueda alargar la vida útil de la misión. Para lo cual previamente se habrían desarrollado dos algoritmos de control por medio de dos paradigmas distintos, buscando maximizar la eficiencia de cada uno y ahora se compararán entre sí, para encontrar el más eficiente entre ambos.

El primer paso será la verificación de los algoritmos, esto pues, en caso de que alguno de los algoritmos no funcione, se tiene que corregir esto en primera instancia para poder continuar con el desarrollo del proyecto.

El segundo paso será la definición de cuales características se utilizarán para realizar la comparación de eficiencia. Este paso es importante pues la definición de esto hará que se tenga una medida cuantitativa de la eficiencia y por lo tanto una comparación cuantitativa también.

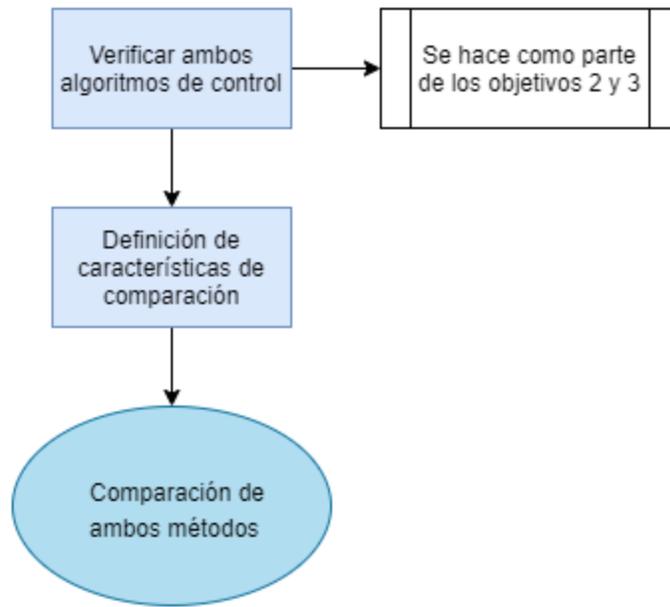


Figura 4-5: Diagrama de flujo del objetivo específico 4

Fuente: Elaboración propia

5 Resultados y análisis

5.1 Ambiente de simulación

Es importante definir las condiciones en las cuales se realizaron las pruebas del sistema de control del satélite, pues estas definen la dinámica del sistema y el marco de referencia inercial y del cuerpo inicial. Estas condiciones se detallan a continuación:

- Altitud: 400 km
- Inclinação: 51.64°
- Excentricidad: 0
- $Q_0 = [0, 0, 0, 1]$
- $\omega_0 = [0.01, 0.01, 0.01]$ rad/s
- Masa = 4 kg
- Dimensiones = 10x10x34 cm
- Perturbaciones:
 - Arrastre Atmosférico
 - J2
 - Gradiente Gravitacional

Para el desarrollo de la red neuronal también fue importante definir una función de *reset* para darle una mayor robustez y variabilidad al conjunto de datos de la red neuronal, logrando de esta manera un entrenamiento más completo. Esta función se define en las ecuaciones (5-1) y (5-2).

$$Q_0 = [rand, rand, rand, 1] \tag{5-1}$$

$$\omega_0 = [rand * 0.01, rand * 0.01, rand * 0.01] \tag{5-2}$$

Por último, para una órbita de 400km se tiene un tiempo orbital de 5600s.

5.2 Antecedentes

En esta sección se muestran los resultados obtenidos por control actual del GW-Sat, el cual como se explicó anteriormente, es un control de tipo PD.

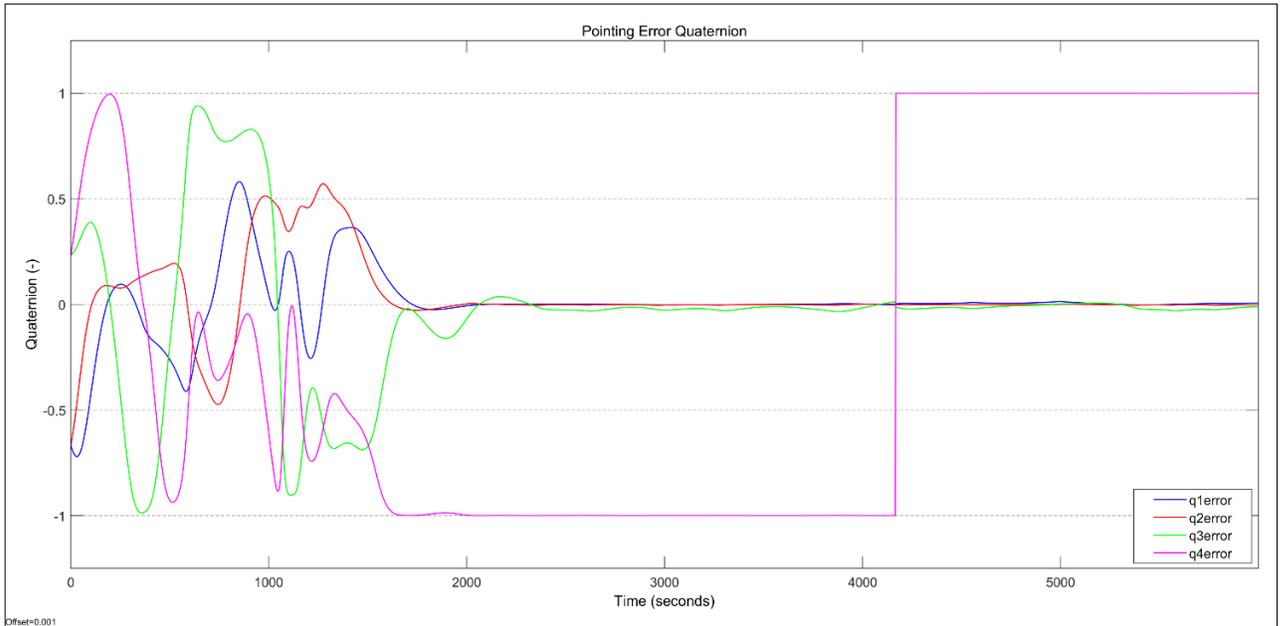


Figura 5-1: Error de los cuaternios del controlador PD. Matlab 2020a

Fuente: [11]

En la Figura 5-1 se puede observar el error en los cuaternios de orientación. En este caso se puede ver como el tiempo de estabilización promedio de estos es cercano a los dos mil segundos. Además, se puede ver como el q4 mantiene su error en un valor absoluto de uno una vez que el sistema estabiliza, esto se da por la propia naturaleza de los cuaternios que cuando el error de un cuaternio es $\mathbf{q} = [0 \ 0 \ 0 \ 1]$ representa que el error es cero y por lo tanto que se tiene la orientación deseada. También es claro como el sistema pasa por varios picos hasta que finalmente logra estabilizarse, mostrando de esta manera un sobreimpulso alto al igual que un tiempo de estabilización alto, generalmente asociado a alguna de dos razones:

- Torque de control muy alto y oscilatorio, sobre-compensando por el error
- Torque de control muy bajo, fallando en corregir a tiempo

Más adelante se determina la razón de este comportamiento. Además, en la Tabla 5-1 se resumen las características de esta gráfica.

Tabla 5-1: Tabla resumen del error de los cuaternios para el control PD.

Fuente: Elaboración propia

Error Cuaternios PD	
Ts Cuaternio 1 (s)	2050
Ts Cuaternio 2 (s)	2050
Ts Cuaternio 3 (s)	2400
ESS Cuaternio 1	0.002
ESS Cuaternio 2	0.003
ESS Cuaternio 3	0.03

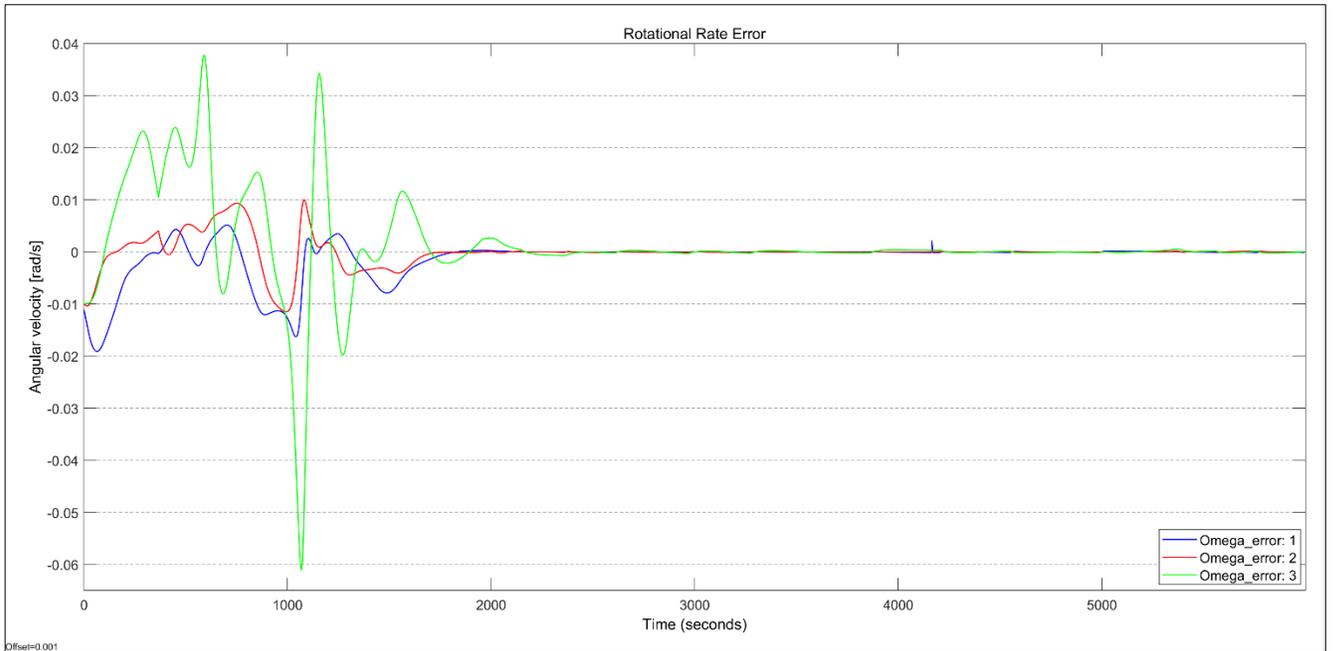


Figura 5-2: Error de la velocidad angular del controlador PD. Matlab 2020a

Fuente: [11]

En la Figura 5-2 se puede observar el error con respecto a la referencia del sistema para el controlador PD. En este caso, al igual que con los cuaternios se puede observar que el tiempo de estabilización en promedio está cerca de los dos mil segundos. Además, es notable como el error de la velocidad angular en el eje Z (Omega_error:3) tiene picos considerables previo a su estabilización. También, al igual que en el caso de los cuaternios el sistema presenta alto sobreimpulso y alto tiempo de estabilización, por las mismas potenciales razones. En la Tabla 5-2 se resumen las características de la gráfica del error de la velocidad angular.

Tabla 5-2: Tabla resumen del error de la velocidad angular para el control PD.

Fuente: Elaboración propia

Error Velocidad Angular PD	
Ts Omega 1 (s)	2100
Ts Omega 2 (s)	2100
Ts Omega 3 (s)	2500
ESS Omega 1	3.6×10^{-5}
ESS Omega 2	9.0×10^{-5}
ESS Omega 3	2.7×10^{-4}

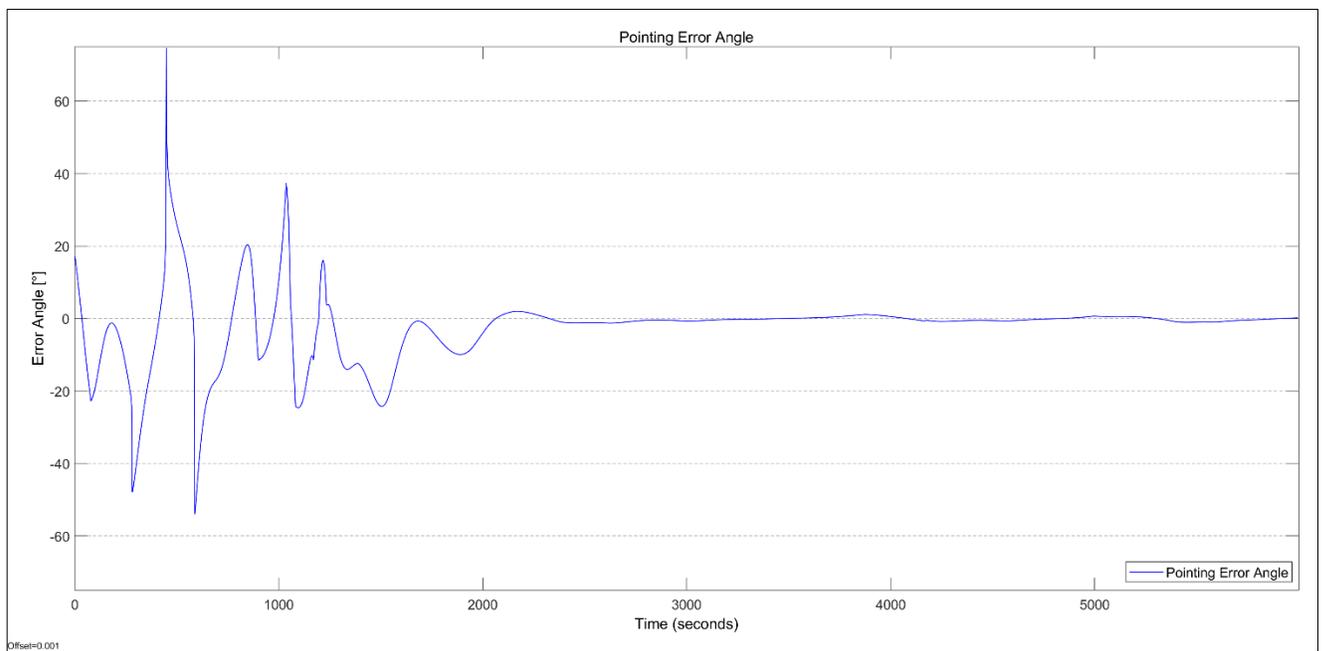


Figura 5-3: Ángulo de error del controlador PD. Matlab 2020a

Fuente: [11]

En la Figura 5-3 se puede observar el ángulo de error promediado entre los 3 ejes. Esta es una manera más simple de ver el error en la orientación que se vio en la Figura 5-1.

Se puede ver un comportamiento similar al del error de los cuaternios en el sentido de que se pueden ver una serie de picos, de los cuales el mayor es un pico de error de aproximadamente setenta grados alrededor de los quinientos segundos. Además, se puede ver un tiempo de estabilización cercano a los 2000 segundos y un sobreimpulso alto. En la Tabla 5-3 se puede ver un resumen de las características del error del ángulo de puntería.

Tabla 5-3: Resumen del error del ángulo de puntería para el control PD.

Fuente: Elaboración propia

Error Angulo Puntería PD	
Ts (s)	2400
ESS	1.05

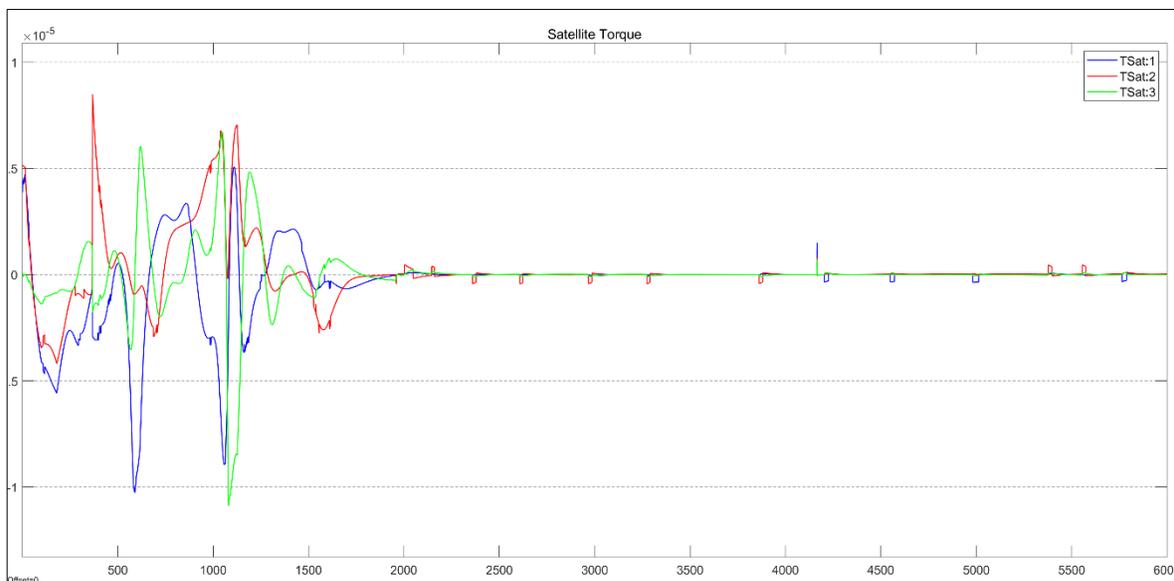


Figura 5-4: Torque generado por el controlador PD. Matlab 2020a

Fuente: [11]

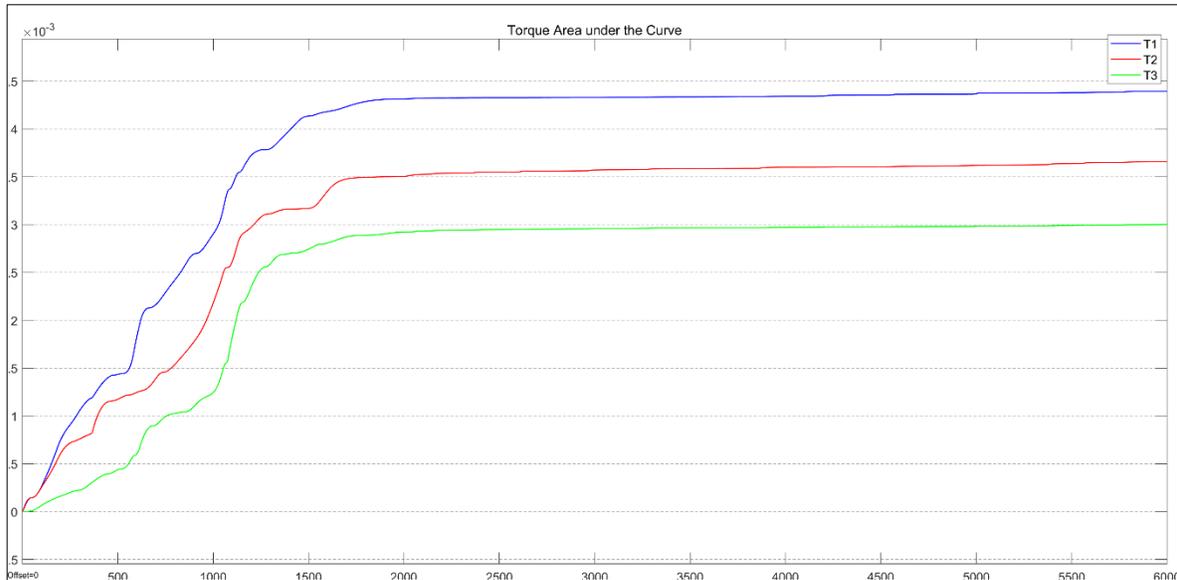


Figura 5-5: Torque acumulado del controlador PD. Matlab 2020a

Fuente: [11]

En la Figura 5-4 y Figura 5-5 se puede observar las gráficas que corresponden al parámetro de consumo energético que interesa para el uso eficiente de recursos. En la Figura 5-4 se puede ver como una vez que el sistema se estabiliza solamente se necesitan pequeños ajustes de torque para mantener estable el sistema. Además, se puede ver como hay picos cercanos a 0.8×10^{-5} Nm antes del punto de estabilización. De esta manera se puede ver como la justificación del alto sobreimpulso y prolongadas oscilaciones era precisamente que el torque de control es muy alto y oscilatorio, por lo tanto, sobre-compensando por el error.

En la Figura 5-5 se puede notar como el torque ejercido sobre el eje x es el mayor de todos cuando se considera el torque ejercido acumulado, seguido del torque en el eje Y y por último el eje Z. Además, se puede ver que el acumulado de todos se encuentra entre 3×10^{-3} y 4.5×10^{-3} Nm. En la Tabla 5-4 se presenta un resumen de las características del torque ejercido sobre el sistema.

Tabla 5-4: Tabla resumen del torque ejercido para el control PD.

Fuente: Elaboración propia

Torque PD	
Ts T1 (s)	2000
Ts T2 (s)	2000
Ts T3 (s)	2000
Acumulado T1 (Nm)	0.0044
Acumulado T2 (Nm)	0.0037
Acumulado T3 (Nm)	0.0030

5.3 Determinación de los métodos de control

En esta sección se muestran los resultados obtenidos en el proceso de la determinación de los métodos de control para el satélite. En primera instancia se muestran las propuestas generadas, luego se explican los parámetros que serán evaluados para la selección de los candidatos. Después, se muestran las tablas morfológicas generadas para seleccionar cuantitativamente al mejor candidato de cada uno de los objetivos.

5.3.1 Propuestas generadas para el control clásico

Como parte del proceso de diseño de ingeniería, la generación de propuestas es un paso importante en el diseño de cualquier solución. Para el caso de este proyecto se planteó que se iban a generar como mínimo 3 propuestas para cada uno de los paradigmas de controladores.

En el caso del controlador clásico se decidió tomar como punto de partida el hecho de que se tiene actualmente un controlador PD, por lo que la primera propuesta fue la de un controlador PID de manera que corrigiera totalmente el error de estado estacionario. Luego, se consideraron los paradigmas de control moderno por medio de realimentación de estados.

Para estos, se tomaron en cuenta la realimentación de estados regular (RE), el regulador cuadrático lineal (LQR) y el regulador cuadrático gaussiano (LQG).

5.3.2 *Parámetros de evaluación de los controladores clásicos*

Dado que la selección de candidatos de manera ideal tiene que ser un proceso cuantitativo para ser lo más objetivo posible, se tienen que seleccionar parámetros que sean cuantificables. Para el caso de los controladores clásicos se plantearon 4 parámetros que se evaluarán y además se le asignó un peso a cada uno dependiendo de la importancia de estos.

Los parámetros planteados son:

- Tiempo de desarrollo
- Coste computacional
- Configurabilidad
- Manejo de la eficiencia del control

Para el tiempo de desarrollo se tomó en cuenta las capacidades y funciones preexistentes del software MATLAB para el diseño de este tipo de controladores. En el caso del coste computacional se hizo una prueba de concepto en la cual se corría una vez la simulación del modelo con los diferentes controladores y se midió su tiempo de ejecución. La configurabilidad se evalúa por medio de la experiencia del diseñador y por medio de consulta bibliográfica. Por último, el manejo de la eficiencia del control se evalúa por medio de consulta bibliográfica.

5.3.3 *Selección de candidatos del controlador clásico*

En esta subsección se muestra el procedimiento de selección de candidatos para el controlador clásico, además se analiza los valores que se obtuvieron para la selección de estos.

La selección de los candidatos se hizo por medio de una tabla morfológica, como se muestra en la Tabla 5-5. Para esta se decidió que para los parámetros previamente definidos se tendría una calificación del 1 al 5 dependiendo de qué tan bueno es dicho paradigma con respecto al rubro que se evaluó.

Tabla 5-5: Tabla morfológica de los candidatos de controladores clásicos.

Fuente: Elaboración Propia

Peor 1	2	Moderado 3	4	Excelente 5	
Tipo de Control	Tiempo de Desarrollo	Coste Computacional	Configurable	Manejo de la eficiencia del control	TOTAL
Peso	6	4	10	6	130
PID	5	2	3	3	86
LQR	3	5	5	5	✓ 118
RE	4	5	4	3	102
LQG	1	5	5	5	106

De la Tabla 5-5 se puede ver como el controlador LQR se considera la mejor estrategia de control para el problema dado. Aunque presenta el segundo peor tiempo de desarrollo, este sobresale con respecto al resto de los candidatos en el resto de los rubros evaluadores.

Con respecto al tiempo de desarrollo se consideró la existencia de la función “lqr” dentro del *toolbox* de control de MATLAB, lo que ahorra mucha programación. Por lo tanto, lo que más afecta el tiempo de desarrollo de este algoritmo es su naturaleza iterativa. En la que hay que modificar los valores de las matrices Q y R hasta que se tenga el desempeño deseado.

Para el caso del coste computacional se hicieron 50 corridas de los controladores PD y LQR para medir su tiempo de ejecución en el modelo del satélite, la duración de las iteraciones se puede encontrar en el apéndice A.1. Dado que los controladores RE y LQG funcionan igual que el LQR a la hora de implementar el controlador en un bloque, se asume

que el tiempo de ejecución de estos será el mismo. Además, se asume que el tiempo de ejecución del controlador PID será el mismo del controlador PD ya implementado. Los tiempos de ejecución se pueden ver en la Tabla 5-6.

Tabla 5-6: Tiempos de ejecución controladores.

Fuente: Elaboración Propia

Controlador	Tiempo de Ejecución (s)
PD	19.45
LQR	16.24

Para determinar la configurabilidad de los diferentes métodos se consideró la cantidad de variables sobre las cuales se tiene control. En el caso del controlador PID se pueden variar solamente tres variables, las cuales son las ganancias K_p , K_i , y K_d . En el método de realimentación de estados se pueden seleccionar 6 polos distintos dado que ese es el orden del sistema. Mientras tanto, para el caso del LQR y LQG se pueden variar los 6 parámetros de la matriz Q , que representan los estados y los 3 parámetros de la matriz R , que representa las acciones; para un total de 9 valores que se pueden variar, dando de esta manera la mayor configurabilidad de todos los algoritmos.

La importancia del manejo de la eficiencia del control se basa en el objetivo de maximizar la eficiencia en el uso del propelente para poder alargar la vida útil del satélite lo más posible. En este caso se analizó como la relación entre las variables que se pueden variar y el esfuerzo de control. Dado que en el LQR lo que se está limitando es el esfuerzo de control de manera directa por medio de las matrices Q y R , este presenta una calificación de excelente en este rubro. Mientras tanto en el caso del RE y PID, aunque se pueden controlar parámetros como el sobreimpulso y tiempo de estabilización, el esfuerzo de control no se puede controlar directamente, solo indirectamente.

5.3.4 *Propuestas generadas para el control inteligente*

Como se explicó anteriormente, en el proceso de diseño de ingeniería, la generación de propuestas es importante para un diseño objetivo y óptimo. En este caso se plantearon nuevamente 4 propuestas para un controlador inteligente.

En primera instancia se tomó el conocimiento adquirido en el curso de Inteligencia Artificial, con el cual se propuso un controlador por medio de aprendizaje supervisado. El cual se encargaría de mapear una diversidad de distintos controladores para el sistema planteado en los escenarios planteados. De esta manera, se obtuvo un controlador que “aprendió” de otros controladores clásicos.

Luego, se exploró el concepto de aprendizaje por refuerzo. Este tipo de aprendizaje se ha visto utilizado en diversas ocasiones para el control de diversos sistemas [27, 34, 26, 25, 24, 35]. Este se encargaría de explorar las diferentes opciones que existen para las acciones y observar y aprender del resultado que arrojan los estados, de aquí se tomaron las técnicas de aprendizaje TD y aprendizaje Q.

También, basándose en el aprendizaje por refuerzo, se tomó en cuenta los paradigmas de aprendizaje profundo por refuerzo. El cual combina las características de manejar modelos complejos por medio del aprendizaje profundo y además de aprender los mejores pares acciones-estados por medio del aprendizaje por refuerzo [27, 26, 25].

5.3.5 *Parámetros de evaluación de los controladores inteligentes*

En este caso se definieron 6 parámetros distintos con sus respectivos pesos. Los pesos se asignaron de manera que los parámetros más importantes para que la consecución de los objetivos sea óptima sean los de mayor peso

Los parámetros definidos son:

- Tiempo de desarrollo
- Coste computacional
- Manejo de la eficiencia del control
- Espacio de acción continuo

- Sobreestimación
- Robustez

Para el tiempo de desarrollo, al igual que en la sección 5.3.2, se tomó en cuenta la existencia de funciones preexistentes dentro de los *toolbox* que nos ofrece MATLAB. Para el coste computacional, también se utilizó la misma técnica de la sección 5.3.2, además de tomar en cuenta el tamaño de las redes neuronales. Para el manejo de la eficiencia del control se utilizó revisión bibliográfica para determinar la capacidad de cada uno de los algoritmos para variar al gusto del diseñador el esfuerzo de control.

El caso del espacio de acción continuo representa el primer parámetro que se evalúa solamente en el caso de los controladores inteligentes. Esto se debe a que en el área de control por medio de IA hay algoritmos que ofrecen espacios continuos de acción, mientras que otros solamente ofrecen campos discretos. Este parámetro se considera el más importante de todos pues el sistema existente funciona en tiempo continuo, y remodelarlo en esta etapa del proyecto no resulta factible.

Para el caso de la sobreestimación, se evalúa por medio de lo que la literatura indica. Este parámetro es importante pues, aunque en muchos casos no sucede, la sobreestimación del valor del agente puede llegar a presentar problemas serios de aprendizaje, como se indica en la sección 3.6.6

Por último, se tiene el rubro de la robustez. Se decidió que este es el segundo parámetro de mayor importancia a la hora de seleccionar un candidato para la consecución del objetivo. Esta decisión se basa en el hecho que gran parte del sentido de utilizar un control inteligente es aumentar considerablemente la robustez de este, en especial en los casos de situaciones impredecibles o inmodelables. Además, un controlador que presente una baja robustez puede presentar problemas que no se verían con un control más robusto.

5.3.6 Selección de candidatos del controlador inteligente

En esta subsección se muestra el procedimiento de selección de candidatos para el controlador clásico, además se analiza los valores que se obtuvieron para la selección de estos.

En este caso, al igual que en el caso anterior, se utilizó una tabla morfológica para tomar una decisión cuantitativa sobre cuál es el mejor algoritmo de IA para lograr los objetivos planteados. Estos resultados de se ven reflejados en la Tabla 5-7.

Tabla 5-7: Tabla morfológica de los candidatos de controladores por IA.

Fuente: Elaboración Propia

Peor 1	2	Moderado 3	4	Excelente 5			
Tipo de Control	Tiempo de Desarrollo	Coste Computacional	Manejo de la eficiencia del control	Espacio de acción continuo	Sobreestimación	Robustez	TOTAL
Peso	6	4	8	15	4	10	235
Aprendizaje Supervisado	5	5	2	5	3	1	163
Q-learning	4	3	4	1	3	3	125
DDPG	4	2	5	5	3	5	209
TD3	4	1	5	5	5	5	✓ 213

De la Tabla 5-7 se puede ver como el mejor candidato es el TD3 solo cuatro puntos por encima del DDPG, pero muy por encima del resto. Lo más notable es que, aunque presenta el peor coste computacional, este sobresale en el resto de los rubros, siendo igual o mejor a cualquiera de los otros.

En el rubro del tiempo de desarrollo se puede ver como el candidato con la mayor puntuación es el de aprendizaje supervisado. Esto se debe a varias razones. Las principales son la cantidad de información que hay sobre este tipo de redes neuronales y las funciones y bibliotecas altamente probadas y de fácil uso para diseñar una de estas redes. En el caso de

los otros se puede ver como igual tienen una calificación alta, esto se debe a que a partir de Matlab r2019a se implementó un *toolbox* que es capaz de crear agentes de aprendizaje por refuerzo de manera simple y amistosa con el usuario. Este *toolbox* se llama *Reinforcement Learning Toolbox*.

Con respecto al coste computacional lo que se toma en cuenta es el tiempo de entrenamiento que necesita cada uno los candidatos para llegar a un valor deseado. Esto, como se sabe, depende en gran manera del tamaño de la red neuronal. Por lo tanto, en el caso del aprendizaje supervisado, el cual una vez más es el de la mejor calificación, éste toma muy poco tiempo de entrenamiento debido a que solo se trata de mapear las acciones y estados actuales de un set de datos preexistentes. En el caso del aprendizaje Q este se trata de una sola red neuronal de un tamaño grande, por lo cual su calificación baja con respecto al aprendizaje supervisado. El DDPG al estar formado por dos redes neuronales distintas, duplica el tamaño total de entrenamiento, por lo que se vuelve aún más lento el entrenamiento. Por último, el TD3 presenta la peor calificación de todos. Esto se debe a que necesita de tres redes neuronales distintas que se tienen que entrenar, por lo tanto, aumentando el coste computacional necesario para su entrenamiento.

Cuando analizamos el rubro del manejo de la eficiencia del control podemos ver como el aprendizaje supervisado tiene la peor calificación, seguido del aprendizaje Q y finalmente DDPG y TD3. Lo que hace que el aprendizaje supervisado tenga esta calificación es el hecho de que no se tiene control sobre cuál será su eficiencia final, pues esto depende solamente del set de datos que se tenga. El aprendizaje Q, el DDPG y el TD3 presentan una buena calificación. Esto se debe a que como parte de las recompensas que se le dan a manera de refuerzo a la red neuronal, se le puede dar la importancia del caso a las acciones de entrada por medio de recompensas si se acerca a un valor bajo y castigos si más bien se aleja mucho. La diferencia en puntaje entre el aprendizaje Q y los otros se basa en que en el caso de este el campo de acción es discreto por lo cual no tendrá tantas opciones para encontrar una manera eficiente de manejar el esfuerzo de control, mientras que los otros, al ser campos de acción continuos tienen una infinidad de opciones.

El rubro del espacio de acción continuo constituye el parámetro de mayor peso de todos. Como se explicó anteriormente, esto se debe a que el sistema está modelado en tiempo

continuo, y un remodelado a tiempo discreto no es factible en esta etapa de desarrollo. Además, en este caso se trata de una evaluación de tipo binaria, en esencia, si funciona en tiempo continuo se le asigna un 5, si no entonces se le asigna un 1.

La sobreestimación del valor de la red neuronal se explicó en la sección 3.6.6. En el caso del aprendizaje supervisado, este se conoce como sobre entrenamiento, y se basa en el hecho que, si se entrena demasiado a una red neuronal sobre el mismo set de datos, esta se adaptará de manera excelente a este set, pero solamente a este, por lo que se vuelve poco versátil. [19]. Por esa razón se le da la calificación que tiene al aprendizaje supervisado. En el caso del aprendizaje Q y el DDPG sucede lo mismo, el valor Q de la red neuronal comienza a estimar que la recompensa es mejor de lo que realmente es y comienza a propagar este error hasta el punto en el que esta sobreestimación causa inclusive un desaprendizaje de la red. En el caso del TD3, este tipo de arquitectura se enfocó en corregir estos errores y entonces se encarga más bien de subestimar este valor, ya que las subestimaciones no se propagan en el aprendizaje de una red neuronal [26]. Por esta razón, el TD3 recibe la calificación más alta.

La robustez se evaluó considerando aspectos claves que se mencionan en la literatura. En el caso del aprendizaje supervisado este tiene una muy baja robustez, esto se debe a que se adapta muy bien a un set de datos. Pero su adaptabilidad en general depende totalmente de la calidad del set de datos, en casos que este no sea de buena calidad la robustez será muy baja y por lo tanto será propensa a errores fácilmente. En el caso del aprendizaje Q la mayor afectación que este posee es que solo tiene una red neuronal para predecir qué tan buena será una acción, mientras que el DDPG y el TD3 tienen dos redes neuronales, una para predecir qué tan buena será una acción y otra para decidir qué acción tomar dado un estado. Por esta razón el DDPG y el TD3 tienen la calificación más alta.

5.4 Controlador LQR

En esta sección se muestran los resultados obtenidos durante el diseño de un controlador de tipo LQR.

Una vez concluido el proceso de selección de candidatos, comenzó el proceso de diseño del controlador seleccionado. En este caso un Regulador Cuadrático Lineal (LQR). Para esto se siguió la serie de pasos detallada en la sección 4.2.

5.4.1 Estudio de las características del satélite

Aunque la mayoría del estudio de las características del satélite se dio como parte del marco teórico, para el desarrollo de un algoritmo de control por medio de realimentación de estados es necesario un modelo del sistema en su espacio de estados. Al momento en el que se comenzó este proyecto, el único modelo que existía era una serie de bloques que describen tanto la cinemática, la dinámica y las perturbaciones del sistema.

El problema de eso es que no se tiene un modelo concreto descrito por matrices, y para poder diseñar un controlador de realimentación de estados es completamente necesario conocer esto. Por lo tanto, se basó en el trabajo de [36] y de la ecuación (3-21) de la sección 3.2, para determinar el modelo en el espacio de estados del satélite. Este se muestra en la ecuación (5-3)

$$\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{\omega}_1 \\ \dot{\omega}_2 \\ \dot{\omega}_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5 \\ f41 & 0 & 0 & 0 & 0 & f46 \\ 0 & f52 & 0 & 0 & 0 & 0 \\ 0 & 0 & f63 & f64 & 0 & 0 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{1}{J_x} & 0 & 0 \\ 0 & \frac{1}{J_y} & 0 \\ 0 & 0 & \frac{1}{J_z} \end{bmatrix} \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \quad (5-3)$$

Donde

$$f41 = \frac{8(J_z - J_y)\omega_0^2}{J_x} \quad (5-4)$$

$$f_{46} = \frac{(-J_x - J_z + J_y)\omega_0}{J_x} \quad (5-5)$$

$$f_{52} = \frac{6(J_z - J_x)\omega_0^2}{J_y} \quad (5-6)$$

$$f_{63} = \frac{(J_x - J_y)\omega_0^2}{J_z} \quad (5-7)$$

$$f_{64} = \frac{(J_x - J_y + J_z)\omega_0}{J_z} \quad (5-8)$$

$$\omega_0 = 1.1288 \times 10^{-3}, \text{ asumiendo órbita ISS} \quad (5-9)$$

Una vez que se determinó el modelo de la ecuación (5-3), se podía proceder a la definición de las características de diseño que se deseaban que cumpliera el control.

5.4.2 Definición de las características de desempeño

Una etapa fundamental en el diseño de un sistema de control es la definición de las características de desempeño. Para el caso de este proyecto este paso se basó en primera instancia en las características mínimas establecidas por GWU. Las cuales se detallan en la Tabla 5-8.

Tabla 5-8: Parámetros de desempeño para diferentes modos de operación.

Fuente: [37]

Modo	Error Ángulo de Puntería (°)	Error Velocidad Angular (rad/s)	Tiempo de Estabilización (s)
Por Defecto	5	0.0087 en X y Y, 0.87 en Z	86 400
Apuntar Brusco	5	0.0087	86 400
Apuntar Fino	1	0.0017	86 400

Por lo tanto, estos se definieron como los parámetros mínimos que se tienen que cumplir para que la solución sea aceptable. Además de esto también es importante analizar el desempeño del control PD. Como se analizó en la subsección anterior el desempeño del sistema PD se resume en la Tabla 5-9.

Tabla 5-9: Parámetros de desempeño control PD.

Fuente: Elaboración propia

Control PD	
Tiempo de estabilización(s)	2000
Error Velocidad Angular (rad/s)	0.0004
Error Ángulo de Puntería (°)	1.126
Consumo de Torque (N*m)	0.01
Índice de Rendimiento	0.19

Es importante mencionar que, para el tiempo de estabilización, así como para error en la velocidad angular se tomó el mayor valor entre los ejes y se tomó como el parámetro que define al sistema.

Ya teniendo definido el desempeño del control PD, se obtuvo una base de parámetros de la cual se tendrá que mejorar. La prioridad siempre fue el consumo energético, por lo tanto, este tuvo un mayor peso que los otros parámetros a la hora de decidir si un algoritmo es mejor que otro, siempre y cuando se cumplan las características mínimas que planteó GWU (Tabla 5-8).

Para comparar el rendimiento de los controladores se creó un parámetro llamado “Índice de Desempeño”. Esto se hace de manera que se pueda comprar directamente los controladores tomando en cuenta todos sus parámetros de manera cuantitativa. El proceso de cómo se calculó e ideó este índice se explica en la próxima sección.

5.4.3 Desarrollo del algoritmo de control

En esta subsección se analizarán las iteraciones más importantes durante el desarrollo del algoritmo de control. La totalidad de las iteraciones que se realizaron durante el diseño de este controlador se pueden encontrar en el apéndice A.2, en este se encontrará las matrices Q y R utilizadas, así como los resultados de todas.

Para explicar el desarrollo del algoritmo de control, es importante recordar que el diseño de un controlador LQR depende de sus matrices Q y R. También, se creó el “Índice de Desempeño” para comparar los diferentes algoritmos.

El proceso de desarrollo del algoritmo de control LQR comenzó con la definición de una matriz Q inicial al igual que una matriz R inicial. Esta se definió según recomienda [28] y se muestra en las ecuaciones (5-10) y (5-11).

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (5-10)$$

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (5-11)$$

Estas matrices dieron los resultados que se muestran en la Tabla 5-10.

Tabla 5-10: Resultados primera iteración LQR.

Fuente: Elaboración propia

Iteración 1	
Tiempo de estabilización(s)	No estabiliza
Error Velocidad Angular (rad/s)	NA
Error Ángulo de Puntería (°)	NA
Consumo de Torque (N*m)	NA
Índice de Rendimiento	0

Como se puede ver en la Tabla 5-10, esta primera iteración no tuvo un comportamiento satisfactorio. Por lo tanto, se tuvo que seguir el procedimiento que se detalla en la sección 4.2, el cual indica que el siguiente paso fue el desarrollo de un nuevo sistema de control. Este proceso iterativo se continuó hasta la quinta iteración en donde se tuvo el primer sistema que pudo estabilizar la planta. Los resultados de esta iteración se muestran en la Tabla 5-11. Es importante recordar que en el apéndice A.2 se encuentran los detalles de cada iteración del diseño de este controlador.

Tabla 5-11: Resultados quinta iteración LQR.

Fuente: Elaboración propia

Iteración 5	
Tiempo de estabilización(s)	1400
Error Velocidad Angular (rad/s)	0.00020
Error Ángulo de Puntería (°)	0.05
Consumo de Torque (N*m)	0.08047
Índice de Rendimiento	0.16

Esta iteración fue crucial para el desarrollo del proyecto pues logró demostrar que el sistema se puede estabilizar por medio de un controlador LQR. Siguiendo el procedimiento que se planteó en la sección 4.2, podemos ver como con esta iteración se logró la validación del algoritmo y por lo tanto el siguiente paso fue la optimización del algoritmo, en especial dado que el rendimiento de este fue peor que el del control PD.

Al comenzar el proceso de optimización, surgió un problema. No había una forma cuantitativa de comparar todos los parámetros de desempeño de un controlador de manera que se pudieran comparar objetivamente. Por esta razón se ideó el Índice de Rendimiento, el cual nos permite eliminar cualquier subjetividad que se pueda presentar al momento de evaluar a los controladores.

Para calcular este valor, lo primero que se realizó fue crear un “Candidato Ideal”, del cual se basaron los cálculos. Sus especificaciones se pueden ver en la Tabla 5-12. Para definir estos parámetros se tomó la decisión de mantener el requerimiento de la velocidad angular que define GWU, mientras que en el caso del ángulo de puntería se redujo su

requerimiento por dos órdenes de magnitud. Finalmente, para definir el consumo de torque ideal, se tomó el consumo del controlador PD y se redujo este en dos órdenes de magnitud.

Tabla 5-12: Características candidato ideal.

Fuente: Elaboración propia

Candidato Ideal	
Tiempo de estabilización(s)	800
Error Velocidad Angular (rad/s)	1.00E-03
Error Ángulo de Puntería (°)	0.01
Consumo de Torque (N*m)	0.001
Índice de Rendimiento	1

Con estos valores definidos y aprobados por el Dr. Adolfo Chávez se procedió a asignarle un peso a cada uno de estos para el consiguiente cálculo para cada controlador. Para esto, se llegó al acuerdo que el consumo de torque sería la mayor prioridad, por lo tanto, se le asignó un peso del 75%, a los otros tres valores se les asignó un 25% en total, para un 8,3% aproximadamente para cada uno. Además, si un valor de un controlador supera el valor ideal, este se tomará como ideal, por lo tanto, con un valor máximo de uno para cualquier parámetro.

De esta manera, la ecuación (5-12) describe el cálculo del Índice de Rendimiento

$$R = \begin{cases} \frac{0.25}{3} \left(\frac{\tau_{ideal}}{\tau_{cont}} + \frac{E\omega_{ideal}}{E\omega_{cont}} + \frac{E\theta_{ideal}}{E\theta_{cont}} \right) + 0.75 \sum |T| \\ \frac{0.25}{3} \left(1 + \frac{E\omega_{ideal}}{E\omega_{cont}} + \frac{E\theta_{ideal}}{E\theta_{cont}} \right) + 0.75 \sum |T|, \tau_{cont} < \tau_{ideal} \\ \frac{0.25}{3} \left(\frac{\tau_{ideal}}{\tau_{cont}} + 1 + \frac{E\theta_{ideal}}{E\theta_{cont}} \right) + 0.75 \sum |T|, E\omega_{cont} < E\omega_{ideal} \\ \frac{0.25}{3} \left(\frac{\tau_{ideal}}{\tau_{cont}} + \frac{E\omega_{ideal}}{E\omega_{cont}} + 1 \right) + 0.75 \sum |T|, E\theta_{cont} < E\theta_{ideal} \\ \frac{0.25}{3} \left(\frac{\tau_{ideal}}{\tau_{cont}} + \frac{E\omega_{ideal}}{E\omega_{cont}} + \frac{E\theta_{ideal}}{E\theta_{cont}} \right) + 0.75, \sum |T|_{cont} < \sum |T|_{ideal} \end{cases} \quad (5-12)$$

Una vez definida esta ecuación se continuó con el proceso de optimización del algoritmo. Esto tomo cinco iteraciones más hasta lograr un desempeño que se considera satisfactorio.

A la décima iteración se probaron las matrices Q y R que se describen en las ecuaciones (5-13) y (5-14).

$$Q = \begin{bmatrix} 10^{-9} & 0 & 0 & 0 & 0 & 0 \\ 0 & 10^{-9} & 0 & 0 & 0 & 0 \\ 0 & 0 & 10^{-9} & 0 & 0 & 0 \\ 0 & 0 & 0 & 10^{-6} & 0 & 0 \\ 0 & 0 & 0 & 0 & 10^{-6} & 0 \\ 0 & 0 & 0 & 0 & 0 & 10^{-6} \end{bmatrix} \quad (5-13)$$

$$R = \begin{bmatrix} 0.05 & 0 & 0 \\ 0 & 0.05 & 0 \\ 0 & 0 & 0.05 \end{bmatrix} \quad (5-14)$$

Lo que dio como resultado la matriz K de ganancias descrita en la ecuación (5-15).

$$K = \begin{bmatrix} 0.0001 & 0 & 0 & 0.0051 & 0 & 0 \\ 0 & 0.0001 & 0 & 0 & 0.0051 & 0 \\ 0 & 0 & 0.0001 & 0 & 0 & 0.0051 \end{bmatrix} \quad (5-15)$$

Estas matrices se obtuvieron por medio de un proceso iterativo basado en la primera iteración exitosa. Se analizó el comportamiento del controlador y se determinó que la ganancia debía ser menor, pues el consumo energético seguía siendo alto en comparación al control PD. De esta manera se logró el desempeño final, el cual se muestra en la Tabla 5-13.

Tabla 5-13: Resultados controlador LQR optimizado.

Fuente: Elaboración propia

Iteración Final	
Tiempo de estabilización(s)	1000
Error Velocidad Angular (rad/s)	0.000015
Error Ángulo de Puntería (°)	0.15
Consumo de Torque (N*m)	0.0017969
Índice de Rendimiento	0.57

5.4.4 Validación del algoritmo

En esta sección se analizarán los resultados obtenidos del controlador LQR optimizado, de manera que se tenga el mejor entendimiento posible de estos.

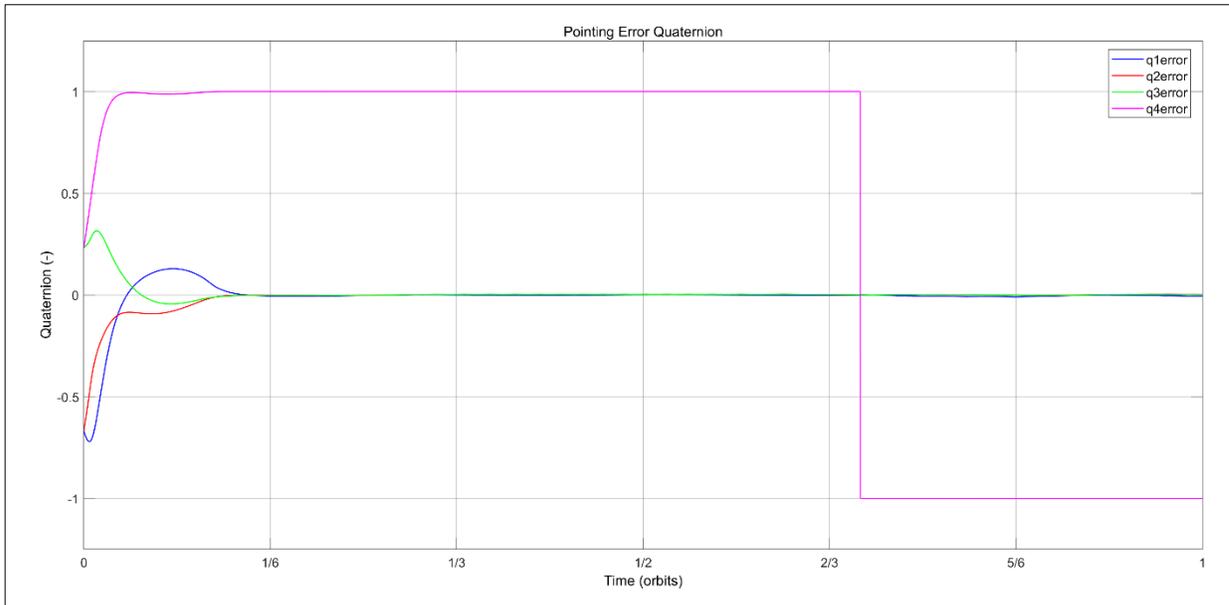


Figura 5-6: Error del cuaternio del controlador LQR.

Fuente: Elaboración propia

En la Figura 5-6 se muestra el error en los cuaternios de orientación que resultan del controlador LQR. Se puede notar como el tiempo de estabilización es cercano a los mil segundos. Además, se puede ver como el sistema presenta un bajo sobreimpulso. Cuando se compara con la Figura 5-1 se aprecia como este sistema presenta una cantidad de picos considerablemente menor. Además, tiene un tiempo de estabilización aproximadamente 50% menor. Por último, presenta un error en cualquiera de los ejes menor al controlador PD. En la Tabla 5-14 se presenta un resumen de las características de esta gráfica.

Tabla 5-14: Tabla resumen del error de los cuaternios para el control LQR.

Fuente: Elaboración propia

Error Cuaternios LQR	
Ts Cuaternio 1 (s)	950
Ts Cuaternio 2 (s)	950
Ts Cuaternio 3 (s)	950
ESS Cuaternio 1	0.0036
ESS Cuaternio 2	0.0015
ESS Cuaternio 3	0.0016

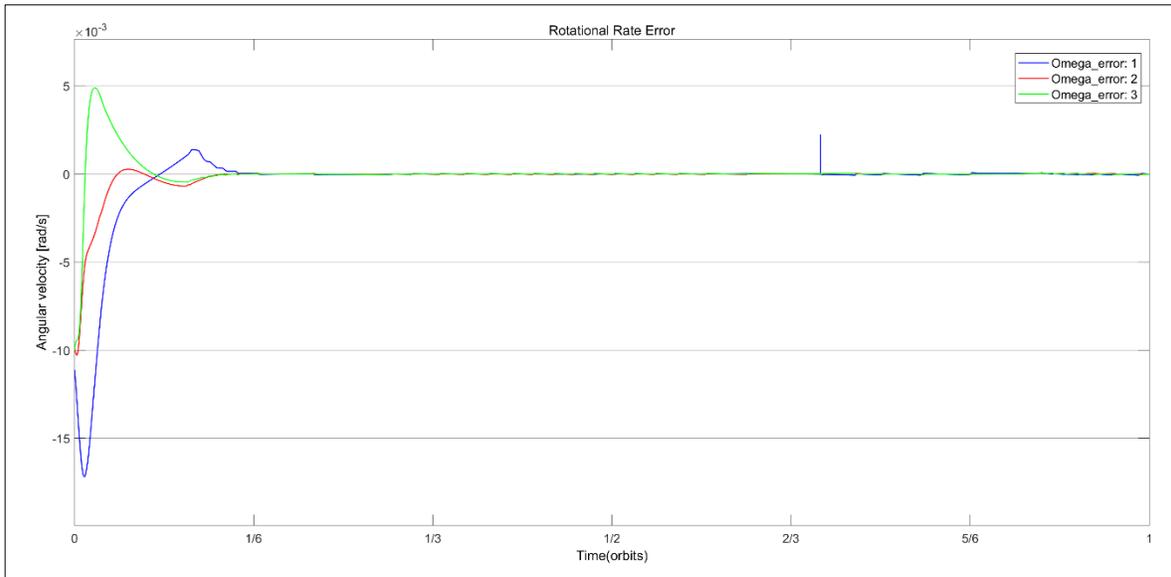


Figura 5-7: Error de la velocidad angular del controlador LQR.

Fuente: Elaboración propia

En la Figura 5-7 se aprecia el error entre la velocidad angular deseada y la velocidad angular actual en el sistema. Se puede ver como el tiempo de estabilización es cercano a los mil segundos. Además, se puede ver como el error de la velocidad angular en el eje Y no presenta sobreimpulso, mientras que el error en el eje X y Z tienen un bajo sobreimpulso. Cuando estos resultados se comparan con los resultados obtenidos en la Figura 5-2 se puede ver como se presenta un sobreimpulso de un orden de magnitud menor. Además, al igual que en la Figura 5-6 un tiempo de estabilización aproximadamente 50% menor y un error de estado estable menor también. En la Tabla 5-15 se resumen las características de la respuesta del error de la velocidad angular del sistema.

Tabla 5-15: Tabla resumen del error de la velocidad angular para el control LQR.

Fuente: Elaboración propia

Error Velocidad Angular LQR	
Ts Omega 1 (s)	950
Ts Omega 2 (s)	950
Ts Omega 3 (s)	950
ESS Omega 1 (rad/s)	2.7×10^{-5}
ESS Omega 2 (rad/s)	2.7×10^{-5}
ESS Omega 3 (rad/s)	3.0×10^{-5}

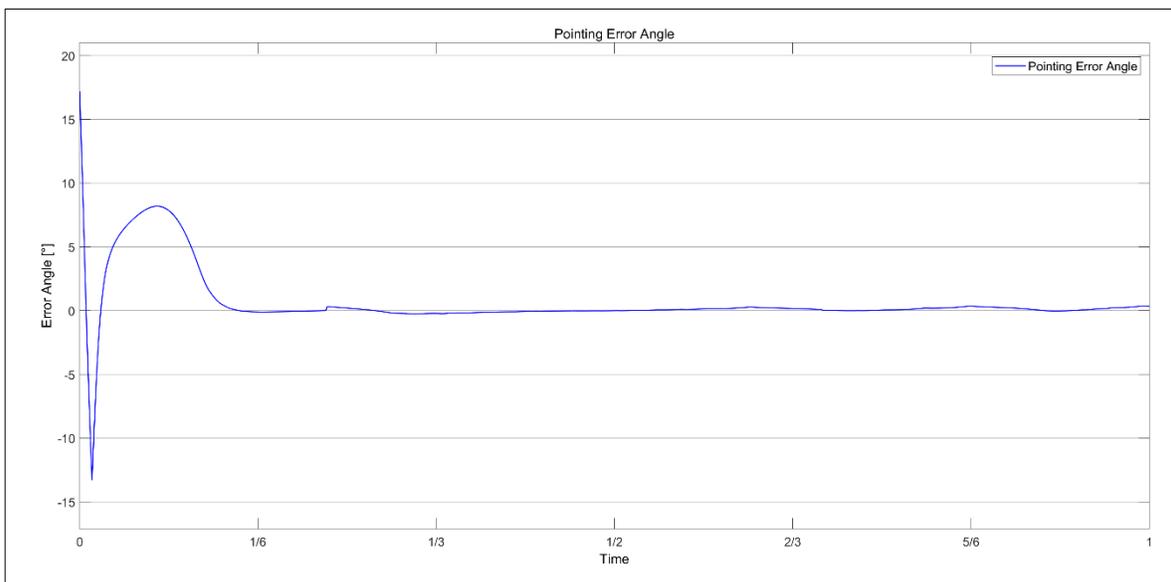


Figura 5-8: Ángulo de error del controlador LQR.

Fuente: Elaboración propia

En la Figura 5-8 se muestra el error entre el ángulo de puntería deseado y el ángulo de puntería actual del sistema. En esta gráfica lo primero que se puede ver es como al inicio esta presenta un pico que llega hasta -10° de error. Seguidamente, se puede ver una curva que

llega hasta 10° como pico para luego estabilizarse suavemente cerca de los 1000 segundos. Cuando se compara con la Figura 5-3 se puede ver que presenta una cantidad considerablemente menor de picos, además tiene un tiempo de estabilización menor en un 50% y un sobre impulso menor también. De esta manera, también presenta un error en estado estable menor que su contraparte PD. En la Tabla 5-16 se muestra un resumen de las características de esta gráfica.

Tabla 5-16: Tabla resumen del error del ángulo de puntería para el control LQR.

Fuente: Elaboración propia

Error Ángulo Puntería LQR	
Ts(s)	950
ESS (°)	0.15

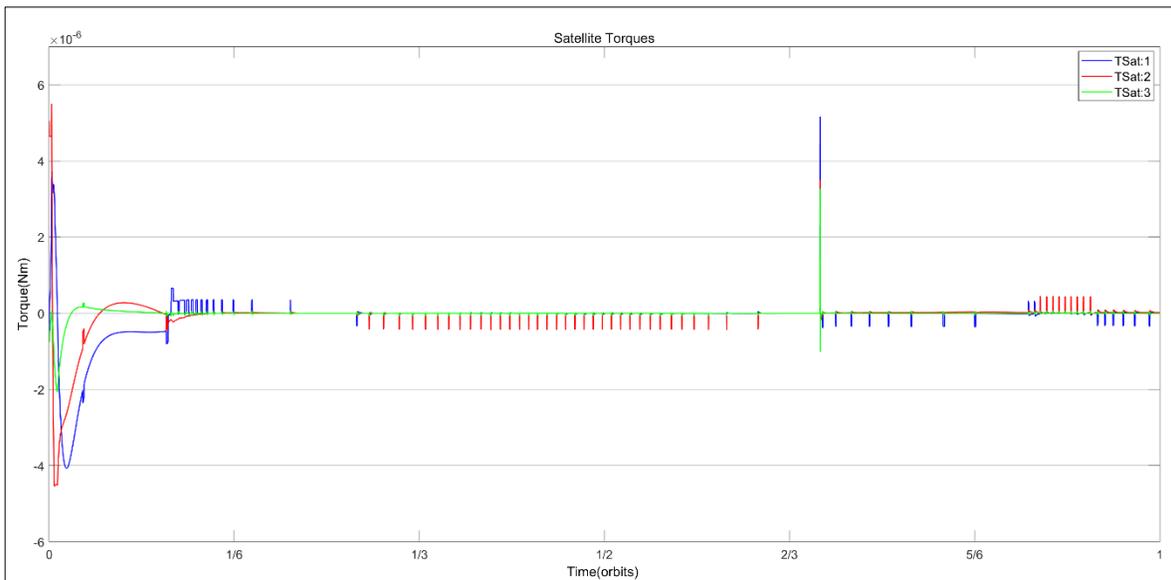


Figura 5-9: Torque generado por el controlador LQR.

Fuente: Elaboración propia.

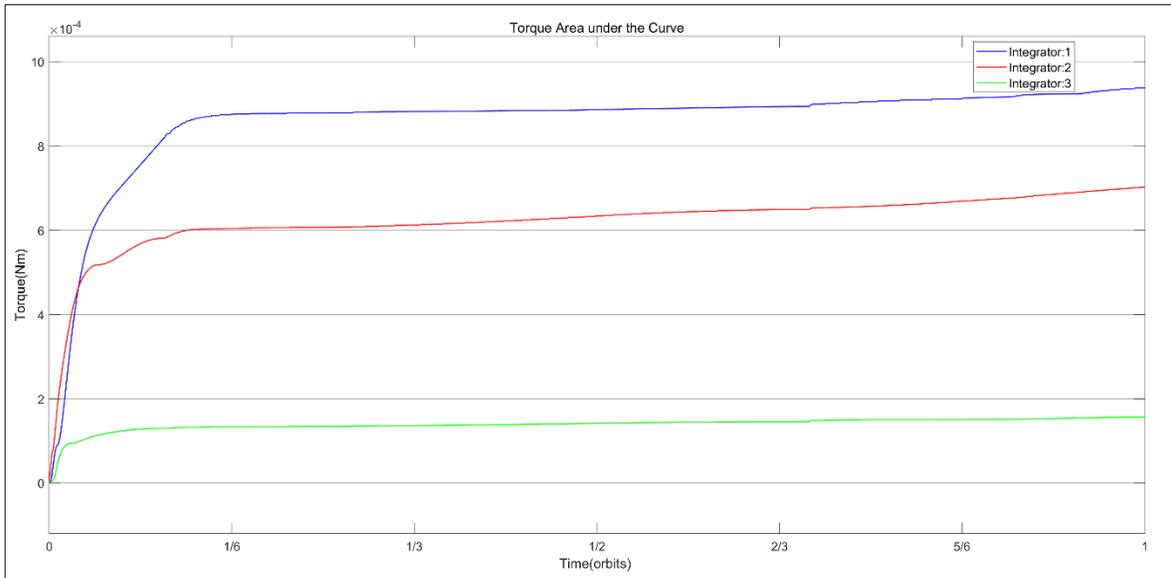


Figura 5-10: Torque acumulado del control LQR.

Fuente: Elaboración propia

En la Figura 5-9 y Figura 5-10 se pueden ver las gráficas asociadas al consumo energético del sistema. En la Figura 5-9 se puede ver en un pico inicial de torque, el cual es el encargado de la mayoría de la corrección de la rotación y velocidad angular. Luego de esto se genera una pequeña corrección suave hasta llegar a la estabilización. Una vez que el sistema está estable se pueden ver pequeños picos durante el resto de la simulación. Esto se debe al hecho que el satélite se encontrará en un entorno espacial, por lo tanto, la velocidad angular es prácticamente imposible que sea cero, por lo que se necesitan pequeños ajustes cada cierto tiempo. Cuando se compara con la Figura 5-4, se puede ver como los picos iniciales tienen un orden de magnitud de un orden de magnitud menor. También se puede apreciar como las pequeñas correcciones son más seguidas, pero con una duración menor.

De la misma forma, cuando comparamos la Figura 5-5 y la Figura 5-10 se puede ver como la suma de los torques a lo largo de toda la simulación es menor en un orden de magnitud, haciendo del controlador LQR un controlador más eficiente en el uso energético.

Es importante mencionar que, en estos casos, se utiliza el torque como el parámetro de consumo energético dado que este es la entrada del sistema en el modelo planteado por las ecuaciones dinámicas. Como este es directamente proporcional con el consumo energético

entonces se justifica su uso para dicho análisis. En la Tabla 5-17 se muestra un resumen de las características de estas gráficas.

Tabla 5-17: Tabla resumen del torque ejercido para el control LQR.

Fuente: Elaboración propia

Torque LQR	
Ts T1 (s)	700
Ts T2 (s)	700
Ts T3 (s)	700
Acumulado T1 (Nm)	9.38×10^{-4}
Acumulado T2 (Nm)	7.00×10^{-4}
Acumulado T3 (Nm)	1.56×10^{-4}

5.5 Controlador por medio de inteligencia artificial

En esta sección se muestran los resultados del proceso de diseño del controlador inteligente. Como se muestra en la Tabla 5-7 se eligió el algoritmo TD3. Para el diseño de este se siguió la serie de pasos detallados en la Figura 4-4.

5.5.1 Estudio de las características del satélite

Como se explicó en la sección anterior, la mayoría de este proceso se llevó a cabo durante el desarrollo del marco teórico. La diferencia en este caso es que para desarrollar un algoritmo de tipo TD3 se necesita definir el campo de observación y el campo de acción del sistema que se desea controlar. Para este caso se necesitan definir dos características principales:

- Espacio continuo o discreto
- Tamaño del campo de acción y observaciones

En el caso de GWSat, se tiene que tanto el espacio de acción, como el espacio de observaciones es continuo. A la hora de definir el tamaño de los campos se tomó la decisión

de que se tomarían las observaciones como los estados del sistema y las acciones como las entradas de este. Para el caso del GWSat, como se muestra en la ecuación (5-3), se tiene un campo de observaciones de tamaño seis, y un campo de acción de tamaño tres.

5.5.2 *Definición de los objetivos de desempeño*

Para cualquier sistema de control es fundamental que se definan los objetivos de desempeño que se tienen que cumplir para que se considere un control exitoso. Para el caso del control inteligente se tomó la decisión de que se siguieran los parámetros establecidos en la Tabla 5-12.

El segundo objetivo fue reducir el consumo energético lo más posible, de manera que se utilizara la menor cantidad de propelente de los propulsores. Esto tiene la intención de alargar lo más posible la vida útil de la misión.

5.5.3 *Determinación de los parámetros y arquitectura inicial de la red neuronal*

Para empezar a implementar un algoritmo inteligente, lo primero que se tiene que hacer es definir la arquitectura inicial de la red neuronal, así como sus hiper-parámetros iniciales. Dado que esto será un proceso iterativo, estos valores iniciales se tomarán como la base con la cual se comenzará a optimizar el algoritmo.

En primera instancia se utilizó el algoritmo DDPG, para el cual se tienen que optimizar los siguientes hiper-parámetros:

1. Factor de Descuento
2. Tamaño del Mini Lote
3. Varianza del Modelo de Exploración
4. Tasa de Decadencia de la Varianza del Modelo de Exploración
5. Tasa de Aprendizaje del Actor
6. Tasa de Aprendizaje del Crítico

Otro parámetro de alta importancia a la hora de implementar una red neuronal es la arquitectura que se utilizará. Esta se define por la cantidad de capas ocultas, además de la cantidad de neuronas en cada una de estas capas.

Por último, un parámetro vital para el entrenamiento de una red neuronal por medio de aprendizaje por refuerzo es la definición de la función de recompensa que tendrá el agente. La importancia de esta se centra en que, con la función de recompensa incorrecta, se estarán recompensando y castigando acciones que no deberían ser, o no se está haciendo con la magnitud necesaria para estimular el aprendizaje.

Con respecto a los hiper-parámetros se tomó como base el trabajo de [38] para la definición de estos. Además, se redujo el tamaño del mini lote y se aumentó la tasa de aprendizaje del actor. Estos cambios se realizaron con el objetivo de reducir el coste computacional. Estos se pueden ver en la Tabla 5-18.

Tabla 5-18: Híper-Parámetros de la primera iteración del controlador inteligente
Fuente: Elaboración propia

Iteración 1	
Algoritmo	DDPG
Factor de Descuento	0.99
Tamaño Mini Lote	64
Varianza del Modelo de Exploración	0.5
Tasa de Decadencia de la Varianza del Modelo de Exploración	1.00E-05
Tasa Aprendizaje Actor	0.001
Tasa Aprendizaje Crítico	0.001

Para definir la arquitectura inicial se tomó el trabajo de [38] y [39]. De esta manera, se definió que la arquitectura inicial, considerando la complejidad del sistema con el cual se está trabajando, sería una de dos capas ocultas con 400 y 300 neuronas respectivamente.

Finalmente, también basándose en el trabajo de [38] se definió la función de recompensa como la suma de funciones de recompensa independientes para cada uno de los parámetros de interés. Esto se puede ver en la ecuación (5-16)

$$R = RQ + R\omega + RT \quad (5-16)$$

Donde

$$RQ = -1.2 * EQ \quad (5-17)$$

$$R\omega = -E\omega \quad (5-18)$$

$$RT = -0.1 * T \quad (5-19)$$

Esta primera función presenta la forma general mostrada en la Figura 5-11.

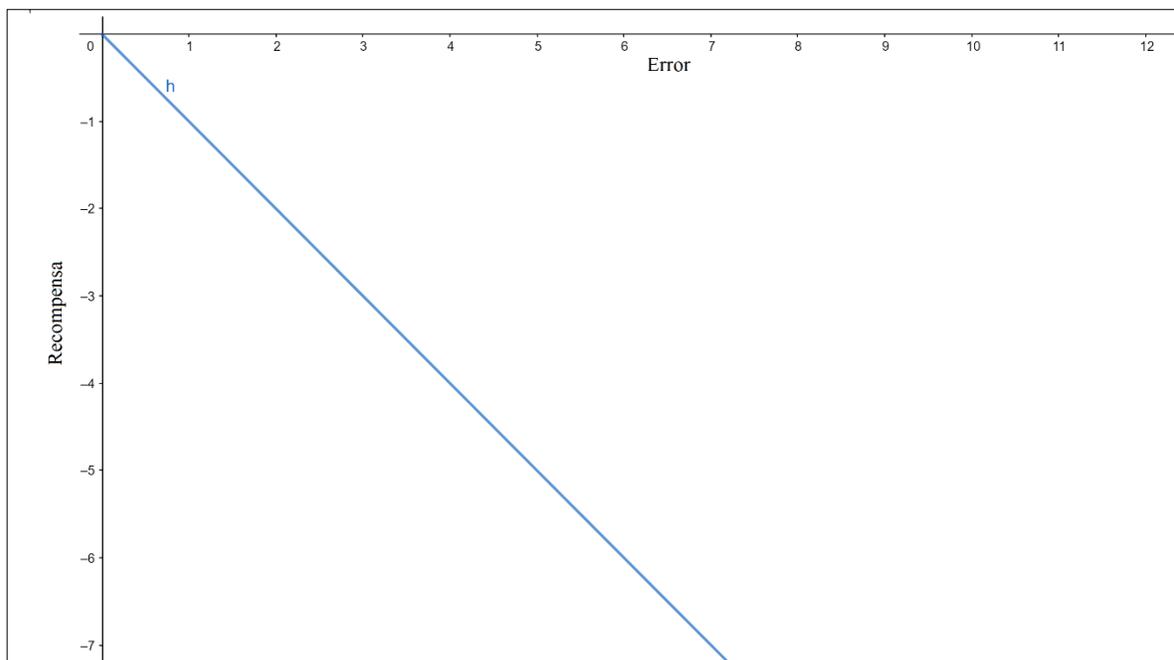


Figura 5-11: Forma general de la primera función de recompensa

Fuente: Elaboración propia

5.5.4 Entrenamiento de la red neuronal

Una vez que se han definido los parámetros iniciales se realiza el entrenamiento de la red neuronal. Para esto se escogió como software de desarrollo MATLAB por dos razones principales:

1. El modelo preexistente del satélite existe en esta plataforma, por lo que remodelarlo en otro software requeriría de mucho tiempo innecesario.
2. MATLAB tiene el *Reinforcement Learning Toolbox* el cual nos permite crear agentes de aprendizaje por refuerzo con facilidad, así como modificar los hiper-parámetros de manera simple.

La primera iteración utilizó los parámetros definidos en la sección anterior. Este entrenamiento tuvo como resultado que no hubo aprendizaje, además de existir divergencia del valor final.

Cuando se habla de que no hubo aprendizaje, esto se refiere al evento de que la recompensa promedio del agente se vuelve cada vez más baja convergiendo a un valor mucho más bajo de donde comenzó, por lo tanto, no logrando aprender. Esto quiere decir que el desempeño del agente es cada vez peor, un comportamiento irreversible dentro de un entrenamiento. Esto se puede ver en la Figura 5-12.

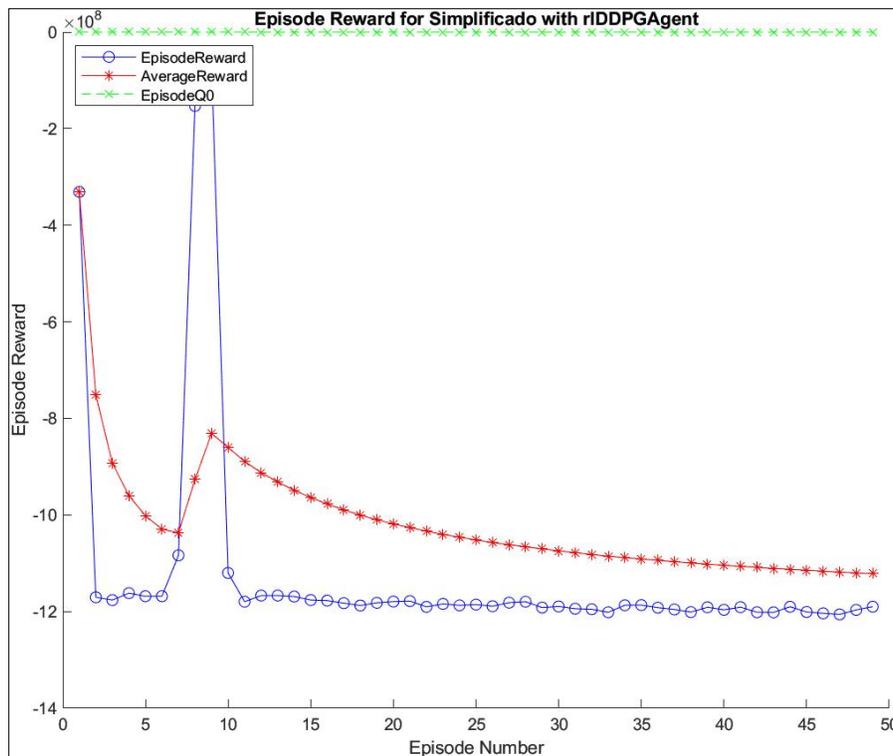


Figura 5-12: Gráfica de Recompensa de la Iteración 1

Fuente: Elaboración propia

En el caso de la divergencia, esto se refiere a que los valores finales, en cada episodio, de tanto los cuaternios como la velocidad angular divergen del valor deseado, un comportamiento claramente no deseado.

Después de analizar los resultados de la primera iteración, en la siguiente se tomó la decisión de aumentar el tamaño del mini lote de 64 a 128 muestras. Esto significó un mejor comportamiento de la curva de la recompensa. Aun así, no hubo aprendizaje y también se presenta divergencia.

Al analizar la función de recompensa que se muestra en la Figura 5-11, se puede notar que esta no tenía valores positivos dentro de su ámbito. Esto causó que la red neuronal no se premiara, sino que solo se castigara, y por lo tanto nunca tuvo un incentivo para aprender.

Por esta razón, para la tercera iteración se decidió que se cambiaría la función de la recompensa para que se pudiera premiar las acciones buenas, así como castigar las acciones malas. Basado en una función lineal se decidió que la función de recompensa sea la que se muestra en la ecuación (5-20), (5-21) y (5-22).

$$RQ = 1.2 * (1 - 2 * EQ) \quad (5-20)$$

$$R\omega = 1 - 2 * E\omega \quad (5-21)$$

$$RT = 0.15 * (1 - 2 * T) \quad (5-22)$$

La forma general de esta función de recompensa se muestra en la Figura 5-13. En esta se puede ver que con esta función cualquier valor del error en los cuaternios, en el error de la velocidad angular y en el torque que esté por debajo de 0.5 se premia, hasta un valor máximo de 1 en caso de que el error sea 0. Además, las acciones que produjeran valores mayores a estos serían castigados proporcionalmente a que tan alto es el error. Cuando se compara con la función anterior se nota como esta tiene la capacidad de premiar un buen comportamiento, a diferencia de la anterior. Asimismo, se puede definir tanto su valor

máximo como su valor de corte de manera simple, mientras que en el pasado el valor máximo siempre sería cero y el corte, también, siempre sería en cero.

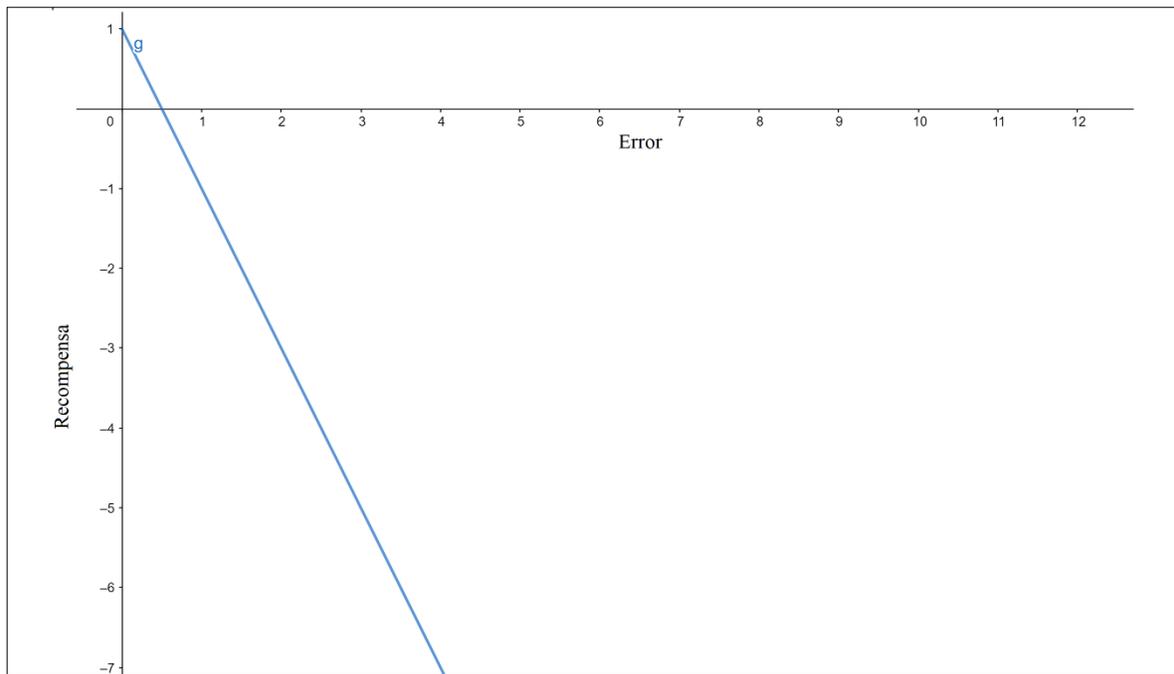


Figura 5-13: Forma general de la segunda función de recompensa

Fuente: Elaboración propia

Estas ecuaciones tienen una recompensa que escala linealmente conforme esta mejora con un valor máximo de 1.2 para la ecuación (5-20), de 1 para la (5-21) y de 0.15 para la (5-22). Esta función hizo que la red tuviera un aprendizaje muy leve. Sin embargo, sigue habiendo divergencia de los valores finales. La curva de aprendizaje se muestra en la Figura 5-14, en azul la recompensa de cada episodio, en rojo la recompensa promedio y en amarillo el Q0 de cada episodio.

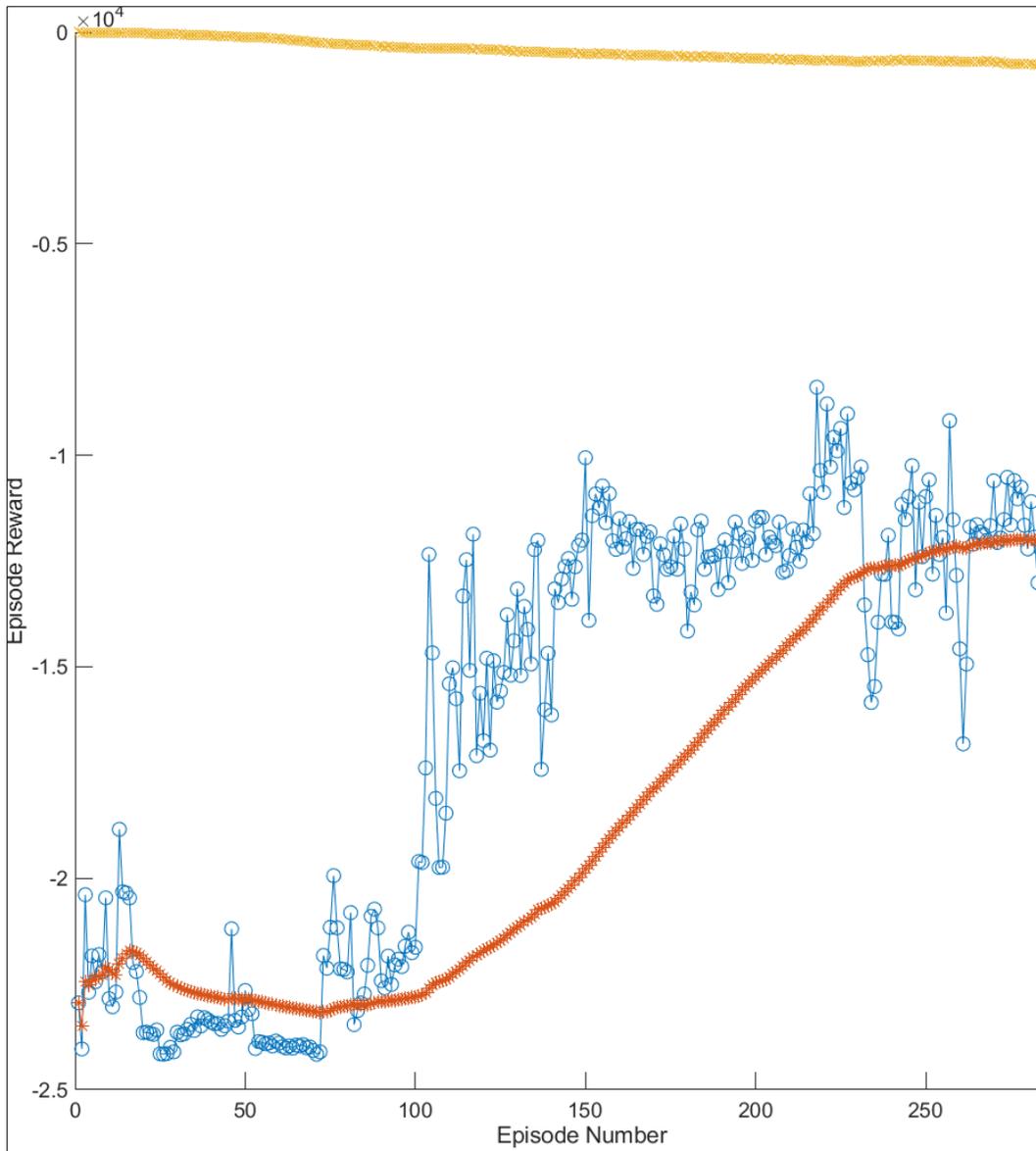


Figura 5-14: Recompensa de la tercera iteración

Fuente: Elaboración propia

Las curvas de los valores finales donde se muestra la divergencia de estos se muestran en la Figura 5-15 y Figura 5-16.

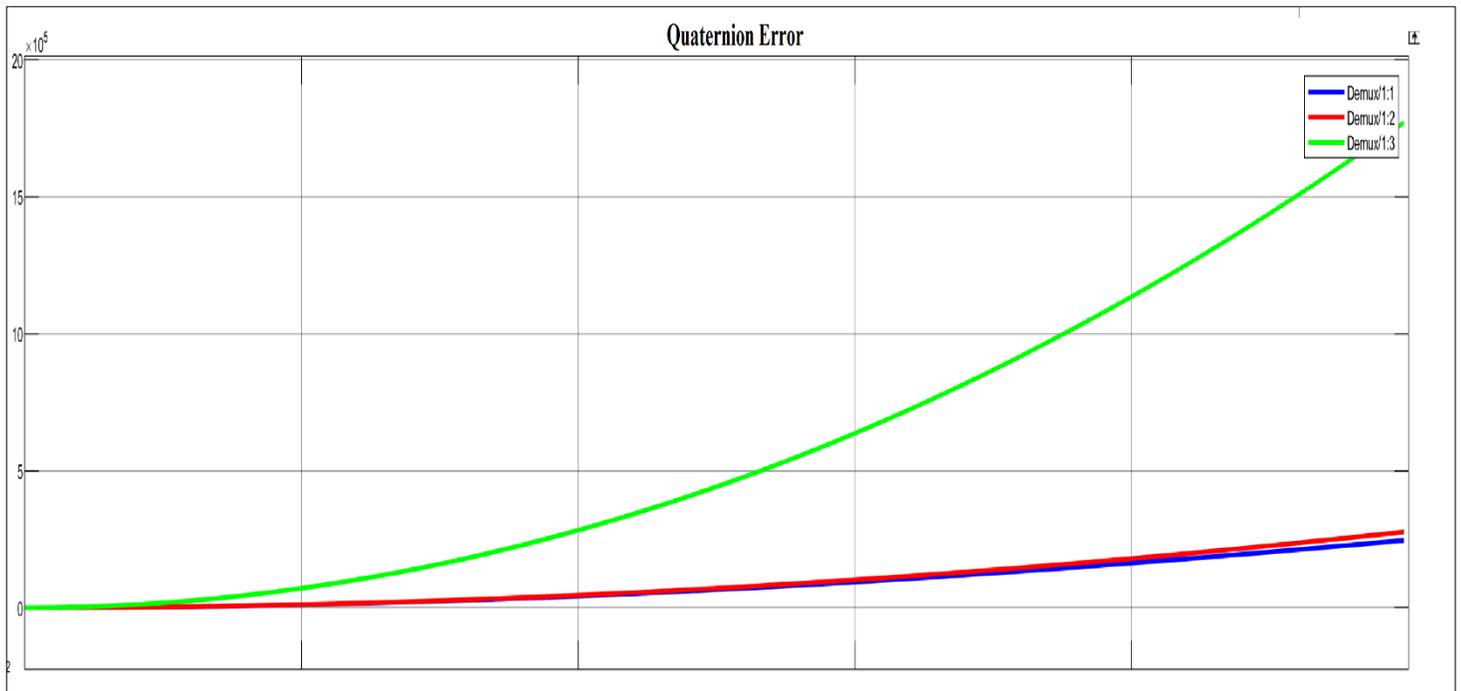


Figura 5-15: Comportamiento divergente del error de los cuaternios

Fuente: Elaboración propia

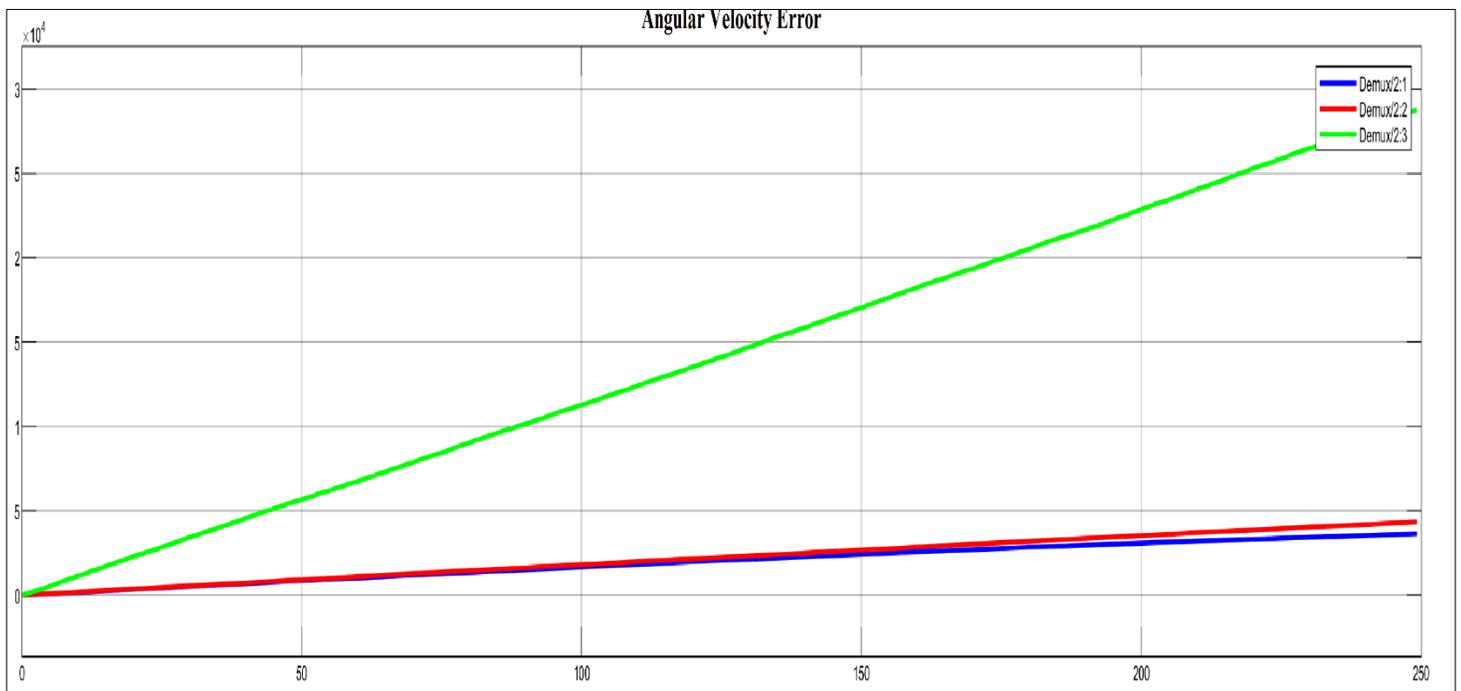


Figura 5-16: Comportamiento divergente del error de la velocidad angular

Fuente: Elaboración propia

Dado el comportamiento mostrado en la Figura 5-14 de un aprendizaje que se estanca casi inmediatamente, se determina que se está ante la presencia de un mínimo local, por lo que la red neuronal necesita una exploración más agresiva para encontrar un agente. Por esta razón, se aumentó la varianza del modelo de exploración, así mismo, la tasa de decadencia de esta se disminuyó para poder explorar por una cantidad de tiempo mayor. Por último, se disminuyó la tasa de aprendizaje del actor para evitar que se quede atrapado en mínimos locales. El resto se mantuvieron iguales. Los nuevos parámetros se muestran en la Tabla 5-19.

Tabla 5-19: Híper-Parámetros de la cuarta iteración del controlador inteligente

Fuente: Elaboración propia

Iteración 4	
Varianza del Modelo de Exploración	0.85
Tasa de Decadencia de la Varianza del Modelo de Exploración	1.00E-06
Tasa Aprendizaje Actor	0.0001
Tasa Aprendizaje Crítico	0.001

Para la determinación de estos nuevos valores se siguieron una serie de principios de manera que se fundamentaran sus distintos valores. Para determinar la nueva varianza se hizo uso de la técnica de búsqueda binaria, para la tasa de decadencia se hizo uso del principio de la vida media de la varianza [40] y para la tasa de aprendizaje del actor se hizo uso del concepto de búsqueda por cuadrícula [41].

Para el caso de la varianza que se utilizó la técnica de búsqueda binaria. Esto se hizo ya que no hay principios matemáticos que se puedan utilizar para encontrar el valor óptimo, sino que es un proceso empírico. Dado que se podían necesitar varias iteraciones se escogió esta técnica, pues es una forma altamente eficiente de encontrar el valor en la menor cantidad de iteraciones.

Previo a la iteración descrita anteriormente, se estaba utilizando una varianza de 0.5, pero para esta iteración se decidió aumentarla a 0.85, pues no se estaba explorando suficiente. En caso que la red no mostrara suficiente variación se subiría a la mitad entre 0.85 y 1; y en caso que la red más bien mostrara desaprendizaje por la alta varianza se bajaría a la mitad entre 0.5 y 0.85. Este proceso se habría continuado de ser necesario. Para el caso preciso de este proyecto, 0.85 se mostró como un valor funcional a la hora de entrenar la red neuronal.

Para el caso de la tasa de decadencia de la varianza se utilizó el concepto de vida media de la varianza, esta se define en la ecuación (5-23)

$$HL = \frac{\ln(0.5)}{\ln(1 - DR)} \quad (5-23)$$

Este concepto nos permite saber cada cuantos pasos se reducirá la varianza a la mitad. Determinar este valor es importante pues permite modificar la relación exploración-explotación a gusto y optimizarla. Para el caso de la tasa propuesta se puede ver la vida media en la ecuación

$$HL = \frac{\ln(0.5)}{\ln(1 - 1 \times 10^{-6})} = 693\ 146 \text{ timesteps} \quad (5-24)$$

Además, sabiendo que se tienen cuatro mil *timesteps* por episodio entonces se puede determinar que la vida media de la varianza con dicha tasa de decadencia es de 173 episodios. Este valor se consideró idóneo pues genera un buen balance entre la exploración y la explotación.

Par definir la nueva tasa de aprendizaje del actor se basó en la técnica de búsqueda por cuadrícula descrita en [41]. Esta busca la mejor tasa de aprendizaje para una red neuronal dada una serie de datos. La técnica consiste en hacer un barrido de diferentes tasas de

aprendizaje, generalmente haciendo saltos de manera logarítmica. Para el caso de este proyecto, dado que cada iteración toma mucho tiempo, se decidió no hacer un barrido, sino simplemente analizar el comportamiento de la curva de aprendizaje, que en este caso muestra claramente un mínimo local, y aumentar o disminuir en un orden de magnitud la tasa de aprendizaje tantas veces como sea necesario. Para este sistema solamente fue necesario hacerlo una sola vez de 1×10^{-5} a 1×10^{-6} .

En este caso, este cambio de parámetros provocó que por primera vez hubiera tanto aprendizaje como convergencia. La gráfica de la recompensa se muestra en la Figura 5-17.

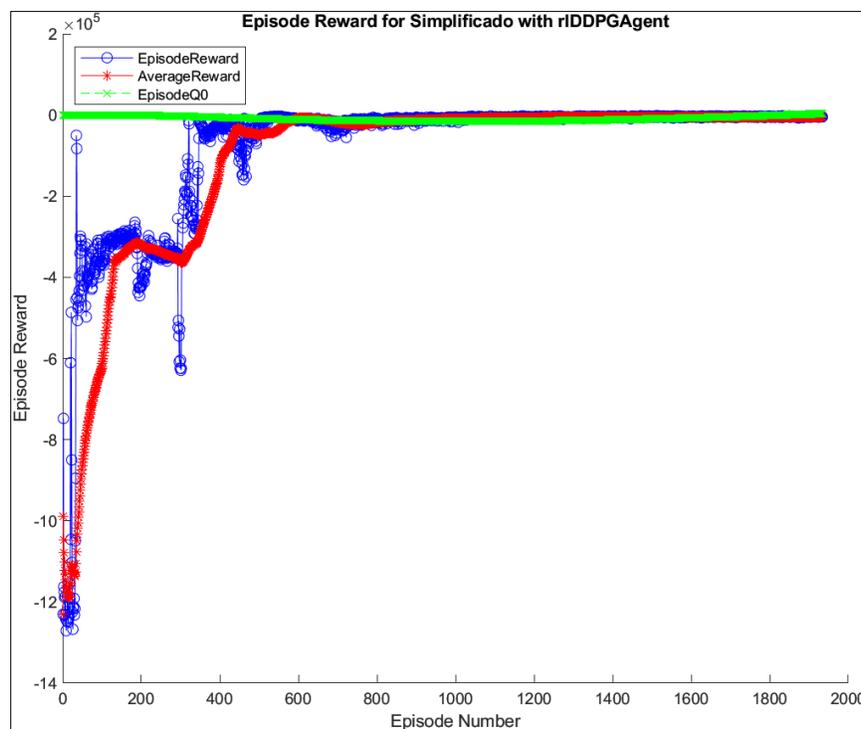


Figura 5-17: Gráfica de Recompensa de la Iteración 4

Fuente: Elaboración propia

Como se puede observar en la Figura 5-17 la red neuronal rápidamente aprende, pero se queda estancada en un valor cercano a cero. Esto sucede pues dada la función de recompensa que se escogió el rango que se tiene para que pueda ser negativo es mucho mayor que el rango de valores positivos. El rango de valores negativos es infinito, mientras que el rango de valores positivos tiene un máximo muy bajo. Por lo tanto, se determina que se debe

cambiar la función de recompensa una vez más. Las nuevas funciones de recompensa se expresan en las ecuaciones (5-25), (5-26) y (5-27)

$$RQ = -Ln(2 * EQ) \quad (5-25)$$

$$R\omega = -Ln(2 * E\omega) \quad (5-26)$$

$$RT = -0.15 * Ln(2 * T) \quad (5-27)$$

En la Figura 5-18 se muestra la forma general de la función de recompensa descrita por las ecuaciones (5-25), (5-26) y (5-27). En esta figura podemos ver como corta el eje X cuando el error alcanza 0.5 y a partir de este punto, si el error disminuye, la recompensa crece de manera exponencial, mientras que, si el error aumenta, la recompensa disminuye de manera logarítmica. Cuando se compara con la función de recompensa representada en la Figura 5-13 se puede ver como el valor máximo de esta es, teóricamente, infinito, cuando el error sea exactamente cero. Además, la pendiente inicia suave, pero rápidamente aumenta, creando un efecto que logra premiar de manera sustancial las acciones conforme mejor sean. Mientras tanto, cuando el error aumenta sucede lo contrario, la pendiente comienza siendo pronunciada y luego se va aplanando de manera asintótica. De esta forma se logra que en el caso de los episodios que presenten errores muy altos no se castigue de manera exagerada, exacerbando los mínimos locales. Cuando se consideran estos dos cambios se puede determinar que esta es una mejor función de recompensa que la pasada.

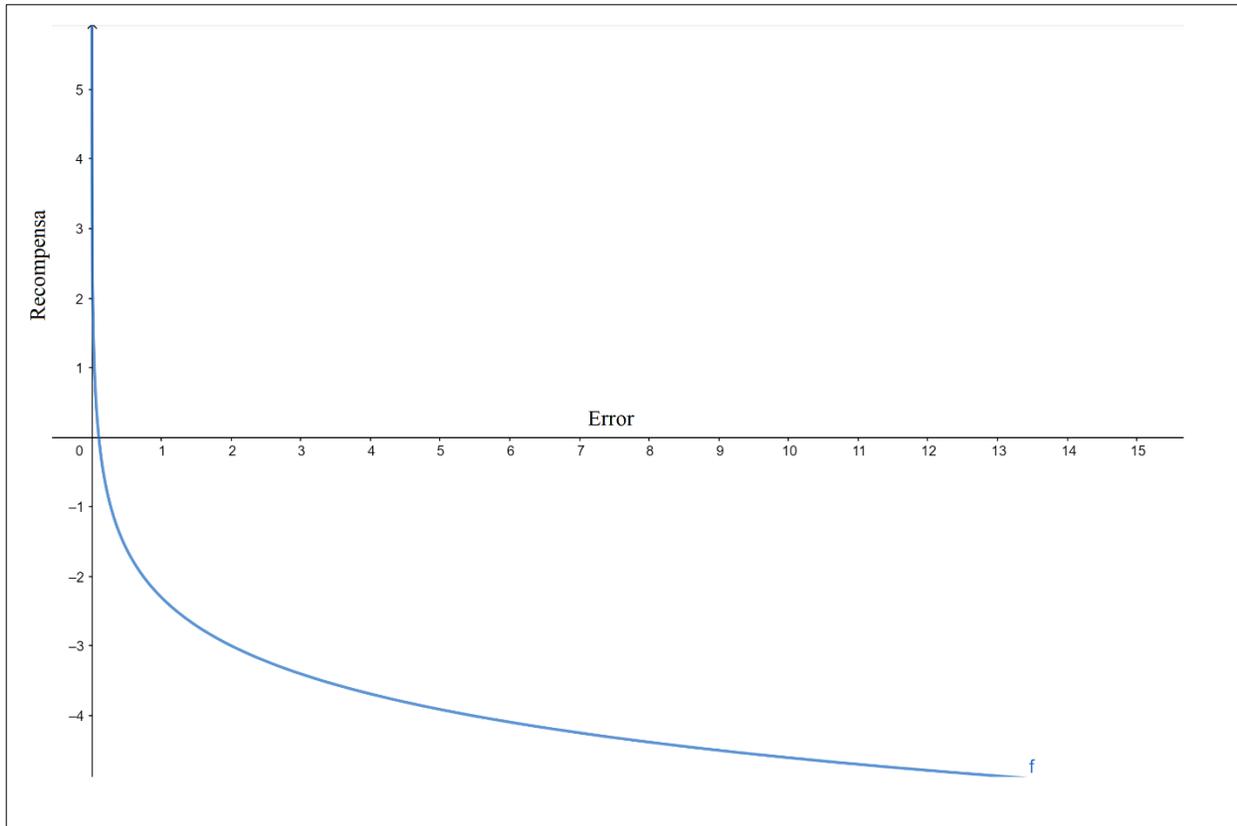


Figura 5-18: Forma general de la tercera iteración de la recompensa

Fuente: Elaboración propia

De esta función de recompensa se obtuvo una mejora del desempeño del sistema, como se muestra en la Figura 5-19. Se determinó que eso se debe al hecho que esta nueva función de recompensa es capaz de premiar el buen desempeño de manera exponencial, al igual que castigar de manera significativa a los agentes con un desempeño pobre.

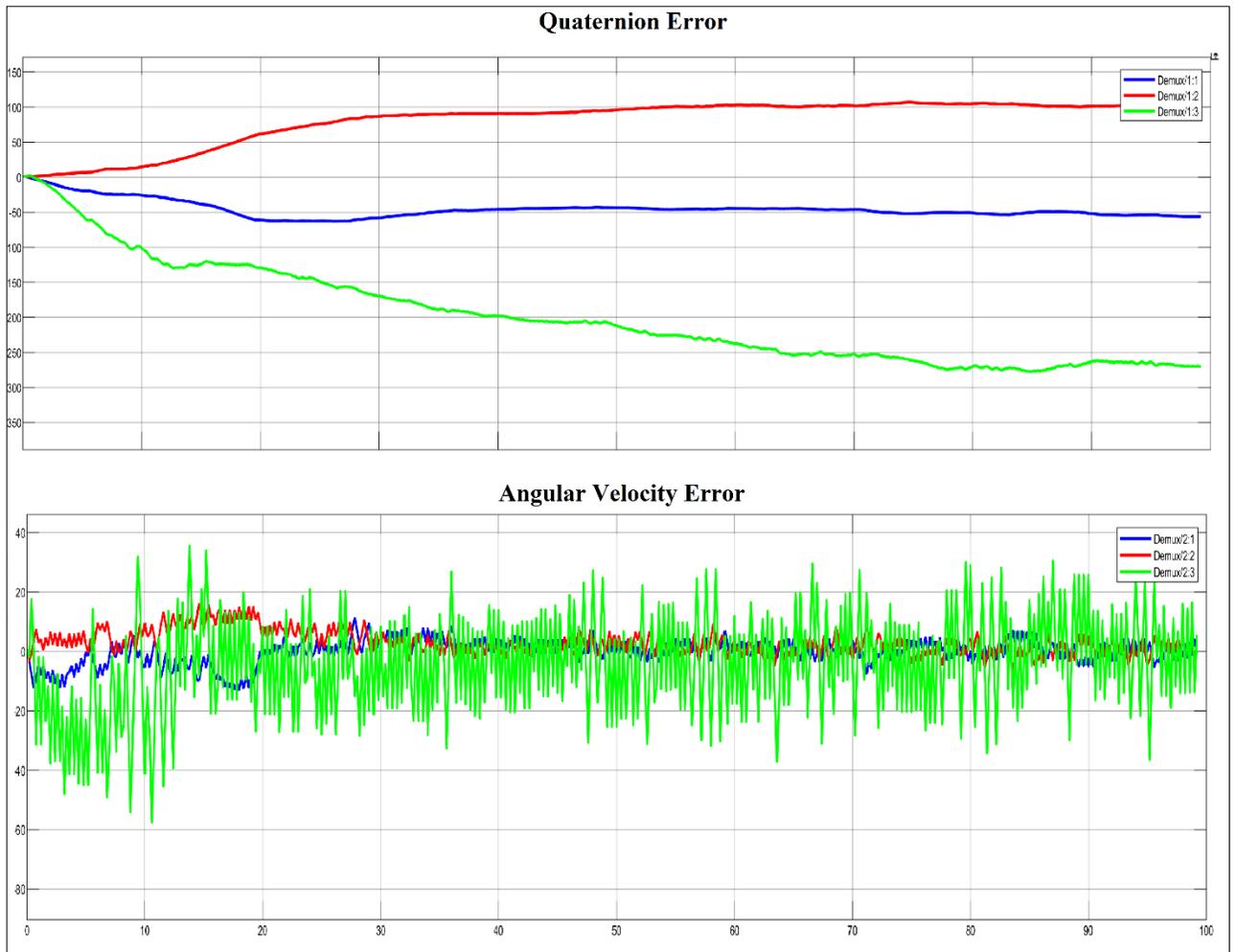


Figura 5-19: Comportamiento convergente de los cuaternios (arriba) y velocidad angular (abajo)

Fuente: Elaboración propia

Durante todas las iteraciones pasadas se utilizó un modelo simplificado del sistema en el espacio de estados para que el entrenamiento fuera rápido y tomara el menor coste computacional posible. El problema que traía consigo el uso del espacio de estados es que los valores de los cuaternios no estaban limitados a un valor de -1 a 1, y por lo tanto podía conseguir rotaciones en una infinidad de valores. Al utilizar el modelo completo utilizado por GWU este problema se soluciona, pues una rotación con un valor absoluto mayor a 1 en un cuaternio simplemente se regresa, como lo haría una onda senoidal. Este cambio se realizó

en la séptima iteración. Es importante recordar que en el apéndice A.3 se pueden encontrar los detalles de cada iteración.

Este cambio de ambiente de simulación hizo que el coste computacional aumentara considerablemente, como se puede observar en la Tabla 5-20, al estar compuesto principalmente de bloques interpretados, los cuales se tienen que compilar en cada paso de la simulación. Para solventar este problema se decidió cambiar la arquitectura de la red neuronal por una que disminuyera el coste computacional lo más posible sin perder complejidad de la red. Tomando los resultados obtenidos por [24], [39] y [42] se decidió que se utilizaría la arquitectura [100 50 25]. Esta decisión se tomó dado que cuando se comparan los resultados en la literatura de la arquitectura [400 300] con los de [100 50 25], se ven resultados similares y en algunos casos incluso superiores por parte de la [100 50 25].

La reducción en coste computacional basado en la arquitectura se puede ver en la **Tabla 5-21** y se puede notar como se da una reducción de un 15% del coste computacional en la simulación. Este cambio es aún más dramático durante el entrenamiento de la red, por lo que se justifica el cambio que se realizó de arquitectura.

Tabla 5-20: Tiempos de ejecución para los diferentes modelos de simulación

Fuente: Elaboración propia

Modelo Utilizado	Tiempo de ejecución (s)
Simplificado	2.31
Modelo Completo	16.5

Tabla 5-21: Tiempos de ejecución para las diferentes arquitecturas

Fuente: Elaboración propia

Arquitectura	Tiempo de ejecución (s)
[400 300]	2.31
[100 50 25]	1.96

Todas las mediciones para encontrar los tiempos de ejecución se pueden encontrar en los apéndices A.4 y A.5.

Tabla 5-22: Ámbitos de los cuaternios y torque para los distintos modelos

Fuente: Elaboración propia

Modelo	Ámbito Cuaternios	Ámbito Torque
Simplificado	$[-\infty, +\infty]$	$[-1, 1]$
Modelo Completo	$[-1, 1]$	$[-1 \times 10^{-5}, 1 \times 10^{-5}]$

Además de este cambio, los valores de los cuaternios y el torque se limitaron, como se expresa en la Tabla 5-22, a los valores reales que tiene el satélite. Por esta razón, se tiene que cambiar la función de recompensa. Las ecuaciones (5-28), (5-29) y (5-30) detallan la función de recompensa.

$$RQ = -1.2 \ln(5 * EQ) \quad (5-28)$$

$$R\omega = -\ln(100 * E\omega) \quad (5-29)$$

$$RT = -0.15 * Ln(2x10^5 * T) \quad (5-30)$$

La diferencia entre esta función de recompensa y la pasada se basa esencialmente en su punto de cambio de signo. En esta nueva función los puntos de cambio de signo son distintos en cada uno de los diferentes parámetros. En la Tabla 5-27 se expresan los cambios realizados a este punto de intersección con el eje X. La importancia de este cambio se da en el hecho de que, de haber dejado la función como estaba antes tanto el torque como la velocidad angular serían premiados de manera constante y los cuaternios por lo menos el 50% del tiempo, y eso haría que el algoritmo asuma que un individuo es mejor de lo que realmente es.

Tabla 5-23: Puntos de cambio de signo entre la recompensa anterior y actual

Fuente: Elaboración propia

Función de Recompensa	Cambio de Signo Cuaternios	Cambio de Signo Velocidad Angular	Cambio de Signo Torque
Anterior	EQ = 0.5	Ew = 0.5	T = 0.5
Actual	EQ = 0.2	Ew = 0.01	T = 5x10 ⁻⁶

Estos dos cambios hicieron que se obtuviera un desempeño mucho mejor que en la iteración anterior con un error en los cuaternios cerca de 400 veces menor y un error en la velocidad angular 2000 veces mejor. Estos resultados se muestran en la Figura 5-20 y Figura 5-21.

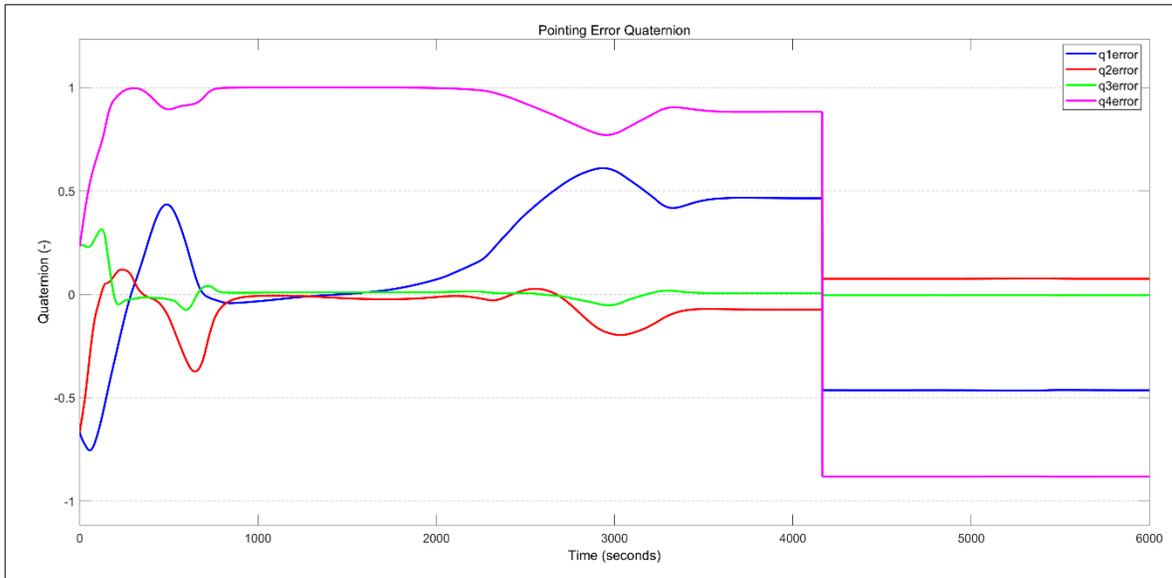


Figura 5-20: Error de los cuaternios de la séptima iteración

Fuente: Elaboración propia

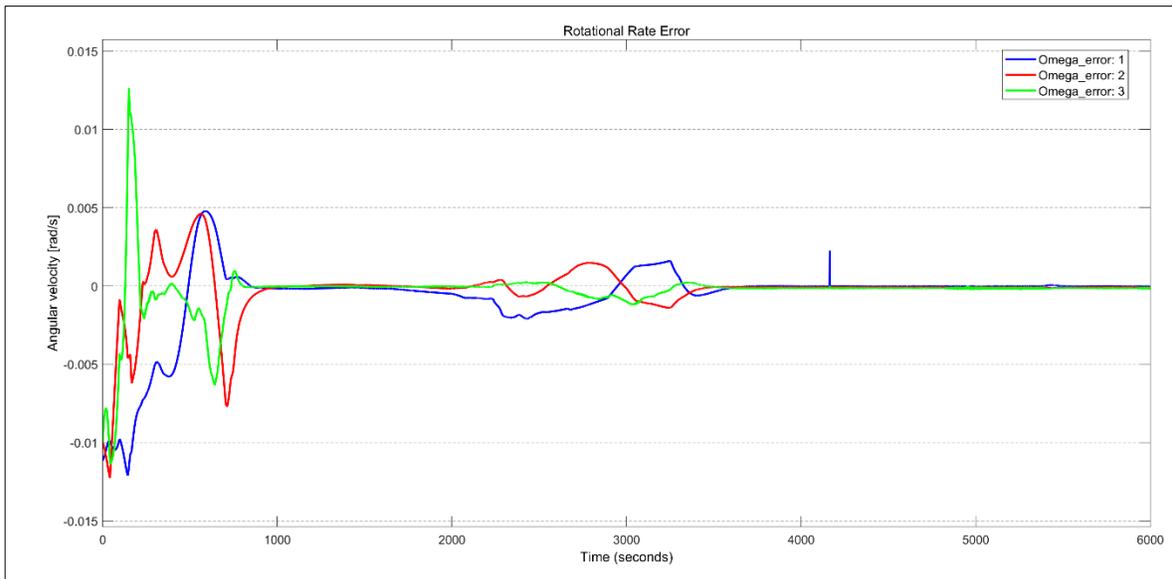


Figura 5-21: Error de la velocidad angular de la séptima iteración

Fuente: Elaboración propia

La siguiente iteración se le aumentó el factor de descuento de 0.99 a 0.9995 para hacer que las recompensas futuras tengan un mayor peso a la hora de decidir qué acción es

mejor en este momento, por lo tanto, quitándole un poco de prioridad a la estabilización inmediata y dándole una mayor a la estabilización eventual. Se decidió utilizar el valor de 0.9995 pues se quería un factor de descuento que permita que la recompensa 2000 pasos en el futuro, lo cual representa medio episodio, tuviera por lo menos una importancia del 20% sobre el estado actual. Para determinar esto se utiliza la ecuación (5-31) [43]. El problema que presentó esta iteración fue que sufrió olvido catastrófico dada la sobreestimación que se muestra en la Figura 5-22, en azul la recompensa de cada episodio, en rojo la recompensa promedio y en amarillo el Q0 de cada episodio.

$$Rf = (DF^{ts}) * R = (0.9995^{2000}) * 100 = 36.78 \quad (5-31)$$

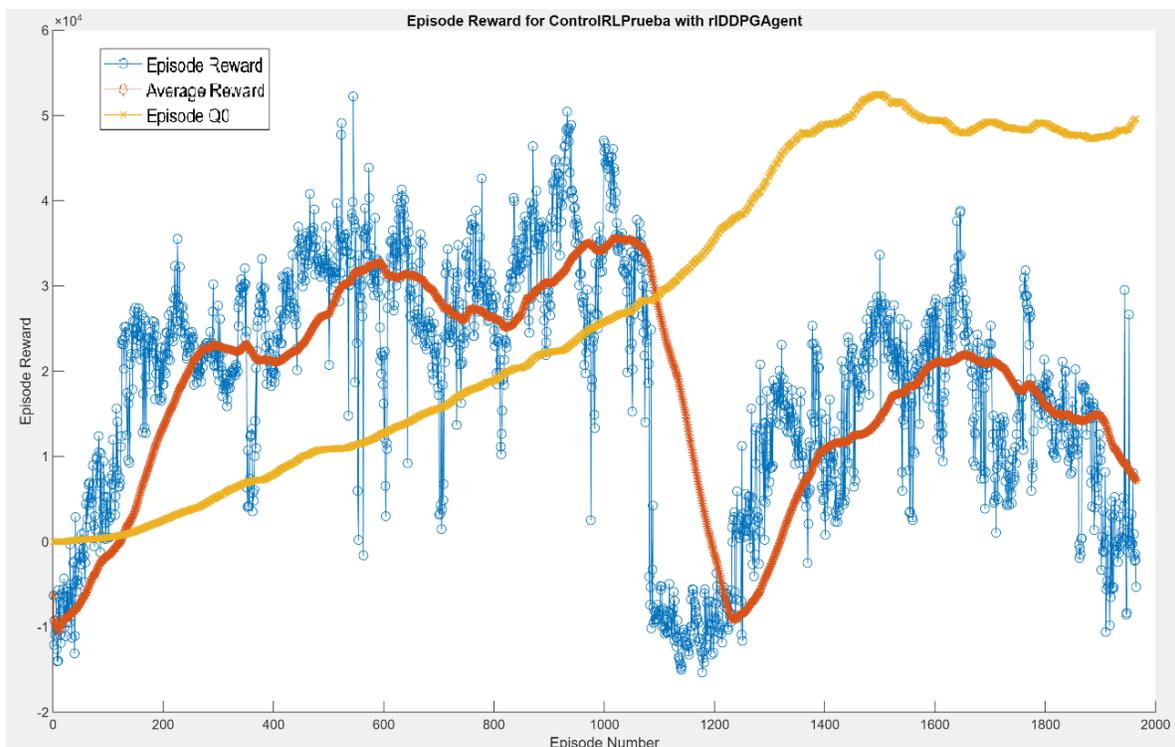


Figura 5-22: Olvido Catastrófico en la recompensa de la séptima iteración

Fuente: Elaboración propia

Este fenómeno hizo necesario que se cambiara el software MATLAB 2019 por la versión 2020, de manera que se pudiera utilizar el algoritmo TD3. Al elegir este algoritmo se eliminó el problema del olvido catastrófico, y por lo tanto se pudo optimizar correctamente la red.

La iteración 11 presenta una mejora sustancial en comparación a la última iteración descrita, la séptima. Con un valor de error en los cuaternios 19 veces más bajo y 137 veces más bajo en el error de la velocidad angular. El único cambio que se realizó cuando se comparan estas dos iteraciones es la complejidad de la arquitectura, la cual contiene 50% más neuronas en la iteración 11. Esto hizo esa gran diferencia entre ambos controladores. Los parámetros utilizados por esta red se pueden ver en la Tabla 5-24.

Tabla 5-24: Parámetros de la red neuronal elegida

Fuente: Elaboración Propia

Iteración	11
Algoritmo	TD3
Limitado Q y T	Sí
Factor de Descuento	0.9995
Tamaño Mini Lote	128
Varianza del Modelo de Exploración	0.85
Tasa de Decadencia de la Varianza del Modelo de Exploración	5.00E-07
Varianza de la Política Objetivo	0.85
Tasa de Decadencia de la Varianza de la Política Objetivo	5.00E-07
Frecuencia de Actualización Política	2
Frecuencia de Actualización Objetivo	2
Tasa Aprendizaje Actor	0.0001
Tasa Aprendizaje Crítico	0.001
Arquitectura de Red	[150 75 38]

De la Figura 5-23 se puede ver como la recompensa promedio se mantuvo en una tendencia ascendente durante los primeros quinientos episodios aproximadamente. Luego de esto se mantuvo cercano al valor de veinte mil con una leve tendencia ascendente. Se puede ver también, como el pico de aprendizaje se obtuvo alrededor del episodio mil doscientos, con un valor cercano a los sesenta mil. También, se puede observar cómo cada vez el Q0 se comienza a acercar cada vez más a los valores de la recompensa del episodio, esto indica que la red del crítico está aprendiendo de manera correcta.

El comportamiento que exhibe la recompensa promedio nos indica que al inicio le es fácil aprender a la red neuronal, pero una vez que llega a un valor promedio cercano a veinte mil, se le complicó aprender. Esto puede tener varias razones. La primera es que la varianza no sea lo suficientemente alta después de una cierta cantidad de episodios y por lo tanto no puede seguir explorando. Otra razón es que la tasa de aprendizaje sea muy alta y por lo tanto se está en un mínimo local del cual no puede salir. También podría ser que la arquitectura de la red no es lo suficientemente compleja como para poder resolver este problema de una mejor manera. Aun así la red logra un desempeño que cumple con dos de los modos de operación requeridos por GWU y por lo tanto se elige como el controlador inteligente a validar.

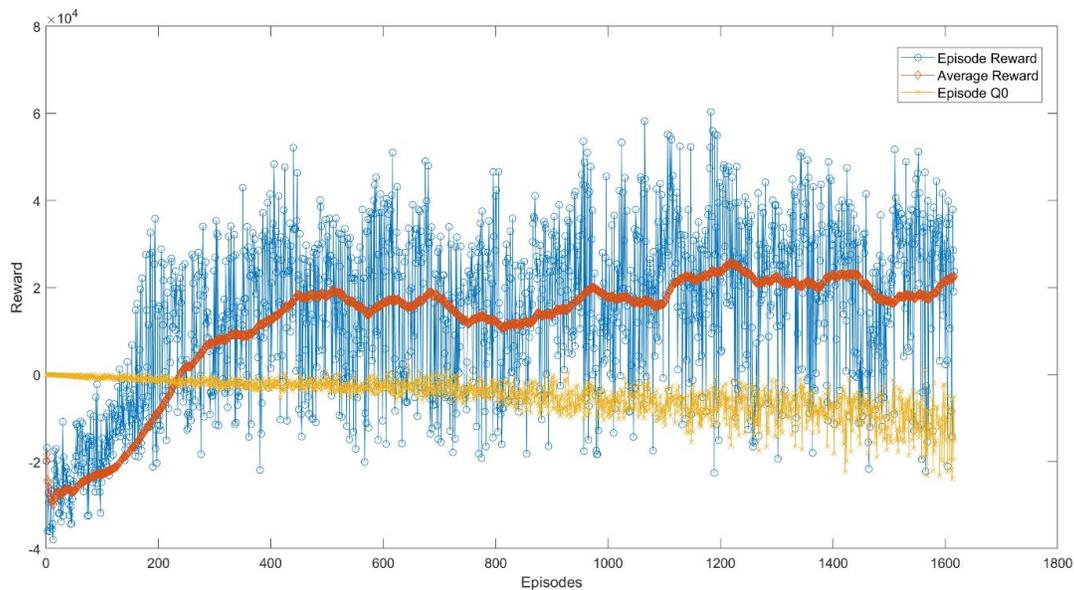


Figura 5-23: Recompensa del algoritmo TD3 escogido

Fuente: Elaboración propia

Debido a esto se decidió entrenar una red neuronal en donde se eliminó la aleatoriedad de las condiciones iniciales, los resultados de este se pueden ver resumidos en la Tabla 5-25 además si se desean ver las gráficas, estas se encuentran en el apéndice A.6. Esto se hizo a manera de prueba de concepto para determinar si con un problema más “simple” la arquitectura seleccionada es capaz de obtener un mejor rendimiento. En esta prueba se comprobó que, en efecto, la misma arquitectura funciona para un problema más simple y permite llegar a todos los parámetros deseados. Es importante notar que esta red neuronal no funciona para controlar el sistema en general pues esta sobre ajustada para un solo escenario. Por lo tanto, una arquitectura más compleja para el modelo aleatorizado permitiría una mejor solución al problema.

Tabla 5-25: Tabla resumen del desempeño de la iteración no aleatorizada.

Fuente: Elaboración propia

Iteracion No Aleatoria	
Ts (s)	500
Error Ángulo de Puntería(°)	0.86
Error Velocidad Angular(rad/s)	1.6×10^{-5}
Torque Acumulado (Nm)	0.00267
Torque Acumulado Propulsores (Nm)	0.0004241

El problema con aumentar la complejidad de la arquitectura se centra en el coste computacional. Ya que el coste computacional escala exponencialmente con respecto al tamaño de la red neuronal. De esta manera, para una red muy compleja se necesitaría una computadora muy poderosa de manera que se pueda entrenar en un tiempo razonable. La razón por la cual no se realizó esto en el caso de este proyecto, es que dados los límites de tiempo que se tenían y cumpliendo con el paso de iterar del proceso de diseño en ingeniería, no se podía implementar un algoritmo que tardara demasiado tiempo entrenando, pues la depuración del algoritmo tomó más de 10 iteraciones, las cuales cada una tardaba una semana en promedio en completarse. Este tiempo podría escalar rápidamente con arquitecturas más complejas.

5.5.5 Validación del algoritmo

En esta subsección se analizarán los resultados obtenidos con respecto a los parámetros de estabilización y desempeño del controlador inteligente. Para cada conjunto de variables se muestra una gráfica mostrando el desempeño y una tabla resumiendo los datos más importantes.

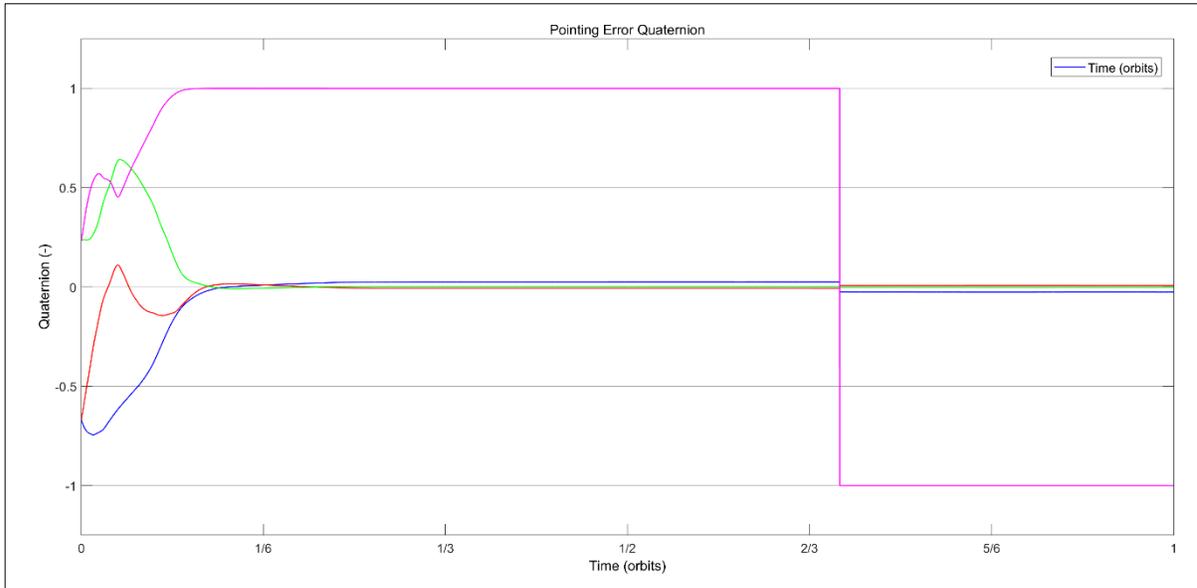


Figura 5-24: Error de los cuaternios del controlador TD3.

Fuente: Elaboración propia

En la Figura 5-24 se muestran los resultados que se obtuvieron para el error de los cuaternios de orientación a la hora de utilizar el controlador inteligente TD3. Lo primero que se puede notar es que el tiempo de estabilización se encuentra entre los 700 y 900 s. También, podemos ver como las curvas de estabilización desde el inicio son suaves, indicando que no se dan cambios bruscos en la orientación, una característica importante para asegurarse de no dañar ningún componente electrónico. Otra característica notable de esta figura es que el error en estado estacionario de q_1 no se encuentra sobre la línea del cero, si no que tiene una magnitud de 0.026, una característica indeseada pues podría hacer que no se cumpla con la precisión de puntería necesaria. En la Tabla 5-26 se resumen las características de desempeño de este controlador para el error de los cuaternios.

Tabla 5-26: Tabla resumen del error de los cuaternios para el control TD3.

Fuente: Elaboración propia

Error Cuaternios TD3	
Ts Cuaternio 1 (s)	700
Ts Cuaternio 2 (s)	700
Ts Cuaternio 3 (s)	700
ESS Cuaternio 1	0.0259
ESS Cuaternio 2	0.0079
ESS Cuaternio 3	0.0029

En la Figura 5-25 se puede ver la gráfica que describe el error de la velocidad angular actual del satélite.

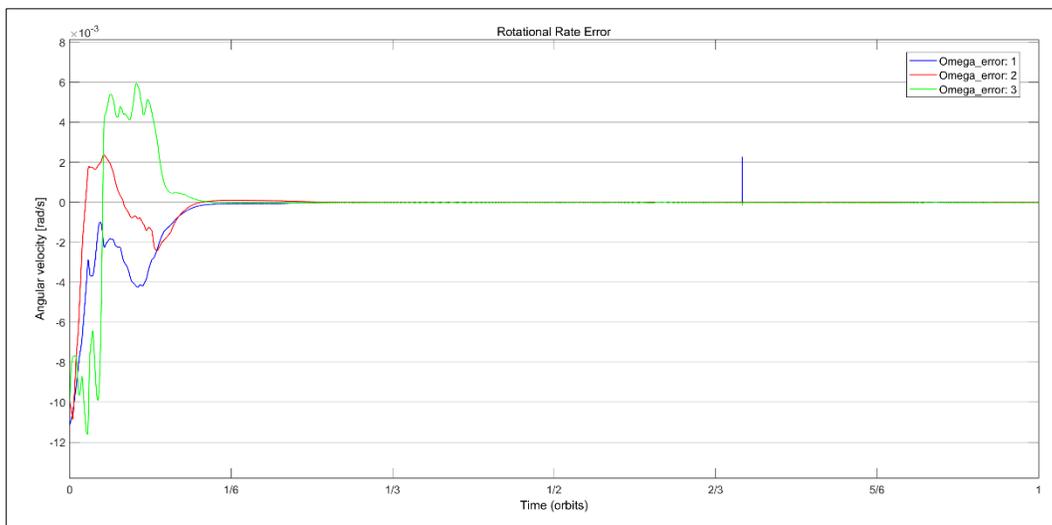


Figura 5-25: Error de la velocidad angular del controlador TD3.

Fuente: Elaboración propia

Se observa como esta, presenta un tiempo de estabilización de las 3 variables cercano a los mil segundos. También, se puede observar como el error en estado estacionario de estas

variables es particularmente, llegando a un valor para efectos prácticos de 0 rad/s, un valor perfectamente dentro del rango deseado por GWU para cualquiera de los tres modos de operación expresados en la Tabla 5-8. Por último, se puede ver un pico que se genera cerca de la marca de los cuatro mil segundos, que se debe a una característica inherente a la programación de la dinámica del satélite en donde la referencia para el cuaternio q4 tiene un salto abrupto por un instante de tiempo dado que cambia de signo, por lo tanto, se puede despreciar en el análisis de la gráfica. En la Tabla 5-27 se presenta un resumen de las características más importantes del error de la velocidad angular por este controlador.

Tabla 5-27: Tabla resumen del error de la velocidad angular para el control TD3.

Fuente: Elaboración propia

Error Velocidad Angular TD3	
Ts Omega 1 (s)	840
Ts Omega 2 (s)	840
Ts Omega 3 (s)	840
ESS Omega 1 (rad/s)	6.3×10^{-6}
ESS Omega 2 (rad/s)	7.3×10^{-6}
ESS Omega 3 (rad/s)	2.1×10^{-5}

En la Figura 5-26 se observa la gráfica del error del ángulo de puntería que se obtuvo por parte de la implementación del controlador inteligente entrenado por medio del algoritmo TD3.

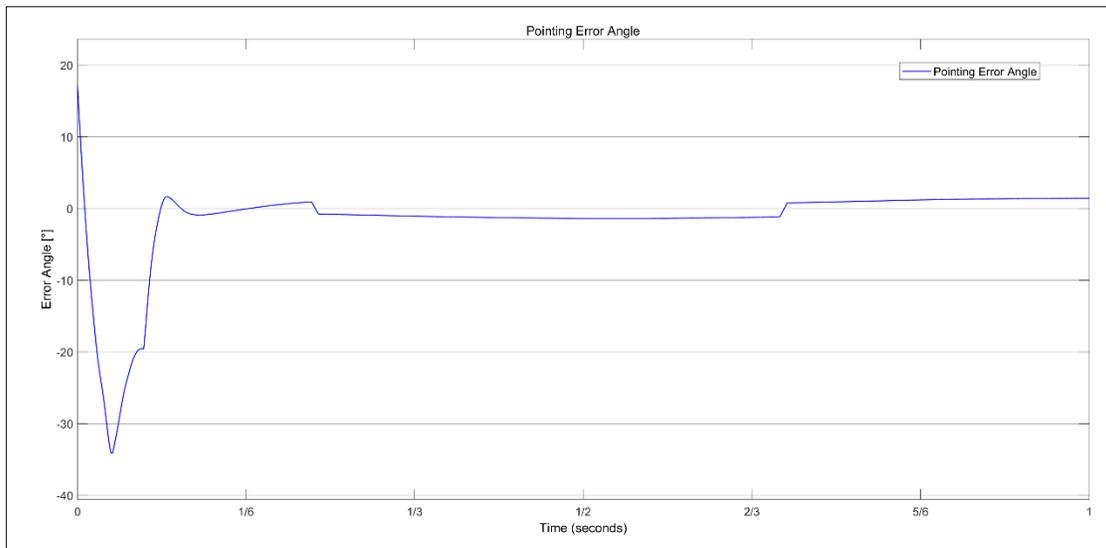


Figura 5-26: Ángulo de error del controlador TD3.

Fuente: Elaboración propia

Se puede apreciar como al inicio de la Figura 5-26 este controlador exhibe un pico pronunciado que comienza aproximadamente en 15° , la cual baja rápidamente hasta un pico de aproximadamente -35° , lo cual muestra que tiene un sobreimpulso que se podría considerar alto dado que tiene una variación del 200%. Sin embargo, este pico tiene un tiempo de subida de 200 s, por lo tanto, aunque el sobreimpulso es alto, la velocidad angular no es muy alta. Esto hace que este pico no represente una característica negativa.

Otra característica que podemos notar de esta gráfica es que el tiempo de estabilización se encuentra cercano a los 700 s, un valor bajo cuando se compara con el controlador PD e incluso con el LQR. Además de esto se puede observar como el error en estado estacionario es un error alto. En la Tabla 5-28 se puede ver como este es un error de 1.4° . Esto hace que no cumpla con los parámetros del modo de apuntamiento fino que se muestran en la Tabla 5-8, entre los cuales está un error de 1° de precisión de puntería. Sin embargo, esto no quiere decir que este controlador no se puede utilizar para el satélite, ya que sigue cumpliendo con las características de dos modos de operación. En la siguiente sección se discutirá la elección del mejor controlador. En la Tabla 5-28 se muestra un resumen de los parámetros más importantes de la gráfica del error en el ángulo de puntería.

Tabla 5-28: Tabla resumen del error del ángulo de puntería para el control TD3.

Fuente: Elaboración propia

Error Ángulo Puntería TD3	
Ts(s)	650
ESS (°)	1.4

En la Figura 5-27 se observa la gráfica del torque de control que se obtuvo por parte de la implementación del controlador inteligente entrenado por medio del algoritmo TD3.

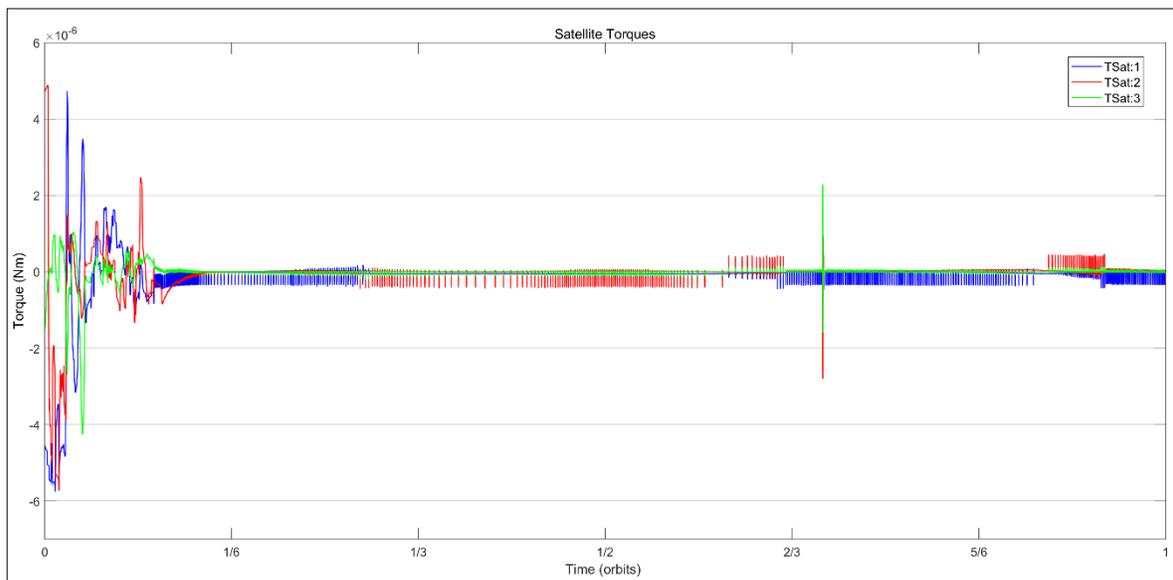


Figura 5-27: Torque generado por el controlador TD3.

Fuente: Elaboración propia

En la Figura 5-27 lo primero que se puede notar es como, en especial durante los primeros quinientos segundos, se dan señales de control elevadas y con una alta frecuencia, un comportamiento que para el caso de este proyecto en especial no es deseado, pues aunque representa un tiempo de estabilización más corto, esto genera un consumo energético mayor.

Lo siguiente que se puede notar en esta gráfica es el tiempo de estabilización de la señal de control. Esta, al igual que en los casos pasados, se encuentra cerca de los 700

segundos. También se puede ver como una vez que la señal llega a su punto de estabilización, las que corresponden al torque en el eje X y en el eje Y no se mantienen en cero, sino más bien siguen realizando pequeños ajustes a lo largo de toda la simulación. Esto se debe a las perturbaciones que se experimentan en el espacio, como los son el arrastre atmosférico, el gradiente gravitacional y la perturbación J2.

En la Figura 5-28 se puede ver la gráfica del torque acumulado por la señal de control del controlador inteligente TD3.

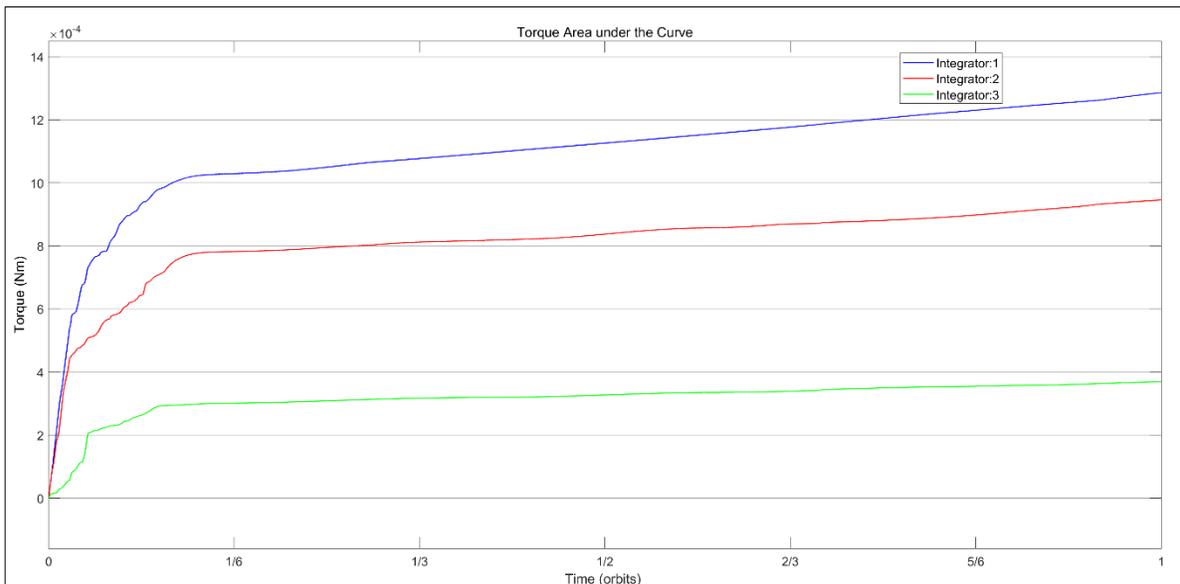


Figura 5-28: Torque acumulado del control TD3.

Fuente: Elaboración propia

En esta figura se puede observar cómo al inicio de la simulación, en la sección donde la señal de control se presenta con las mayores oscilaciones es donde se presenta la subida más brusca de esta gráfica llegando a valores de aproximadamente 10×10^{-4} Nm, 8×10^{-4} Nm y 3×10^{-4} Nm, para el acumulado en el eje X, Y y Z respectivamente, en cuestión de solamente 500 segundos. Luego de esto se puede observar como el torque en el eje Z prácticamente no crece, mientras que el torque en X y Y crece linealmente con una leve pendiente. Este crecimiento lineal se debe a las pequeñas actuaciones que se presentan en la Figura 5-27. Por

último, podemos ver como el mayor torque utilizado es por parte del eje X, en segundo lugar, el eje Y y por último el eje Z.

La gran diferencia entre el eje Z y los otros dos ejes se explica porque el eje Z tiene un momento de inercia menor que el de los otros dos, necesita de una cantidad menor de torque para romper o contrarrestar la inercia en el movimiento del satélite. En la Tabla 5-29 se puede ver un resumen de las características más importantes de la Figura 5-27 y Figura 5-28.

Tabla 5-29: Tabla resumen del torque ejercido para el control TD3.

Fuente: Elaboración propia

Torque TD3	
Ts T1 (s)	720
Ts T2 (s)	720
Ts T3 (s)	720
Acumulado T1 (Nm)	1.38×10^{-3}
Acumulado T2 (Nm)	9.46×10^{-4}
Acumulado T3 (Nm)	3.69×10^{-4}

Por último, en la Tabla 5-30, se puede observar el desempeño general de este controlador y su índice de rendimiento.

Tabla 5-30: Parámetros de desempeño del controlador TD3

Fuente: Elaboración propia

Control TD3	
Tiempo de estabilización(s)	700
Error Velocidad Angular (rad/s)	7.30E-06
Error Ángulo de Puntería (°)	1.4
Consumo de Torque (N*m)	0.002695
Índice de Rendimiento	0.46

En este caso, cuando se compara con los parámetros del control PD, actualmente implementado en el satélite, de la Tabla 5-9 se puede ver como el controlador inteligente es mejor que el controlador PD, con un índice de rendimiento 0.27 puntos por arriba de este.

5.6 Comparación entre controladores

En esta sección se trata la comparación de los dos controladores desarrollados para el nanosatélite GWSat. En primera instancia se definieron los parámetros que se usaron para comparar los controladores, luego de esto se comparan por los diferentes rubros y se ofrece la mejor solución al problema planteado.

5.6.1 Parámetros de comparación

Para determinar que controlador se elegirá para la solución del problema es esencial definir que parámetros se utilizarán para evaluar a los candidatos. Hasta el momento se ha utilizado el índice de rendimiento para comparar los controladores desarrollados con el controlador PD previamente diseñado. Éste toma en cuenta cuatro parámetros, los cuales se pueden ver en la Tabla 5-31, así como sus pesos.

Tabla 5-31: Parámetros del índice de rendimiento

Fuente: Elaboración Propia

Parámetro	Peso
Tiempo de estabilización(s)	0.083
Error Velocidad Angular (rad/s)	0.083
Error Ángulo de Puntería (°)	0.083
Consumo de Torque (N*m)	0.75

Dados estos valores, esto nos permite comparar los índices de rendimiento entre los distintos controladores en la Tabla 5-32.

Tabla 5-32: Índice de rendimiento de los distintos controladores

Fuente: Elaboración propia

Controlador	Índice de Rendimiento
PD	0.19
LQR	0.57
TD3	0.46

De la Tabla 5-32 podemos ver como los dos controladores propuestos superan al controlador PD en sus índices de rendimiento.

Este índice de rendimiento solo toma en cuenta el torque utilizado por todo el sistema, un parámetro que no es tan importante como el torque acumulado por los propulsores, en especial cuando se considera que estos tienen un propelente limitado. Aunque el consumo energético de los magnetorquers es importante, no es vital maximizarlo, pues estos utilizan la carga eléctrica del satélite, la cual se recarga por medio de paneles solares.

Cuando se considera la importancia de la utilización de torque solamente por los propulsores, se descubre que éste es el parámetro más importante para definir si un controlador es mejor que otro, pues a menor utilización de torque por estos, mayor será la vida útil del satélite. De esta manera se propuso una estrategia de dos pasos para determinar cuál controlador se escoge. Esta se detalla a continuación:

1. Se verifica que cumpla con los requisitos, para los diferentes modos de operación del satélite.
2. Se toma la utilización de torque por los propulsores de los controladores diseñados y se escoge el que tenga el valor menor.

5.6.2 Comparación desempeño con respecto a referencia de GWU

El primer paso para la escogencia del mejor controlador es asegurarse que cumpla con los requisitos de desempeño. Para este caso solamente se verificará que estén dentro de las tolerancias que planteó GWU, los cuales se muestran en la Tabla 5-8. Los resultados de los controladores se pueden ver en la Tabla 5-33.

Tabla 5-33: Cumplimiento de los requisitos por parte de los controladores diseñados

Fuente: Elaboración Propia

Modo	Requerimiento (° rad/s s)	Controlador Actual (PD)	Controlador LQR	Controlador TD3
Por Defecto	5 0.0087 86 400	1.13 0.0004 2000	0.15 3×10^{-5} 1000	1.4 7.3×10^{-6} 700
Apuntar Grueso	5 0.0087 86 400	1.13 0.0004 2000	0.15 3×10^{-5} 1000	1.4 7.3×10^{-6} 700
Apuntar Fino	1 0.0017 86 400	1.13 0.0004 2000	0.15 3×10^{-5} 1000	1.4 7.3×10^{-6} 700

De esta tabla se puede ver como el controlador LQR cumple con los requisitos para todos los modos de operación, mientras que el controlador TD3 cumple solamente con los requisitos para dos modos de operación. Dado que ambos cumplen por lo menos con uno de los modos de operación, ninguno se descarta en este primer paso y se tendrán que terminar de analizar en el segundo paso.

5.6.3 Comparación de la utilización del torque de los controladores

En esta subsección se analiza la eficiencia en la utilización del torque de los controladores de manera que se pueda determinar el controlador a utilizar en el sistema del GWSat. Primeramente, en la Figura 5-29 se puede ver el torque utilizado por el controlador actual, de tipo PD.

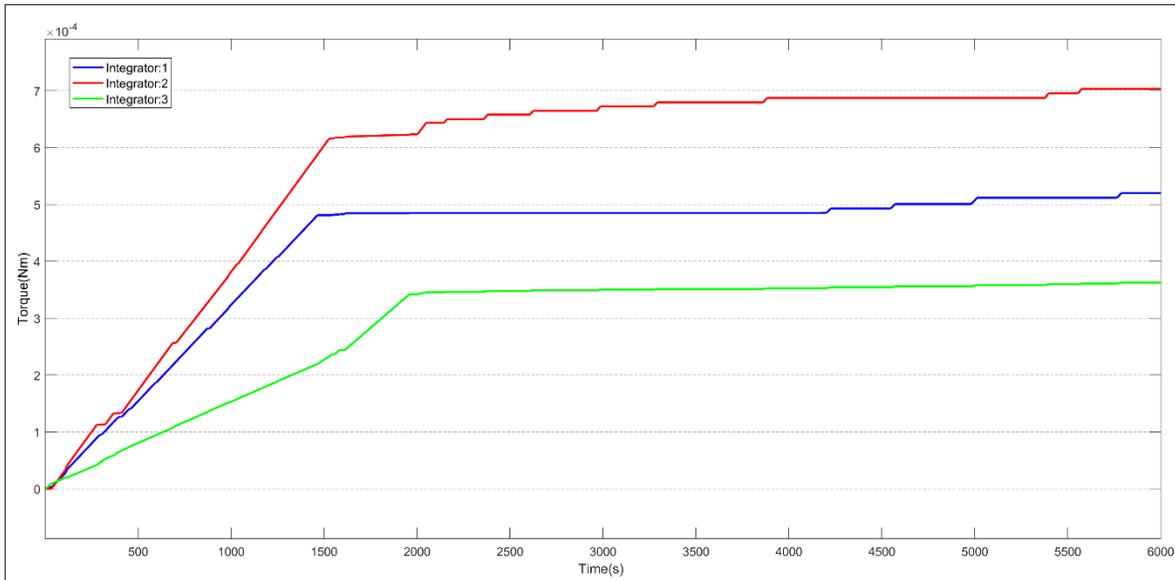


Figura 5-29: Torque utilizado por los propulsores con el controlador PD

Fuente: Elaboración propia

En Figura 5-29 podemos notar como durante los primeros mil quinientos segundos el torque aumenta rápidamente y luego sube paulatinamente hasta sus valores finales. La Tabla 5-34 muestra un resumen de lo observable en la figura. Además, En la Figura 5-30 se puede observar la gráfica del torque utilizado por el controlador LQR.

Tabla 5-34: Tabla resumen del torque ejercido por los propulsores para el control PD.

Fuente: Elaboración propia

Torque Propulsores PD	
Acumulado T1 (Nm)	5.20x10 ⁻⁴
Acumulado T2 (Nm)	7.00x10 ⁻⁴
Acumulado T3 (Nm)	3.63x10 ⁻⁴
TOTAL (Nm)	1.58x10⁻³

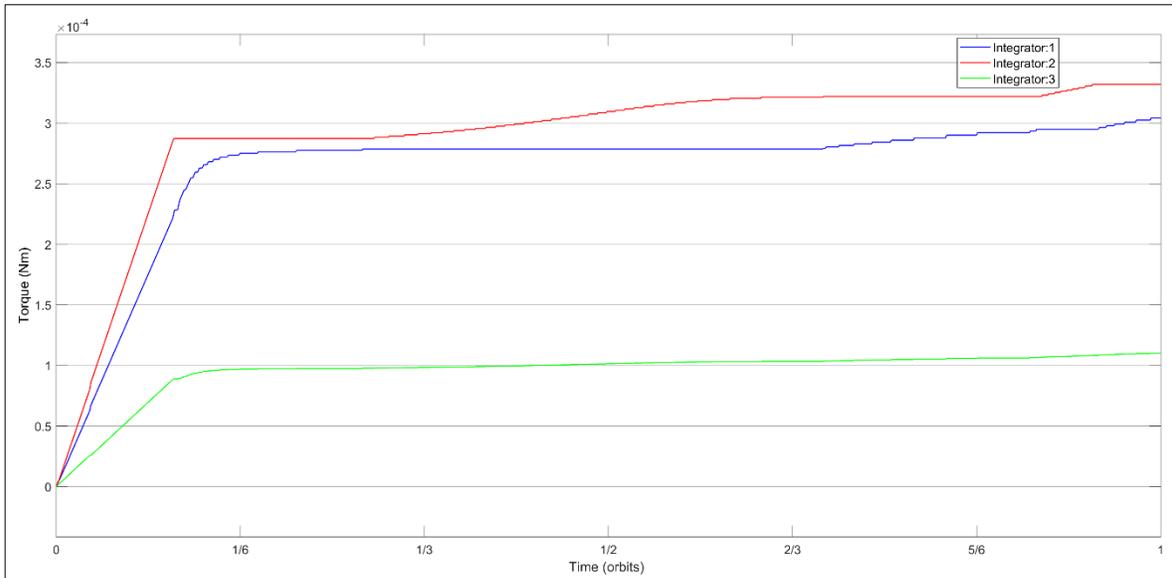


Figura 5-30: Torque utilizado por los propulsores con el controlador LQR

Fuente: Elaboración propia

En esta gráfica se puede notar como al inicio el consumo energético es alto y por lo tanto escala rápidamente, mientras que una vez que el sistema se estabiliza este crecimiento se vuelve mucho más lento hasta llegar a los valores descritos en la Tabla 5-35.

Tabla 5-35: Tabla resumen del torque ejercido por los propulsores para el control LQR.

Fuente: Elaboración propia

Torque Propulsores LQR	
Acumulado T1 (Nm)	3.05x10 ⁻⁴
Acumulado T2 (Nm)	3.35x10 ⁻⁴
Acumulado T3 (Nm)	1.15x10 ⁻⁴
TOTAL (Nm)	7.55x10⁻⁴

De la Tabla 5-35 se puede ver como la mayor utilización de torque se da por parte del T2, en segundo lugar, el T1 y por último el T3. La suma de estos nos da un total de 7.55x10⁻⁴ Nm.

En la Figura 5-31 se observa la utilización del torque por los propulsores a la hora de utilizar el controlador inteligente TD3.

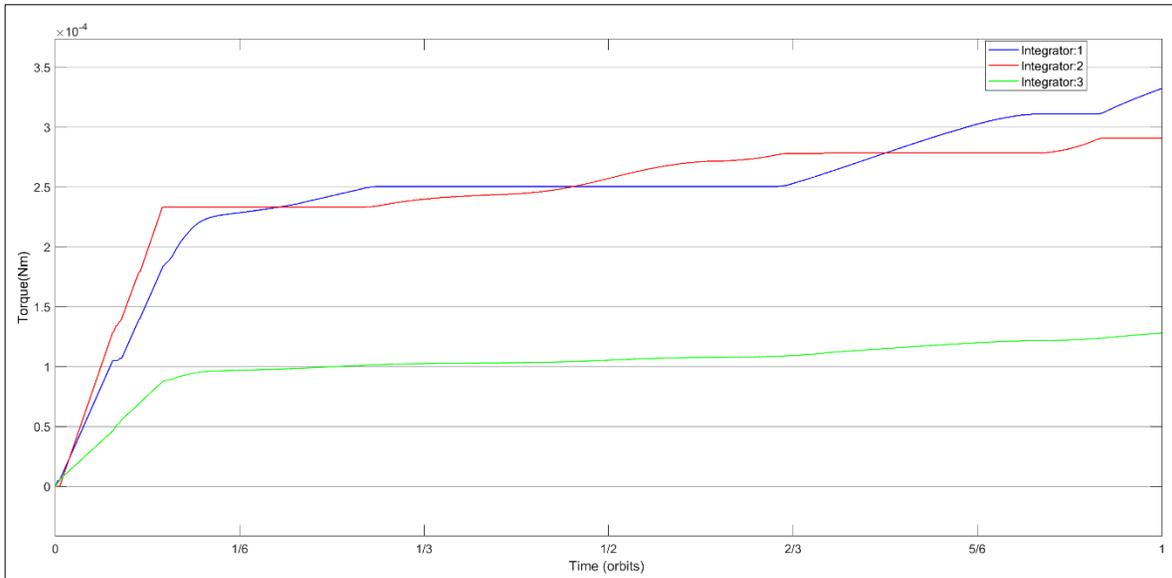


Figura 5-31: Torque utilizado por los propulsores con el controlador TD3

Fuente: Elaboración propia

De esta figura se puede apreciar cómo, al igual que en el caso del LQR, se presenta un aumento veloz al inicio de la simulación, y luego el crecimiento es lento. Los valores finales se describen en la Tabla 5-36.

Tabla 5-36: Tabla resumen del torque ejercido por los propulsores para el control TD3.

Fuente: Elaboración propia

Torque Propulsores TD3	
Acumulado T1 (Nm)	3.32×10^{-4}
Acumulado T2 (Nm)	2.91×10^{-4}
Acumulado T3 (Nm)	1.28×10^{-4}
TOTAL (Nm)	7.51×10^{-4}

De la Tabla 5-36 se puede observar como a diferencia del controlador LQR el T1 presenta el mayor torque acumulado, en segundo lugar, el T2 y por último el T3. Estos valores generan el total de torque acumulado de 7.51×10^{-4} Nm. En la Tabla 5-37 se presenta la comparación del torque acumulado de los diferentes controladores.

Tabla 5-37: Tabla resumen del torque ejercido por los propulsores para los distintos controladores.

Fuente: Elaboración propia

Controlador	Torque (Nm)	Mejoría con respecto al actual (%)
Actual (PD)	1.58×10^{-3}	0
LQR	7.55×10^{-4}	52
TD3	7.51×10^{-4}	53

Como se puede percibir cuando se comparan los totales de los controladores en la Tabla 5-37, el controlador TD3 es el más eficiente en el uso de sus recursos energéticos con una mejoría con respecto al PD de un 53%.

Para elegir entre el controlador LQR y TD3 es necesario analizar cuanto más eficiente es uno con respecto al otro. En la ecuación (5-32) se muestra cuanto más eficiente es el controlador TD3 relativo al controlador LQR.

$$\eta = 1 - \frac{\sum T_{TD3}}{\sum T_{LQR}} = 1 - \frac{7.51 \times 10^{-4}}{7.55 \times 10^{-4}} = 0.0053 = 0.53\% \quad (5-32)$$

Un 0.53% de eficiencia adicional quiere decir que se estarían adicionando a la misión, asumiendo que la duración sea de un año en la órbita de la ISS, un total de 1.93 días a la misión.

Esta mejora en eficiencia no se considera como una mejora suficientemente significativa como para elegir al TD3 solo por esta razón. Además, el TD3 no cumple con los requisitos para uno de los modos de operación. Por estas dos razones se tomó la decisión de que se implemente el LQR para los tres modos de operación, haciendo así más sencillo el sistema de control. Esto se resume en la Tabla 5-38.

Tabla 5-38: Controlador elegido por modo de operación

Fuente: Elaboración Propia

Modo	Controlador Elegido
Por Defecto	LQR
Apuntar Grueso	LQR
Apuntar Fino	LQR

6 Conclusiones

En este proyecto se logró diseñar dos controladores distintos para el nanosatélite GWSat, los cuales se centran en la optimización del consumo energético para alargar la vida útil de la misión.

1. Se determinaron por medio de tablas morfológicas un controlador clásico, de tipo LQR, y un controlador inteligente de tipo TD3 para el algoritmo de control avanzado del GWSat.
2. Se diseñó un controlador clásico de tipo LQR, el cual presenta un ángulo de error de 0.15° , un error de la velocidad angular de 3×10^{-5} rad/s y un tiempo de estabilización de 1000 s. Además, este controlador es 52% más eficiente en el uso de energía de los propulsores que el controlador actual de tipo PD. En los apéndices A.7, A.8, A.9 y A.10 se pueden encontrar los diagramas de bloques, así como las ecuaciones y código que describen a este controlador y su desarrollo.
3. Se diseñó un controlador inteligente de tipo TD3, el cual presenta un ángulo de error de 1.4° , un error de la velocidad angular de 7.3×10^{-6} rad/s y un tiempo de estabilización de 700 s. Además, este controlador es 53% más eficiente en el uso de energía de los propulsores que el controlador actual de tipo PD. En los apéndices A.11, A.12 y A.13 se pueden encontrar los diagramas de bloques y el código que componen este controlador.
4. Se compararon los controladores LQR y TD3 basado en su utilización energética y cumplimiento de requerimientos. Esta comparación mostró que el TD3 cumple con los requisitos para el modo de apuntar brusco y el modo por defecto, mientras que el LQR además de esos dos, también cumple con los requisitos para el modo de apuntar fino. Con respecto al uso energético que el TD3 es 0.53% más eficiente que el LQR. Para reducir la complejidad del sistema de control se seleccionó el controlador LQR para los 3 modos de operación.

7 Recomendaciones

Como parte del desarrollo de este proyecto se identificaron aspectos que pueden contribuir a mejorar los resultados en futuros proyectos.

Con respecto a la selección de controladores, se recomienda considerar los campos de algoritmos de control robusto y control no lineal, de manera que se pueda tener información sobre todo el espectro de posibles controladores para un satélite.

Se recomienda utilizar un modelo que contemple las perturbaciones y la dinámica completa del sistema para seguir optimizando el LQR de la mejor manera.

Para el entrenamiento de la red, se recomienda fuertemente el entrenamiento con una arquitectura mucho más compleja, de manera que se pueda encontrar un algoritmo que presente el mejor rendimiento en todos los parámetros de interés. Para esto también se recomienda obtener equipo de alto poder computacional que se pueda dedicar al entrenamiento de algoritmos inteligentes en tiempos razonables.

Dada la gran cantidad de combinaciones posibles que presentan los hiper parámetros, para encontrar la mejor combinación de estos para la red neuronal se recomienda el uso de un algoritmo de optimización, como un algoritmo genético en el cual se toman los valores promedios finales de la recompensa promedio como la función de calidad. En este caso también se recomienda fuertemente el uso de computadoras de alto poder computacional dedicadas a esto.

Por último, se recomienda explorar la posibilidad de un controlador híbrido, el cual implemente tanto un controlador clásico como el controlador inteligente, aprovechando de esta manera las fortalezas de cada uno de estos paradigmas.

8 Bibliografía

- [1] C. Rodríguez, «Lanzan Cluster Aeroespacial en Costa Rica,» *Semanario Universidad*, 8 Marzo 2016.
- [2] SETEC-Lab, «Acerca del TEC: Departamentos: Laboratorio de Sistemas Espaciales,» [En línea]. Available: <https://www.tec.ac.cr/unidades/laboratorio-sistemas-espaciales>.
- [3] J. Umaña, «Histórico: el primer satélite costarricense funciona en el espacio y hace contacto,» Cartago, 2018.
- [4] J. Umaña, «UNIVERSIDAD GEORGE WASHINGTON Y TEC SON SOCIOS EN NUEVA MISIÓN ESPACIAL,» Cartago, 2018.
- [5] K. Schwab, «The Fourth Industrial Revolution: what it means, how to respond,» World Economic Forum, 2016.
- [6] A. R. Fazylab, F. F. Saberi y M. Kabganian, «Adaptive attitude controller for a satellite based on neural network,» *Advances in Space Research*, nº 57, pp. 367-377, 2016.
- [7] J. Kolbeck, «GW-Sat Preliminary Design Review Document,» Washington D.C, 2017.
- [8] NASA CubeSat Launch Initiative, «CubeSat101: Basic Concepts and Processes for First-Time CubeSat Developers,» 2017.
- [9] California Polytechnic State University CubeSat Program, «CubeSat Design Specification,» 2014.
- [10] Members of the Technical Staff of the Attitude Systems Operation and Computer Sciences Corporation, *Spacecraft Attitude Determination and Control*, Dordrecht: Kluwer Academic Publishers, 2002.
- [11] C. Fernandez-Cerdas, «Attitude Control Design for Station Keeping Mode in the GW-Sat nanosatellite,» 2020.
- [12] J. Reijneveld, «Design of the Attitude Determination and Control Algorithms for the Delfi-n3Xt,» 2012.
- [13] G. H. J. v. Vuuren, «Design and Simulation Analysis of an Attitude Determination and Control System for a Small Earth Observation Satellite,» Stellenbosch, 2015.
- [14] J. Crassidis, *Fundamentals of Spacecraft Attitude Determination and Control*.
- [15] B. Wie, *Space Vehicles Dynamics and Control*, American Institute of Aeronautics and Astronautics, 2008.

- [16] ai-solutions, «FreeFlyer University Guide: J2 Perturbation,» [En línea]. Available: https://ai-solutions.com/_freelyeruniversityguide/j2_perturbation.htm.
- [17] J. Elfving, «Attitude and Orbit Control for Small Satellites,» Linköping, 2002.
- [18] M. Keidar, «Micro-Cathode Arc Thruster for Small Satellite Propulsion,» 2016.
- [19] A. P. Engelbrecht, Computational Intelligence: An Introduction, John Wiley and Sons, 2007.
- [20] W. Z. X.-h. L. M.-h. L. Ping Guan, «The Direct Adaptative Fuzzy Control for Satellite Attitude Control,» de *Chinese Control and Decision Conference*, 2011.
- [21] N. Siddique y H. Adeli, Computational Intelligence: Synergies of Fuzzy Logic, Neural Networks and Evolutionary Computing, Wiley, 2013.
- [22] University of Illinois, «Syllabus: CS357,» 25 November 2017. [En línea]. Available: <https://courses.engr.illinois.edu/cs357/fa2019/references/ref-12-opt-nd/>.
- [23] S. T. Mohammed, A. Bytyn, G. Ascheid y G. Dartmann, «Reinforcement learning and deep neural network for autonomous driving,» de *Big Data Analytics for Cyber-Physical Systems*, Elsevier, 2019, pp. 187-213.
- [24] D. Du y M. Fei, «A two-layer networked learning control system using actor-critic neural network,» *Applied Mathematics and Computation*, n° 205, pp. 26-36, 2008.
- [25] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver y D. Wierstra, «Continuous Control with Deep Reinforcement Learning,» de *International Conference on Learning Representations*, London, 2016.
- [26] S. Fujimoto, H. van Hoof y D. Meger, «Addressing Function Approximation Error in Actor-Critic Methods,» de *International Conference on Machine Learning*, Stockholm, 2018.
- [27] D. Byrne, «TD3: Learning to Run with AI,» Towards Data Science, 15 June 2019. [En línea]. Available: <https://towardsdatascience.com/td3-learning-to-run-with-ai-40dfc512f93>.
- [28] K. Ogata, Ingeniería de Control Moderna, 5ta ed., M. Martín-Romo, Ed., Madrid: Pearson Educación, 2010.
- [29] C. Meza, Interviewee, *Curso Control Automático*. [Entrevista]. 2019.
- [30] «PID ¿Como sintonizar un lazo PID?: Control Real Español,» 4 Noviembre 2017. [En línea]. Available: <https://controlreal.com/es/pid-como-sintonizar-un-lazo-pid/>.

- [31] Y. Li, S. Zhaowei y Y. Dong, «Time Efficient Robust PID Plus Controller for Satellite Attitude Stabilization Control Considering Angular Velocity and Control Torque Constraint,» *Journal of Aerospace Engineering*, 2017.
- [32] A. Ghaedi y M. A. Nekoui, «3-Axes Satellite Attitude Control Based on Biased Angular Momentum,» de *International Power Electronics and Motion Control Conference*, 2008.
- [33] M. C. Mahdi y M. J. Al-Bermani, «LQR Controller for Kufasat,» *JOURNAL OF KUFA – PHYSICS*, vol. 6, n° 1, pp. 13-20, 2014.
- [34] X.-t. Cui y X.-d. Liu, «Fuzzy Neural Control of Satellite Attitude by TD Based Reinforcement Learning,» de *World Congress on Intelligent Control and Automation*, Dalian, 2006.
- [35] D. Yadava, R. Hosangadi, S. Krishna, P. Paliwal y A. Jain, «Attitude Control of a Nanosatellite system using Reinforcement Learning and Neural Networks,» Karnataka, 2018.
- [36] I. Ofodile, H. Ehrpais, A. Slavinskis y G. Anbarjafari, «Stabilised LQR Control and Optimised Spin Rate Control for Nanosatellites,» Tartu, 2019.
- [37] GWU, *GWSat Operational Modes*, Washington, 2019.
- [38] S. Castro, «Deep Reinforcement Learning for Walking Robots,» MathWorks, 2020.
- [39] R. Islam, P. Henderson, M. Gomrokchi y D. Precup, «Reproducibility of Benchmarked Deep Reinforcement Learning Tasks for Continuous Control,» Montreal, 2017.
- [40] D. Davis, «Noise parameters in Reinforcement learning DDPG: MATLAB Answers,» Mathworks, 19 June 2019. [En línea]. Available: <https://www.mathworks.com/matlabcentral/answers/467153-noise-parameters-in-reinforcement-learning-ddpg>.
- [41] I. Goodfellow, Y. Bengio y A. Courville, de *Deep Learning*, MIT Press, 2016, p. 434.
- [42] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup y D. Meger, «Deep Reinforcement Learning that Matters,» Montreal, 2019.
- [43] Z. Salloum, «Basics of Reinforcement Learning, the Easy Way,» Medium, 29 Agosto 2018. [En línea]. Available: <https://medium.com/@zsalloum/basics-of-reinforcement-learning-the-easy-way-fb3a0a44f30e#:~:text=Discount%20factor%20is%20a%20value,ahead%20of%20our%20current%20state..>
- [44] A. Rao, *Dynamics of Particles and Rigid Bodies: A Systematic Approach*, 1st ed., Boston, Massachusetts: Cambridge University Press, 2006.

[45] B. Dunbar, 14 Febrero 2018. [En línea]. Available:
https://www.nasa.gov/mission_pages/cubesats/overview.

[46] J. L. Crespo-Mariño, «Introducción a la Lógica Difusa,» Cartago, 2018.

Apéndices

A.1 Tiempos de ejecución de los algoritmos de control

Tiempos de Ejecucion (ms)		
Iteración	PD	LQR
1	19.2941498	16.386808
2	19.2920431	16.2161452
3	19.2111267	15.943067
4	19.1810709	16.3923453
5	19.3097144	16.285663
6	19.4729509	15.9891849
7	19.2876429	16.1178062
8	19.1398504	16.2412033
9	19.4568868	16.0166783
10	19.4131664	16.2706695
11	19.3275851	15.9994078
12	19.4332952	16.1698024
13	19.6934716	16.4190322
14	19.4567467	16.4545751
15	19.5107136	16.4556432
16	19.9839324	16.0275887
17	19.7496098	16.3429905
18	19.5693397	16.3642365
19	19.3815396	16.2844861
20	19.324789	15.9787955
21	19.2754692	16.0000786
22	19.2064402	16.4536162
23	19.4086485	16.5043494
24	19.3653397	15.970364
25	19.5929968	16.0123914
26	19.3190878	16.4008637
27	19.4872989	16.4202142
28	19.3664727	16.3299044
29	19.4045523	16.3474036
30	19.3968895	16.041006
31	19.5384592	15.9486606
32	19.4655682	16.4201861
33	19.5214875	16.6383608
34	19.3703233	16.2117171
35	19.439613	15.9825301
36	19.5910484	16.3867354
37	19.4640625	16.4392436
38	19.4847201	16.7838635
39	19.4028333	16.5661679
40	19.8025346	16.043337
41	19.6457286	16.1245759
42	19.5902241	16.5016957
43	19.4467875	16.405967
44	19.4876537	16.3348638
45	19.303825	16.1224503
46	19.3573156	16.0404419
47	19.8626342	16.0633835
48	19.5182901	15.9497744
49	19.5285421	16.3014732
50	19.4099099	16.0337914
Promedio	19.45088763	16.24271079

A.2 Iteraciones de diseño LQR

Iteración 1	
Matriz Q	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$
Matriz R	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$
Matriz K	$\begin{pmatrix} 1 & 0 & 0 & 1.0208 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1.0208 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1.0033 \end{pmatrix}$
Tiempo de estabilización(s)	No estabiliza
Error Velocidad Angular (rad/s)	NA
Error Ángulo de Puntería (°)	NA
Consumo de Torque (N*m)	NA
Indice de Rendimiento	0

Iteración 2	
Matriz Q	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$
Matriz R	$\begin{pmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.5 \end{pmatrix}$
Matriz K	$\begin{pmatrix} 3.1623 & 0 & 0 & 3.1832 & 0 & 0 \\ 0 & 3.1623 & 0 & 0 & 3.1832 & 0 \\ 0 & 0 & 3.1623 & 0 & 0 & 3.1832 \end{pmatrix}$
Tiempo de estabilización(s)	No estabiliza
Error Velocidad Angular (rad/s)	NA
Error Ángulo de Puntería (°)	NA
Consumo de Torque (N*m)	NA
Indice de Rendimiento	0

Iteración 3	
Matriz Q	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 10 \end{pmatrix}$
Matriz R	$\begin{pmatrix} 0.05 & 0 & 0 \\ 0 & 0.05 & 0 \\ 0 & 0 & 0.05 \end{pmatrix}$
Matriz K	$\begin{pmatrix} 4.4721 & 0 & 0 & 14.1488 & 0 & 0 \\ 0 & 4.4721 & 0 & 0 & 14.1488 & 0 \\ 0 & 0 & 4.4721 & 0 & 0 & 14.1432 \end{pmatrix}$
Tiempo de estabilización(s)	No estabiliza
Error Velocidad Angular (rad/s)	NA
Error Ángulo de Puntería (°)	NA
Consumo de Torque (N*m)	NA
Indice de Rendimiento	0

Iteración 6	
Matriz Q	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 10000 \end{pmatrix}$
Matriz R	$\begin{pmatrix} 0.05 & 0 & 0 \\ 0 & 0.05 & 0 \\ 0 & 0 & 0.05 \end{pmatrix}$
Matriz K	$\begin{pmatrix} 4.4721 & 0 & 0 & 447.2138 & 0 & 0 \\ 0 & 4.4721 & 0 & 0 & 447.2138 & 0 \\ 0 & 0 & 4.4721 & 0 & 0 & 447.2138 \end{pmatrix}$
Tiempo de estabilización(s)	2000
Error Velocidad Angular (rad/s)	0.00025
Error Ángulo de Puntería (°)	0.2
Consumo de Torque (N*m)	0.08047
Indice de Rendimiento	0.13

Iteración 7	
Matriz Q	$\begin{pmatrix} 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \end{pmatrix}$
Matriz R	$\begin{pmatrix} 0.05 & 0 & 0 \\ 0 & 0.05 & 0 \\ 0 & 0 & 0.05 \end{pmatrix}$
Matriz K	$\begin{pmatrix} 1.4142 & 0 & 0 & 44.7220 & 0 & 0 \\ 0 & 1.4142 & 0 & 0 & 44.7220 & 0 \\ 0 & 0 & 1.4142 & 0 & 0 & 44.7215 \end{pmatrix}$
Tiempo de estabilización(s)	1400
Error Velocidad Angular (rad/s)	0.00020
Error Ángulo de Puntería (°)	0.1
Consumo de Torque (N*m)	0.08049
Indice de Rendimiento	0.15

Iteración 8	
Matriz Q	$\begin{pmatrix} 0.001 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.001 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.001 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$
Matriz R	$\begin{pmatrix} 0.05 & 0 & 0 \\ 0 & 0.05 & 0 \\ 0 & 0 & 0.05 \end{pmatrix}$
Matriz K	$\begin{pmatrix} 0.1414 & 0 & 0 & 4.4728 & 0 & 0 \\ 0 & 0.1414 & 0 & 0 & 4.4728 & 0 \\ 0 & 0 & 0.1414 & 0 & 0 & 4.4722 \end{pmatrix}$
Tiempo de estabilización(s)	1400
Error Velocidad Angular (rad/s)	0.00020
Error Ángulo de Puntería (°)	0.15
Consumo de Torque (N*m)	0.08018
Indice de Rendimiento	0.15

Iteración 4	
Matriz Q	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 100 & 0 & 0 \\ 0 & 0 & 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \end{pmatrix}$
Matriz R	$\begin{pmatrix} 0.05 & 0 & 0 \\ 0 & 0.05 & 0 \\ 0 & 0 & 0.05 \end{pmatrix}$
Matriz K	$\begin{pmatrix} 4.4721 & 0 & 0 & 44.7235 & 0 & 0 \\ 0 & 4.4721 & 0 & 0 & 44.7235 & 0 \\ 0 & 0 & 4.4721 & 0 & 0 & 44.7217 \end{pmatrix}$
Tiempo de estabilización(s)	No estabiliza
Error Velocidad Angular (rad/s)	NA
Error Ángulo de Puntería (°)	NA
Consumo de Torque (N*m)	NA
Índice de Rendimiento	0

Iteración 5	
Matriz Q	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1000 \end{pmatrix}$
Matriz R	$\begin{pmatrix} 0.05 & 0 & 0 \\ 0 & 0.05 & 0 \\ 0 & 0 & 0.05 \end{pmatrix}$
Matriz K	$\begin{pmatrix} 4.4721 & 0 & 0 & 141.4220 & 0 & 0 \\ 0 & 4.4721 & 0 & 0 & 141.4220 & 0 \\ 0 & 0 & 4.4721 & 0 & 0 & 141.4215 \end{pmatrix}$
Tiempo de estabilización(s)	1400
Error Velocidad Angular (rad/s)	0.00020
Error Ángulo de Puntería (°)	0.05
Consumo de Torque (N*m)	0.08047
Índice de Rendimiento	0.16

Iteración 9	
Matriz Q	$\begin{pmatrix} 10^{-5} & 0 & 0 & 0 & 0 & 0 \\ 0 & 10^{-5} & 0 & 0 & 0 & 0 \\ 0 & 0 & 10^{-5} & 0 & 0 & 0 \\ 0 & 0 & 0 & 10^{-2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 10^{-2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 10^{-2} \end{pmatrix}$
Matriz R	$\begin{pmatrix} 0.05 & 0 & 0 \\ 0 & 0.05 & 0 \\ 0 & 0 & 0.05 \end{pmatrix}$
Matriz K	$\begin{pmatrix} 0.1414 & 0 & 0 & 4.4728 & 0 & 0 \\ 0 & 0.1414 & 0 & 0 & 4.4728 & 0 \\ 0 & 0 & 0.1414 & 0 & 0 & 4.4722 \end{pmatrix}$
Tiempo de estabilización(s)	1400
Error Velocidad Angular (rad/s)	0.00010
Error Ángulo de Puntería (°)	0.15
Consumo de Torque (N*m)	0.08035
Índice de Rendimiento	0.15

LQR	
Matriz Q	$\begin{pmatrix} 10^{-9} & 0 & 0 & 0 & 0 & 0 \\ 0 & 10^{-9} & 0 & 0 & 0 & 0 \\ 0 & 0 & 10^{-9} & 0 & 0 & 0 \\ 0 & 0 & 0 & 10^{-6} & 0 & 0 \\ 0 & 0 & 0 & 0 & 10^{-6} & 0 \\ 0 & 0 & 0 & 0 & 0 & 10^{-6} \end{pmatrix}$
Matriz R	$\begin{pmatrix} 0.05 & 0 & 0 \\ 0 & 0.05 & 0 \\ 0 & 0 & 0.05 \end{pmatrix}$
Matriz K	$\begin{pmatrix} 0.0001 & 0 & 0 & 0.0051 & 0 & 0 \\ 0 & 0.0001 & 0 & 0 & 0.0051 & 0 \\ 0 & 0 & 0.0001 & 0 & 0 & 0.0051 \end{pmatrix}$
Tiempo de estabilización(s)	1000
Error Velocidad Angular (rad/s)	0.000015
Error Ángulo de Puntería (°)	0.15
Consumo de Torque (N*m)	0.0017969
Índice de Rendimiento	0.57

A.3 Iteraciones del diseño del controlador inteligente

Iteración	1	2	3	4
Algoritmo	DDPG	DDPG	DDPG	DDPG
Limitado Q y T	No	No	No	No
Factor de Descuento	0.99	0.99	0.99	0.99
Tamaño Mini Lote	64	128	128	128
Varianza del Modelo de Exploración	0.5	0.5	0.5	0.85
Tasa de Decadencia de la Varianza del Modelo de Exploración	1.00E-05	1.00E-05	1.00E-05	1.00E-06
Varianza de la Política Objetivo	NA	NA	NA	NA
Tasa de Decadencia de la Varianza de la Política Objetivo	NA	NA	NA	NA
Frecuencia de Actualización Política	NA	NA	NA	NA
Frecuencia de Actualización Objetivo	NA	NA	NA	NA
Tasa Aprendizaje Actor	0.001	0.001	0.001	0.0001
Tasa Aprendizaje Crítico	0.001	0.001	0.001	0.001
Arquitectura de Red	[400 300]	[400 300]	[400 300]	[400 300]
Funciones de Recompensa	$\begin{cases} RQ = -EQ * 1.2 \\ R\omega = -E\omega \\ RT = -0.1ET \end{cases}$	$\begin{cases} RQ = -EQ * 1.2 \\ R\omega = -E\omega \\ RT = -0.1ET \end{cases}$	$\begin{cases} RQ = 1.2(1 - 2(EQ)) \\ R\omega = 1 - 2(E\omega) \\ RT = 0.15(1 - 2(ET)) \end{cases}$	$\begin{cases} RQ = 1.2(1 - 2(EQ)) \\ R\omega = 1 - 2(E\omega) \\ RT = 0.15(1 - 2(ET)) \end{cases}$
Resultado	No hubo aprendizaje, hay divergencia	Se ve un comportamiento un poco mejor, todavía no hay aprendizaje, hay divergencia	Sigue sin haber convergencia, hay aprendizaje muy leve	Mejora el aprendizaje, hay divergencia

Iteración	5	6	7	8
Algoritmo	DDPG	DDPG	DDPG	DDPG
Limitado Q y T	No	No	Sí	Sí
Factor de Descuento	0.99	0.99	0.99	0.9995
Tamaño Mini Lote	128	128	128	128
Varianza del Modelo de Exploración	0.85	0.5	0.85	0.85
Tasa de Decadencia de la Varianza del Modelo de Exploración	1.00E-06	1.00E-06	1.00E-06	1.00E-06
Varianza de la Política Objetivo	NA	NA	NA	NA
Tasa de Decadencia de la Varianza de la Política Objetivo	NA	NA	NA	NA
Frecuencia de Actualización Política	NA	NA	NA	NA
Frecuencia de Actualización Objetivo	NA	NA	NA	NA
Tasa Aprendizaje Actor	0.0001	0.0001	0.0001	0.0003
Tasa Aprendizaje Crítico	0.001	0.001	0.001	0.001
Arquitectura de Red	[400 300]	[400 300]	[100 50 25]	[100 50 25]
Funciones de Recompensa	$\begin{cases} RQ = -Ln(2 * EQ) \\ R\omega = -Ln(2 * E\omega) \\ RT = -0.15Ln(2 * ET) \end{cases}$	$\begin{cases} RQ = -Ln(2 * EQ) \\ R\omega = -Ln(2 * E\omega) \\ RT = -0.15Ln(2 * ET) \end{cases}$	$\begin{cases} RQ = -1.2Ln(5 * EQ) \\ R\omega = -Ln(100 * E\omega) \\ RT = -0.15Ln(2 * 10^5 * ET) \end{cases}$	$\begin{cases} RQ = -1.2Ln(5 * EQ) \\ R\omega = -Ln(100 * E\omega) \\ RT = -0.15Ln(2 * 10^5 * ET) \end{cases}$
Resultado	Hay convergencia y aprendizaje. Se mejoró el desempeño significamente, todavía se está lejos de los parámetros requeridos	Hay convergencia y aprendizaje. El desempeño empeoró	Mejora considerablemente el desempeño, se encuentra moderadamente cerca a los requerimientos mínimos. Además, bajó el coste computacional	El pico de aprendizaje llega a un valor muy cercano a los objetivos, luego sufre sobreestimación y por lo tanto, desaprendizaje. Se opta por cambiar a TD3

Iteración	9	10	11
Algoritmo	TD3	TD3	TD3
Limitado Q y T	Sí	Sí	Sí
Factor de Descuento	0.9995	0.9995	0.9995
Tamaño Mini Lote	128	128	128
Varianza del Modelo de Exploración	0.85	0.85	0.85
Tasa de Decadencia de la Varianza del Modelo de Exploración	1.00E-06	2.00E-07	1.00E-06
Varianza de la Política Objetivo	0.85	0.85	0.85
Tasa de Decadencia de la Varianza de la Política Objetivo	1.00E-06	2.00E-07	1.00E-06
Frecuencia de Actualización Política	2	2	2
Frecuencia de Actualización Objetivo	2	2	2
Tasa Aprendizaje Actor	0.0003	0.0003	0.0001
Tasa Aprendizaje Crítico	0.001	0.001	0.001
Arquitectura de Red	[100 50 25]	[100 50 25]	[150 75 38]
Funciones de Recompensa	$\begin{cases} RQ = -1.2\text{Ln}(10 * EQ) \\ R\omega = -\text{Ln}(100 * E\omega) \\ RT = -0.01\text{Ln}(2 \times 10^5 * ET) \end{cases}$	$\begin{cases} RQ = -1.2\text{Ln}(10 * EQ) \\ R\omega = -\text{Ln}(100 * E\omega) \\ RT = -0.01\text{Ln}(2 \times 10^5 * ET) \end{cases}$	$\begin{cases} RQ = -1.2\text{Ln}(10 * EQ) \\ R\omega = -\text{Ln}(100 * E\omega) \\ RT = -0.01\text{Ln}(2 \times 10^5 * ET) \end{cases}$
Resultado	El pico de aprendizaje llega a un valor cercano a los requerimientos mínimos, luego no logra salir de un mínimo local y por lo tanto, no logra seguir aprendiendo	Resultados prácticamente iguales a la iteración 9	Se logra una estabilización exitosa prácticamente dentro de los parámetros deseados. El consumo energético es un poco menor al LQR.

Iteración	12	13
Algoritmo	TD3	TD3
Limitado Q y T	Sí	Sí
Factor de Descuento	0.9995	0.9995
Tamaño Mini Lote	128	128
Varianza del Modelo de Exploración	0.85	0.85
Tasa de Decadencia de la Varianza del Modelo de Exploración	5.00E-07	5.00E-07
Varianza de la Política Objetivo	0.85	0.85
Tasa de Decadencia de la Varianza de la Política Objetivo	5.00E-07	5.00E-07
Frecuencia de Actualización Política	2	2
Frecuencia de Actualización Objetivo	2	2
Tasa Aprendizaje Actor	0.00005	0.0001
Tasa Aprendizaje Crítico	0.001	0.001
Arquitectura de Red	[150 75 38]	[150 75 38]
Funciones de Recompensa	$\begin{cases} RQ = -1.2\text{Ln}(10 * EQ) \\ R\omega = -\text{Ln}(100 * E\omega) \\ RT = -0.05\text{Ln}(2 \times 10^5 * ET) \\ RTt = -0.1\text{Ln}(2 \times 10^6 * Tt) \end{cases}$	$\begin{cases} RQ = -1.2\text{Ln}(10 * EQ) \\ R\omega = -\text{Ln}(100 * E\omega) \\ RT = -0.05\text{Ln}(2 \times 10^5 * ET) \\ RTt = -0.1\text{Ln}(2 \times 10^6 * Tt) \end{cases}$
Resultado	Se logra una estabilización exitosa prácticamente dentro de los parámetros deseados. El consumo energético es un poco mayor al LQR.	Se hizo a modo de prueba sin aleatoriedad al inicio de cada episodio. Logró una estabilización excelente dentro de los parámetros requerido. Esto quiere decir que con una red suficientemente compleja y con una computadora suficientemente poderosa es posible alcanzar la estabilización en un tiempo razonable

A.4 Mediciones tiempo de ejecución por modelo

Tiempos de Ejecución (s)		
Iteració	Simplificado	Completo
1	2.6884653	16.3410947
2	2.8558367	16.4834052
3	2.8468415	16.3967439
4	2.4630406	16.3577631
5	2.2273471	16.4133123
6	2.2264542	16.3364556
7	2.4160874	16.3967255
8	2.130607	16.3297892
9	2.2432291	16.3698382
10	2.2645918	16.7283277
11	2.4698169	16.4178371
12	2.0606732	16.4330012
13	2.012928	17.1337188
14	2.1131777	17.1389034
15	2.3562702	16.5078841
16	2.1758425	16.902745
17	2.3424758	16.4517204
18	2.2370043	16.3666782
19	2.2822755	16.4438365
20	2.4961974	16.3491118
21	2.3992822	16.4223887
22	2.442599	16.4943064
23	2.0021017	16.5934838
24	2.3254028	16.5300638
25	2.2134213	16.4594557
26	2.2561033	16.3916516
27	2.2268038	16.3803754
28	2.2748329	16.3203048
29	2.5712658	16.4161824
30	2.7164475	16.5540305
31	2.2646524	16.2430896
32	2.1899693	16.3591603
33	2.3918779	16.3961738
34	2.2405445	16.5895966
35	2.1152232	16.3987509
36	2.2018572	16.3453745
37	2.1321574	16.3687862
38	2.4991026	16.3870374
39	2.1177431	16.3881219
40	2.1713401	16.3304097
41	2.2959671	16.5232552
42	2.2897022	16.3886968
43	2.2513076	16.3496932
44	2.3983865	16.3626914
45	2.0781579	16.3000435
46	2.1325478	16.3803
47	2.5395147	16.436714
48	2.4405403	16.8915701
49	2.2127026	16.5456248
50	2.2862354	16.651772
Promedio	2.311739046	16.46995994

A.5 Mediciones tiempo de ejecución por arquitectura

Tiempos de Ejecución (s)		
Iteració	[400 300]	[100 50 25]
1	2.69	2.25
2	2.86	1.99
3	2.85	1.97
4	2.46	1.87
5	2.23	2.06
6	2.23	2.20
7	2.42	1.93
8	2.13	2.03
9	2.24	1.83
10	2.26	1.94
11	2.47	2.00
12	2.06	1.91
13	2.01	1.99
14	2.11	1.94
15	2.36	1.89
16	2.18	1.93
17	2.34	1.98
18	2.24	1.91
19	2.28	2.00
20	2.50	1.93
21	2.40	1.90
22	2.44	1.99
23	2.00	1.94
24	2.33	2.05
25	2.21	1.95
26	2.26	1.97
27	2.23	1.99
28	2.27	2.00
29	2.57	1.95
30	2.72	1.95
31	2.26	2.05
32	2.19	1.97
33	2.39	2.02
34	2.24	1.87
35	2.12	2.00
36	2.20	1.81
37	2.13	2.01
38	2.50	2.01
39	2.12	2.01
40	2.17	1.93
41	2.30	1.92
42	2.29	1.88
43	2.25	1.93
44	2.40	1.92
45	2.08	2.02
46	2.13	1.95
47	2.54	1.93
48	2.44	2.00
49	2.21	1.95
50	2.29	1.91
Promedio	2.31	1.97

A.6 Resultados prueba sin aleatoriedad

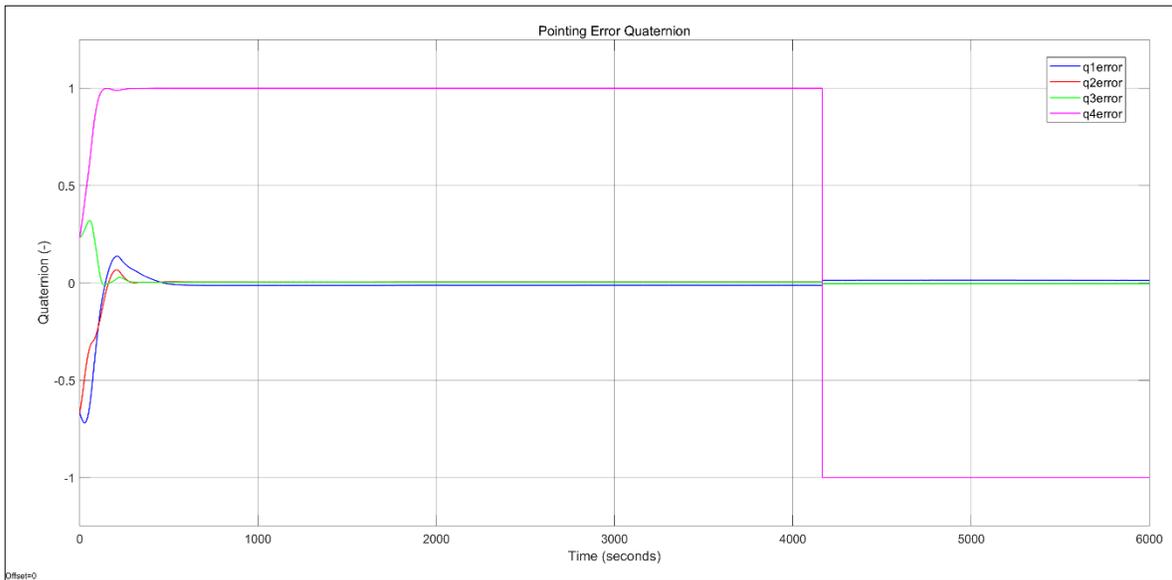


Figura A 1: Error de los cuaternios en iteración de prueba

Fuente: Elaboración Propia

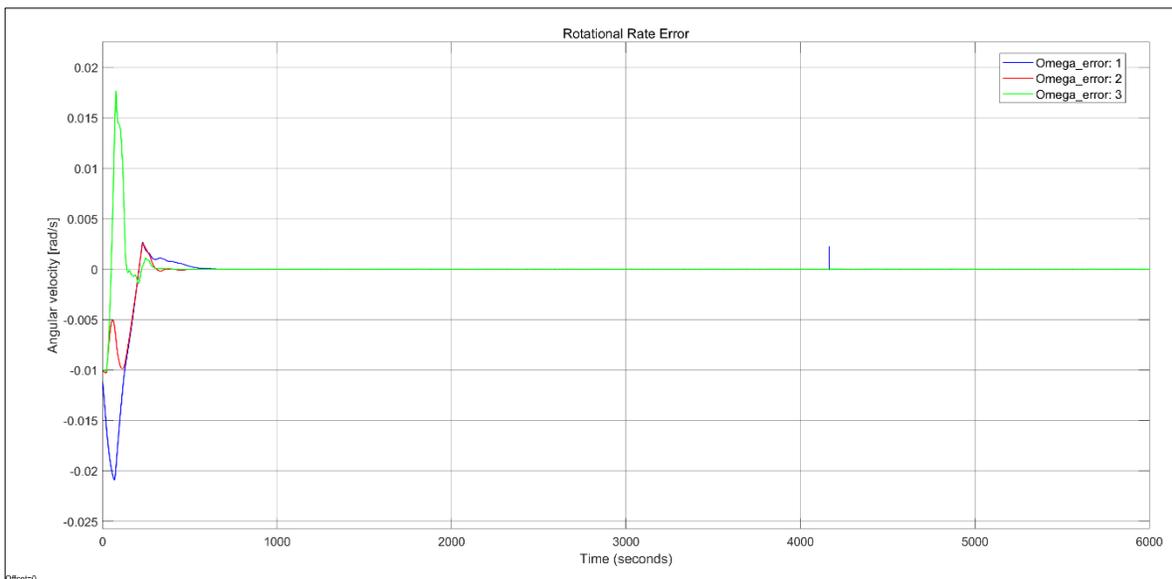


Figura A 2: Error en la velocidad angular de la iteración de prueba

Fuente: Elaboración propia

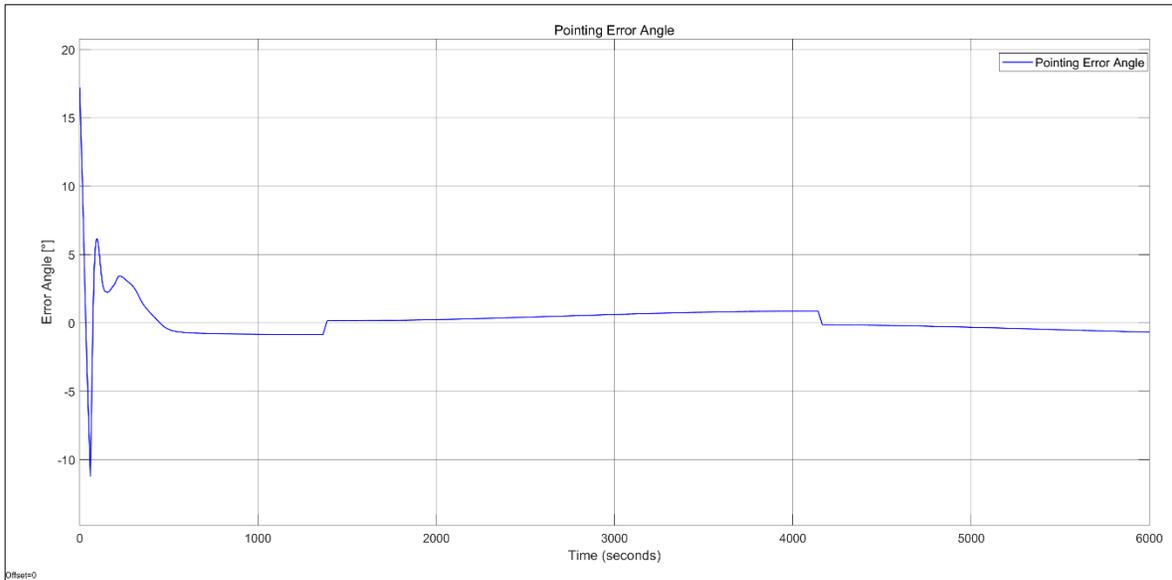


Figura A 3: Error en el ángulo de puntería de la iteración de prueba.

Fuente: Elaboración propia

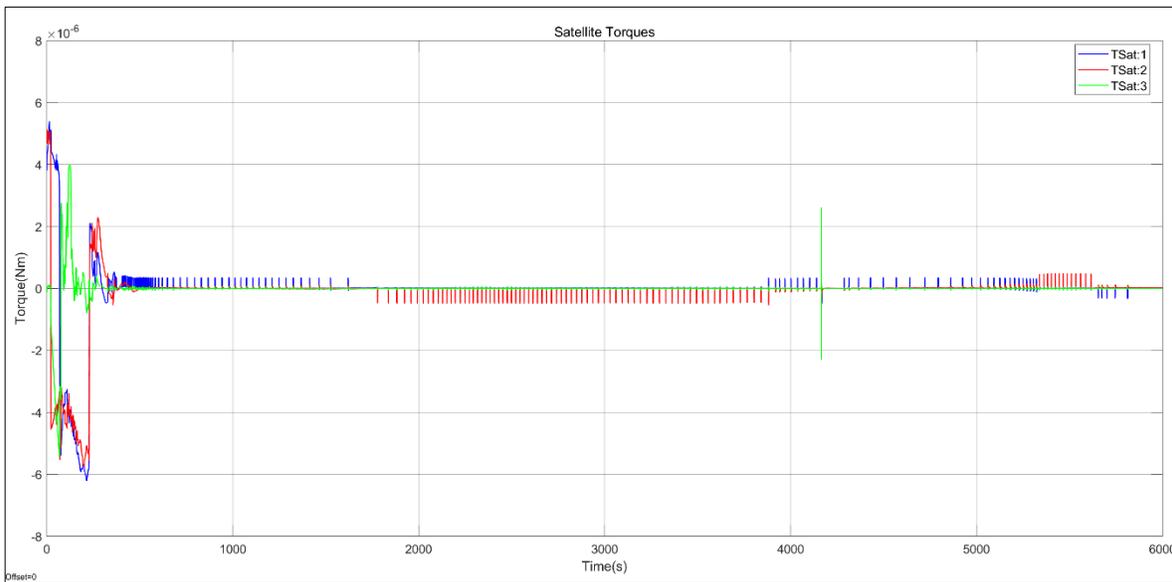


Figura A 4: Torque generado por el satélite con la iteración de prueba

Fuente: Elaboración propia

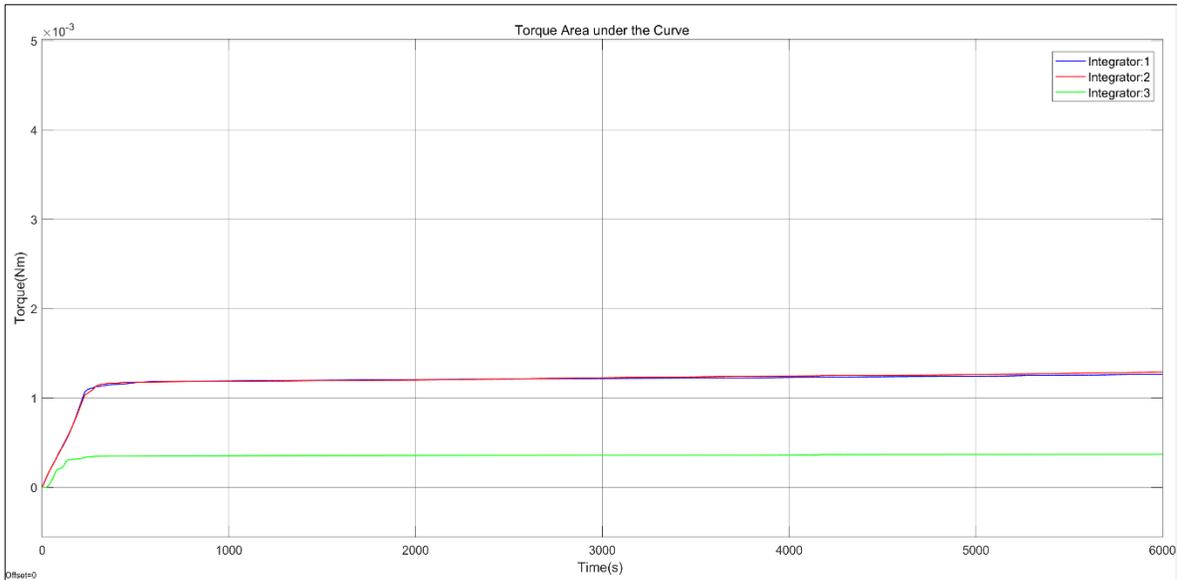


Figura A 5: Torque acumulado del satélite de la iteración de prueba

Fuente: Elaboración propia

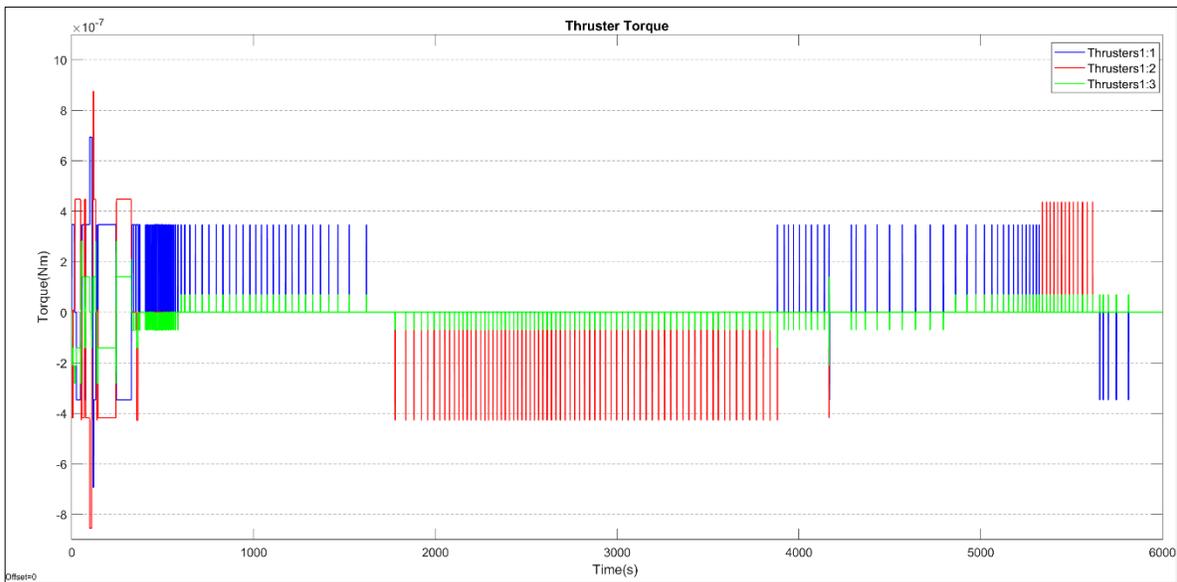


Figura A 6: Torque generado por los propulsores de la iteración de prueba

Fuente: Elaboración propia

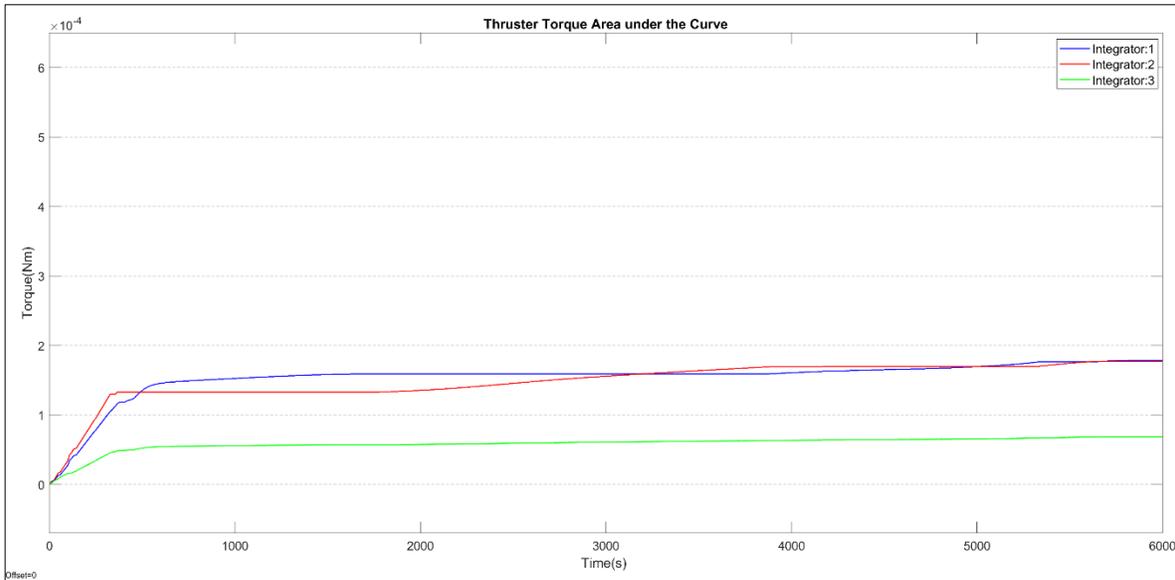


Figura A 7: Torque acumulado por los propulsores de la iteración de prueba

Fuente: Elaboración propia

A.7 Diagramas de bloques del sistema con el controlador LQR

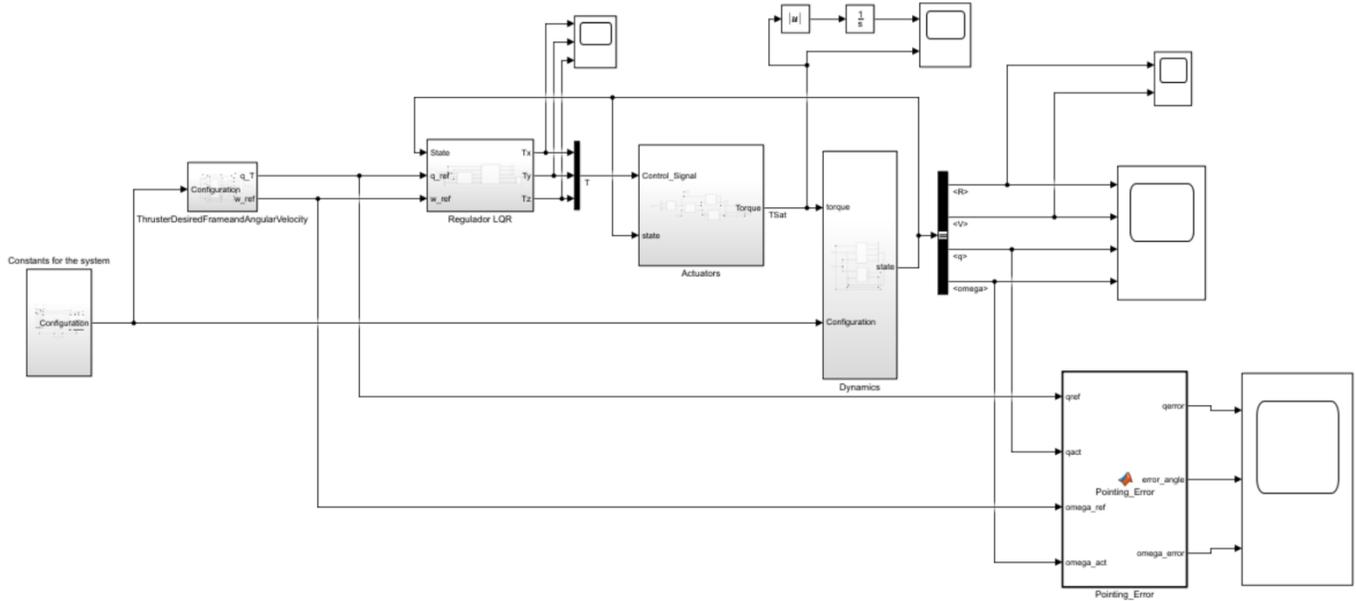


Figura A 8: Diagrama de bloques del sistema completo con el regulador LQR

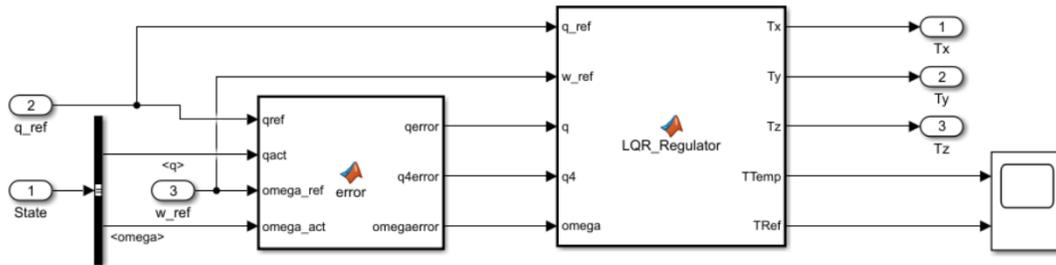


Figura A 9: Diagrama de bloques del subsistema del regulador LQR

A.8 Código del Regulador LQR

```
function [Tx,Ty,Tz] = LQR_Regulator(q, q4,omega)

% This function defines the output of the LQR controller given the K
matrix

%% Variable Definition
Tx = 0 ;
Ty = 0 ;
Tz = 0 ;

%% LQR Gain Matrix
K = [0.0001    0.0000    0.0000    0.0051    0.0000    0.0000;
      0.0000    0.0001    0.0000    0.0000    0.0051    0.0000;
      0.0000    0.0000    0.0001    0.0000    0.0000    0.0051];

%% Outputs
Tx = -1*sign(q4)*(q(1)*K(1,1) + q(2)*K(1,2) + q(3)*K(1,3)) -
1*(omega(1)*K(1,4) + omega(2)*K(1,5) + omega(3)*K(1,6)) ;

Ty = -1*sign(q4)*(q(1)*K(2,1) + q(2)*K(2,2) + q(3)*K(2,3)) -
1*(omega(1)*K(2,4) + omega(2)*K(2,5) + omega(3)*K(2,6)) ;

Tz = -1*sign(q4)*(q(1)*K(3,1) + q(2)*K(3,2) + q(3)*K(3,3)) -
1*(omega(1)*K(3,4) + omega(2)*K(3,5) + omega(3)*K(3,6)) ;

end
```

A.9 Código error de los cuaternios y velocidad angular

```
function [qerror,q4error,omegaerror] =  
error(qref,qact,omega_ref,omega_act)  
  
%Compute of the error between the actual quaternion and the reference  
%quaternion  
  
qmu=[qref(4) -qref(3) qref(2);  
      qref(3) qref(4) -qref(1);  
      -qref(2) qref(1) qref(4);  
      -qref(1) -qref(2) -qref(3)];  
  
qerror=transpose(qmu)*qact; %error of q1,q2 and q3  
q4error=transpose(qact)*qref; %error of q4  
  
% omegaerror=omega_act;  
omegaerror=omega_act-omega_ref; %error of angular velocity  
  
end
```

A.10 Código para la determinación de la matriz de ganancias

```

% Gain Matrix Determination

% Code dedicated to the determination of the gain matrix K for an LQR
% controller for the nanosatellite GWSat, given a certain Q and R matrix

%% Constants Definition
format short ;
mass1=4; %Mass of spacecraft 1 (kg)
a=0.1;
b=0.1;
c=0.3408;
Jx=(mass1/12)*(a^2+c^2); %Rotational Inertia in x axis
Jy=(mass1/12)*(b^2+c^2); %Rotational Inertia in y axis
Jz=(mass1/12)*(a^2+b^2); %Rotational Inertia in z axis
omega0 = 1.1288e-3 ; %ISS Orbit angular velocity
f41 = 8*omega0^(2)*(Jz-Jy)/Jx ;
f46 = omega0*(-Jx-Jz+Jy)/Jx ;
f52 = 6*omega0^(2)*(Jz-Jx)/Jy ;
f63 = 2*omega0^(2)*(Jx-Jy)/Jz ;
f64 = omega0*(Jx+Jz-Jy)/Jz ;

%% System Model

A = [0 0 0 0.5 0 0;
     0 0 0 0 0.5 0;
     0 0 0 0 0 0.5;
     f41 0 0 0 0 f46;
     0 f52 0 0 0 0;
     0 0 f63 f64 0 0];

B = [0 0 0;
     0 0 0;
     0 0 0;
     1/Jx 0 0;
     0 1/Jy 0;
     0 0 1/Jz];

C = [1 0 0 0 0 0;
     0 1 0 0 0 0;
     0 0 1 0 0 0;
     0 0 0 1 0 0;
     0 0 0 0 1 0;
     0 0 0 0 0 1] ;

```

```

D = [0 0 0;
     0 0 0;
     0 0 0;
     0 0 0;
     0 0 0;
     0 0 0] ;
satellite = ss(A,B,C,D) ;
figure
step(satellite)                                %Show step response without any control

%% LQR Gain Matrix Determination
Q = eye(6) ;
Q = [1e-9    0        0        0        0        0;
     0    1e-9    0        0        0        0;
     0    0    1e-9    0        0        0;
     0    0    0    1e-6    0        0;
     0    0    0    0    1e-6    0;
     0    0    0    0    0    1e-6];
R = [0.05 0 0;
     0 0.05 0;
     0 0 0.05];
K = zeros(3,6);
K = lqr(A,B,Q,R);
satellite2 = ss(A-B*K,B,C,D);
figure
step(satellite2)                                %Show step response given the LQR controller

```

A.11 Diagramas de bloques del sistema del controlador inteligente

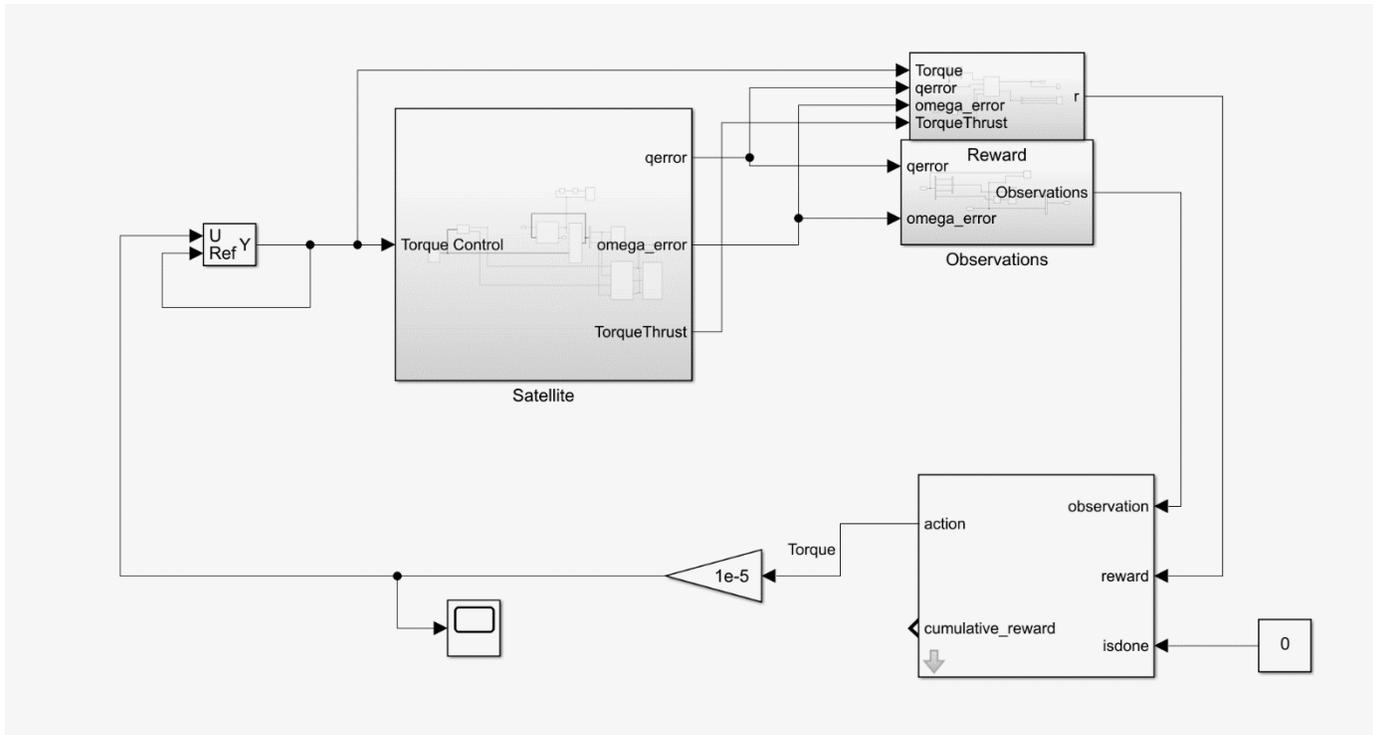


Figura A 10: Diagrama de bloques del sistema completo con el regulador TD3

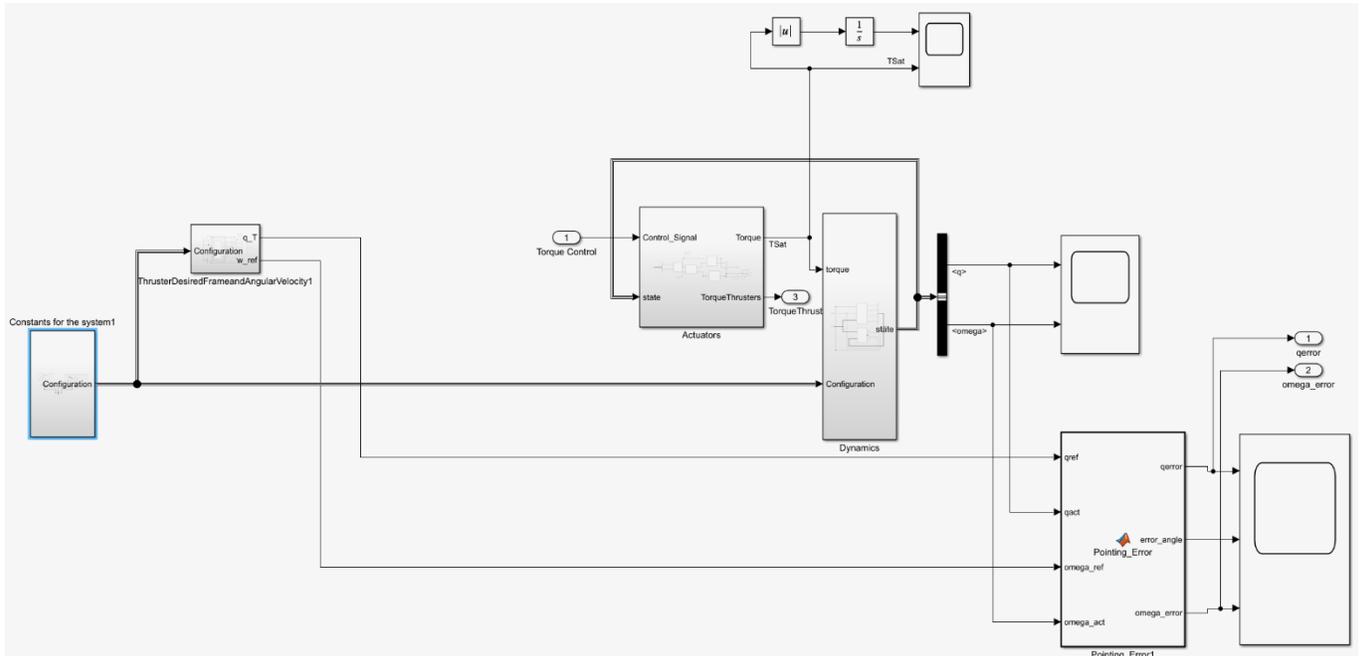


Figura A 11: Diagrama de bloques del subsistema del satélite

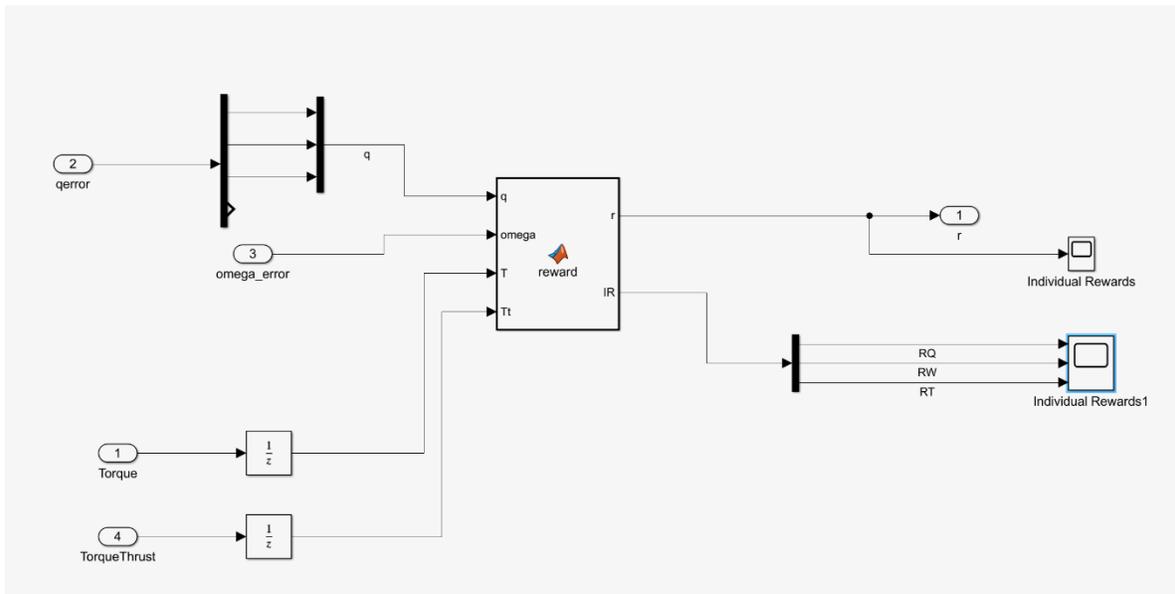


Figura A 12: Diagrama de bloques del subsistema de la recompensa

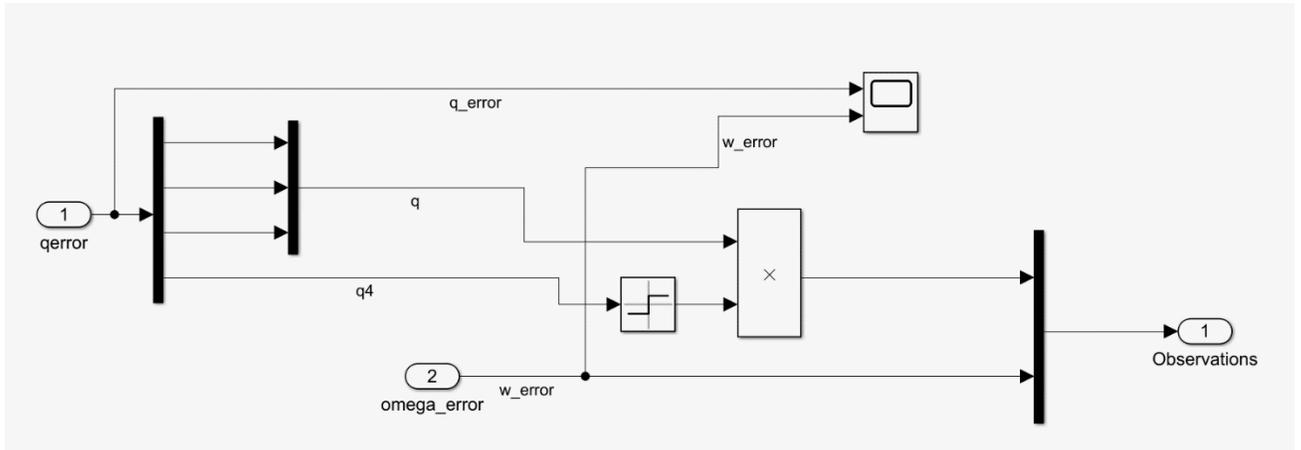


Figura A 13: Diagrama de bloques del subsistema de las observaciones

A.12 Código para el entrenamiento de la red neuronal por medio del algoritmo TD3

```
% TD3 Agent Training
% Author: Gabriel Alba

% This code is used to train a neural network with the TD3 algorithm. You
% can set up options such as the different parameters for training, the
% network architecture as well as the agent parameters. Also, there are
% options for speeding up the process with gpu use and parallelization

%% SET UP ENVIRONMENT
% Speedup options
useFastRestart = true;
useGPU = false;
useParallel = false;
% Create the observation info
numObs = 6;
observationInfo = rlNumericSpec([numObs 1]);
observationInfo.Name = 'observations';
% create the action info
numAct = 3;
actionInfo = rlNumericSpec([numAct 1], 'LowerLimit', -1, 'UpperLimit', 1);
actionInfo.Name = 'torque';
% Environment
mdl = 'RLModeloCompleto';
load_system(mdl);
blk = [mdl, '/RL Agent'];
env = rlSimulinkEnv(mdl, blk, observationInfo, actionInfo);
env.ResetFcn = @(in) localResetFcn(in);
if ~useFastRestart
    env.UseFastRestart = 'off';
end
rng('shuffle')
Ts = 0.25;
Tf = 1000;

%% TD3 CRITICS
criticLayerSizes = [300 150 75];
statePath = [
    imageInputLayer([numObs 1 1], 'Normalization', 'none', 'Name',
'observation')
    fullyConnectedLayer(criticLayerSizes(1), 'Name', 'CriticStateFC1');
    reluLayer('Name', 'CriticStateRelu1')
    fullyConnectedLayer(criticLayerSizes(2), 'Name', 'CriticStateFC2');
    reluLayer('Name', 'CriticStateRelu2')
    fullyConnectedLayer(criticLayerSizes(3), 'Name', 'CriticStateFC3')];
actionPath = [
    imageInputLayer([numAct 1 1], 'Normalization', 'none', 'Name', 'action')
    fullyConnectedLayer(criticLayerSizes(3), 'Name', 'CriticActionFC1')];
commonPath = [
    additionLayer(2, 'Name', 'add')
    reluLayer('Name', 'CriticCommonRelu1')
    fullyConnectedLayer(1, 'Name', 'CriticOutput')];
```

```

% Connect the layer graph
criticNetwork = layerGraph(statePath);
criticNetwork = addLayers(criticNetwork, actionPath);
criticNetwork = addLayers(criticNetwork, commonPath);
criticNetwork = connectLayers(criticNetwork, 'CriticStateFC3', 'add/in1');
criticNetwork = connectLayers(criticNetwork, 'CriticActionFC1', 'add/in2');
% Create critic representation
criticOptions =
rlRepresentationOptions('Optimizer','adam','LearnRate',1e-3, ...
'GradientThreshold',1,'L2RegularizationFactor',2e-4);
if useGPU
    criticOptions.UseDevice = 'gpu';
end
critic1 =
rlQValueRepresentation(criticNetwork,env.getObservationInfo,env.getActionInfo,...
    'Observation',{'observation'},...
    'Action',{'action'},criticOptions);
critic2 =
rlQValueRepresentation(criticNetwork,env.getObservationInfo,env.getActionInfo,...
    'Observation',{'observation'},...
    'Action',{'action'},criticOptions);

%% TD3 ACTOR
% Create the actor network layers
actorLayerSizes = [300 150 75];
actorNetwork = [
    imageInputLayer([numObs 1
1], 'Normalization', 'none', 'Name', 'observation');
    fullyConnectedLayer(actorLayerSizes(1), 'Name', 'ActorFC1');
    reluLayer('Name', 'ActorRelu1');
    fullyConnectedLayer(actorLayerSizes(2), 'Name', 'ActorFC2');
    reluLayer('Name', 'ActorRelu2');
    fullyConnectedLayer(actorLayerSizes(3), 'Name', 'ActorFC3');
    reluLayer('Name', 'ActorRelu3');
    fullyConnectedLayer(numAct, 'Name', 'ActorFC4');
    tanhLayer('Name', 'action') ;
];
% Create actor representation
actorOptions = rlRepresentationOptions('Optimizer','adam','LearnRate',1e-4, ...
'GradientThreshold',1,'L2RegularizationFactor',1e-5);
if useGPU
    actorOptions.UseDevice = 'gpu';
end
actor =
rlDeterministicActorRepresentation(actorNetwork,env.getObservationInfo,...
    env.getActionInfo, ...
    'Observation',{'observation'}, 'Action', {'action'},...
    actorOptions);

```

```

%% TD3 Agent Options
agentOptions = rlTD3AgentOptions;
agentOptions.SampleTime = Ts;
agentOptions.DiscountFactor = 0.9995;
agentOptions.MiniBatchSize = 256;
agentOptions.ExperienceBufferLength = 10e4;
agentOptions.TargetSmoothFactor = 1e-3;
agentOptions.ExplorationModel.Variance = 0.85 ;
agentOptions.ExplorationModel.VarianceDecayRate = 5e-7 ;
agentOptions.TargetPolicySmoothModel.Variance = 0.85 ;
agentOptions.TargetPolicySmoothModel.VarianceDecayRate = 5e-7 ;
agentOptions.PolicyUpdateFrequency = 2 ;
agentOptions.TargetUpdateFrequency = 2 ;
agentOptions.ResetExperienceBufferBeforeTraining = true;
agentOptions.SaveExperienceBufferWithAgent = true ;
agentOptions.NumStepsToLookAhead = 1 ;
agent = rlTD3Agent(actor,[critic1 critic2],agentOptions);

%% Training Options
trainingOptions = rlTrainingOptions;
trainingOptions.MaxEpisodes = 50000;
trainingOptions.MaxStepsPerEpisode = Tf/Ts;
trainingOptions.ScoreAveragingWindowLength = 125;
trainingOptions.StopTrainingCriteria = 'AverageReward';
trainingOptions.StopTrainingValue = 50000000;
trainingOptions.SaveAgentCriteria = 'EpisodeReward';
trainingOptions.SaveAgentValue = 50000;
trainingOptions.Plots = 'training-progress';
trainingOptions.Verbose = true;
if useParallel
    trainingOptions.Parallelization = 'async';
    trainingOptions.ParallelizationOptions.StepsUntilDataIsSent = 32;
end

% Train the agent
trainingStats = train(agent,env,trainingOptions);

% %% SAVE AGENT
%reset(agent); % Clears the experience buffer
curDir = pwd;
saveDir = 'C:\Users\Gabriel\Documents\MATLAB\Reinforcement Learning';
cd(saveDir)
save(['trainedAgent_' datestr(now,'mm_DD_YYYY_HHMM')], 'agent');
save(['trainingResults_'
datestr(now,'mm_DD_YYYY_HHMM')], 'trainingStats');
cd(curDir)

```

A.13 Código para la función de reset aleatorio del ambiente

```
function in = localResetFcn(in)
%This function resets the environment in a randomized way for training the
%neural network for the nanosatellite GWSat

    rng('shuffle');
    in=in.setVariable('q0', [rand; rand; rand; 1]);
    in=in.setVariable('omega0', [0.01*rand; 0.01*rand; 0.01*rand]);

end
```

Anexos

A.1 Teorema del Transporte

El teorema del transporte se puede describir de la siguiente manera. Dados dos marcos de referencia A y B tales que la velocidad angular del marco de referencia B visto por un observador en el marco de referencia A es ${}^A\omega^B$, la tasa de cambio del vector \mathbf{b} visto por un observador en el marco de referencia A es igual a la suma de la tasa de cambio de \mathbf{b} visto por un observador en el marco de referencia B y el producto cruz de ${}^A\omega^B$ con el vector \mathbf{b} . Esto se puede ver matemáticamente de la siguiente manera. [44]

$$\frac{{}^A d\mathbf{b}}{dt} = \frac{{}^B d\mathbf{b}}{dt} + {}^A\omega^B \times \mathbf{b}$$

Es importante mencionar que \mathbf{b} puede ser cualquier vector y es conocido en términos de un marco de referencia B que está rotando relativo a otro marco de referencia A. Consecuentemente, la tasa de cambio de \mathbf{b} visto por un observador en el marco de referencia A debe tomar en cuenta la rotación del marco de referencia B relativo al marco de referencia A. [44]