Universidad de Cádiz
Escuela Superior de Ingeniería

TRABAJO DE FIN DE GRADO

GRADO EN INGENIERÍA AEROESPACIAL

ESTUDIO TEÓRICO DEL APRENDIZAJE POR REFUERZO Y SU APLICACIÓN PRÁCTICA AL POSICIONAMIENTO DE UN DRON CUADRICÓPTERO

AUTORA: CLAUDIA ALDANA BILBAO

Puerto Real, febrero 2022

Universidad de Cádiz
Escuela Superior de Ingeniería

TRABAJO DE FIN DE GRADO

GRADO EN INGENIERÍA AEROESPACIAL

ESTUDIO TEÓRICO DEL APRENDIZAJE POR REFUERZO Y SU APLICACIÓN PRÁCTICA AL POSICIONAMIENTO DE UN DRON CUADRICÓPTERO

DIRECTOR: LUIS ANTONIO MARISCAL RICO
CODIRECTOR: PABLO LÓPEZ OSORIO
AUTORA: CLAUDIA ALDANA BILBAO

Puerto Real, febrero 2022

Resumen

El propósito de este proyecto es la aplicación de técnicas de Aprendizaje por Refuerzo a un problema robótico. En particular, al control de la posición de un dron cuadricóptero.

Por un lado, estas técnicas forman parte de la Inteligencia Artificial. Por ello, lo primero será conocer un poco este campo, concretamente de la rama del Aprendizaje Automático, y hacer hincapié en el Aprendizaje por Refuerzo. Este consiste en aprender de la interacción con el entorno, es decir, de la experiencia, gracias a la obtención de una serie de recompensas en función de las acciones que se lleven a cabo y las observaciones que se reciban.

Por otro lado, se hará una introducción a los drones y se indagará más en el dron cuadricóptero, explicando sus movimientos y las ecuaciones físicas que lo gobiernan.

Finalmente, se pondrán en conjunto ambos bloques de conocimiento para realizar una simulación en el software Matlab-Simulink del control de la altitud de un cuadricóptero y evaluar sus resultados.

Palabras clave: Aprendizaje por Refuerzo, Cuadricóptero, Matlab-Simulink, DDPG

Abstract

The purpose of this project is the application of Reinforcement Learning techniques to a robotic problem. Specifically, to the position control of a quadcopter drone.

On the one hand, these techniques are part of the Artificial Intelligence. That is why the first thing to do is to know a little about this field, explicitly about the branch of Machine Learning, emphasizing in Reinforcement Learning. This consists of learning from the interaction with the environment, that is, from experience, thanks to obtaining rewards based on the actions performed and the observations received.

On the other hand, an introduction to drones is made and the quadcopter drone is studied further, explaining its movements and the physical equations that govern it.

Finally, both blocks of knowledge are put together to perform a simulation in the software Matlab-Simulink to control the altitude of a quadcopter and evaluate its results.

Key words: Reinforcement Learning, Quadcopter, Matlab-Simulink, DDPG

Índice de contenidos

Índice de figuras

Índice de tablas

Simbología y Nomenclatura

Capítulo 3

$\mathcal{A}$	Conjunto de acciones disponibles
$\mathcal{A}(s)$	Conjunto de acciones disponibles para un estado concreto
A_i	Cada una de las acciones de $\mathcal{A}$, donde $i = 1, 2, 3, ..., K$
$a.\ a'$	Una acción y la del estado siguiente
a_t	Acción llevada a cabo en el instante t
$\mathcal{D}$	Conjunto de experiencias del buffer
$J(\theta)$	Rendimiento de la política
$\mathcal{N}$	Generador de ruido
N	Número de transiciones del *minibatch*
$\mathcal{P}$	Matriz de probabilidad de transición de estados
$\mathcal{P}_{ss'}$	Probabilidad de pasar de un estado al siguiente
$\mathcal{P}^a_{ss'}$	Probabilidad de pasar de un estado al siguiente por medio de una acción
$Q^\pi(s, a)$	Valor de tomar una acción en un estado bajo una política cualquiera
$Q^*(s, a)$	Valor de tomar una acción en un estado bajo una política óptima
$\mathcal{R}_s$	Recompensa esperada por estar en un estado
$\mathcal{R}_{ss'}$	Recompensa esperada por pasar de un estado al siguiente
$\mathcal{R}^a_{ss'}$	Recompensa esperada por pasar de un estado al siguiente por medio de una acción
R_t	Recompensa acumulada a partir del instante t
r	Una recompensa
r_t	Recompensa esperada en el instante t
$\mathcal{S}$	Conjunto de estados disponibles
S_i	Cada uno de los estados de $\mathcal{S}$, donde $i = 0, 1, 2, ..., N$
$s,\ s'$	Un estado y el que le sigue

s_t	Estado correspondiente al instante t
T	Dinámica de transición
t	Un instante de tiempo
$V^{\pi}(s)$	Valor de un estado bajo una política cualquiera
$V^{*}(s)$	Valor de un estado bajo una política óptima
α	Tasa de aprendizaje, $0 \leq \alpha \leq 1$
ϕ, ϕ'	Parámetros del crítico principal y objetivo
γ	Factor de descuento, $0 \leq \gamma \leq 1$
μ	Política determinista
π	Una política
π^{*}	Una política óptima
$\pi(s)$	Acción tomada en un estado bajo una política determinista
$\pi(s, a)$	Probabilidad de tomar una acción en un estado bajo una política estocástica
θ, θ'	Parámetros del actor principal y objetivo
τ	Parámetro de actualización, $0 \leq \tau \leq 1$
∇	Gradiente

Capítulo 5

$+$	Configuración en cruz del cuadricóptero
$\times$	Configuración en equis del cuadricóptero
d_i	Par de arrastre ejercido por cada rotor, $i = 1, 2, 3, 4$
F_G	Fuerza ejercida por la gravedad
F_i	Fuerza ejercida por cada rotor, $i = 1, 2, 3, 4$
F_T	Fuerza de empuje total
F_x	Fuerza en el eje x
F_y	Fuerza en el eje y
F_z	Fuerza en el eje z
g	Aceleración de la gravedad
h_c	Momento cinético
I	Matriz de inercia
I_{xx}	Inercia en el eje x
I_{yy}	Inercia en el eje y
I_{zz}	Inercia en el eje z
k_d	Constante de arrastre

k_t	Constante de empuje
l	Distancia desde cada rotor al centro de masas del cuadricóptero
m	Masa
M_d	Par de arrastre total
M_x	Momento alrededor del eje x
M_y	Momento alrededor del eje y
M_z	Momento alrededor del eje z
p_m	Momento lineal
R_{ab}	Matriz de cambio de ejes de un sistema a a otro b
T_{ab}	Matriz de cambio de ejes de la variación angular de un sistema a a otro b
T_i	Fuerza de empuje ejercida por cada rotor, $i = 1, 2, 3, 4$
X_B	Magnitud referida al sistema de ejes cuerpo
X_E	Magnitud referida al sistema de ejes tierra
X_H	Magnitud referida al sistema de ejes horizonte local
δ_i	Ángulo de rotación de Euler, $i = 1, 2, 3$
Ω_i	Velocidad de giro de cada rotor, $i = 1, 2, 3, 4$
Ω_H	Velocidad de giro de cada rotor en estado de *hovering*
Ω_H^*	Velocidad de giro de cada rotor si la inclinación es muy pequeña, $\approx \Omega_H$

---

p	Velocidad angular alrededor del eje x_B
q	Velocidad angular alrededor del eje y_B
r	Velocidad angular alrededor del eje z_B
$\dot{p}$	Aceleración angular alrededor del eje x_B
$\dot{q}$	Aceleración angular alrededor del eje y_B
$\dot{r}$	Aceleración angular alrededor del eje z_B
u	Velocidad lineal en la dirección del eje x_B
v	Velocidad lineal en la dirección del eje y_B
w	Velocidad lineal en la dirección del eje z_B
x	Posición sobre el eje x_E
y	Posición sobre el eje y_E
z	Posición sobre el eje z_E
$\ddot{x}$	Aceleración lineal en la dirección del eje x_E
$\ddot{y}$	Aceleración lineal en la dirección del eje y_E
$\ddot{z}$	Aceleración lineal en la dirección del eje z_E
ϕ	Ángulo de rotación alrededor el eje x_E
θ	Ángulo de rotación alrededor el eje y_E
ψ	Ángulo de rotación alrededor el eje z_E

$\dot{\phi}$	Velocidad de rotación alrededor del eje x_E
$\dot{\theta}$	Velocidad de rotación alrededor del eje y_E
$\dot{\psi}$	Velocidad de rotación alrededor del eje z_E
$\vec{\xi}$	Vector de posición en ejes tierra
$\dot{\xi}$	Vector de velocidad lineal en ejes tierra
$\ddot{\xi}$	Vector de aceleración lineal en ejes tierra
$\vec{\eta}$	Vector de orientación en ejes tierra
$\vec{v}$	Vector de velocidad lineal en ejes cuerpo
$\vec{\omega}$	Vector de velocidad angular en ejes cuerpo

Siglas

AGL	Above Ground Level
BLOS	Beyond Line Of Sight
CR	Close Range
DDPG	Deep Deterministic Policy Gradient
DEC	Decoys
DL	Deep Learning
DP	Dynamic Programming
DRL	Deep Reinforcement Learning
DT	Temporal-Difference
EXO	Exo-stratospheric
GPS	Global Positioning System
HALE	High Altitude Long Endurance
HTOL	Horizontal Take-Off and Landing
IA	Inteligencia Artificial
LADP	Low Altitude Deep Penetration
LALE	Low Altitude Long Endurance
LET	Lethal
LOS	Line Of Sight
MALE	Medium Altitude Long Endurance
MC	Monte Carlo
MDP	Markov Decision Process
ML	Machine Learning
MR	Medium Range
MRE	Medium Range Endurance
MTOW	Maximum Take-Off Weight
OTAN	Organización del Tratado del Atlántico Norte

DQN	Deep Q-Network
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
RPA	Remotely Piloted Aircraft
RPAS	Remotely Piloted Aircraft System
SR	Short Range
Strato	Stratospheric
UAS	Unmanned Aerial System
UAV	Unmanned Aerial Vehicle
UCAV	Unmanned Combat Aerial Vehicle
VANT	Vehículo Aéreo No Tripulado
VTOL	Vertical Take-Off and Landing

1 Introducción

1.1. Motivación y Objetivos

La idea de este proyecto surge de la necesidad de encontrar una alternativa a las técnicas de posicionamiento de un dron respecto de un punto determinado situado en un barco. Ya que, tanto el Sistema de Posicionamiento Global (GPS) —del inglés *Global Positioning System*— como el sistema por comparación de imágenes podrían presentar carencias en alta mar. El GPS puede sufrir problemas de cobertura, mientras que el método visual no es útil en condiciones adversas como niebla o oleaje.

Por ello, se ha pensado en aplicar técnicas de Inteligencia Artificial (IA), con las que se pretende que el dron aprenda a llegar a un punto del que se conocen sus coordenadas y mantenerse sobre él (Figura 1.1).

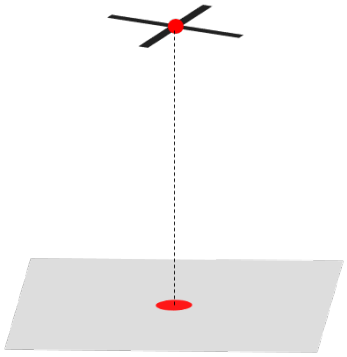


Figura 1.1: Esquema del posicionamiento del dron sobre un punto del suelo.

En este trabajo se persiguen dos objetivos fundamentales:

- Obtener una visión general de la IA y, más concretamente, del Aprendizaje por Refuerzo (RL) —del inglés *Reinforcement Learning*— y su aplicación al control robótico.
- Aplicar esta técnica a un dron cuadricóptero para controlar su posición.

1.2. Alcance

Aquí se marcan las pautas que limitan el alcance de este trabajo. En primera instancia, se hace un repaso teórico general de la IA presentando los distintos tipos hasta llegar al RL, tema de interés, en el que se profundiza más. Después, se pretende implementar la tecnología presentada previamente con los softwares matemáticos Matlab-Simulink de forma que se muestre su funcionamiento mediante una simulación limitada al movimiento en el eje z.

1.3. Estructura del trabajo

En este apartado se va a hacer un recorrido por los distintos capítulos de este documento comentando brevemente de qué tratan.

En el presente capítulo, el Capítulo 1, se ha hecho una introducción al cometido de este Trabajo de Fin de Grado. En las secciones previas se han expuesto los motivos que han llevado a la realización del proyecto y los objetivos que se quieren alcanzar con él; y se han definido los límites hasta los que se va a abarcar.

El Capítulo 2 comienza con un repaso por la evolución histórica de la IA y se comentan aspectos sobre el impacto que está teniendo en la actualidad, respaldándose en la mención de algunas aplicaciones. Continúa con definiciones importantes, partiendo de la Informática, pasando por la IA y el Aprendizaje Automático (ML) —del inglés *Machine Learning*— y estrechando el círculo hasta llegar al tema objetivo, el RL. Finaliza introduciendo brevemente las redes neuronales.

En el Capítulo 3 se profundiza en el RL, presentando una serie de conceptos básicos, describiendo el problema y métodos de solución.

En el Capítulo 4 se habla de los drones, comentando su nomenclatura y definición, distintas clasificaciones de los tipos de drones que se pueden encontrar actualmente y algunos campos de aplicación.

El Capítulo 5 está dedicado al dron de tipo cuadricóptero. Se introduce explicando su funcionamiento, se presentan los dos tipos de configuraciones que pueden presentar, se exponen sus movimientos y se desarrollan sus ecuaciones.

El Capítulo 6, y último, se dedica al proceso de realización de la simulación, con los pasos para su implementación y la exposición de los resultados obtenidos.

Antecedentes y desarrollo actual de la IA

2

2.1. Evolución histórica y situación actual

El tema de los dispositivos inteligentes es algo muy actual que parece que nació ayer, pero esto no es así. Oficialmente, no llega al siglo de edad, aunque el ser humano, ya desde épocas antiguas, se ha planteado crear artefactos a su imagen y semejanza que automaticen procesos propios de él. A continuación, se va a ver un breve recorrido histórico de los antecedentes hasta su nacimiento, muy ligados al desarrollo de la Informática, y los avances que se han sucedido desde entonces hasta la actualidad en un período no exento de altibajos. [1][2][3]

Para ver los antecedentes de la IA, se retrocede hasta la época de la Antigua Grecia, donde aparecieron los primeros proyectos de autómatas mecánicos y otros diseños de máquinas mecánicas. También en ese tiempo, Aristóteles formuló un sistema de mecanización del razonamiento lógico que permitía sacar conclusiones dadas unas premisas iniciales. Más tarde, ya en el siglo XIV, Ramón Llull ideó una máquina que combinaba afirmaciones mediante operaciones lógicas y así poder producir todo el conocimiento posible. Esto influyó notablemente en Leibniz quien, en el siglo XVII, enunció la existencia de una máquina y un lenguaje universales para automatizar tanto el razonamiento matemático como el humano, de forma que mejoró los diseños de calculadoras anteriores tratando de buscar esta máquina. Poco después, Jacquard construyó un telar que reproducía los patrones codificados en tarjetas perforadas. En el siglo XIX, cabe destacar el diseño de la *Máquina Analítica* de Babbage en colaboración con Lady Lovelace, considerada el primer ordenador programable, que pretendía ser una máquina capaz de actuar de forma distinta según el problema que se le planteara; así como los desarrollos en torno al lenguaje lógico de los matemáticos Boole y Frege que se utiliza actualmente en las ciencias formales.

Con la llegada del siglo XX, se produjeron diversos resultados teóricos que asentaron las bases de las Ciencias de la Computación. Y, a mediados de siglo, tuvieron lugar una serie de publicaciones, de las que se destacan algunas: McCulloch y Pitts (1943) establecían las bases de las redes neuronales artificiales; Turing (1950) proponía una evaluación, llamada *Test de Turing*, para determinar si una máquina

se puede considerar inteligente o no; Shannon (1950) describía un jugador de ajedrez automático que buscara la respuesta más apropiada considerando las distintas respuestas posibles; Samuel (1952) desarrollaba un jugador de damas automático basado en el modelo de Shannon, pero con la peculiaridad de que aprendía los movimientos más adecuados a medida que iba jugando contra humanos; y Newell, Shaw y Simon (1955) presentaban un programa que simulaba el razonamiento seguido por un matemático para demostrar teoremas. Este panorama lleno de logros dio lugar a la Conferencia de Dartmouth de 1956, un encuentro de verano con los nombres más reconocidos del campo. En este evento se bautizó la materia con el nombre de *Inteligencia Artificial* (IA), se definieron sus bases y líneas de trabajo a seguir y se hizo una predicción de avances en 10 años.

De ahí se salió con un optimismo excesivo que duraría sólo unos pocos años, hasta los años 70, cuando se comenzó a ver que se habían generado unas expectativas demasiado altas, ya que los resultados previstos no se alcanzaban, desembocando en una crisis basada en críticas y en la pérdida de interés de los inversores, que retiraron la financiación. Es lo que se conoce como invierno de la IA.

Rosenblatt (1958) diseñó el perceptrón, una forma de red neuronal con el fin de explicar las habilidades humanas para el reconocimiento de patrones y a la que él le auguró grandes logros como la toma de decisiones o la traducción de idiomas. Sin embargo, Minsky (1968) publicó un artículo de crítica demoledor para esta parte del campo que hizo que se olvidara por muchos años.

También, en esos años de auge, se inventaron varios lenguajes de programación de alto nivel y tuvieron lugar algunos desarrollos donde la comunicación en lenguaje natural tomaba protagonismo. Por ejemplo, *STUDENT* (1964) era un programa capaz de entender y resolver problemas de álgebra de nivel de instituto dados en lenguaje natural; *ELIZA* (1966) fue el primer chatbot, es decir, un programa interactivo capaz de mantener una conversación en inglés, aunque no sabía de qué hablaba y se limitaba a dar respuestas triviales y reformular las preguntas con estructuras de respuesta; y *WABOT-1* (1972) fue el primer robot humanoide que podía trasladarse, manipular objetos y comunicarse en japonés.

Después de varios años de un período de latencia, con la llegada de los años 80 hubo unos años de crecimiento gracias a los sistemas expertos que, aunque ya habían habido algunos antes, aquí tuvieron su período de explosión. Sin embargo, al poco tiempo se volvió a estancar la evolución debido a que las máquinas especializadas en IA tenían una mala relación resultados-coste.

Los sistemas expertos consistían en programas capaces de resolver problemas específicos y tomar decisiones en base a reglas lógicas derivadas del conocimiento como si fueran expertos en la materia. Por mencionar algunos de los primeros, que estuvieron en uso durante años y que sirvieron de base para los siguientes: *DENDRAL* (1965) para identificar la estructura de compuestos químicos complejos; *MACSYMA* (1968) para la resolución de ecuaciones complejas; *MYCIN* (1972) diagnóstico de enfermedades infecciosas de la sangre.

También en este período de bonanza, se construyó el primer vehículo autónomo que funcionó con éxito, el *Stanford Cart* (1980). Y las redes neuronales volvieron a tenerse en cuenta gracias a Hopfield (1982) que tomó un nuevo enfoque, y a Hinton y

Rumelhart que popularizaron un método de entrenamiento basado en la transmisión de errores hacia atrás, *backpropagation*, previamente desarrollado por Werbos (1974).

Ya entrados los años 90 surgió el concepto de agente inteligente, sistemas que perciben su entorno y actúan de forma que sus probabilidades de éxito sean las más altas posibles. A partir de ahí se volvió a repuntar en un crecimiento que aún perdura en la actualidad gracias a los avances, sobre todo, en la rama de ML y los algoritmos que se combinan con redes neuronales.

Algunos hitos importantes de esos últimos años son: el superordenador *DeepBlue* (1997) de IBM derrotó del campeón de ajedrez Gary Kasparov; el robot limpiador *Roomba* (2002) fue el primer robot doméstico comercializado; el sistema de IA *Watson* (2011) de IBM ganó en el concurso estadounidense de preguntas y respuestas *Jeopardy!* de forma que era capaz de entender las preguntas en lenguaje natural; aparecen las aplicaciones asistentes de voz; un chatbot que se hacía llamar *Eugene Goostman* (2014) logró convencer a más del 30 % de los jueces de que era humano; el programa de IA *AlphaGo* (2017) de Google *DeepMind* venció al campeón del mundo del juego de mesa chino Go.

Como se ha visto en los párrafos anteriores, en los últimos tiempos se ha venido notando, cada vez más, la presencia de la tecnología de la IA en la sociedad, aunque no sea palpable para la mayoría de las personas. Pero lo cierto es que la IA es la base en muchas de las actividades de la vida diaria, como son los sistemas de recomendaciones personalizadas de las plataformas en línea tan populares actualmente. Y es que uno de los temas recurrentes en las películas de ciencia ficción está tomando fuerza gracias a los nuevos desarrollos tecnológicos, los sistemas inteligentes, fruto de la competencia entre las grandes empresas por liderar el control de los datos.

Estos sistemas inteligentes combinan tres actores: software, hardware y datos. Un software novedoso, ya que no consta de un programa informático tradicional y que es capaz de aprender para reconocer patrones en los datos y encontrar la información de interés. Un hardware, por un lado, con un poder de procesamiento masivo y, por otro, con una amplia red de sensores. Y, finalmente, cantidades colosales de datos que procesar, obtenidos de dichos sensores, integrados en los dispositivos móviles y las redes inalámbricas que recogen cualquier movimiento que se lleve a cabo. [4]

Todo esto está dando lugar a cambios en los procesos industriales, como ya ocurriera en los siglos pasados en varias ocasiones, lo que indica que la época actual está a las puertas de una nueva Revolución Industrial, la Cuarta Revolución Industrial. Primero, se produjo la mecanización de los procesos industriales impulsados por la invención de la máquina de vapor; luego, se implementó el modelo de producción masiva en cadenas de montaje electrificadas; más tarde, se introdujo la automatización de los procesos por medio de sistemas programables; y, por último, en la actualidad está tomando fuerza el concepto de *Industria 4.0*, o lo que es lo mismo, *Industria Inteligente*. [5]

De esta forma, que el campo de la IA esté en auge es una realidad que no tiene discusión, las estadísticas lo avalan. Estadísticas como las recogidas en el Informe Anual [6] llevado a cabo por la Universidad de Stanford, en colaboración con la Universidad de Harvard y el MIT, así como con varias organizaciones sin ánimo de lucro, todos ellos estadounidenses. Aquí se muestra que tanto a nivel académico y

de investigación como a nivel económico y empresarial, los números en torno a la IA se han multiplicado año tras año.

También queda patente que el estallido de la IA ha dado lugar a algo que ha venido para quedarse, y no es una moda pasajera, en que ciertos organismos internacionales han visto la necesidad de crear una normativa que la regulen, ampliando las 3 leyes fundamentales de la robótica que, el visionario escritor, Isaac Asimov ya enunciara en una de sus obras a mediados del pasado siglo XX. El primer organismo que ha propuesto una normativa regularizadora de la IA es el Parlamento Europeo, quien ha definido un conjunto de 6 leyes con las que pretende reducir el impacto que tendrá en el empleo la convivencia con las máquinas inteligentes. [7][8]

2.1.1. Aplicaciones

Ya ha quedado claro que la IA está fuertemente instalada en la sociedad, implícita en gran cantidad de actividades y dispositivos, desde aspectos generales comunes al conjunto de la población hasta herramientas de apoyo para expertos de distintos campos. En lo que sigue se van a comentar algunos ejemplos recogidos en [8], [9], [10] y [11].

La gente de a pie tiene contacto con la IA en el momento en que utiliza un dispositivo con conexión a Internet, mayoritariamente. Esa frase muy común de oír, *parece que los dispositivos nos espían*, es verdad hasta cierto punto y tiene mucho que ver con las aplicaciones de la IA.

En la actualidad, en el mundo occidental, es prácticamente imposible encontrar una casa en la que no haya un dispositivo equipado con un asistente virtual, ya sea bajo el nombre de *Siri*, *Cortana*, *Google Assistant* o *Alexa*. Se fundamentan en el reconocimiento de voz y en el procesamiento del lenguaje natural para permitir la interacción usuario-máquina. El dispositivo tiene un micrófono que recoge la voz del usuario, que se interpreta para ejecutar las órdenes.

También, son muy familiares para los usuarios de las plataformas online, ya sea de compras o streaming, las recomendaciones personalizadas. Estas se hacen en función de lo que el usuario haya consumido o buscado, comparando esta actividad con la de otros usuarios con tendencias similares para, así, hacer sugerencias con posibilidades de éxito. De igual forma funcionan los anuncios publicitarios de la red. Además, siguiendo en este entorno, se tiene en cuenta la cantidad de clics que reciben los resultados de las búsquedas, de forma que el orden de aparición de los resultados sea el óptimo.

Otro aspecto son las redes sociales como *Twitter*, *Facebook* o *Instagram*, que utilizan la IA para evitar la publicación de contenido inapropiado, así como en las sugerencias de etiquetado de caras en las fotos y de posibles contactos. Lo mismo se aplica en el correo electrónico para clasificar un mensaje como spam.

Igualmente, merecen una mención los sistemas de domótica, los coches y robots autónomos y los videojuegos. En estos últimos se da inteligencia a los enemigos del protagonista para conseguir actuaciones más realistas y menos previsibles.

Por otra parte, están las aplicaciones a nivel profesional. En el ámbito de la seguridad, el reconocimiento de imágenes permite detectar la presencia de personas en cámaras de seguridad e, incluso, saber quienes son gracias a las técnicas de reconocimiento facial. En el terreno de las finanzas, se utilizan algoritmos de IA para detectar patrones con los que hacer predicciones de mercado y tomar decisiones en base a ellas, como, por ejemplo, estimar cuándo realizar movimientos en bolsa.

Por último, son muy significativos los avances en medicina. Ya existen robots especializados para realizar un diagnóstico primario, mediante el reconocimiento de patrones tras realizar unas preguntas al paciente. También, se han desarrollado herramientas para detectar el cáncer de mama con más antelación, así como enfermedades oculares relacionadas con la diabetes.

2.2. Informática

El ser humano tiende a minimizar el esfuerzo realizado en las distintas tareas de la vida, es por eso que, desde siempre, ha tratado de crear máquinas que lleven a cabo ciertas actividades de forma automática, reduciendo su esfuerzo y quedando relevado a tareas de supervisión. Como una de las bases de dicha automatización se tiene el conocimiento proporcionado por la Ingeniería Informática.

Según las definiciones proporcionadas por [12] y [13], la *Ingeniería Informática* es la ciencia que engloba los conocimientos, métodos y técnicas que permiten el tratamiento automático de información y datos en formato digital por medio de computadoras. Esto se hace implementando algoritmos en dichos dispositivos electrónicos de forma que se cumplan tres tareas básicas: la entrada de la información, su almacenamiento y procesamiento, y la salida o transmisión de los resultados.

La tecnología desarrollada por esta disciplina no sirve únicamente para autoalimentarse y crecer ella misma, sino que es una herramienta aprovechada y aplicada por infinidad de campos ajenos, de forma que les permite avanzar más rápido con gran variedad de propósitos. De hecho, hoy en día, es difícil pensar algún área que no esté influenciada por la informática y que no la use como herramienta de trabajo.

La Informática abarca un campo de conocimiento muy amplio dentro del cual se sitúa la IA, ámbito en el que se va a profundizar en este trabajo.

2.3. Inteligencia Artificial

El nombre científico que el ser humano se ha otorgado a sí mismo es el de *homo sapiens*, lo que significa *hombre sabio*, en honor a sus capacidades y habilidades mentales sobresalientes por encima del resto de seres vivos. El cerebro es el órgano más importante del cuerpo humano por darle este privilegio, ya que sin él no sería más que un organismo primitivo que no actuaría más allá de simples reflejos. Fascinado por el poder de su cerebro, desde hace muchos años, el hombre ha tratado de entender cómo se lleva a cabo la acción de pensar, tanto en él mismo como en otras formas

de inteligencia más sencillas como los animales. También, ha soñado con construir máquinas a su imagen y semejanza, capaces de realizar tareas de la misma forma en que las haría un humano. En respuesta a estos anhelos surgió el campo de la IA, cuya cuestión fundamental es *¿pueden las máquinas pensar y comportarse como los humanos?*, a la que trata de buscar una respuesta afirmativa [14]. [4][15]

Lo primero que se tiende a hacer ante un concepto nuevo es buscar su significado, y eso es lo que se va a tratar de hacer a continuación. Aunque para la IA no es algo tan sencillo. Por un lado, no se puede definir con exactitud qué es *inteligencia*, ya que hay distintos tipos; y, por otro, al ser un campo joven y experimental, los objetivos van cambiando a medida que la tecnología avanza. Por esto, no hay una definición única y aceptada por la comunidad, sino que a lo largo de la historia han ido surgiendo distintas definiciones con los distintos investigadores y según el enfoque que se ha tomado.

Para tratar de dar una explicación al término, en primer lugar, se va a ver qué es lo que hace al ser humano inteligente, ya que él es el pilar en torno al que crece la disciplina. Las características por las que el ser humano destaca sobre las demás especies son (Figura 2.1): capacidad de razonamiento, aprendizaje, percepción, resolución de problemas y toma de decisiones, e inteligencia lingüística. [14]

Figura 2.1: Cualidades propias del intelecto humano. [14]

Otra forma de intentar encontrar su significado, es buscar la definición de cada palabra por separado y, luego, formar una definición con la unión de ambas.

En la misma línea de lo anterior, [12] muestra cualidades que se le atribuyen al ser humano entre las acepciones de *inteligencia*. Por su parte, define *artificial* como aquello hecho por mano o arte del hombre. Y del concepto de *Inteligencia Artificial* dice que es la disciplina científica que se ocupa de crear programas informáticos que ejecutan operaciones comparables a las que realiza la mente humana.

Por otra parte, según un diccionario en línea, ambas palabras vienen del latín. Dice que *inteligencia*, en su significado más básico, hace referencia a quien sabe elegir la mejor opción o alternativa entre las distintas posibilidades para la resolución de un problema [16]. Mientras que *artificial* hace alusión a aquello fabricado por el hombre [17]. De aquí, se puede concluir, a grandes rasgos, que la IA es el estudio y desarrollo de sistemas creados por el ser humano con la capacidad para valorar qué hacer en distintas situaciones.

En consonancia con lo dicho, dos definiciones generales para IA dadas por dos de los pioneros del campo son:

> "La ciencia e ingeniería de crear máquinas inteligentes, especialmente programas informáticos inteligentes." [14] —John McCarthy, 1955
>
> "La ciencia de construir máquinas para que hagan cosas que, si las hicieran los humanos, requerirían inteligencia." [18] —Marvin Minsky, 1968

Hecho esto, es interesante indagar en las agrupaciones que se pueden hacer de los sistemas según algunas de sus características. De esta forma, los sistemas de IA se pueden clasificar de varias formas, en las que sólo los escalones más bajos existen en la actualidad.

Por un lado, aunque esta tecnología se puede apreciar en multitud de formas, algunas muy sorprendentes y alucinantes, se distinguen sólo tres niveles de complejidad: IA Débil o Estrecha, IA Fuerte o General y Súper Inteligencia, descritas en [19] y [20].

Por otro lado, Arend Hintze, profesor de la Universidad Estatal de Michigan, ha propuesto una clasificación de los desarrollos de IA en cuanto a su funcionalidad, basada en dos aspectos: la memoria y la conciencia. Su diferenciación consta de cuatro tipos de máquinas: reactivas, con memoria limitada, con una teoría de la mente y con autoconciencia, que se pueden encontrar en [19] y [21].

Otro aspecto curioso de conocer es saber las fuentes de conocimiento de las que se sirve esta disciplina. Aunque la IA nació a raíz de la Informática, como se ha mencionado, es una ciencia multidisciplinar, esto es, tiene influencias de distintos campos con más bagaje como son: la Filosofía, que engloba distintas teorías sobre el comportamiento humano; las Matemáticas, necesarias para el desarrollo e implementación de los algoritmos; la Economía, que aporta la experiencia de la toma de decisiones en un entorno probabilístico y de incertidumbre para maximizar el resultado; la Neurociencia, que estudia el cerebro, inspira muchos modelos de IA; y la Psicología, da las bases de los procesos mentales que se llevan a cabo en el conocimiento; entre otras aportaciones. [15]

Finalmente, hay que tener en cuenta que la IA es un campo con una materia de estudio muy amplia y que puede dividirse en ramas de conocimiento de muchas formas. Una manera común es agrupar el contenido de una forma parecida a las características que el Test de Turing dice que tiene que tener una máquina para superarlo y considerarse inteligente. Las ramas que se distinguen de esta forma se esquematizan en la Figura 2.2 y se describen en [22], [23] y [24].

Este trabajo se va a enfocar en la parte de ML, cuyo contenido se va a desarrollar de aquí en adelante.

2.4. Machine Learning

El *Aprendizaje Automático* o *Aprendizaje Máquina*, como se llama en castellano, es mucho más universal por su nombre en inglés, *Machine Learning* (ML), por ello, es la denominación que se va a utilizar a partir de ahora.

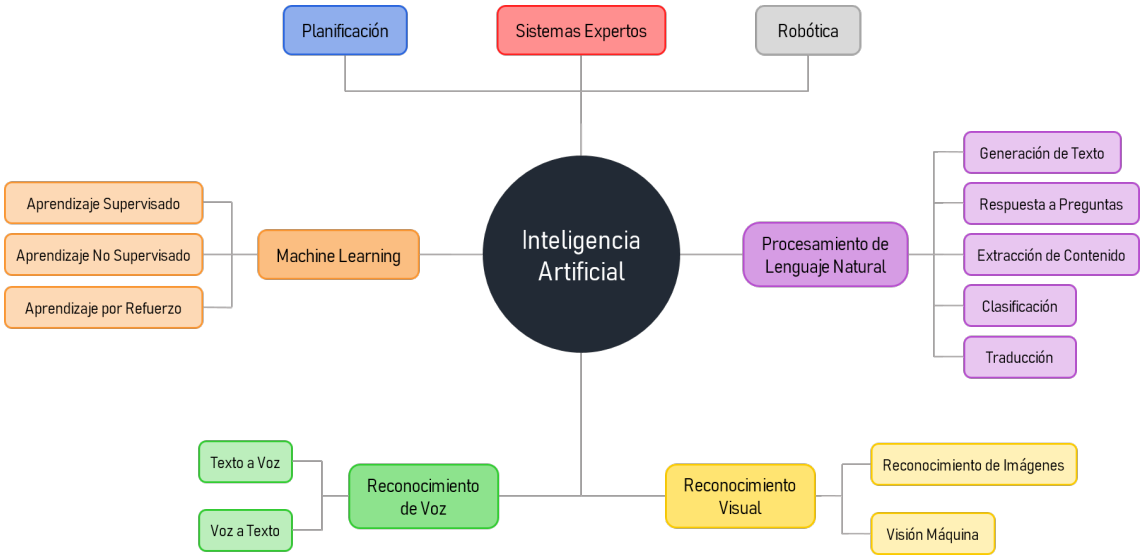


Figura 2.2: Esquema de las ramas de la IA.

Como se ha mencionado en lo anterior, el ML es una de las ramas que conforman la IA. Se puede decir que es el responsable de la popularidad en alza y el éxito que está teniendo la IA actualmente, gracias a los avances tan rápidos que viene cosechando desde hace unos años.

Lo primero, se va a tratar de definir el ML, de la misma forma que se hizo anteriormente con la IA. Mediante una búsqueda en el diccionario, una explicación de carácter más informal y algunas definiciones de expertos en la materia.

Desglosando el concepto en los dos términos que lo componen, se busca cada uno en [12]. Por un lado, *aprendizaje* es la adquisición de conocimientos y habilidades por medio del estudio, la enseñanza o la experiencia. Y, por otro lado, *automático* hace referencia a un mecanismo o aparato que funciona total o parcialmente por sí solo.

El ML se fija en el comportamiento humano a la hora de aprender. Se dice que una persona aprende cuando modifica su forma de relacionarse con el mundo que le rodea para obtener mejores consecuencias de sus actos, basando sus decisiones en conocimientos aprendidos y en experiencias propias o transmitidas por otros. Cuanta más experiencia se tiene, más sencillo es realizar predicciones, de ahí que los padres suelan decirle a los hijos cómo actuar, ya que ellos ya han pasado por situaciones similares. De la misma forma, una máquina aprende cuando el algoritmo por el que se rige altera la forma de procesar los datos para obtener resultados más convenientes, ajustando sus actuaciones a las distintas situaciones en las que se ve involucrada. Ahora bien, ante una situación desconocida, la máquina tiene dificultades para hacer la predicción, dando lugar a una probabilidad de éxito menor. [25][26]

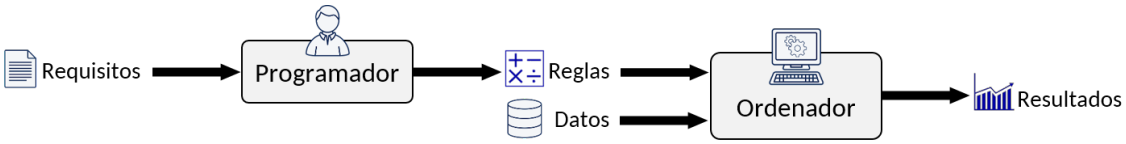
Todo esto que se ha comentado está recogido en algunas definiciones formales que han dado expertos en torno al ML: [27]

> "El campo de estudio que da a las máquinas la capacidad de aprender sin ser explícitamente programadas." —Arthur Samuel, 1959

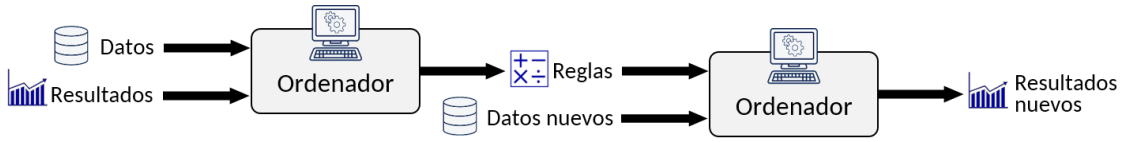
> "Se dice que un programa informático aprende de la experiencia E con respecto a alguna tarea T y alguna medida de rendimiento P, si su actuación en T, medida por P, mejora con la experiencia E." —Tom Mitchell, 1997

La implementación de una aplicación basada en el ML es un proceso iterativo. Es decir, no se trata de un proceso estático en el que una vez que se desarrolle el modelo se mantiene fijo todo el tiempo, sino que él mismo se va actualizando con los nuevos datos que va recibiendo para realizar sus predicciones. Aunque sí es cierto que no es necesario modificar las reglas ni realizar un entrenamiento exhaustivo de nuevo. Esto es así porque, según a qué campo se aplique, los datos pueden evolucionar, así como las preferencias y necesidades. [26][28]

Esta manera de trabajar del ML hace que difiera de la programación tradicional en el proceso de alcanzar el objetivo perseguido. En ambos casos, este es el de realizar una tarea, facilitando a las personas su trabajo o apoyándolas en él. Ambos procesos se encuentran esquematizados a continuación, en la Figura 2.3.



(a) Esquema de la programación tradicional.



(b) Esquema del ML.

Figura 2.3: Esquemas de la programación tradicional y el ML. [29]

Por una parte, en la programación tradicional, el programador informático se encarga de construir el programa que recoge todas las órdenes, escritas siguiendo una secuencia lógica, que la máquina tiene que seguir para realizar la tarea requerida. En este programa se tienen que recoger todas las posibilidades que se puedan dar sin dejar ningún cabo suelto. Luego, una vez implementado, la máquina solamente se encarga de ejecutarlo y, siguiendo exhaustivamente todos los pasos, dará una salida. La desventaja de este método es que a medida que el sistema crece y se hace más complejo o es necesario realizar alguna modificación, es el programador quien debe reescribir el código, lo cual lo hace un trabajo tedioso y difícil de mantener. [26][29]

En cambio, con el ML no es necesario tener un código perfectamente escrito ni conocer la relación que existe entre unos datos y sus resultados correspondientes, sino que la máquina es capaz de escribir sus propias reglas y relaciones a partir de esos dos conjuntos de datos. Y, una vez que se tenga dicho modelo, se puede aplicar a otro conjunto de datos para obtener el resultado que corresponda entonces. Como desventaja, hay que mencionar que se tiene que disponer de un gran conjunto de datos para lograr un modelo fiable. Sin embargo, la gran ventaja de este método es que a medida que se incorporan conjuntos de datos y resultados, el ordenador es capaz de adaptarse y modificar el modelo por sí mismo, mejorando automáticamente

con la experiencia a través de la autosuperación o, lo que es lo mismo, aprendiendo por sí misma. [26][29]

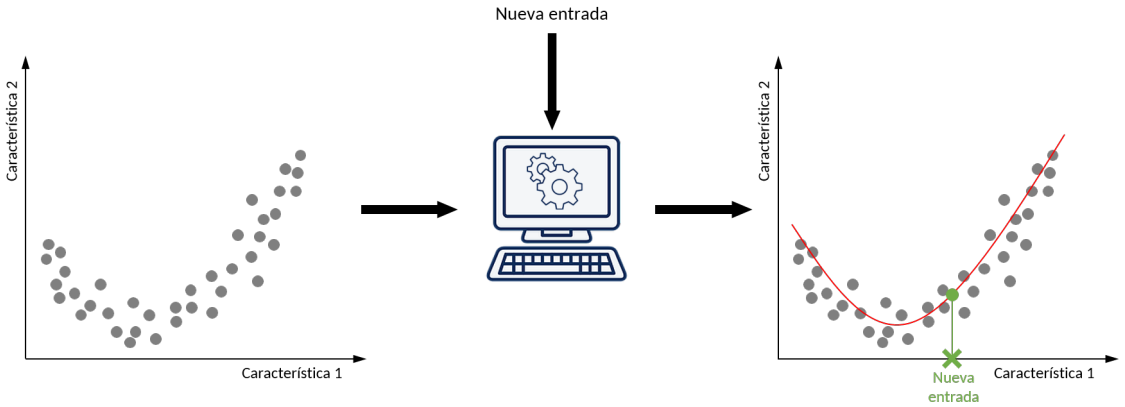
Al hablar de ML, también se pueden hacer divisiones para diferenciar distintos tipos. Una clasificación común de los sistemas de ML se basa en la supervisión que reciben durante su entrenamiento, lo que hace que tengan algoritmos y objetivos diferentes. De esta forma, se suelen distinguir 3 categorías: Aprendizaje Supervisado, Aprendizaje No Supervisado y Aprendizaje por Refuerzo.

En muchos sitios se suele destacar una cuarta categoría, el Aprendizaje Semisupervisado. Sin embargo, aquí se va a mencionar solamente, ya que, al fin y al cabo, está a medio camino entre Supervisado y No Supervisado.

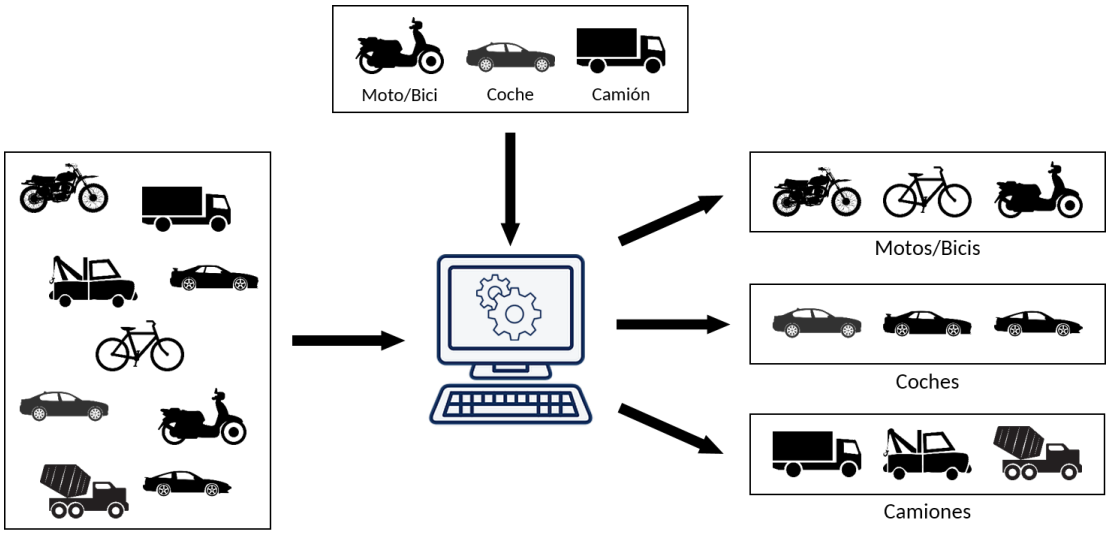
2.4.1. Aprendizaje Supervisado

Como bien se deduce de su nombre, el Aprendizaje Supervisado requiere la presencia de un tutor que le enseñe para poder dar respuestas correctas. Se asemeja al aprendizaje en las escuelas donde el maestro enseña los métodos para resolver una serie de problemas de forma que, luego, el alumno sea capaz de aplicar lo aprendido en situaciones similares. En este caso, lo que se aporta a la máquina es un conjunto de datos de entradas y sus correspondientes salidas, de forma que ella aprenda la relación que hay entre las dos partes y genere un modelo que, en el futuro, pueda proporcionar resultados correctos aunque no se hayan visto antes. Es por esto que el Aprendizaje Supervisado también se conoce por aprendizaje predictivo [30]. [4][31]

Tal como aparece en [30], el objetivo de esta clase es aprender la relación entre entradas $\vec{x}$ y salidas y, dado un conjunto de pares de datos entrada-salida etiquetados $\mathcal{D} = \{(\vec{x}_i, y_i)\}_{i=1}^{N}$. Ahí, $\mathcal{D}$ es el conjunto de entrenamiento formado por N ejemplos, los cuales constan de una entrada $\vec{x}_i$, denominada vector de características, y una salida y_i, en forma de número. Esa salida puede tomar valores de dos formas: una variable categórica de un conjunto finito $y_i \in \{1, \ldots, C\}$ o un valor escalar real $y_i \in \mathbb{R}$. Según sea, se distinguen dos tipos de funciones que se pueden llevar a cabo con Aprendizaje Supervisado: Regresión y Clasificación, ambas mostradas en la Figura 2.4.



(a) Regresión. El algoritmo crea un modelo de tendencia con los datos de entrenamiento y, luego, estima la salida de una nueva entrada.



(b) Clasificación. El algoritmo se entrena con datos etiquetados y, luego, es capaz de etiquetar nuevos datos.

Figura 2.4: Tareas de Aprendizaje Supervisado.

En la Regresión, el resultado consiste en un valor cuantitativo, es decir, un número de naturaleza continua, $y_i \in \mathbb{R}$.

Por su parte, en la Clasificación el resultado objetivo es cualitativo, dado por un valor discreto, ya que consiste en darle a cada entrada una etiqueta de salida que la adjudique a una de las categorías existentes, de forma que $y_i \in \{1, \ldots, C\}$, siendo C el número total de clases. Dependiendo cuántas categorías tenga el problema, se distinguen: clasificación binaria cuando $C = 2$, de forma que $y_i \in \{0, 1\}$, y clasificación multiclase cuando $C > 2$. A veces, el resultado se puede dar con probabilidades.

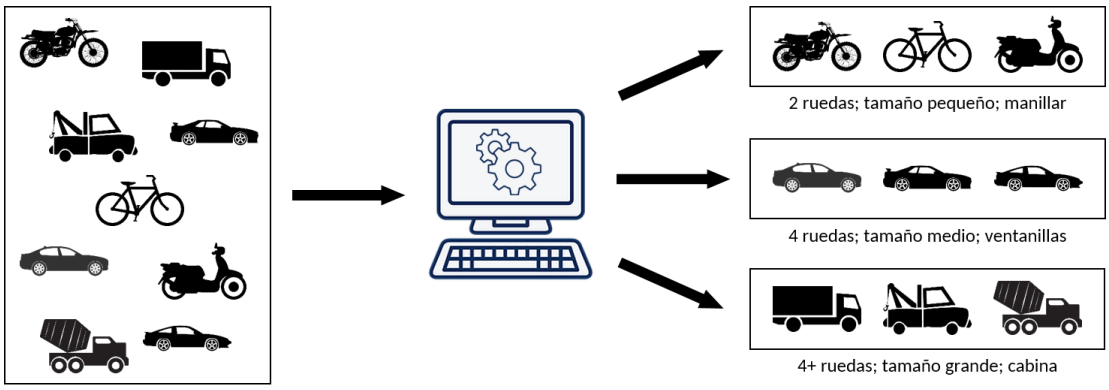
2.4.2. Aprendizaje No Supervisado

Al contrario que en el caso anterior, en el Aprendizaje No Supervisado sólo se cuenta con un conjunto de datos de entrada, $\mathcal{D} = \{\vec{x}_i\}_{i=1}^{N}$, sin tener ninguna información sobre las salidas correspondientes, de ahí el porqué de su nombre. De esta forma, se trata de un aprendizaje más anárquico, ya que no se define detalladamente el problema ni se le da información de lo que representan los datos. Es un método de carácter exploratorio que sirve para conocer la estructura de los datos y dar información que permita su comprensión, lo cual hace que, también, se conozca como aprendizaje descriptivo. [4][30][31]

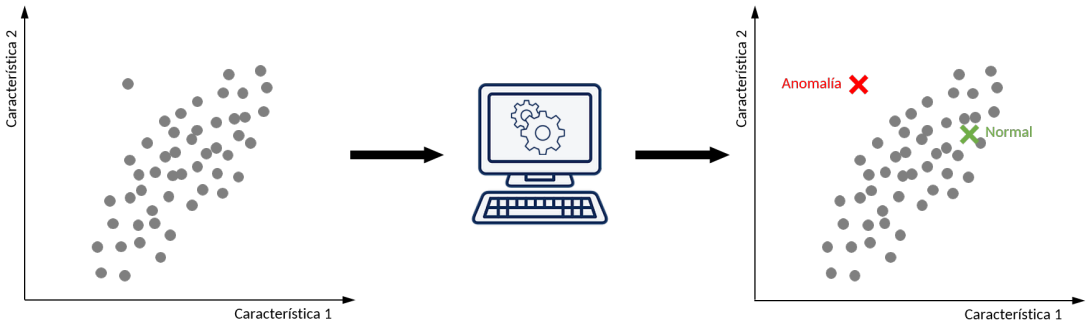
En este caso, no siempre es fácil saber cómo de bien está funcionando el modelo, debido a que no se tienen datos de referencia con los que compararlos y es una tarea más bien subjetiva [31]. Por ejemplo, un niño después de una cabalgata donde se repartan caramelos, puede ver todo lo que ha cogido y puede agruparlos de distintas maneras: por tamaño, colores, tipos... Esto no quiere decir que esté bien o mal, dependiendo de cómo lo haga, simplemente son distintas maneras de hacerlo.

Los algoritmos de Aprendizaje No Supervisado se utilizan para llevar a cabo distintas tareas: Clustering o Agrupación, Detección de Anomalías, Visualización y Reducción de Dimensión y Aprendizaje de Reglas de Asociación. [27]

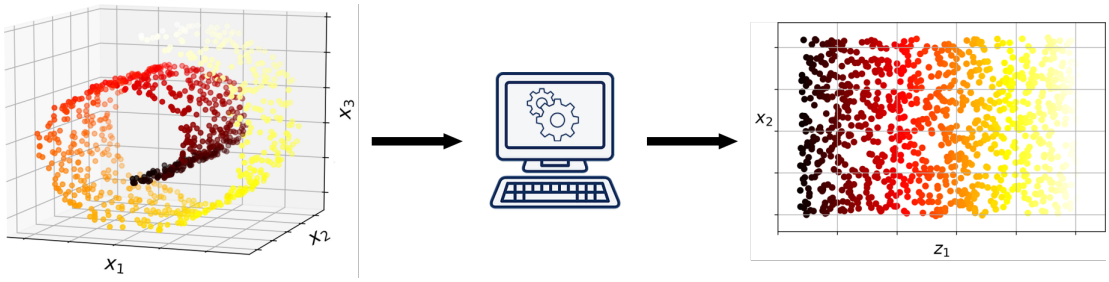
En la Figura 2.5 se muestran imágenes explicativas de cada tarea. El Clustering o Agrupación consiste en detectar conjuntos de datos con características similares. La Detección de Anomalías, como es de esperar, es encontrar datos atípicos dentro de un conjunto. La Visualización trata de ver la organización y la estructura que tienen los datos para detectar posibles patrones, mientras que la Reducción de Dimensión tiene como objetivo simplificar los datos sin perder demasiada información, así como fusionar varias características que estén relacionadas y que tengan comportamientos similares. Por último, el Aprendizaje de Reglas de Asociación sirve para descubrir relaciones entre los distintos atributos.



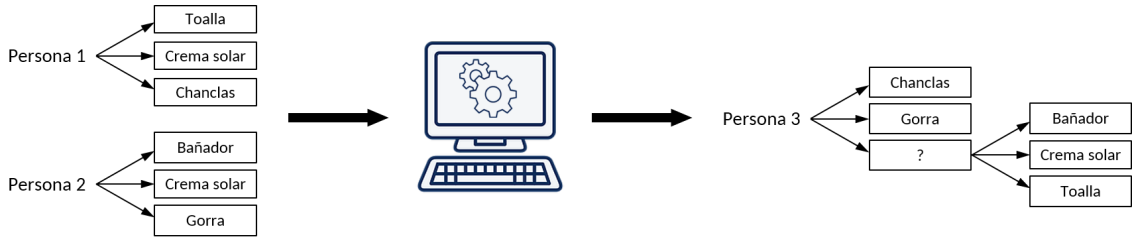
(a) Clustering o Agrupación. El algoritmo agrupa los vehículos con características similares.



(b) Detección de Anomalías. El algoritmo detecta datos que están fuera del conjunto considerado normal.



(c) Reducción de Dimensión. El algoritmo realiza un cambio de coordenadas para facilitar la comprensión de los datos. [27]



(d) Reglas de Asociación. Las personas 1 y 2 hacen las compras para ir a la playa. Como la persona 3 compra artículos que también han elegido las otras personas, el algoritmo los relaciona y le sugiere los otros artículos que también ha adquirido las personas 1 y 2.

Figura 2.5: Tareas de Aprendizaje No Supervisado.

2.4.3. Aprendizaje por Refuerzo

El RL es la técnica elegida para el desarrollo de este trabajo, como ya se sabe. Ya que se profundizará más en el siguiente capítulo, aquí se va a comentar de forma introductoria solamente.

En este método, el sistema está inmerso en el mundo que le rodea y aprende de la experiencia que va adquiriendo a base de explorarlo e interactuar con él. Partiendo de unas reglas preestablecidas, el propio algoritmo es el que evalúa las distintas opciones y decide qué acción llevar a cabo para acercarse al objetivo deseado. [4][27][31]

Es un aprendizaje mediante ensayo y error, esto es, el sistema realiza una acción y recibe una respuesta, una recompensa, del medio según el efecto que ha causado en él. Esta respuesta puede ser positiva, en forma de premio, o negativa, como una penalización, dependiendo de la idoneidad de la acción realizada. Según estas, el sistema va aprendiendo qué acciones le han llevado al éxito y cuáles no, de forma que va creando su estrategia con el fin de maximizar la recompensa a largo plazo. Como se puede ver, es un método orientado a objetivos. [4][27][31]

Se van a comentar dos claros ejemplos de este tipo. Por un lado está el proceso de aprender a jugar a un juego, donde se progresa desde la mera asimilación de las reglas, en un principio, hasta ir creando una estrategia que lleve a la victoria. Y por otro lado, se considera el proceso de enseñarle a un perro a hacer un truco o acatar una orden (Figura 2.6). Cuando se consigue que el perro haga lo que se le ha enseñado, se le recompensa con un premio, mientras que si no lo hace, se le penaliza con una voz alta y quedándose sin premio. Mediante la repetición, poco a poco, el perro aprenderá qué acciones conllevan recompensa y cuáles no. [4][32]

Según qué aspectos se podría pensar que el RL es un caso concreto de Aprendizaje Supervisado o que forma parte del Aprendizaje No Supervisado. Sin embargo, esto no es así y se trata de un tipo de aprendizaje con unas características propias que lo desmarcan de ambos. En primer lugar, en los dos anteriores se busca un comportamiento correcto en situaciones individuales; mientras que en este se persigue maximizar la recompensa de una secuencia de situaciones, aunque, a veces, los pasos seguidos para lograrlo no sean los óptimos individualmente [33][34]. Luego, a diferencia del Aprendizaje Supervisado, en el RL, en ningún momento se provee al

sistema de una respuesta correcta concreta, pero sí de una señal de recompensa, lo que lo distancia del Aprendizaje No Supervisado [35].

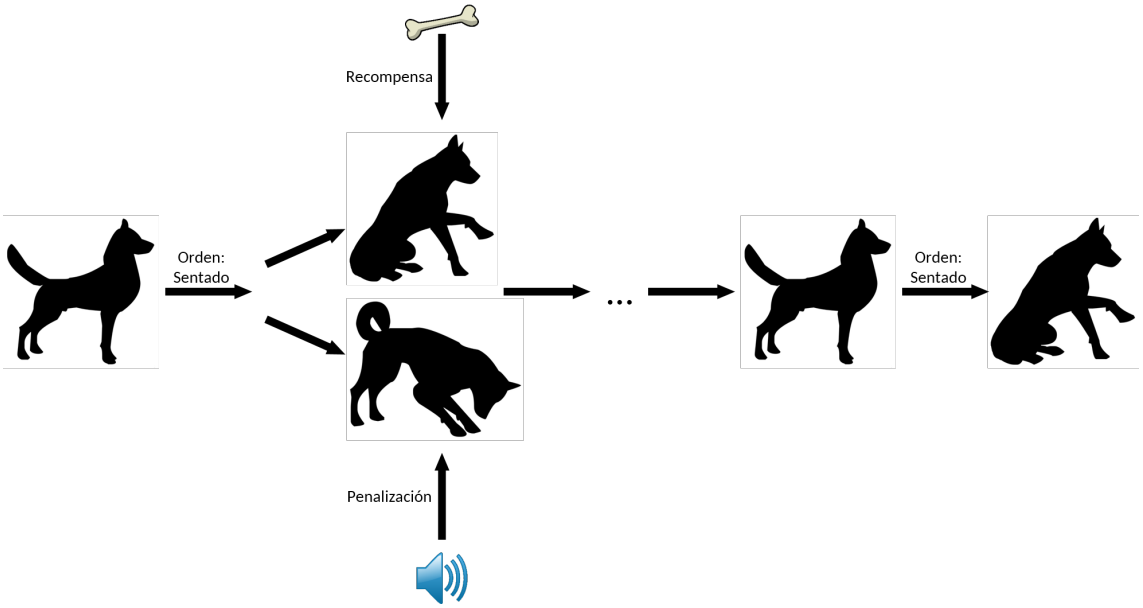


Figura 2.6: Ejemplo de Aprendizaje por Refuerzo: perro aprendiendo una orden. El perro aprende la orden que se le da a base de asociar la voz con una recompensa.

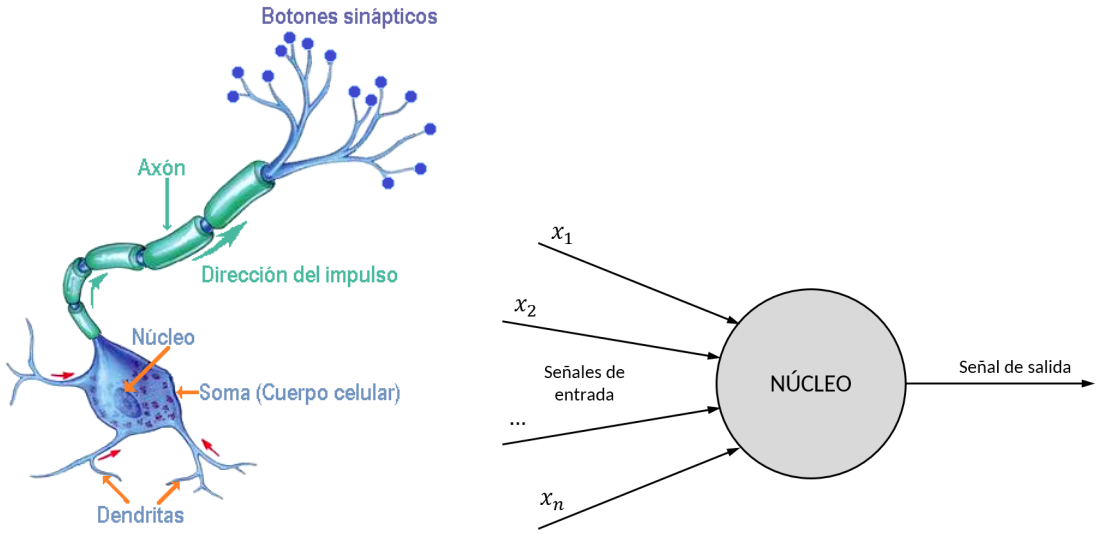
2.5. Deep Learning

Un paso más allá, está lo que se conoce como *Aprendizaje Profundo* (DL) —del inglés *Deep Learning*—, una técnica de aprendizaje basada en la estructura del cerebro humano, donde el término *profundo* hace referencia a las capas centrales de la red. El DL trata de resolver el trabajo del ML por medio de la aplicación de redes neuronales artificiales. [4][36]

Aprende gracias al diseño de estructuras neuronales artificiales inspiradas en los cerebros biológicos. La unidad más básica es el perceptrón, que es la representación matemática de una neurona (Figura 2.7). Las neuronas biológicas tienen un núcleo con diversas ramificaciones llamadas dentritas, que son las entradas de información procedentes de otras neuronas. En el núcleo se procesa dicha información y se genera una señal de salida que es transmitida a hacia el axón y los botones sinápticos que crean las conexiones sinápticas con las dentritas de otras neuronas. Una neurona por sí sola no tiene relevancia, pero sí cuando forma parte de una gran red. Lo mismo ocurre con los perceptrones y las redes neuronales artificiales. Un perceptrón individualmente sólo realiza un cálculo simple de combinación lineal de sus entradas ponderadas con sus respectivos pesos, pero su fuerza radica en las estructuras de redes que pueden formar. [36][37]

Como se ha dicho, lo realmente relevante es la secuenciación de perceptrones creando redes. Cada uno de ellos está conectado con muchos otros formando una estructura jerárquica de distintas capas interconectadas. Esta estructura (Figura 2.8) consiste en una capa de entrada que recibe los datos del exterior y los transmite

a una serie de capas de número variable, llamadas capas ocultas, donde se realizan los cálculos necesarios hasta obtener un resultado, que es comunicado a la capa de salida para su devolución. Cada capa hace transformaciones simples que van dando un resultado cada vez más complejo. [36][38]



(a) Neurona biológica. [37] **(b)** Perceptrón. [36]

Figura 2.7: Comparación de la neurona biológica y el perceptrón.

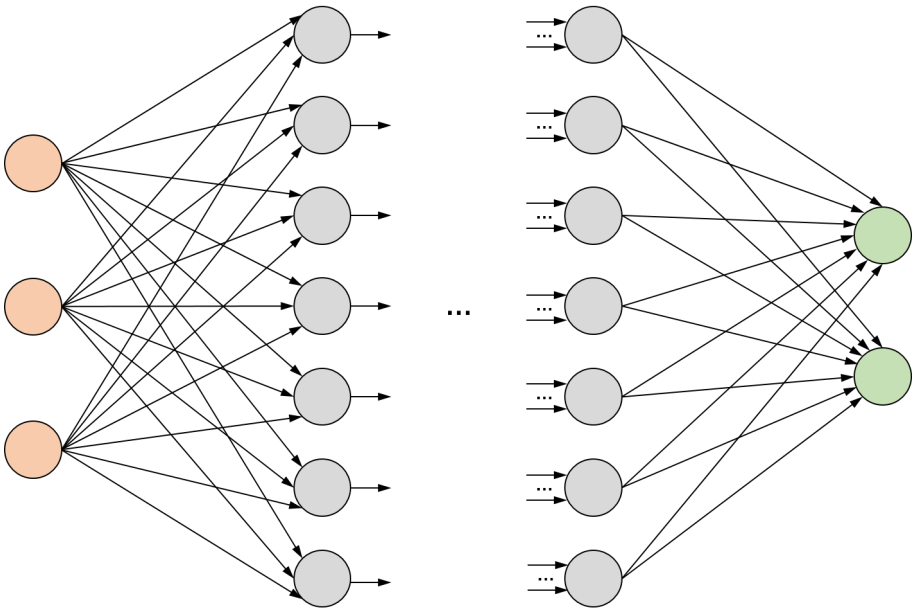
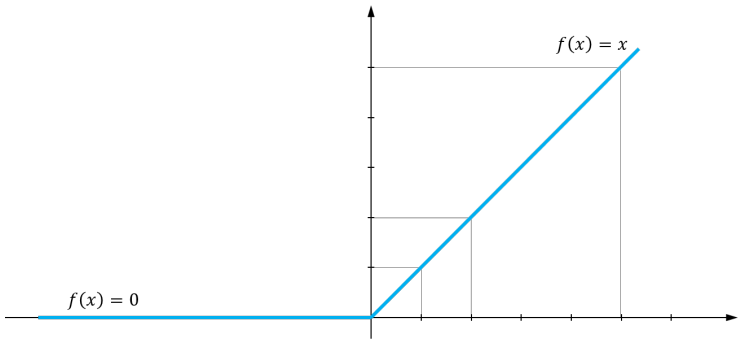


Figura 2.8: Esquema de una red neuronal profunda.

Pero aún falta algo, definir cómo transmite cada perceptrón su información. Este es el trabajo de las funciones de activación que añaden no linealidad al modelo. Hay distintos tipos, de entre los cuales la más común es la función rectificadora (ReLU) —del inglés *Rectified Linear Unit*—, que toma un valor de 0 para valores de entrada negativos y el propio valor de la entrada cuando este es positivo. [36]



$$f(x) = \begin{cases} 0, & x \leq 0 \\ x, & x > 0 \end{cases}$$

Figura 2.9: Función de activación ReLU.

El entrenamiento de la red consiste en el ajuste de los pesos de tal forma que a partir de unas entradas se obtenga la salida deseada. Al principio, estos pesos se asignan aleatoriamente y a medida que se van ajustando la red va aprendiendo. [4][36]

Estas estructuras se combinan con las técnicas descritas en los apartados anteriores. Por ejemplo, se tiene el *Aprendizaje por Refuerzo Profundo* —del inglés *Deep Reinforcement Learning* (DRL).

3 Aprendizaje por Refuerzo

El *Aprendizaje por Refuerzo* se conoce como RL, que son sus siglas en inglés, provenientes de *Reinforcement Learning*. En el Apartado 2.4.3 anterior se introdujo este tipo de aprendizaje, tema que se va ampliar de forma más técnica en el presente capítulo, como ya se adelantó.

3.1. Conceptos básicos

En primer lugar, se van a definir una serie de conceptos básicos imprescindibles para la comprensión del RL, de acuerdo con [39] y [40].

- **Agente**. Es la entidad que aprende gracias a la percepción y la exploración de su entorno por medio de llevar a cabo acciones sobre él con la esperanza de obtener recompensas.
- **Entorno**. Es el medio en el que se encuentra inmerso el agente. Engloba a todo lo que es externo al agente. Contiene limitaciones y reglas en cada momento.
- **Acción** (a). Se refiere a cada movimiento que puede llevar a cabo el agente.
- **Estado** (s). Se trata de cada situación en la que se encuentra el entorno, esto es, sus propiedades en un instante.
- **Recompensa** (r). Corresponde a la respuesta que da el entorno evaluando el éxito o el fracaso de las acciones llevadas a cabo por el agente sobre él.
- **Factor de descuento** (γ). En algunos casos, se utiliza para ponderar las recompensas, de forma que se vaya reduciendo el efecto que tienen las recompensas inmediatas sobre la acumulada a lo largo del tiempo.
- **Política** (π). Es la estrategia que define el comportamiento del agente en cada momento para decidir qué acción realizar según sea el estado actual.
- **Valor-estado** (V). Es el rendimiento esperado a largo plazo de la estancia en un estado. La función valor indica cómo de buena será en el futuro una

decisión tomada en base al estado actual. Se busca aumentar este parámetro, ya que es el que maximiza la recompensa a largo plazo.

- **Valor-acción** (Q). Es similar a V, pero teniendo en cuenta también la acción actual. Su función da una idea de cómo será la recompensa a largo plazo de una acción tomada en el estado actual, teniendo en cuenta que la acción que se lleve a cabo puede condicionar situaciones futuras del agente.
- **Trayectoria**. Es la secuencia de estados por los que pasa el agente y las acciones que llevan a ellos.
- **Modelo**. Es una representación que imita el comportamiento del entorno y ayuda a conocer cómo se comportaría ante una situación, sirviendo de ayuda en la planificación de los movimientos que se llevarán a cabo. Dados un estado y una acción, el modelo predice cuáles serán la recompensa y el siguiente estado. No existe necesariamente, ya que hay métodos de resolución libres de modelo.

3.2. Descripción del problema

Hay una frase muy común que dice *de los errores se aprende*, esta podría ser parte de una definición del RL sin problemas. Si se piensa en la forma más básica de aprendizaje en el reino animal, viene a la cabeza la interacción de un individuo con el mundo que le rodea para conseguir un objetivo. Este es un proceso cíclico en el que, ante una determinada situación, el primero realiza acciones y ve cómo afectan al segundo por cómo reacciona. Así, mediante una serie de intentos en forma de prueba y error, el sujeto va actuando de distintas formas ante situaciones similares y reteniendo aquellas a las que les ha sacado más partido, de manera que su comportamiento se ve modificado por su experiencia. En el momento en el que esto ocurre, se habla de aprendizaje. [32][35]

Da igual si se trata de un depredador perfeccionando su técnica de caza; o la forma en que los herbívoros han aprendido a reconocer qué plantas son capaces de tolerar sus organismos y cuáles no; o cómo los primeros humanos, a partir de hacer fuego por primera vez, lo siguieron haciendo. Todos estos ejemplos tienen como nexo común la técnica de aprendizaje que llevan a cabo. Con base en ella, se ha tratado de trasladar la naturaleza a la forma de aprender de la tecnología, dando como resultado el RL [33].

Por tanto, el RL propone un enfoque para comprender y automatizar cómo una máquina aprende a alcanzar sus objetivos mediante la interacción y por medio de la toma de decisiones secuencial, aprendiendo a elegir qué hacer en cada momento [35]. Como ya se intuye, el RL se basa en dos elementos: agente y entorno, cuyas relaciones son el objeto de interés.

Estableciendo la situación, el agente está situado en el entorno, como se ve en la Figura 3.1. En cualquier momento, dicho entorno tiene un estado determinado, dentro de un conjunto de estados posibles, sobre el cual el agente recibe información. A raíz de esto, el agente toma una acción, elegida de un conjunto de acciones posibles, lo que causa la modificación del estado del entorno. Luego, el agente vuelve a recibir

la información del entorno, además de una posible señal de recompensa evaluando cuán apropiada ha sido la acción realizada para alcanzar el objetivo, de forma que se refuerce o penalice su comportamiento. Todas las decisiones que se tomen tendrán sus consecuencias y el agente irá aprendiendo de ellas a guardar o desechar sus movimientos, para crear su política de aprendizaje, a medida que vaya realizando iteraciones. Por tanto, esta secuencia de estado-acción-recompensa es la que trata de optimizar el RL por medio de la política, maximizando la recompensa obtenida a largo plazo.[1] [33][34]

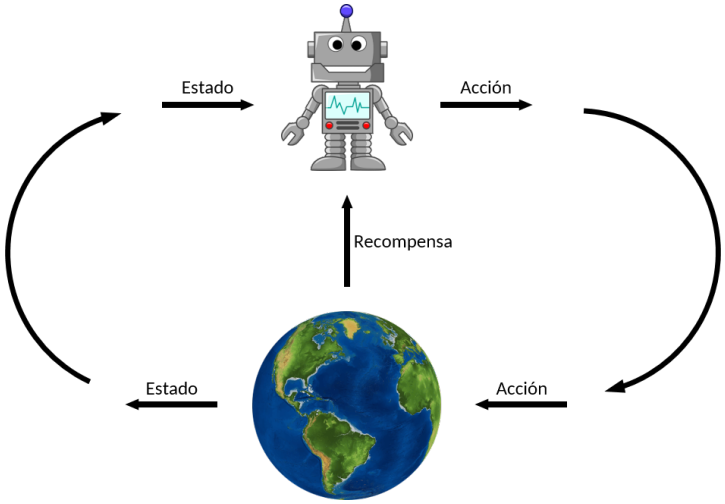


Figura 3.1: Interacción agente-entorno. [41]

3.2.1. Proceso de Decisión de Markov

El *Proceso de Decisión de Markov* (MDP) —del inglés *Markov Decision Process*—, es la teoría bajo la cual se formalizan matemáticamente los problemas de toma de decisiones secuencial en entornos totalmente observables y, por extensión, el RL aquí estudiado. Estos forman un modelo (Figura 3.2) para el aprendizaje por interacción entre un agente y el entorno, en base a: estados, acciones, función de recompensa y transición entre estados. [42][43]

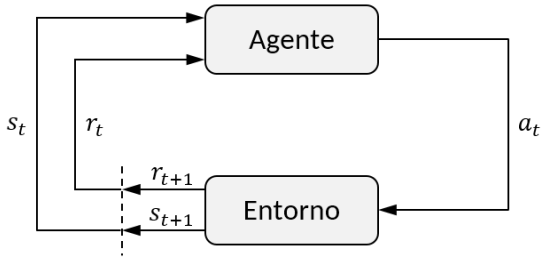


Figura 3.2: Interacción agente-entorno en un MDP. [42]

[1]Hay que distinguir entre: estado del entorno, estado del agente y observación. Los dos primeros son los respectivos estados internos y la observación es la información que recibe el agente por parte del entorno. Los problemas pueden ser de entorno total o parcialmente observable, sin embargo, en este trabajo sólo se va a considerar el caso primero, en el que los tres términos coinciden. Es por eso que se habla de que el agente recibe el estado del entorno y no una observación. [41]

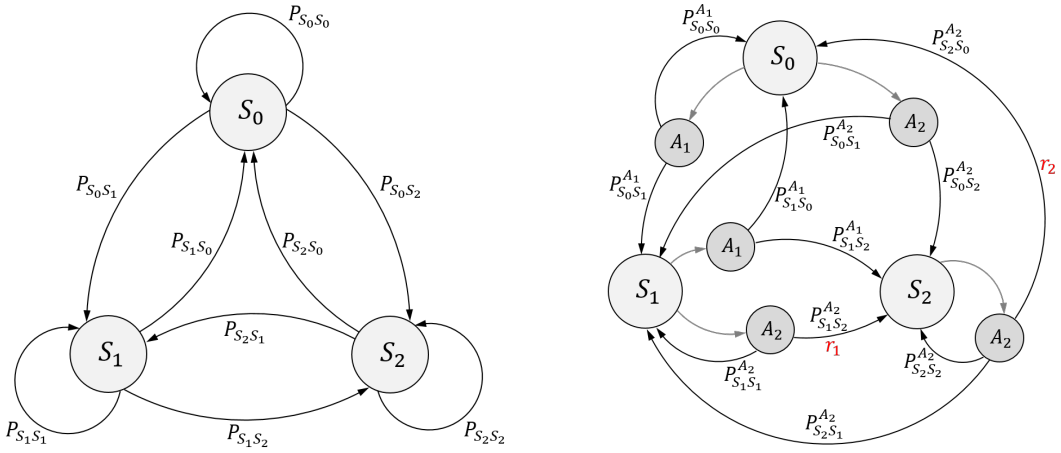
Para simplificar, este desarrollo se va a limitar a considerar lapsos de tiempo discretos para cada iteración, aunque cabe saber que bien se puede extender a casos de tiempo continuo. Del mismo modo, se va a tratar con conjuntos de acciones y estados finitos. [42][43]

$$t = 0, 1, 2, 3, ...$$
$$\mathcal{S} : \{S_1, S_2, ..., S_N\}$$
$$\mathcal{A} : \{A_1, A_2, ..., A_K\}$$

Los MDP son una extensión de las Cadenas de Markov (Figura 3.3), también conocidas como Procesos de Markov, que no son más que una sucesión de estados que cumplen la Propiedad de Markov. Esto es, cada estado contiene toda la información útil de la interacción histórica entre el agente y el entorno que pueda afectar a estados futuros, por tanto, la probabilidad de pasar al siguiente estado es independiente del pasado, dado el presente. [44][45][46]

Así, se define que un estado cumple la propiedad de Markov si y sólo si:

$$P(s_{t+1}|s_t) = P(s_{t+1}|s_0, s_1, ..., s_{t-1}, s_t) \tag{3.1}$$



(a) Representación de Cadena de Markov. **(b)** Representación de MDP.

Figura 3.3: Representaciones gráficas.

A partir de esto, se define una Cadena de Markov, cuya secuencia se muestra en la Figura 3.4, como una tupla $< \mathcal{S}, T >$. [45][46]

- $\mathcal{S}$ es el conjunto de estados.
- $T \rightarrow \mathcal{P}$ es la dinámica de transición, que consiste en una tabla de probabilidad dada por la matriz (3.3) de probabilidad de transición de estados. [47]

$$\mathcal{P}_{ss'} = P(s_{t+1}|s_0, s_1, ..., s_{t-1}, s_t) = P(s_{t+1} = s'|s_t = s) \tag{3.2}$$

$$\mathcal{P} = \begin{pmatrix} P_{S_0S_0} & P_{S_0S_1} & \cdots & P_{S_0S_N} \\ P_{S_1S_0} & P_{S_1S_1} & \cdots & P_{S_1S_N} \\ \vdots & \vdots & \ddots & \vdots \\ P_{S_NS_0} & P_{S_NS_1} & \cdots & P_{S_NS_N} \end{pmatrix} \tag{3.3}$$

- $\mathcal{P}_{ij} = P(s_{t+1}|s_t) \leq 1$
- $\mathcal{P} \in \mathcal{M}_{n \times n}$
- $\sum_{j=1}^{n} \mathcal{P}_{ij} = 1$

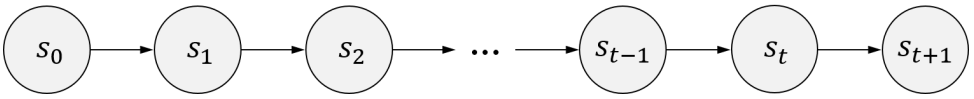


Figura 3.4: Secuencia de una Cadena de Markov. [45]

Ampliando lo anterior de forma que la secuencia de estados queda como se muestra en la Figura 3.5, un MDP se define como una tupla $< \mathcal{S}, \mathcal{A}, R, T, \gamma >$. [43][44][46]

- $\mathcal{S}$ es el conjunto de estados.
- $\mathcal{A}$ es el conjunto de acciones. Puede ocurrir que no todas ellas puedan llevarse a cabo en todos los estados, entonces se denota como $\mathcal{A}(s)$ para cada estado.
- $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathcal{P}^a_{ss'} \in [0, 1]$ es la dinámica de transición, que consiste en una tabla de probabilidad (Tabla 3.1) que se puede entender como un conjunto de matrices de probabilidad, una por cada acción posible, $\mathcal{P}^a$.

$$\begin{aligned} \mathcal{P}^a_{ss'} &= P(s_{t+1}|s_0, a_0, s_1, a_1, ..., s_{t-1}, a_{t-1}, s_t, a_t) = \\ &= P(s_{t+1} = s'|s_t = s, a_t = a) = T(s, a, s') \end{aligned} \tag{3.4}$$

	S_0	S_1	...	S_N
$T(S_0, A_1, s')$	$P^{A_1}_{S_0S_0}$	$P^{A_1}_{S_0S_1}$	...	$P^{A_1}_{S_0S_N}$
$T(S_0, A_2, s')$	$P^{A_2}_{S_0S_0}$	$P^{A_2}_{S_0S_1}$	...	$P^{A_2}_{S_0S_N}$
⋮	⋮	⋮	⋱	⋮
$T(S_0, A_K, s')$	$P^{A_K}_{S_0S_0}$	$P^{A_K}_{S_0S_1}$	...	$P^{A_K}_{S_0S_N}$
⋮	⋮	⋮	⋱	⋮
$T(S_1, A_1, s')$	$P^{A_1}_{S_1S_0}$	$P^{A_1}_{S_1S_1}$	...	$P^{A_1}_{S_1S_N}$
⋮	⋮	⋮	⋱	⋮
$T(S_N, A_K, s')$	$P^{A_K}_{S_NS_0}$	$P^{A_K}_{S_NS_1}$	...	$P^{A_K}_{S_NS_N}$

$$\sum_{s' \in \mathcal{S}} T(s, a, s') = 1$$

Tabla 3.1: Tabla de probabilidad. [48]

- $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathcal{R}^a_{ss'} \in \mathbb{R}^2$ es la función de recompensa, que representa la recompensa inmediata esperada en cada momento. [42]

$$\mathcal{R}^a_{ss'} = E(r_{t+1} = r|s_{t+1} = s', s_t = s, a_t = a) = R(s, a, s') \tag{3.5}$$

- $\gamma \in [0, 1]$ es el factor de descuento.

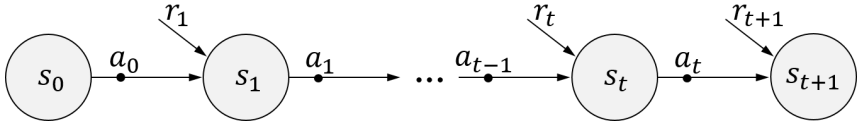


Figura 3.5: Secuencia de un MDP.

[2]En otras formulaciones depende sólo del estado presente o del estado y de la acción tomada, $R : \mathcal{S} \to \mathcal{R}_s \in \mathbb{R}$ o $R : \mathcal{S} \times \mathcal{A} \to \mathcal{R}^a_s \in \mathbb{R}$.

3.2.2. Recompensa acumulada

Ya se ha mencionado con anterioridad que el objetivo de estos problemas es encontrar una política óptima, es decir, aquella que maximice la recompensa acumulada que el agente espera recibir a largo plazo. Esta recompensa acumulada acoge a la secuencia de recompensas recibidas a partir de un determinado momento. En el caso más básico, se puede ver como una simple suma: [42]

$$R_t = r_{t+1} + r_{t+2} + \cdots + r_{t+T} = \sum_{i=1}^{T} r_{t+i} \tag{3.6}$$

Es común que las tareas que lleva a cabo un agente se puedan dividir en episodios con una duración determinada entre un estado inicial y un estado objetivo. No obstante, a menudo, el número de acciones que se va a realizar antes de alcanzar el objetivo es desconocido o, incluso, se dan casos de tareas continuas que perduran en el tiempo sin un fin. De esta forma, se puede hablar de tres tipos de problemas: de horizonte fijo, de horizonte indefinido y de horizonte infinito, respectivamente. [42][43][44]

La ecuación (3.6) es aplicable cuando se conoce la duración del episodio y se puede garantizar que la suma de recompensas será finita. En cambio, para el resto de casos se podría obtener una suma infinita, lo que hace que este método no sirva a la hora de comparar las distintas alternativas. Es por eso que entra en juego el concepto de descuento, como se muestra en la ecuación (3.7), con un factor $0 \leq \gamma \leq 1$. [42][44]

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{i=1}^{\infty} \gamma^{i-1} r_{t+i} \tag{3.7}$$

De esta forma, se va reduciendo el peso de las recompensas a medida que es más lejano el momento en el que se recibirán, de manera que se garantiza que la recompensa acumulada se mantenga en valores finitos aunque se trate de tareas continuas infinitas. Analizando los casos extremos: para $\gamma = 1$, se está en el caso de la ecuación (3.6); y, para $\gamma = 0$, se suele decir que el agente es miope porque sólo tiene en cuenta la recompensa inmediata, ignorando el resto. [42][43][44]

La ecuación (3.7) se puede expresar de la siguiente forma:

$$R_t = r_{t+1} + \gamma(r_{t+2} + \gamma r_{t+3} + \cdots) = r_{t+1} + \gamma R_{t+1} \tag{3.8}$$

3.2.3. Políticas y Funciones Valor

Una política determina el comportamiento del agente, a través de relacionar las acciones y los estados. La modificación de estas según su experiencia es la base de los métodos de aprendizaje por refuerzo. [42]

Las políticas pueden ser estacionarias, si las probabilidades son independientes del tiempo manteniéndose invariables durante todo el proceso; o no estacionarias, de forma que la probabilidad de realizar una acción dependa de la localización en

el tiempo en el que se produzca [46]. También pueden diferenciarse en políticas determinísticas o estocásticas. Las primeras son de elección fijada, es decir, siempre que se esté en un estado se tomará la misma acción $\pi(s) : \mathcal{S} \to \mathcal{A}$; mientras que las segundas lo hacen en base a una distribución de probabilidad $\pi(s, a) : \mathcal{S} \times \mathcal{A} \to [0, 1]$, donde $\sum_{a \in \mathcal{A}(s)} \pi(s, a) = 1$ [43][48]. Así, estas últimas se expresan [46]:

$$\pi(s, a) = P\big(a_t = a | s_t = s\big) \tag{3.9}$$

Para vincular los criterios de optimización con las políticas se recurre a las funciones valor [43]. Son funciones valor de una política las funciones de valor-estado (3.10) y de valor-acción (3.11).

$$V^{\pi}(s) = E_{\pi}\big(R_t | s_t = s\big) = E_{\pi}\left(\sum_{i=1}^{\infty} \gamma^{i-1} r_{t+i} | s_t = s\right) \tag{3.10}$$

$$Q^{\pi}(s, a) = E_{\pi}\big(R_t | s_t = s, a_t = a\big) = E_{\pi}\left(\sum_{i=1}^{\infty} \gamma^{i-1} r_{t+i} | s_t = s, a_t = a\right) \tag{3.11}$$

Ellas dan una idea de cómo de bueno ha sido para el agente haber pasado por un estado o haber realizado una acción desde un estado, respectivamente, en términos de las recompensas esperadas, en base a las cuales se puede definir el rendimiento esperado como criterio de optimización. De esta forma, las funciones valor cuyos rendimientos esperados sean máximos, pertenecerán a políticas óptimas. [42][43]

Una política es mejor que otras cuando su rendimiento esperado es mayor o igual que el de las otras políticas para todos los estados, esto es, $\pi(s) \geq \pi'(s)$ si y sólo si $V^{\pi}(s) \geq V^{\pi'}(s)$. Siempre hay al menos una (pueden ser más) mejor que todas las demás, a la cual se la denota como política óptima (π^*), cuyas funciones valor serán las máximas posibles. [42]

$$V^{\pi^*}(s) = V^*(s) = \max_{\pi} V^{\pi}(s) \tag{3.12}$$

$$Q^{\pi^*}(s, a) = Q^*(s, a) = \max_{\pi} Q^{\pi}(s, a) \tag{3.13}$$

Ambas ecuaciones, (3.12) y (3.13), se relacionan de la siguiente forma. De manera que el valor de un estado es igual al valor de la mejor acción posible. [48]

$$V^*(s) = \max_{a} Q^*(s, a) \tag{3.14}$$

A partir de $Q^*(s, a)$ se puede obtener la política óptima. [48]

$$\pi^*(s) = \arg\max_{a} Q^*(s, a) \tag{3.15}$$

3.2.4. Dificultades

Los problemas enmarcados bajo la teoría desarrollada anteriormente tienen algunas dificultades. [35][43][49]

Dilema exploración-explotación. Es normal que en un principio, el agente no conozca la dinámica del problema, de forma que debe explorar el entorno para ir construyendo su política de actuación. Pasado un tiempo, tendrá una política con un cierto rendimiento. A partir de ahí, podrá explotar esa política que ya conoce o explorar nuevas opciones. He ahí el problema, la dificultad reside en encontrar un balance entre la exploración de nuevas acciones y la explotación de las acciones que ya ha probado y que le han resultado exitosas. Realizar cualquier técnica exclusivamente sería un error. Por un lado, si tiene una actitud meramente de explotación, nunca sabrá si hay opciones mejores; y, por otro lado, si la actitud sostenida es de completa exploración, nunca usará lo que se ha aprendido. Es por ello que el agente debe intentar una variedad de acciones y favorecer progresivamente a aquellas han dado mejores resultados.

Hay distintas estrategias de exploración que deciden cuándo realizar una cosa u otra, una de las más sencillas y utilizadas es la estrategia ϵ-greedy. Esta consiste en que el agente explota una acción aprendida con una probabilidad de $(1 - \epsilon)$ y explora otra acción con una probabilidad ϵ, donde $0 \leq \epsilon \leq 1$. Lo ideal es que la exploración decrezca conforme se tiene más experiencia, por lo que se va reduciendo el parámetro ϵ con el paso del tiempo, hasta un cierto punto.

Problema de atribución de culpa. Dentro de una secuencia de decisiones no se puede saber de inmediato cómo contribuirá una acción a la hora de alcanzar el objetivo final. Hasta que este no se alcance, es difícil evaluar cómo de buena ha sido cada acción, ya que, al principio, una acción ha podido parecer mala en algunos casos y, sin embargo, ha terminado siendo óptima gracias al efecto de las decisiones tomadas por el agente en el futuro.

3.3. Solución al problema

Ahora, se van a buscar técnicas para encontrar políticas óptimas o, lo que es lo mismo, para resolver un MDP. La elección de una u otra técnica depende de si se parte o no de un modelo exhaustivo del entorno.

Por un lado, cuando se dispone de un modelo del entorno, se pueden aplicar unas ecuaciones matemáticas para calcular directamente las funciones valor y políticas. Estos algoritmos se conocen como Programación Dinámica (DP) —del inglés *Dynamic Programming*. [43]

Por otro lado, no conocer el modelo, o no conocerlo por completo, genera la necesidad de explorar el entorno y recopilar muestras con las que realizar aproximaciones sobre el modelo desconocido, para lo que hay que recurrir a la interacción con el entorno. Estas son técnicas de RL, como ya se ha comentado. En este caso, se pueden tomar dos caminos: métodos basados en un modelo, que consisten en aprender el modelo mediante la interacción con el entorno para, luego, aplicar DP; y métodos libres de modelo, que tratan de estimar los valores sin necesidad de conocer el modelo. [43]

Los algoritmos libres de modelo pueden ser de dos tipos: *on-policy* u *off-policy*. Los algoritmos *on-policy* toman las acciones en base a la propia política que se está evaluando y mejorando; mientras que los algoritmos *off-policy* utilizan dos políticas separadas, hay una que define el comportamiento al mismo tiempo que se evalúa otra en búsqueda de la política óptima. [50]

Dentro de los métodos libres de modelo se encuentran el Método de Monte Carlo (MC) y el Método de Diferencias Temporales (TD) —del inglés *Temporal-Difference*—. El segundo es el que interesa para este trabajo, que tiene tintes tanto del primero como de la DP. Por ello, lo que sigue se va a centrar en ellos. Por contra, no se van a tratar los métodos basados en un modelo.

3.3.1. Programación Dinámica

Con *Programación Dinámica* se hace referencia a una serie de algoritmos que se utilizan para el cálculo de políticas óptimas cuando es conocido el modelo del entorno del problema. Se basan en las ecuaciones de optimalidad de Bellman (3.18) y (3.19).

Jugando con las expresiones del apartado 3.2.3, se llega a lo siguiente.

$$\begin{aligned} Q^*(s,a) &= E\big(r_{t+1} + \gamma R_{t+1} | s_t = s, a_t = a\big) \\ &= E\big(r_{t+1} + \gamma V^*(s_{t+1}) | s_t = s, a_t = a\big) \end{aligned} \tag{3.16}$$

$$\begin{aligned} V^*(s) &= \max_a Q^*(s,a) \\ &= \max_a E\big(r_{t+1} + \gamma V^*(s_{t+1}) | s_t = s, a_t = a\big) \end{aligned} \tag{3.17}$$

Expresándolas de la siguiente forma, resultan lo que se conoce como ecuaciones de optimalidad de Bellman, que se han mencionado antes.

$$V^*(s) = \max_a \sum_{s'} T(s,a,s') \Big(R(s,a,s') + \gamma V^*(s') \Big) \tag{3.18}$$

$$\begin{aligned} Q^*(s,a) &= \sum_{s'} T(s,a,s') \Big(R(s,a,s') + \gamma V^*(s') \Big) \\ &= \sum_{s'} T(s,a,s') \Big(R(s,a,s') + \gamma \max_{a'} Q^*(s',a') \Big) \end{aligned} \tag{3.19}$$

Como se puede apreciar, estas expresiones hacen que los valores de un estado dependan de los valores del estado siguiente, lo que dificulta el cálculo directamente. Para ello, se han desarrollado algoritmos que se basan en la iteración, de los que los más importantes son: iteración de política e iteración de valores. Todos los métodos tienen en común que realizan operaciones que actualizan el valor de una función en base a los valores de los posibles estados siguientes hasta que ya no se notan cambios y se ha dado la convergencia, satisfaciendo las ecuaciones de Bellman. Esta forma de actualización se denomina *bootstrapping* y es una propiedad muy importante. [51]

3.3.2. Método Monte Carlo

El método de *Monte Carlo* es un método libre de modelo basado en la exploración. Lo que hace es recoger secuencias completas de episodios de muestra y promediar las recompensas que va recibiendo de interactuar con el entorno. Sólo es cuando ha concluido un episodio que se evalúan y actualizan los valores y las políticas. [50]

Una expresión sencilla de este método es la siguiente: [52]

$$V(s_t) \leftarrow V(s_t) + \alpha\Big[R_t - V(s_t)\Big] \tag{3.20}$$

3.3.3. Método de Diferencias Temporales

Con el fin de subsanar el problema de atribución de culpa, visto en el apartado 3.2.4, aparece el método de aprendizaje de *Diferencias Temporales*. Cuando los efectos de una acción se perciben pasado un tiempo desde que se realiza, se podría esperar hasta que termine el episodio para evaluarlos. Sin embargo, esto requiere gran capacidad de memoria, además de la incertidumbre de no conocer dicho final, incluso si lo habrá. Así, se opta por ir ajustando el valor estimado de un estado según la recompensa inmediata y el valor estimado del siguiente estado. [43]

El TD aúna las ventajas de los dos anteriores. Por un lado, no se nutre de un modelo del entorno previamente dado, sino que va obteniendo información a base de exploración, como el MC. Y, por otro lado, se basa en el *bootstrapping* como la DP, esto es que actualiza las estimaciones de valores basándose en otras estimaciones aprendidas viendo el error que hay entre ellas, sin tener que esperar al resultado final. [43][52]

Aquí se va a ver el método TD más sencillo: TD(0). Se trata de un caso particular de TD(λ), más complejo que no se va a desarrollar aquí.

Viendo la expresión (3.20), se tiene que la actualización se realiza cuando acaba el episodio, que es cuando se conoce R_t y se puede determinar $V(s_t)$. En cambio, ahora, se modifica para realizar la actualización en cada paso de tiempo usando la recompensa observada r_{t+1} y la estimación $V(s_{t+1})$. Así, queda la expresión de la función de actualización de TD(0). [52]

$$V(s_t) \leftarrow V(s_t) + \alpha\Big[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)\Big] \tag{3.21}$$

El parámetro $0 \leq \alpha \leq 1$ es la tasa de aprendizaje que indica cuánta nueva información actualizamos sobre los valores calculados anteriormente. Si $\alpha = 1$ toda la nueva información sobreescribirá a la antigua; mientras que si $\alpha = 0$ la nueva información no se tendrá en cuenta. Lo deseable es que al principio se actualice más información y se vaya reduciendo con el paso del tiempo. [43]

Por ahora, en las expresiones sólo han aparecido los estados, pero también es útil tener en cuenta la intervención de las acciones a la hora de tomar decisiones. La

forma más sencilla es transformar en la función de actualización anterior los términos valor-estado por términos valor-acción. De esta forma, se obtiene el algoritmo Sarsa, que al igual que TD(0), es un algoritmo *on-policy*. [52]

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\Big[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\Big] \tag{3.22}$$

Con la misma base, sólo cambiando la estimación dada una política por la mejor estimación posible, se tiene el algoritmo *off-policy* llamado *Q-Learning*, que es uno de los desarrollos más importantes del RL y muy utilizado. Su función de actualización es la siguiente.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\Big[r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\Big] \tag{3.23}$$

A continuación, se va a desarrollar el pseudocódigo de este algoritmo.

Q-Learning

- Inicializar la tabla $Q(s, a)$ arbitrariamente
- Repetir para cada episodio:
  - Inicializar s
  - Repetir para cada paso del episodio:
    - Elegir una acción $a \in \mathcal{A}(s)$ usando la política derivada de Q con una estrategia de exploración
    - Realizar la acción a
    - Observar la recompensa r y el estado siguiente s'
    - $Q(s, a) \leftarrow Q(s, a) + \alpha\Big[r + \gamma \max_{a'} Q(s', a') - Q(s, a)\Big]$
    - $s \leftarrow s'$

  hasta llegar al estado terminal

3.3.4. Deep Q-Network

El problema del *Q-Learning* llega cuando la dimensión del problema crece y no se puede aplicar la tabulación. En ese caso, se puede combinar con las redes neuronales, y es entonces cuando aparece el *Deep Q-Network* (DQN), donde la aproximación de la función Q tratando de minimizar la función de pérdida se hace mediante redes neuronales.

Este método trae consigo dos aspectos importantes. El primero es el almacenamiento de experiencias en un buffer, creando el conjunto $\mathcal{D}$, de donde tomar muestras durante el entrenamiento para actualizar el algoritmo de forma que una interacción

puede evaluarse en episodios y políticas diferentes. Y el segundo son las redes objetivo con las que tener una referencia para poder minimizar la pérdida (o maximizar la recompensa esperada, que es lo mismo). Estas redes son copias de la red principal, pero actualizada cada más tiempo, por eso se dice que la red objetivo se congela. [53][54]

La función Q queda parametrizada con parámetros ϕ. Mientras que ϕ denota a la red principal, ϕ' denota a la red objetivo.

$$Q_\phi(s,a) \leftarrow Q_\phi(s,a) + \alpha\Big[r + \gamma \max_{a'} Q_{\phi'}(s',a') - Q_\phi(s,a)\Big] \tag{3.24}$$

Con esto, se ha resuelto el aspecto de la dimensionalidad del problema. Sin embargo, esta solución solamente es factible cuando el espacio de acciones es discreto. Con el fin de resolver este aspecto, se va a seguir ahondando en los algoritmos.

3.3.5. Método de Optimización de Política

Concretamente, se ve el Gradiente de una Política Determinista. En este tipo de métodos se trabaja con una política concreta. Se tiene una política determinista μ parametrizada con parámetros θ de la que se quiere maximizar la recompensa esperada. Esto se hace con el gradiente de la política. [53]

$$J(\theta) = E_{s\sim\rho^\mu}\Big[r\big(s,\mu_\theta(s)\big)\Big] \tag{3.25}$$

$$\nabla_\theta J(\theta) = E_{s\sim\rho^\mu}\Big[\nabla_a Q^\mu(s,a)\nabla_\theta \mu_\theta(s)\Big] \tag{3.26}$$

3.3.6. Deep Deterministic Policy Gradient

El algoritmo buscado es el *Deep Deterministic Policy Gradient* (DDPG) que es básicamente es el DQN para problemas continuos. Aúna las características de los expuestos anteriormente, de forma que aprende a la vez una política determinista y una función Q. Como el espacio de acción es continuo, se supone que la función $Q(s,a)$ es diferenciable respecto de la acción y se puede aproximar $\max_a Q(s,a)$ como $\max_a Q(s,\mu(s))$. [53][54]

La función Q queda parametrizada con parámetros ϕ y la política μ con parámetros θ. Mientras que ϕ y θ denotan a las redes principales, ϕ' y θ' denotan a las redes objetivos.

$$Q_\phi(s,a) \leftarrow Q_\phi(s,a) + \alpha\Big[r + \gamma \max_{a'} Q_{\phi'}(s',\mu_{\theta'}(s')) - Q_\phi(s,a)\Big] \tag{3.27}$$

Esto toma aspecto de actor-crítico (Figura 3.6). El actor es quien se refiere a la política, es decir, realiza las acciones; y el crítico se refiere a la función valor, por tanto, es el que evalúa los estados y las acciones de la política. De esta forma, el

DDPG utiliza cuatro redes: dos principal (actor y crítico) y dos objetivo (actor y crítico). [53][55]

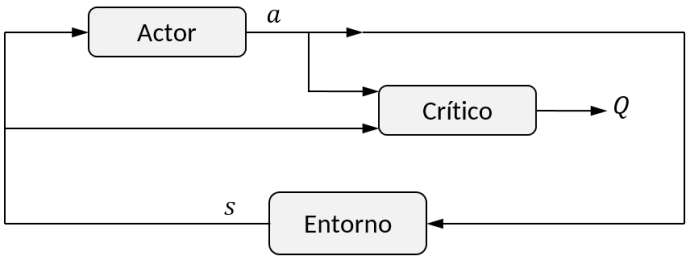


Figura 3.6: Esquema de un actor-crítico.

A continuación, se va a desarrollar el pseudocódigo de este algoritmo. [53][55]

DDPG

- Inicializar aleatoriamente la red del crítico, $Q(s, a)$, con parámetros ϕ y su objetivo con parámetros $\phi' \leftarrow \phi$
- Inicializar aleatoriamente la red del actor, $\mu(s)$, con parámetros θ y su objetivo con parámetros $\theta' \leftarrow \theta$
- Inicializar el buffer $\mathcal{D}$ vacío
- Repetir para cada episodio:
  - Inicializar el generador de ruido $\mathcal{N}$ para exploración de acciones
  - Inicializar s
  - Repetir para cada paso del episodio:
    - Elegir una acción $a = \mu_\theta(s) + \mathcal{N}$
    - Realizar la acción a
    - Observar la recompensa r y el estado siguiente s'
    - Almacenar la transición (s, a, r, s') en $\mathcal{D}$
    - Muestrear un *minibatch* aleatorio de N transiciones (s_i, a_i, r_i, s'_i) de $\mathcal{D}$
    - Actualizar el crítico minimizando la pérdida: $L = \frac{1}{N}\sum_{i=1}^{N}(r_i + \gamma Q_{\phi'}(s'_i, \mu_{\theta'}(s'_i)) - Q_\phi(s_i, a_i))^2$
    - Actualizar la política del actor con su gradiente muestreado: $\nabla_\theta J \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_a Q_\phi(s_1, \mu(s_i))\nabla_\theta \mu_\theta(s_i)$
    - Actualizar las redes objetivo $\phi' \leftarrow \tau\phi + (1-\tau)\phi'$ $\theta' \leftarrow \tau\theta + (1-\tau)\theta'$

  hasta llegar al estado terminal

Antecedentes y desarrollo actual de los drones 4

4.1. Definición y nomenclatura

En la nomenclatura de este ámbito hay distintos términos que se suelen usar como sinónimos y pueden dar lugar a confusión. En esta sección se van a comentar los que más relevancia han tenido.

El término *Vehículo Aéreo No Tripulado* (VANT) es mucho más conocido por sus siglas provenientes de su nombre en inglés, *Unmanned Aerial Vehicle* (UAV). Para explicarlo, no hay una definición única y universal, sino que cada una de las distintas organizaciones pertinentes tiene la suya con unos u otros matices. Aunque, es cierto, que todas giran en torno a unos puntos comunes que se recogen en un intento de definición propia a continuación.

Se puede decir que un UAV es cualquier vehículo motorizado y controlado en 3 ejes que vuela sin tripulación a bordo, ya sea por ser completamente autónomo, por estar programado con un plan de vuelo o porque se pilote de forma remota. Está pensado para ser recuperable o reutilizable y puede transportar cargas, tanto letales como no letales. De esta forma, se incluyen todo tipo de aviones, helicópteros y dirigibles y, por contra, se quedan fuera sistemas como los globos aerostáticos, las armas guiadas, los blancos aéreos no reutilizables o los satélites. [56][57][58][59]

El UAV es el elemento principal de un *Sistema Aéreo No Tripulado*, en inglés *Unmanned Aerial System* (UAS), que está formado por el segmento aire, el segmento tierra y el enlace de comunicaciones. [56][59][60]

- **Segmento aire**. Lo conforman los vehículos en sí, que pueden ser uno o más, incluido el sistema de comunicaciones para su control y la transmisión de datos con el que va equipado, el sistema de propulsión y la carga útil (por ejemplo, armas o cámaras).
- **Segmento tierra**. Lo constituyen la estación de control de tierra, donde está incluido el piloto, los equipos de comunicaciones para el control y la recepción, el análisis y la transmisión de información, así como las plataformas de lanzamiento y recuperación.

- **Enlace de comunicaciones**. La comunicación en tiempo real entre los dos anteriores es crucial, ya que es necesario el envío continuo de datos entre las dos partes para poder llevar a cabo una misión satisfactoria. El elemento principal es la radio ya sea para transmitir las órdenes, en un sentido, o la telemetría, en el contrario. Para ampliar el alcance de la señal se utilizan antenas, sin embargo, no siempre es posible establecer contacto directo (comunicación dentro de la línea de visión (LOS) —del inglés *Line Of Sight* [59]) entre el emisor y el receptor, debido a la propia curvatura terrestre o por la presencia de obstáculos y es necesario utilizar algún intermediario, ya sean repetidores o vía satélite (comunicación BLOS —*Beyond Line Of Sight* [59]). Un esquema se puede ver en la Figura 4.1.

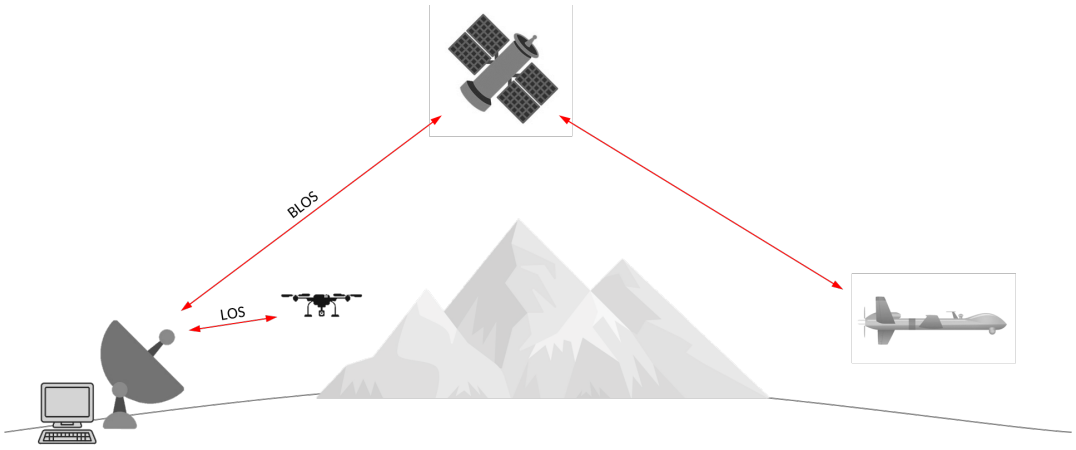


Figura 4.1: Enlace de comunicaciones. Adaptado de [60].

En el caso de que el UAV sea pilotado de forma remota, recibe el nombre concreto de RPA, por las siglas en inglés de *Remotely Piloted Aircraft*. De igual forma que se ha mencionado antes, un RPA forma parte de un RPAS, del inglés *Remotely Piloted Aircraft System*.

Por último, coloquialmente, se les denomina como *drone*, o *dron* en español. Su origen es militar, significa *zángano*, y era usado para hacer referencia a los UAV en general, aunque este término se ha visto extendido tanto a los entornos terrestre y marino como al ámbito civil, ya sea para llevar a cabo tareas profesionales o para recreo, simplemente. [56]

4.2. Clasificaciones

Decir a qué grupo pertenece un UAV es algo realmente complejo. Hay infinidad de configuraciones y prestaciones distintas de UAV destinadas a desempeñar un gran abanico de misiones, así como distintas formas de integrarse en un UAS. Además, no hay una clasificación universal reglamentada en la que situar cada modelo, más bien se han ido estableciendo diferentes tablas conforme a ciertos parámetros. [59][61]

Pero la necesidad de diseñar una clasificación que permita distinguir los sistemas existentes no se basa en algo tan banal como un antojo académico, sino que es de

gran importancia regulatoria al ser un paso crucial para el desarrollo de normativa específica de estos sistemas. Esto se debe a que, con tal amplia variedad de configuraciones, resulta difícil establecer una normativa general aplicable a todos los UAV y es más práctico normalizar según categorías dadas por las características. [59][61]

El espacio de operación es compartido con otros vehículos con regulaciones propias establecidas, así que desarrollar una normativa para los UAS se hace indispensable para poder operar con seguridad y permitir que el resto de vehículos que conviven con ellos también lo hagan. [59]

Se pueden utilizar diversos parámetros para realizar las distinciones de categorías, ya sea uno sólo o una combinación de varios. No obstante, algunos pueden ser más relevantes según lo que se quiera destacar, así como pueden estar relacionados con otros. En el caso del ámbito de la reglamentación, cabe destacar el peso máximo en el despegue (MTOW) —del inglés *Maximum Take-Off Weight*—, por ser un factor determinante en la energía cinética en el momento de un hipotético impacto con el suelo, que se usa para la estimación de muertes. Además, está íntimamente relacionado con características tan importantes como la autonomía, el alcance y la altitud operativa de vuelo. [59][61]

También pueden ser de interés normativo clasificaciones conforme a la altitud de operación, por tener influencia obvia en el riesgo de colisiones; y según la autonomía del control, desde estar pilotado remotamente a ser completamente autónomo. [61]

Como ya se ha visto, hay toda la variedad de tablas que se quiera. Algunos ejemplos conforme a los parámetros que se han mencionado se recogen en [61]. Una clasificación común conforme a algunos de estos parámetros que se han mencionado es la que se muestra en la Tabla 4.1.

Si bien es cierto que la Organización del Tratado del Atlántico Norte (OTAN), ha visto la necesidad de unificar el lenguaje para facilitar la estandarización y las labores de cooperación en el ámbito militar. De esta forma, propuso una clasificación (Tabla 4.2) basada en el MTOW y otros parámetros del perfil de vuelo (altitud operativa y alcance), así como su empleo. Esto último se refiere al nivel en el que prestan servicio que corresponden a los tres niveles de guerra, en orden creciente de complejidad de la contienda: táctico (varios niveles de jerarquía), operacional y estratégico [62].

[3]Alcance cercano —*Close Range*
[4]Alcance corto —*Short Range*
[5]Alcance medio —*Medium Range*
[6]Alcance medio autonomía —*Medium Range Endurance* / Autonomía polivalente —*Multi Role Endurance*. Combina el medio alcance con gran autonomía. [**?**]
[7]Baja altitud penetración profunda —*Low Altitude Deep Penetration*
[8]Baja altitud gran autonomía —*Low Altitude Long Endurance*
[9]Media altitud gran autonomía —*Medium Altitude Long Endurance*
[10]Alta altitud gran autonomía —*High Altitude Long Endurance*
[11]Estratosférico —*Stratospheric*
[12]Exo-estratosférico —*Exo-stratospheric*
[13]UAV de Combate —*Unmanned Combat Aerial Vehicle*
[14]Ofensivo —*Lethal*
[15]Señuelo —*Decoys*

Categoría	MTOW (kg)	Alcance (km)	Altitud de operación (m)	Autonomía (h)
Micro	$< 1 - < 5$	< 10	< 250	1
Mini	$< 20 - < 150$	< 10	< 300	< 2
CR[3]	$25 - 150$	$10 - 30$	3,000	$2 - 4$
SR[4]	$50 - 250$	$30 - 70$	3,000	$3 - 6$
MR[5]	$150 - 500$	$70 - 200$	5,000	$6 - 10$
MRE[6]	$500 - 1{,}500$	> 500	8,000	$10 - 18$
LADP[7]	350	> 250	$50 - 9{,}000$	$0{,}5 - 1$
LALE[8]	< 30	> 500	3,000	> 24
MALE[9]	1,500	> 500	14,000	$24 - 48$
HALE[10]	$2{,}500 - 15{,}000$	$> 2{,}000$	20,000	$24 - 48$
Strato[11]	$2{,}500 - 3{,}000$	$> 2{,}000$	$20{,}000 - 30{,}000$	> 48
EXO[12]	Por deter.	Por deter.	$> 30{,}000$	Por deter.
UCAV[13]	10,000	1,500	$10{,}000 - 12{,}000$	2
LET[14]	250	300	4,000	$3 - 4$
DEC[15]	250	$0 - 500$	$50 - 5{,}000$	< 4

Tabla 4.1: Clasificación general. [61]

Clase (MTOW)	Categoría	Empleo	Altitud de operación AGL[16]	Radio de misión
Clase I < 150 kg	Micro < 2 kg	Táctico (Sección)	< 200 ft	5 km (LOS)
	Mini 2 - 20 kg	Táctico (Compañía)	$< 3{,}000$ ft	25 km (LOS)
	Pequeño > 20 kg	Táctico (Batallón)	$< 5{,}000$ ft	50 km (LOS)
Clase II 150 - 600 kg	Táctico	Táctico (Brigada)	$< 10{,}000$ ft	200 km (LOS)
Clase III > 600 kg	MALE	Operacional	$< 45{,}000$ ft	Sin límite (BLOS)
	HALE	Estratégico	$< 65{,}000$ ft	Sin límite (BLOS)
	Combate	Estratégico	$< 65{,}000$ ft	Sin límite (BLOS)

Tabla 4.2: Clasificación de los UAV dada por la OTAN. [59][61]

Otra forma muy común y visual es la clasificación según su apariencia, que condiciona a la forma de generar sustentación y a la forma de despegar y aterrizar. Se distinguen dos grupos principales más extendidos en los que se hará hincapié: drones de ala fija y drones de ala rotatoria, y otras configuraciones menos estudiadas, que se mencionarán. A su vez, dentro de estas clases se pueden encontrar estructuras de diversos tamaños y características. [58][63][64]

Drones de ala fija. Tienen una fisonomía basada en los aviones convencionales, con todas sus variedades de formas y posiciones de alas y cola. De esta forma, el principio de sustentación es el mismo, esto es, su aerodinámica aprovecha la velocidad y el ángulo de incidencia del aire sobre la aeronave para generar la fuerza de sustentación y mantenerse en vuelo, a la vez que las superficies del timón de dirección, los alerones y el timón de profundidad controlan la orientación de la aeronave. Lo mismo ocurre con la elección del tipo y posición de la planta propulsora, que puede ser tuborreactor o turbohélice, según la velocidad de operación. Como es de esperar, su manera de despegar y aterrizar es horizontal —en inglés *Horizontal Take-Off and Landing* (HTOL)—, por lo que necesitan que se les imprima una velocidad inicial. Tanto el impulso de despegue como para la maniobra de recuperación se pueden hacer de varias formas, dependiendo de las características de cada UAV. Se puede alzar el vuelo mediante el impulso de la propia planta motora en pista, de una catapulta o de una persona que lo lance; y se puede aterrizar sobre la pista de forma convencional, sobre el fuselaje o siendo atrapados por una red. Como carencia tienen que no pueden mantener el vuelo sobre un punto fijo, sin embargo, destacan por alcanzar grandes velocidades y tener una gran autonomía.

(a) Sitaria, UAVOS Inc. [65]

(b) MQ-4C Triton, Northrop Grumman. [66]

Figura 4.2: Ejemplos de drones de ala fija.

Drones de ala rotatoria. Siguen el mismo principio de sustentación que los helicópteros, de manera que se mantienen en el aire gracias a la acción de la fuerza de sustentación generada por las hélices al girar que compensan la fuerza de la gravedad. Los modelos son variados según el número de rotores que tengan, desde configuraciones con un solo rotor, similares a los helicópteros convencionales que deben contar con algún mecanismo anti-par, hasta sistemas multirrotores con una estructura que recuerda a una araña en los que el número de brazos suele variar entre 3 y 8, aunque hay más variaciones. También hay variaciones donde se pueden encontrar configuraciones coaxiales en los que cada punto tiene dos motores cuyas

[16]Sobre el nivel del suelo —del inglés *Above Ground Level*—. Es la altura real sobre un punto.

hélices giran en sentidos opuestos. Todos ellos aterrizan y despegan verticalmente —en inglés *Vertical Take-Off and Landing* (VTOL)— sin necesidad de ninguna ayuda externa, ya que la planta motriz hace las funciones de elevación y propulsión. Al contrario que los anteriores, estos UAV pueden realizar vuelos estáticos (*hover*) y realizar cambios de dirección con agilidad, en cambio, su punto débil reside en el gran consumo que limita la autonomía.

(a) MQ-8B Fire Scout, Northrop Grumman. [67]

(b) Tornado H920, Yuneec. [68]

Figura 4.3: Ejemplos de drones de ala rotatoria.

Además de estas, hay otras configuraciones que no son tan populares por el momento debido, sobre todo, a su complejidad. Por un lado, se sitúan los drones híbridos, que combinan ser VTOL con la capacidad de propulsión horizontal. Y, por otro lado, se encuentran los drones de alas batientes, que son configuraciones bio-inspiradas en los movimientos de aleteo de las alas flexibles y ligeras de los insectos y las aves. Para conocer más, ver [69].

Sin duda son los cuadricópteros los más famosos y más extendidos entre la población gracias a su fácil manejo y su precio asequible. En este trabajo es el que se va a desarrollar más adelante.

4.3. Aplicaciones

Hay gran cantidad de ámbitos en los que utilizar los drones que, cada vez más, se están extendiendo, de hecho es difícil pensar tareas dónde no serían útiles. Se van a comentar algunas aplicaciones a continuación. [69][70][71]

En defensa y seguridad civil se suelen utilizar para tareas de reconocimiento y vigilancia fronteriza y de multitudes. También, se suelen enviar de avanzadilla en situaciones de riesgo, como son los desastres naturales (terreomotos, erupciones volcánicas o avalanchas) para evaluar el estado de la situación o a lugares difíciles de acceder donde haya que realizar rescates.

En el entorno agrícola y forestal se suelen usar para hacer el seguimiento del ganado y la monitorización de los campos para detectar posibles plagas. De la misma forma, permiten detectar posibles focos de incendio.

También sirven de apoyo en la supervisión de obras e inspecciones de alto riesgo, como podría ser una refinería o el tendido eléctrico. En el caso de las telecomunicaciones, con drones se pueden detectar problemas de cobertura a la vez que pueden emplearse como estaciones repetidoras para proporcionar cobertura en zonas de difícil acceso o en caso de emergencias.

Como todo, tiene su parte de ocio. Estas aeronaves son cada vez más populares gracias a las fotografías y vídeos que son capaces de tomar, por lo que son muy utilizados en televisión. Son destacables, también, las competiciones de carreras con ellos.

5 Cuadricóptero

Para este trabajo se ha decidido utilizar un dron cuadricóptero, ya que es el más extendido para uso civil por ser el más económico y más sencillo de utilizar. Es por eso que en este capítulo se va a profundizar en este tipo concreto.

5.1. Descripción física

Un dron cuadricóptero consiste, básicamente, en un cuerpo central del que salen 4 brazos de forma simétrica cada cual con un motor y su correspondiente hélice en el extremo, de forma que quedan equidistantes del centro y con sus ejes fijos y paralelos. Con esta simetría y el grueso del peso concentrado en el cuerpo se tiene un sistema bien equilibrado. [72]

Figura 5.1: Ejemplo de dron cuadricóptero comercial: *Mavic Air 2*. [73]

Las fuerzas que actúan sobre este tipo de aeronaves son: el peso (W), el empuje horizontal (T), la sustentación o empuje vertical (L) y la resistencia (D). Mientras el peso y la resistencia son fruto de la mera presencia del vehículo en el medio, las otras dos están generadas a propósito y son fruto de descomponer la fuerza (F) generada por las hélices (Figura 5.2). Como se mencionó en el capítulo anterior, estos multirrotores generan sustentación mediante sus hélices al acelerar el aire cuando pasa a través de ellas perpendicularmente, entrando por la parte superior y saliendo

por la inferior [72]. Según la 2ª Ley de Newton[17], esta aceleración lleva consigo una fuerza y, conforme a la 3ª Ley de Newton o Principio de acción-reacción[18], se genera una fuerza de igual magnitud y dirección pero de sentido opuesto sobre el vehículo que permite su vuelo y que es lo que se ha denominado antes fuerza F. Dichas fuerzas y los momentos, a los que dan lugar cuando no están equilibradas, son quienes rigen el movimiento del aparato.

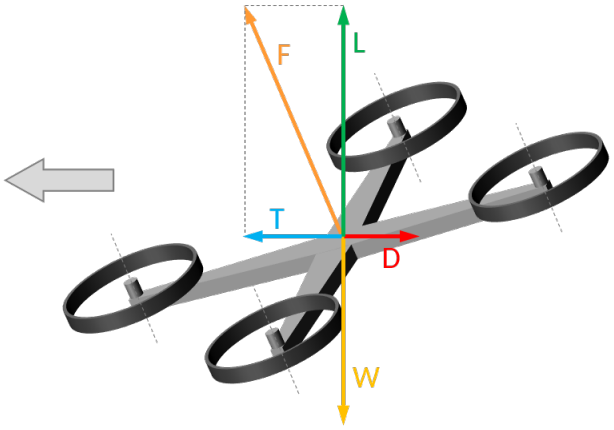


Figura 5.2: Esquema de las fuerzas que actúan sobre el cuadricóptero.[19]

Pero hay que tener cuidado, todas las hélices no pueden girar en el mismo sentido, ya que producirían un momento que haría girar al dron en sentido contrario como respuesta. Esta es la razón del rotor de cola de los helicópteros. Para solventar el problema en este caso, gracias a la simetría, basta con que las hélices giren en sentidos horario y antihorario 2 y 2, de forma que se contrarresten entre ellas los momentos y sólo queden las fuerzas perpendiculares. Las hélices son de paso fijo y están montadas de tal manera que todas generen una fuerza hacia arriba independientemente de su sentido de giro.

Todo cuerpo en el espacio lleva ligado un sistema de referencia móvil y se percibe que está en movimiento cuando es observado desde un sistema de referencia fijo y se ve que cambia de posición y orientación con respecto a él. El cuadricóptero, tratado como un sólido rígido, no es menos y tiene 6 grados de libertad: 3 de posición (traslación a lo largo de los ejes) y 3 de orientación en el espacio (rotación alrededor de los ejes) [74][75]. Sin embargo, su control se realiza mediante 4 entradas solamente, que corresponden a la fuerza proporcionada por cada uno de los 4 rotores, dadas por sus velocidades de giro. Por lo que el sistema está sub-actuado. Esto es posible porque los movimientos de traslación y rotación están acoplados, ya que la modificación de la velocidad de uno de los rotores da lugar a un movimiento en 3 grados de libertad. Esto se puede comprobar recordando la Figura 5.2 donde se ve que para que se produzca un movimiento de traslación es necesaria la inclinación de la plataforma. Y en la Figura 5.3 se observa cómo varía el reparto de la fuerza F entre la sustentación y el empuje horizontal según dicha inclinación. [76]

[17] $\vec{F} = m\vec{a}$

[18] $\vec{F_{AB}} = -\vec{F_{BA}}$

[19] La nomenclatura de esta imagen es simplemente esquemática y no se corresponde completamente con la utilizada en el posterior desarrollo matemático.

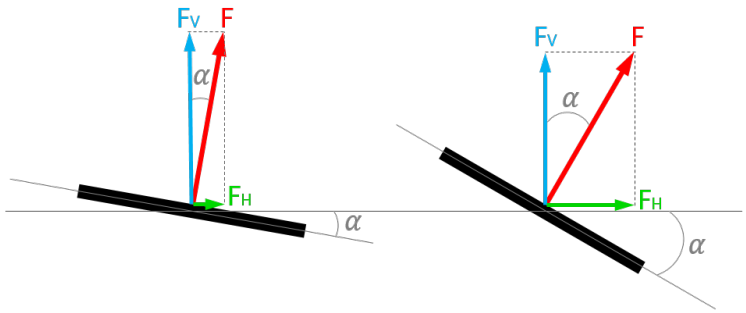


Figura 5.3: Variación de la magnitud de la descomposición en fuerzas horizontal y vertical de la fuerza total generada en función del ángulo.

En aeronáutica, los 3 grados de libertad correspondientes a la orientación se denominan: alabeo o balanceo —*roll*— (en torno al eje x), cabeceo —*pitch*— (en torno al eje y) y guiñada —*yaw*— (en torno al eje z) [74][75]. En la Figura 5.4 se pueden ver cuándo estas rotaciones son positivas sobre un avión, ya que es una configuración más familiar para todos y más fácil de observar al ser simétrica respecto a un eje solamente, pero totalmente aplicable a los cuadricópteros.

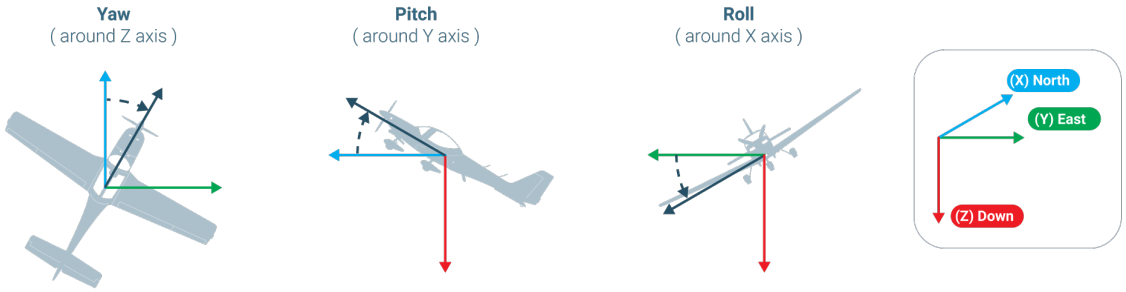


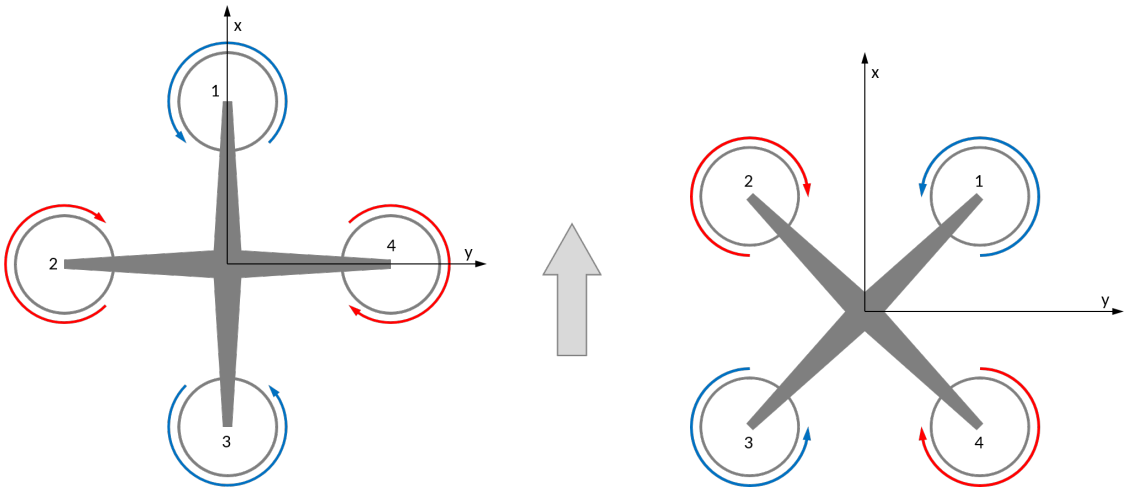
Figura 5.4: Grados de libertad de orientación sobre un avión. [77]

5.1.1. Configuraciones

Los cuadricópteros se pueden presentar en dos configuraciones principales: configuración en cruz (+) o configuración en equis (×), según sea la orientación de los brazos en relación al sistema de ejes del vehículo, basado en la dirección principal de avance. Ambas configuraciones se muestran en la Figura 5.5. [76]

Como se ve, en la configuración en cruz cada par de brazos coincide con un eje de coordenadas, habiendo un rotor frontal y uno trasero. Mientras que en la configuración en equis los brazos están rotados 45° respecto de los ejes, quedando dos rotores delanteros y dos traseros.

Comparando ambas configuraciones, para un mismo movimiento (a lo largo de los ejes x o y), la configuración en cruz sólo requiere modificar la actuación de un rotor, mientras que en la configuración en equis intervienen dos rotores en cada movimiento. Sin embargo, esta complejidad se traduce en mayor potencia y menor agresividad en las hélices, ya que el trabajo que realiza un solo rotor en cruz se reparte entre dos rotores en equis. [76]



(a) Configuración +. **(b)** Configuración ×.

Figura 5.5: Configuraciones más comunes de un cuadricóptero orientadas según la dirección de avance.

5.2. Movimientos

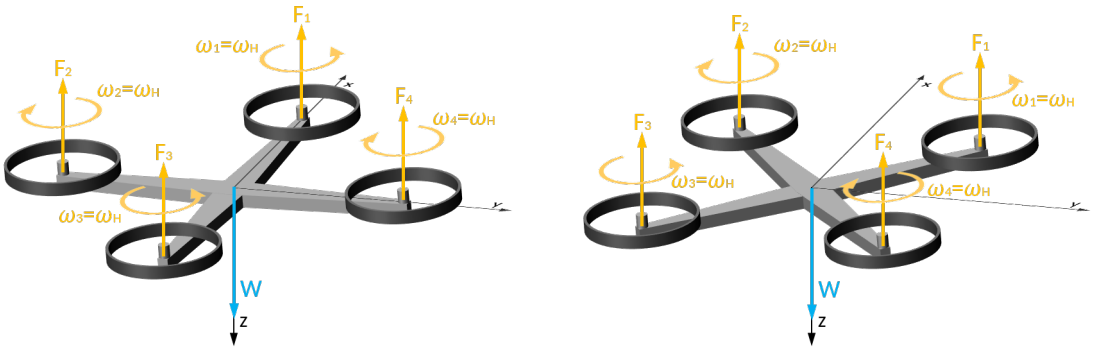
Como se ha mencionado antes, el cuadricóptero conforma un sistema de 6 grados de libertad controlado por 4 variables de entrada. De esta forma, no se puede asociar el control de cada variable a un movimiento, sino que hace falta la colaboración de las 4 variables para realizar los movimientos en los distintos grados de libertad. Se pueden distinguir 4 estados independientes, es decir, 4 combinaciones de control de las entradas que den lugar a distintos movimientos.

Estos movimientos principales se van a describir en lo que sigue en base a [72] y [78]. Junto a ellos, conviene mencionar el estado de equilibrio del que parte el dron en todos los casos, correspondiente al estado de vuelo estático o *hovering*.

Hovering. Corresponde al estado de equilibrio en el que el cuadricóptero está realizando un vuelo estático en posición horizontal. Las hélices realizan un trabajo equitativo para que no aparezca ningún momento. Así, todas giran a la misma velocidad (Ω_H) y generan la misma fuerza, la suficiente para que su suma compense al peso. Un esquema del estado se muestra en la Figura 5.6.

Existe equilibrio de fuerzas en magnitud y dirección, de sus momentos asociados y de las velocidades de giro de las hélices.

$$\sum \vec{F_x} = 0 \qquad \sum \vec{F_y} = 0$$
$$\sum \vec{F_z} = 0 \;\rightarrow\; F_1 + F_2 + F_3 + F_4 = mg$$
$$\sum \vec{M_x} = 0 \;\rightarrow\; (+)\; F_2 = F_4 \quad (\times)\; F_2 + F_3 = F_1 + F_4$$
$$\sum \vec{M_y} = 0 \;\rightarrow\; (+)\; F_1 = F_3 \quad (\times)\; F_1 + F_2 = F_3 + F_4$$
$$\sum \vec{\Omega} = 0 \;\rightarrow\; \Omega_1 + \Omega_3 = \Omega_2 + \Omega_4$$



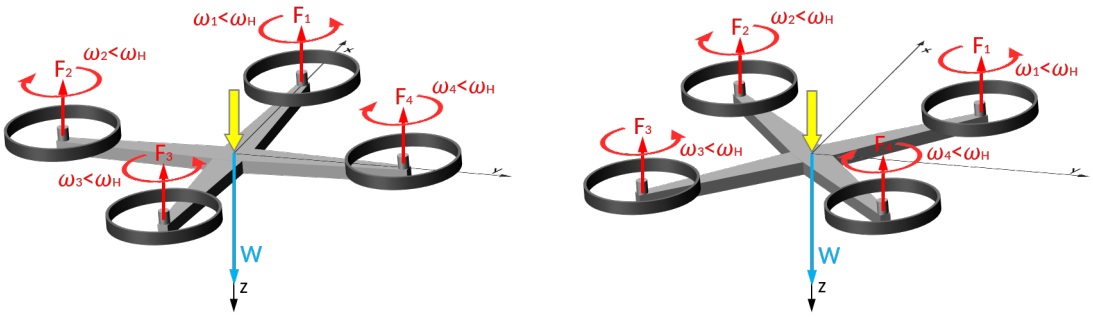
(a) *Hovering* en configuración +. **(b)** *Hovering* en configuración ×.

Figura 5.6: Esquema del estado de *hovering*.

Empuje. Es una modificación del caso de equilibrio en la que el cuadricóptero se mantiene en posición horizontal mientras realiza un movimiento vertical a lo largo del eje z, ya sea de descenso o ascenso. El único cambio respecto al equilibrio es que el sumatorio de fuerzas no es nulo, es decir, que la fuerza generada no llega a compensar al peso o, por el contrario, lo sobrecompensa, y el dron desciende o asciende, respectivamente. Esto se debe a que la velocidad de giro de las hélices ha variado respecto de la de *hovering*. Ambos casos se ven en las Figuras 5.7 y 5.8.

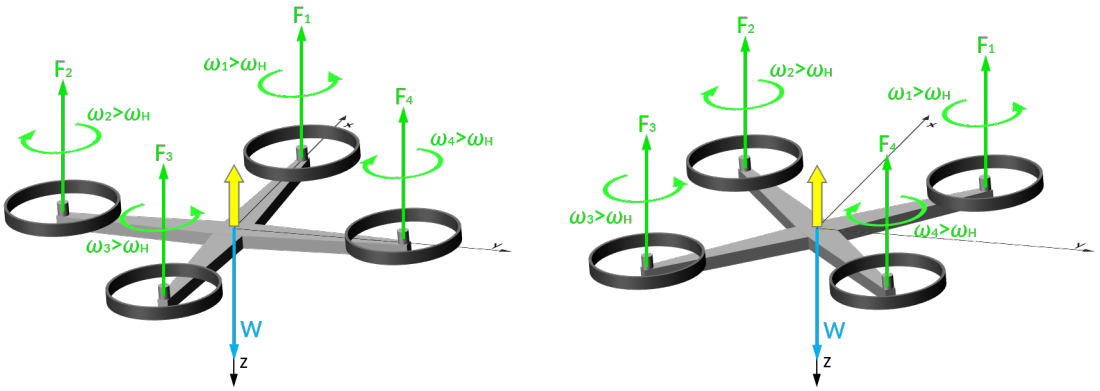
Hay equilibrio de los momentos asociados a las fuerzas y de las velocidades de giro de las hélices, pero no de fuerzas en magnitud, aunque siguen manteniendo la misma dirección.

$$\sum \vec{F}_x = 0 \qquad \sum \vec{F}_y = 0$$
$$\sum \vec{F}_z \neq 0 \;\rightarrow\; F_1 + F_2 + F_3 + F_4 < mg \text{ (descenso)}$$
$$\rightarrow\; F_1 + F_2 + F_3 + F_4 > mg \text{ (ascenso)}$$
$$\sum \vec{M}_x = 0 \;\rightarrow\; (+)\; F_2 = F_4 \quad (\times)\; F_2 + F_3 = F_1 + F_4$$
$$\sum \vec{M}_y = 0 \;\rightarrow\; (+)\; F_1 = F_3 \quad (\times)\; F_1 + F_2 = F_3 + F_4$$
$$\sum \vec{\Omega} = 0 \;\rightarrow\; \Omega_1 + \Omega_3 = \Omega_2 + \Omega_4$$



(a) Descenso en configuración +. **(b)** Descenso en configuración ×.

Figura 5.7: Esquema del movimiento de descenso.



(a) Ascenso en configuración +. **(b)** Ascenso en configuración ×.

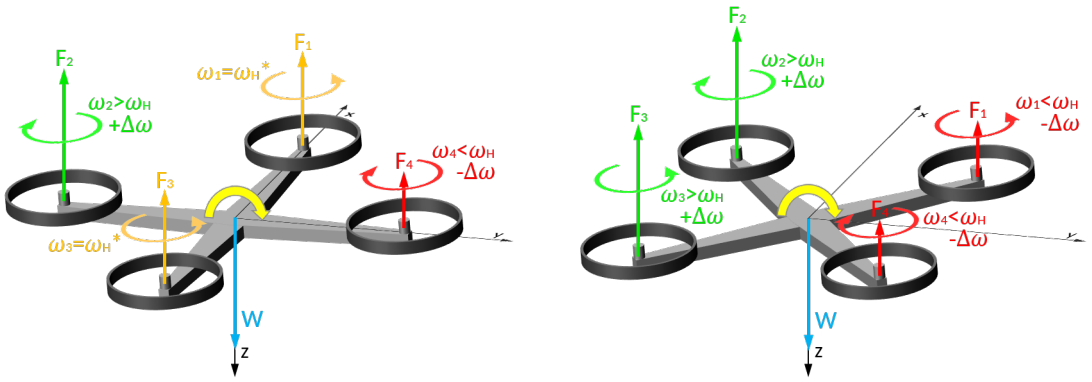
Figura 5.8: Esquema del movimiento de ascenso.

Alabeo. Es una variación del estado de equilibrio en la que el cuadricóptero adopta una posición inclinada debido a un giro alrededor del eje x que le permite realizar un movimiento de traslación horizontal lateral a lo largo del eje y. Esto se logra por una descompensación de las fuerzas imprimidas por los rotores a un lado y a otro del eje de giro, que da lugar a un momento alrededor del eje x. Solamente si la variación de la fuerza en ambos lados es igual en magnitud pero de signo distinto, es decir, que la velocidad en unos rotores incrementa lo mismo que desciende en los otros ($\Delta\Omega = \Omega_i - \Omega_H$), no hay cambio de altitud (movimiento a lo largo del eje z). Hay que tener en cuenta que, como la plataforma está inclinada, la fuerza generada necesaria para contrarrestar el peso será mayor que en el estado de *hovering*[20]. Como se ve, la diferencia que tendrán con las velocidades del estado de equilibrio dependerán de la inclinación, aunque para ángulos pequeños será aproximadamente igual, por eso se han denotado con asterisco (Ω_H^*). En la Figura 5.9 se establece el esquema del movimiento de alabeo positivo para las dos configuraciones.

Aunque no hay paralelismo entre las fuerzas, existe equilibrio en magnitud y dirección entre la componente vertical de la fuerza total generada y el peso. También existe equilibrio de momentos en torno al eje y y de las velocidades de giro de las hélices, ya que las sumas de las que giran en un sentido y otro son iguales. Sin embargo, no hay equilibrio de momentos en torno al eje x.

$$\sum \vec{F}_x = 0 \qquad \sum \vec{F}_y \neq 0$$
$$\sum \vec{F}_z = 0 \;\rightarrow\; (F_1 + F_2 + F_3 + F_4)\cos\phi = mg$$
$$\sum \vec{M}_x \neq 0 \;\rightarrow\; (+)\; F_2 > F_4 \quad (\times)\; F_2 + F_3 > F_1 + F_4$$
$$\sum \vec{M}_y = 0 \;\rightarrow\; (+)\; F_1 = F_3 \quad (\times)\; F_1 + F_2 = F_3 + F_4$$
$$\sum \vec{\Omega} = 0 \;\rightarrow\; \Omega_1 + \Omega_3 = \Omega_2 + \Omega_4$$

[20]Recordar la Figura 5.3.



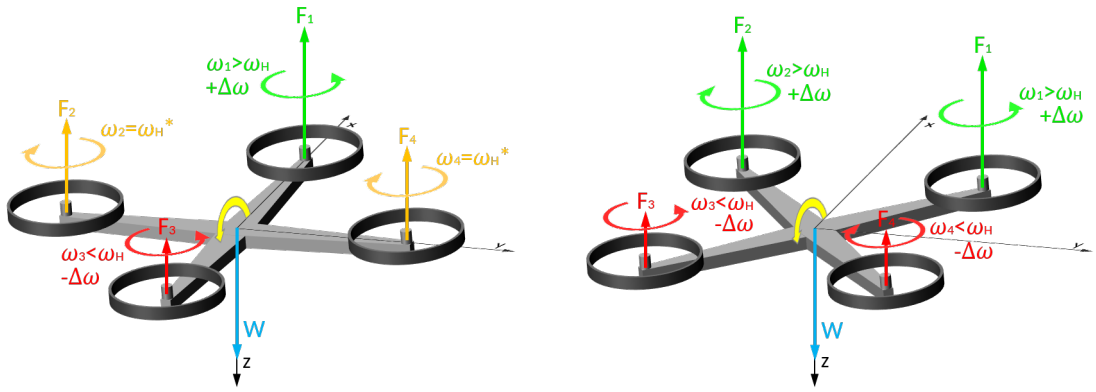
(a) Alabeo en configuración +. **(b)** Alabeo en configuración ×.

Figura 5.9: Esquema del movimiento de alabeo.

Cabeceo. Consiste en la misma mecánica que el alabeo. Pero esta vez el giro se produce alrededor del eje y, lo que le permite realizar un movimiento de traslación horizontal hacia delante o atrás a lo largo del eje x. En la Figura 5.10 se muestra el esquema del movimiento del cabeceo positivo en las dos configuraciones.

A pesar de no haber paralelismo entre las fuerzas, se produce equilibrio en magnitud y dirección entre la componente vertical de la fuerza total generada y el peso. También se tiene compensación de momentos en torno al eje x y de las velocidades de giro de las hélices, ya que las sumas de las que giran en un sentido y otro son iguales. Sin embargo, no hay equilibrio de momentos en torno al eje y.

$$\sum \vec{F}_x \neq 0 \qquad \sum \vec{F}_y = 0$$
$$\sum \vec{F}_z = 0 \;\rightarrow\; (F_1 + F_2 + F_3 + F_4)\cos\theta = mg$$
$$\sum \vec{M}_x = 0 \;\rightarrow\; (+)\; F_2 = F_4 \quad (\times)\; F_2 + F_3 = F_1 + F_4$$
$$\sum \vec{M}_y \neq 0 \;\rightarrow\; (+)\; F_1 > F_3 \quad (\times)\; F_1 + F_2 > F_3 + F_4$$
$$\sum \vec{\Omega} = 0 \;\rightarrow\; \Omega_1 + \Omega_3 = \Omega_2 + \Omega_4$$



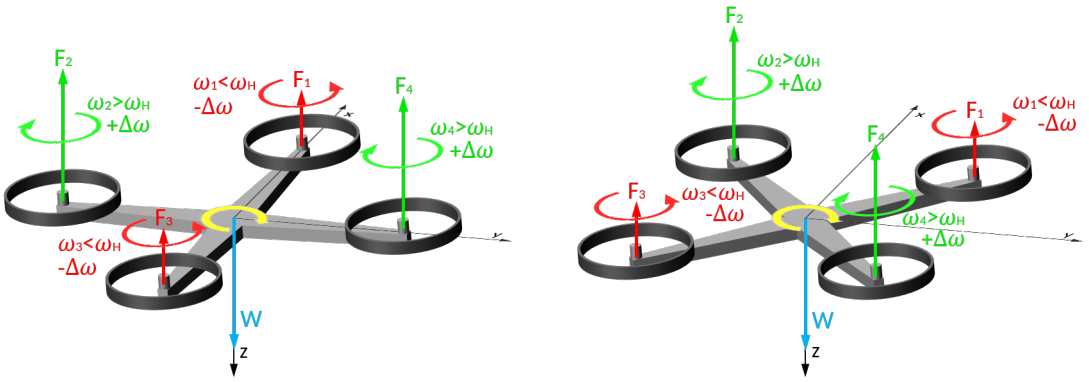
(a) Cabeceo en configuración +. **(b)** Cabeceo en configuración ×.

Figura 5.10: Esquema del movimiento de cabeceo.

Guiñada. Se trata de una modificación del caso de equilibrio en la que el cuadricóptero se mantiene en posición horizontal a la vez que ejecuta un movimiento de rotación en el plano xy en torno al eje z. Esto ocurre porque la velocidad de giro de los rotores en un sentido y en otro no es la misma, no obstante, gracias a su configuración simétrica, no se producen diferencias de momentos alrededor de los otros ejes que inclinen la plataforma.

Se da equilibrio de las fuerzas en magnitud y dirección, así como de los momentos asociados a ellas. Pero no lo hay de las velocidades de giro de las hélices.

$$\sum \vec{F}_x = 0 \qquad \sum \vec{F}_y = 0$$
$$\sum \vec{F}_z = 0 \;\rightarrow\; (F_1 + F_2 + F_3 + F_4) = mg$$
$$\sum \vec{M}_x = 0 \;\rightarrow\; (+)\; F_2 = F_4 \quad (\times)\; F_2 + F_3 = F_1 + F_4$$
$$\sum \vec{M}_y = 0 \;\rightarrow\; (+)\; F_1 = F_3 \quad (\times)\; F_1 + F_2 = F_3 + F_4$$
$$\sum \vec{\Omega} = 0 \;\rightarrow\; \Omega_1 + \Omega_3 < \Omega_2 + \Omega_4$$



(a) Guiñada en configuración +. **(b)** Guiñada en configuración ×.

Figura 5.11: Esquema del movimiento de guiñada.

5.3. Modelo matemático

Como ya se ha visto, las aeronaves en general, y por tanto los drones, son sistemas definidos por 6 grados de libertad que dan su posición y orientación en el espacio. Para obtener las expresiones matemáticas que los gobiernan es necesario tener en cuenta que hay distintos sistemas de referencia, según sea la posición del observador, sobre los que proyectar las fuerzas y momentos, posiciones y rotaciones, y velocidades y aceleraciones tanto lineales como angulares. [79]

5.3.1. Sistemas de referencia

Se pueden definir infinidad de sistemas de referencia, todos ellos pertenecientes a dos categorías: inercial o no inercial. Los sistemas de referencia inerciales son

sistemas fijos o con movimiento rectilíneo uniforme en los que se cumplen las Leyes de Newton. Por su parte, los sistemas de referencia no inerciales son aquellos que están sometidos a una aceleración y en los que para poder explicar los movimientos y aplicar las Leyes de Newton hay que recurrir a fuerzas ficticias. [79][80]

A continuación, se describen algunos sistemas de referencia útiles [79] para el desarrollo y comprensión de este trabajo. Estos se pueden ver en la Figura 5.12.

Sistema de ejes tierra (E). Tiene su origen fijado en un punto de la superficie terrestre definido por su longitud y su latitud, sus ejes x_E e y_E están contenidos en un plano horizontal apuntando al Norte y al Este, respectivamente, y completa el triedro el eje z_E apuntando hacia el centro de la Tierra.

Sistema de ejes horizonte local (H). Con origen en un punto del plano de simetría de la aeronave, normalmente su centro de masas, sus ejes son paralelos a los del sistema de ejes tierra.

Sistema de ejes cuerpo (B). Su origen se sitúa en un punto del plano de simetría de la aeronave, normalmente en su centro de masas, y sus ejes se mantienen fijos. x_B y z_B están contenidos en el plano de simetría del avión apuntando hacia delante y abajo, respectivamente y el eje y_B completa el triedro apuntando hacia la derecha.

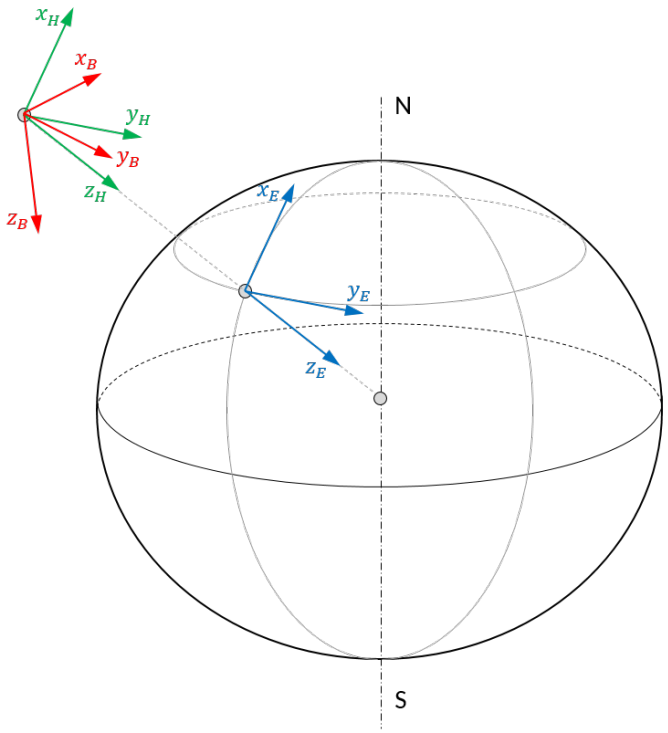


Figura 5.12: Sistemas de referencia.

5.3.2. Ángulos de Euler

Una vez que se han presentado los distintos sistemas de referencia con los que se va a trabajar, es necesario encontrar una relación que permita transformar las

coordenadas entre ellos y, así, poder expresarlas en el sistema que sea más conveniente en cada momento. Para ubicar un sistema de referencia respecto a otro con orígenes comunes hay que hacer una rotación hasta que coincidan los ejes de ambos. Si, además, los orígenes no coinciden hay que realizar una traslación hasta hacerlos coincidir. En este caso, los ejes cuerpo y los ejes tierra no tienen un origen común, sin embargo, para relacionar la rotación entre ellos se recurre a los ejes de horizonte local, que sí tienen su origen común al de los ejes cuerpo. [79][81]

Existen distintos métodos para orientar entre sí dos sistemas de referencia con origen común. Aquí se verán los Ángulos de Euler. Este método consiste en 3 rotaciones sucesivas finitas dadas en un orden especificado (las rotaciones finitas no son magnitudes vectoriales y por tanto, no son conmutativas). Este orden riguroso se conoce como convención zyx, 321 o Tait-Bryan (Figura 5.13) y es universalmente utilizado para orientar aeronaves y vehículos espaciales. Dichas rotaciones son: rotación alrededor del eje z (δ_3), rotación alrededor del eje y (δ_2) y rotación alrededor del eje x (δ_1). [79][81]

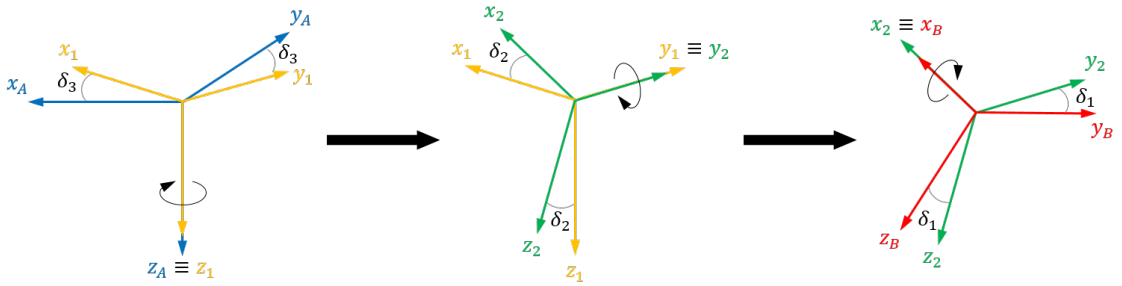


Figura 5.13: Rotación Tait-Bryan.

De forma genérica, se obtienen las siguientes matrices de rotación intermedias y la final, cuya notación R_{ab} indica la matriz de rotación de un sistema a a otro b y donde c y s representan el coseno y el seno, respectivamente. [79][81]

$$\mathrm{R}_{a1} = \mathrm{R}(\delta_3) = \begin{pmatrix} \cos\delta_3 & \operatorname{sen}\delta_3 & 0 \\ -\operatorname{sen}\delta_3 & \cos\delta_3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.1}$$

$$\mathrm{R}_{12} = \mathrm{R}(\delta_2) = \begin{pmatrix} \cos\delta_2 & 0 & -\operatorname{sen}\delta_2 \\ 0 & 1 & 0 \\ \operatorname{sen}\delta_2 & 0 & \cos\delta_2 \end{pmatrix} \tag{5.2}$$

$$\mathrm{R}_{2b} = \mathrm{R}(\delta_1) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\delta_1 & \operatorname{sen}\delta_1 \\ 0 & -\operatorname{sen}\delta_1 & \cos\delta_1 \end{pmatrix} \tag{5.3}$$

$$\mathrm{R}_{ab} = \mathrm{R}(\delta_3)\mathrm{R}(\delta_2)\mathrm{R}(\delta_1) = \begin{pmatrix} \mathrm{c}\delta_2\mathrm{c}\delta_3 & \mathrm{c}\delta_2\mathrm{s}\delta_3 & -\mathrm{s}\delta_2 \\ \mathrm{s}\delta_1\mathrm{s}\delta_2\mathrm{c}\delta_3 - \mathrm{c}\delta_1\mathrm{s}\delta_3 & \mathrm{s}\delta_1\mathrm{s}\delta_2\mathrm{s}\delta_3 + \mathrm{c}\delta_1\mathrm{c}\delta_3 & \mathrm{s}\delta_1\mathrm{c}\delta_2 \\ \mathrm{c}\delta_1\mathrm{s}\delta_2\mathrm{c}\delta_3 + \mathrm{s}\delta_1\mathrm{s}\delta_3 & \mathrm{c}\delta_1\mathrm{s}\delta_2\mathrm{s}\delta_3 - \mathrm{s}\delta_1\mathrm{c}\delta_3 & \mathrm{c}\delta_1\mathrm{c}\delta_2 \end{pmatrix} \tag{5.4}$$

En este caso, como se vio en la Figura 5.4, δ_1 es el ángulo de balanceo ϕ ($-\pi \leq \phi < \pi$), δ_2 es el ángulo de cabeceo θ ($-\pi/2 \leq \theta \leq \pi/2$) y δ_3 es el ángulo de guiñada ψ ($0 \leq \psi < 2\pi$), quedando la matriz de rotación ejes horizonte local a ejes cuerpo de la siguiente forma: [79][81]

$$\mathrm{R}_{HB} = \mathrm{R}_{EB} = \begin{pmatrix} \mathrm{c}\theta\mathrm{c}\psi & \mathrm{c}\theta\mathrm{s}\psi & -\mathrm{s}\theta \\ \mathrm{s}\phi\mathrm{s}\theta\mathrm{c}\psi - \mathrm{c}\phi\mathrm{s}\psi & \mathrm{s}\phi\mathrm{s}\theta\mathrm{s}\psi + \mathrm{c}\phi\mathrm{c}\psi & \mathrm{s}\phi\mathrm{c}\theta \\ \mathrm{c}\phi\mathrm{s}\theta\mathrm{c}\psi + \mathrm{s}\phi\mathrm{s}\psi & \mathrm{c}\phi\mathrm{s}\theta\mathrm{s}\psi - \mathrm{s}\phi\mathrm{c}\psi & \mathrm{c}\phi\mathrm{c}\theta \end{pmatrix} \tag{5.5}$$

Las matrices de rotación entre sistemas son ortogonales, por tanto, cumplen que:

$$\mathrm{R}_{ab} = \mathrm{R}_{ba}^{-1} = \mathrm{R}_{ba}^{\mathrm{T}}$$

$$\det(\mathrm{R}_{ab}) = 1$$

Así, para pasar de ejes cuerpo a ejes tierra, se tiene la siguiente matriz:

$$\mathrm{R}_{BH} = \mathrm{R}_{BE} = \mathrm{R}_{HB}^{-1} = \begin{pmatrix} \mathrm{c}\theta\mathrm{c}\psi & \mathrm{s}\phi\mathrm{s}\theta\mathrm{c}\psi - \mathrm{c}\phi\mathrm{s}\psi & \mathrm{c}\phi\mathrm{s}\theta\mathrm{c}\psi + \mathrm{s}\phi\mathrm{s}\psi \\ \mathrm{c}\theta\mathrm{s}\psi & \mathrm{s}\phi\mathrm{s}\theta\mathrm{s}\psi + \mathrm{c}\phi\mathrm{c}\psi & \mathrm{c}\phi\mathrm{s}\theta\mathrm{s}\psi - \mathrm{s}\phi\mathrm{c}\psi \\ -\mathrm{s}\theta & \mathrm{s}\phi\mathrm{c}\theta & \mathrm{c}\phi\mathrm{c}\theta \end{pmatrix} \tag{5.6}$$

Para conseguir las transformaciones de movimiento rotacional hay que relacionar las velocidades angulares locales con la variación de los ángulos de alabeo, cabeceo y guiñada en ejes tierra, ya que las tasas de variación respecto a un sistema y otro no son las mismas. Para representar cada ángulo hay que transformarlo a partir del sistema de referencia intermedio de la transformación de Euler. De esta forma, la rotación de guiñada ($\dot{\psi}$) está sujeta a tres rotaciones: alrededor de z, y, x; la de cabeceo ($\dot{\theta}$) está sujeta a dos rotaciones: alrededor de y, x; y la de alabeo ($\dot{\phi}$) está sujeta a una rotación: alrededor de x. [82]

$$\mathrm{T}_{BH} = \mathrm{T}_{BE} = \mathrm{R}(\phi)\mathrm{R}(\theta) \begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix} + \mathrm{R}(\phi) \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix} + \begin{pmatrix} \dot{\phi} \\ 0 \\ 0 \end{pmatrix} \tag{5.7}$$

De forma que se obtiene la matriz de rotación buscada y se hace su inversa para hacer la transformación en dirección contraria.

$$\mathrm{T}_{BH} = \mathrm{T}_{BE} = \begin{pmatrix} 1 & \operatorname{sen}\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\operatorname{sen}\phi \\ 0 & \operatorname{sen}\phi/\cos\theta & \cos\phi/\cos\theta \end{pmatrix} \tag{5.8}$$

$$\mathrm{T}_{HB} = \mathrm{T}_{EB} = \mathrm{T}_{BH}^{-1} = \begin{pmatrix} 1 & 0 & -\operatorname{sen}\theta \\ 0 & \cos\phi & \operatorname{sen}\phi\cos\theta \\ 0 & -\operatorname{sen}\phi & \cos\phi\cos\theta \end{pmatrix} \tag{5.9}$$

5.3.3. Dinámica

Para estudiar la dinámica de un cuadricóptero se pueden aplicar dos métodos: Newton-Euler o Euler-Lagrange. En este caso se va a utilizar el primero. Los desarrollos que se van a exponer a continuación de la dinámica traslacional y rotacional son válidos tanto para el modelo de cuadricóptero en cruz como en equis y en los pasos donde se diferencien se indicará. Antes de entrar de lleno con las ecuaciones propiamente dichas, se exponen una serie de preliminares.

5.3.3.1. Consideraciones

Antes de nada, es importante enumerar una serie de consideraciones que definen el marco del posterior desarrollo. [74]

- El cuadricóptero se trata como un sólido rígido en 3 dimensiones con 6 grados de libertad, simétrico respecto a los ejes x e y.
- Aunque se han definido los sistemas de referencia con los ejes z hacia abajo, a partir de ahora se van a rotar para que este eje tenga dirección positiva hacia arriba, ya que es más natural pensar que la altura crece positivamente a medida que se aleja del suelo.
- De acuerdo con la hipótesis de Tierra plana, el sistema de ejes tierra se puede suponer como sistema inercial. Esta hipótesis es válida para alturas de vuelo pequeñas, mucho menores que el radio de la Tierra, y para velocidades de vuelo bajas, mucho menores que las de vuelo orbital.
- Las alturas que se tratan son pequeñas en comparación con el radio de la Tierra, por tanto, la gravedad es constante.
- La masa de la aeronave no sufre ninguna variación.
- Se supone que en las hélices no existe rozamiento con sus ejes.
- No se considera el efecto de la resistencia entre el vehículo y la atmósfera ni en las hélices.
- Se hablará los sistemas de ejes tierra o de horizonte local indistintamente.

5.3.3.2. Notación

También, se requiere destacar la notación que se va a usar según el sistema de referencia en el que se trabaje. [83]

- Sistema de ejes tierra (E). A él se refieren las fuerzas y con ello las posiciones, velocidades y aceleraciones lineales.

$$\vec{\xi} = [x \ y \ z]$$
$$\vec{\eta} = [\phi \ \theta \ \psi]$$
$$\vec{q} = [\vec{\xi} \ \vec{\eta}]$$

- Sistema de ejes cuerpo (B). A él se refieren los momentos y con ello las orientaciones, velocidades y aceleraciones angulares.

$$\vec{v} = [u\ v\ w]$$
$$\vec{\omega} = [p\ q\ r]$$
$$\vec{\nu} = [\vec{v}\ \vec{\omega}]$$

5.3.3.3. Fuerzas y momentos

Dadas las consideraciones anteriores, se presentan las fuerzas y los momentos que afectan al dron. [84]

Fuerza de la gravedad (F_G). Fuerza de atracción de la Tierra dirigida en el eje z del sistema de ejes tierra, es decir, el peso.

$$F_G = mg \tag{5.10}$$

Fuerza de empuje (F_T). Fuerza generada por los rotores, está dirigida hacia arriba en el eje z de los ejes cuerpo y es proporcional a la velocidad angular de los rotores (Ω_i).

$$F_T = \sum T_i = \sum k_t \Omega_i^2 \tag{5.11}$$

Par de arrastre (M_d). Momento en el plano de la hélice de sentido contrario a su giro que surge para contrarrestar dicho giro.

$$M_d = \sum d_i = \sum k_d \Omega_i^2 \tag{5.12}$$

Momentos principales (M). Se miden en los ejes cuerpo. Son los momentos generados debido a la disposición de los rotores y las diferencias en las velocidades de giro entre unos y otros. Mientras que los momentos alrededor de los ejes x e y se generan por diferencias de empuje, el momento alrededor del eje z se debe a la diferencia en sus pares de arrastre.

5.3.3.4. Dinámica de traslación

Se basa en el teorema de cantidad de movimiento aplicado al sistema en ejes tierra, de donde se obtienen 3 ecuaciones de componente lineal. [74][83]

$$\vec{F} = \frac{\mathrm{d}\vec{p_m}}{\mathrm{d}t} = \frac{\mathrm{d}}{\mathrm{d}t}(m\dot{\vec{\xi}}) = m\frac{\mathrm{d}\dot{\vec{\xi}}}{\mathrm{d}t} = m\ddot{\vec{\xi}} \tag{5.13}$$

$$\vec{F}_G + \vec{F}_T = m\ddot{\vec{\xi}} \tag{5.14}$$

Ahí, cada término está referido a unos ejes distintos.

$$\vec{F}_G = \begin{pmatrix} 0 \\ 0 \\ -mg \end{pmatrix}_E \quad (5.15) \qquad \vec{F}_T = \begin{pmatrix} 0 \\ 0 \\ \sum T_i \end{pmatrix}_B \quad (5.16)$$

Por tanto, se traslada todo a los mismos ejes, los ejes tierra, con el cambio de ejes que se muestra y quedan las siguientes 3 ecuaciones.

$$m\ddot{\vec{\xi}} = \vec{F}_G + R_{BE}\vec{F}_T \quad (5.17)$$

$$m \begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -mg \end{pmatrix} + R_{BE} \begin{pmatrix} 0 \\ 0 \\ \sum T_i \end{pmatrix} \quad (5.18)$$

$$\ddot{x} = (\cos\psi \operatorname{sen}\theta \cos\phi + \operatorname{sen}\psi \operatorname{sen}\phi)\frac{\sum T_i}{m} \quad (5.19)$$

$$\ddot{y} = (\operatorname{sen}\psi \operatorname{sen}\theta \cos\phi - \cos\psi \operatorname{sen}\phi)\frac{\sum T_i}{m} \quad (5.20)$$

$$\ddot{z} = (\cos\theta \cos\phi)\frac{\sum T_i}{m} - g \quad (5.21)$$

5.3.3.5. Dinámica de rotación

Se fundamenta en el teorema del momento cinético aplicado al sistema en ejes cuerpo, de donde se obtienen 3 ecuaciones de componente angular. [74][83]

$$\vec{M} = \frac{\mathrm{d}\vec{h_c}}{\mathrm{d}t} + \vec{\omega} \times \vec{h_c} = m\frac{\mathrm{d}(I\vec{\omega})}{\mathrm{d}t} + \vec{\omega} \times (I\vec{\omega}) \quad (5.22)$$

Como el cuadricóptero es un cuerpo simétrico, la matriz de inercia es una matriz diagonal.

$$\mathrm{I} = \begin{pmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{pmatrix} \quad (5.23)$$

Las 3 ecuaciones de componente rotacional son:

$$\begin{pmatrix} M_x \\ M_y \\ M_z \end{pmatrix} = \begin{pmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{pmatrix} \begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} + \begin{pmatrix} p \\ q \\ r \end{pmatrix} \times \begin{pmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{pmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix} \quad (5.24)$$

$$\dot{p} = \frac{M_x}{I_{xx}} + \frac{(I_{yy} - I_{zz})}{I_{xx}} qr \tag{5.25}$$

$$\dot{q} = \frac{M_y}{I_{yy}} + \frac{(I_{zz} - I_{xx})}{I_{yy}} pr \tag{5.26}$$

$$\dot{r} = \frac{M_z}{I_{zz}} + \frac{(I_{xx} - I_{yy})}{I_{zz}} pq \tag{5.27}$$

Hasta aquí todo ha sido común a las dos configuraciones, pero en el desarrollo de las expresiones de los momentos que siguen se diferencia entre la configuración en cruz y la configuración en equis.

$$\begin{pmatrix} M_x \\ M_y \\ M_z \end{pmatrix} = \begin{pmatrix} l(T_2 - T_4) \\ l(T_1 - T_3) \\ M_d = d_1 - d_2 + d_3 - d_4 \end{pmatrix}_+ \tag{5.28}$$

$$\begin{pmatrix} M_x \\ M_y \\ M_z \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} l(-T_1 + T_2 + T_3 - T_4) \\ \frac{\sqrt{2}}{2} l(T_1 + T_2 - T_3 - T_4) \\ M_d = d_1 - d_2 + d_3 - d_4 \end{pmatrix}_\times \tag{5.29}$$

Al margen de las dos dinámicas, el empuje y los momentos generados por las hélices son las entradas de control.

$$\begin{pmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{pmatrix} = \begin{pmatrix} F_T \\ M_x \\ M_y \\ M_z \end{pmatrix} \tag{5.30}$$

Finalmente, en total, se han obtenido 6 ecuaciones pertenecientes a las 3 aceleraciones lineales en ejes tierra y las 3 aceleraciones angulares en ejes cuerpos, que quedarían de la siguiente forma:

$$\ddot{x} = (\cos\psi \operatorname{sen}\theta \cos\phi + \operatorname{sen}\psi \operatorname{sen}\phi)\frac{U_1}{m}$$

$$\ddot{y} = (\operatorname{sen}\psi \operatorname{sen}\theta \cos\phi - \cos\psi \operatorname{sen}\phi)\frac{U_1}{m}$$

$$\ddot{z} = (\cos\theta \cos\phi)\frac{U_1}{m} - g$$

$$\dot{p} = \frac{U_2}{I_{xx}} + \frac{(I_{yy} - I_{zz})}{I_{xx}} qr$$

$$\dot{q} = \frac{U_3}{I_{yy}} + \frac{(I_{zz} - I_{xx})}{I_{yy}} pr$$

$$\dot{r} = \frac{U_4}{I_{zz}} + \frac{(I_{xx} - I_{yy})}{I_{zz}} pq$$

El objetivo de todo esto es conseguir son las posiciones $[x\ y\ z]$ y las orientaciones $[\phi\ \theta\ \psi]$. Esto se consigue integrando las ecuaciones obtenidas y teniendo en cuenta también la matriz de rotación (5.8) en el caso de las ecuaciones angulares. En la Figura 5.14 se muestra un esquema del proceso.

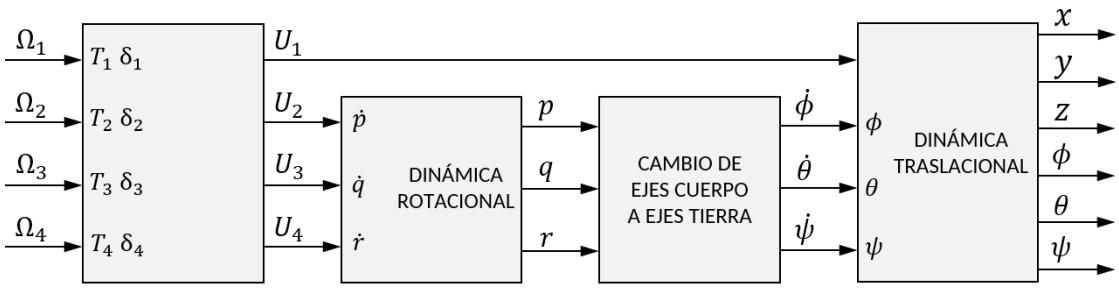


Figura 5.14: Esquema del proceso para obtener el estado del cuadricóptero.

6 Aplicación práctica

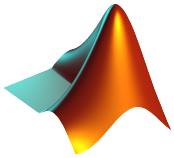
En este capítulo se va a realizar una aplicación donde se van a poner en conjunto los dos bloques teóricos descritos en los capítulos anteriores.

6.1. Descripción del problema

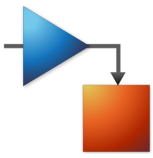
El problema que se va a llevar a cabo es el del posicionamiento de un dron cuadricóptero en el aire. Se trata de que, dada una posición objetivo, el dron aprenda a llegar y a mantenerse sobre ella. Para simplificar el problema se ha elegido realizar el posicionamiento solamente en altura, a lo largo del eje z, ya que es la coordenada más fácil de controlar por el desacople del movimiento vertical respecto a todos los demás.

6.2. Herramientas informáticas

Las herramientas informáticas elegidas son la dupla Matlab®-Simulink® de la corporación MathWorks®, especializada en software de informática matemática, en su versión 2021b. [85]



(a) Logotipo de Matlab [86].



(b) Logotipo de Simulink [87].

Figura 6.1: Logotipos del software usado.

Concretamente, se ha utilizado una paquete llamado *Reinforcement Learning Toolbox*™. Este proporciona recursos de entrenamiento, usando redes neuronales o no, mediante algoritmos de RL a través de la interacción con entornos modelados en Matlab o Simulink que se pueden emplear en controladores, por ejemplo. [88]

6.3. Implementación

Ahora, se explican los archivos creados para llevar a cabo la simulación. Por un lado, la implementación en Simulink del modelo cíclico del RL descrito en el Capítulo 3, con las ecuaciones de la dinámica que gobiernan la aeronave, así como las observaciones y recompensas que recibe del entorno. Por otro lado, se tiene el script de Matlab con el código donde están las órdenes y las redes neuronales del actor-crítico.

Para aprender el funcionamiento del software se han estudiado algunos ejemplos proporcionados por su sitio web oficial [89], también se ha echado un vistazo al curso oficial introductorio *Reinforcement Learning Onramp* [90].

Tras mucho indagar, finalmente, las referencias principales han sido [91] y [92].

6.3.1. Ecuaciones del cuadricóptero en Simulink

Lo primero es construir las ecuaciones vistas en la Sección 5.3 en el entorno de Simulink. Lo que queda es un esquema (Figura 6.2) con la estructura del que se ha mostrado en la Figura 5.14 en el que ingresan las velocidades angulares de los 4 rotores $[\Omega_1\ \Omega_2\ \Omega_3\ \Omega_4]$ para obtener las 3 posiciones y las 3 orientaciones respecto del sistema de ejes tierra $[x\ y\ z\ \phi\ \theta\ \psi]$. A continucación, se comentará cada bloque de esta construcción por separado.

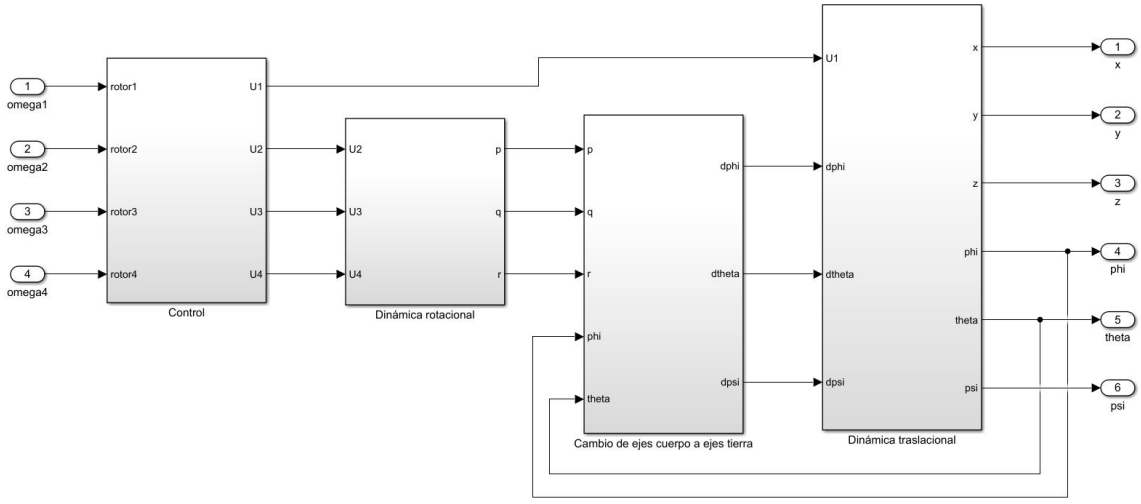


Figura 6.2: Esquema general de bloques de la dinámica del cuadricóptero.

En primer lugar, se encuentra el bloque de control, mostrado en la Figura 6.3, donde a partir de $[\Omega_1\ \Omega_2\ \Omega_3\ \Omega_4]$ se calcula $[U_1\ U_2\ U_3\ U_4]$. Dentro de él hay dos opciones que corresponden a las dos configuraciones que puede presentar el cuadricóptero: cruz (Figura 6.4(a)) o equis (Figura 6.4(b)), representando las ecuaciones (5.28) y (5.29), respectivamente.

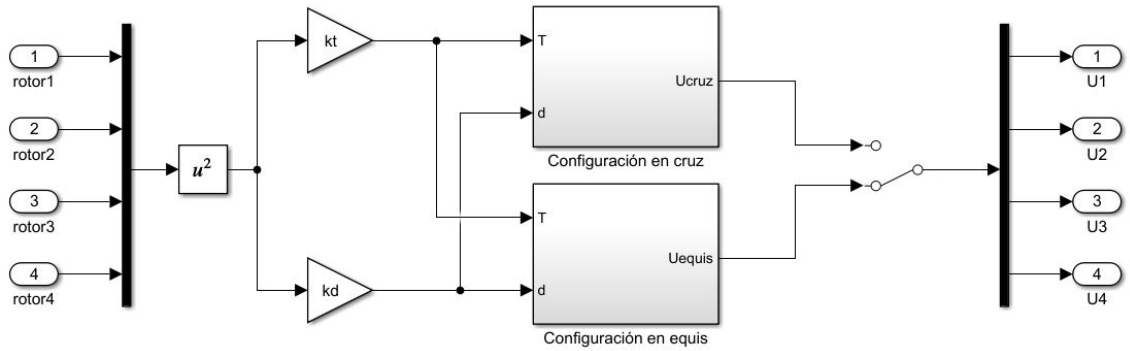
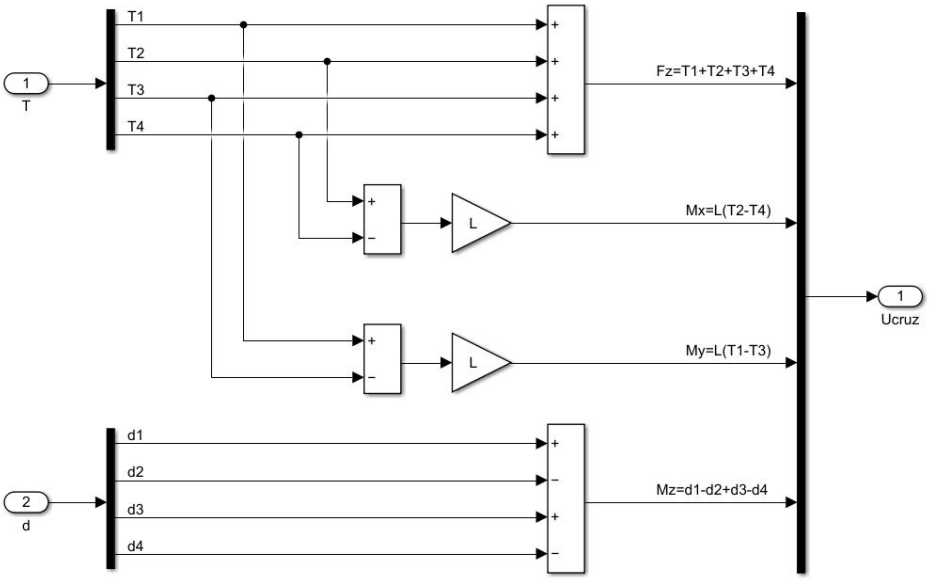
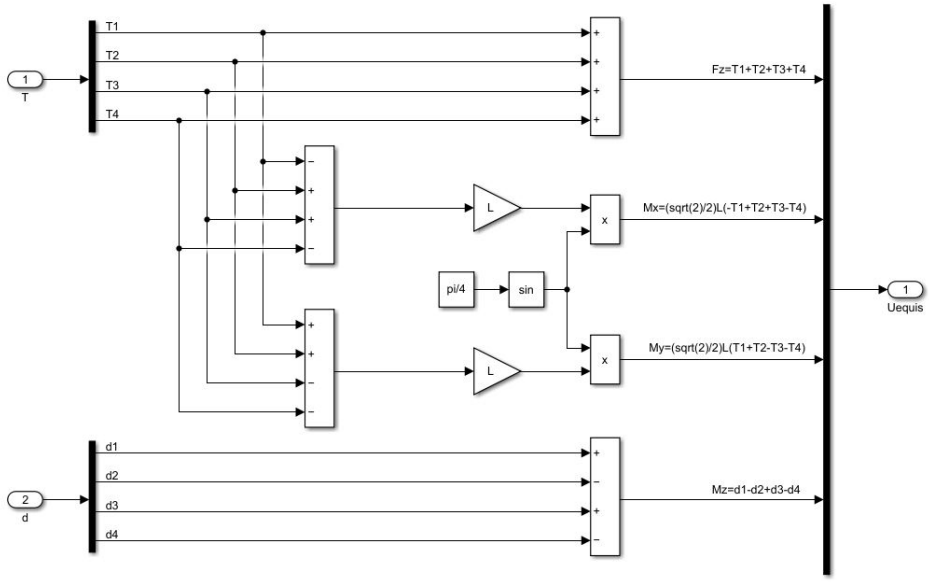


Figura 6.3: Bloque de control.



(a) Configuración en cruz.



(b) Configuración en equis.

Figura 6.4: Distinción entre configuraciones en el bloque de control.

El siguiente bloque representa la dinámica rotacional, en el que se han construido las ecuaciones (5.25), (5.26) y (5.27). Con $[U_2\ U_3\ U_4]$ se calcula $[\dot{p}\ \dot{q}\ \dot{r}]$ y se integra para obtener $[p\ q\ r]$.

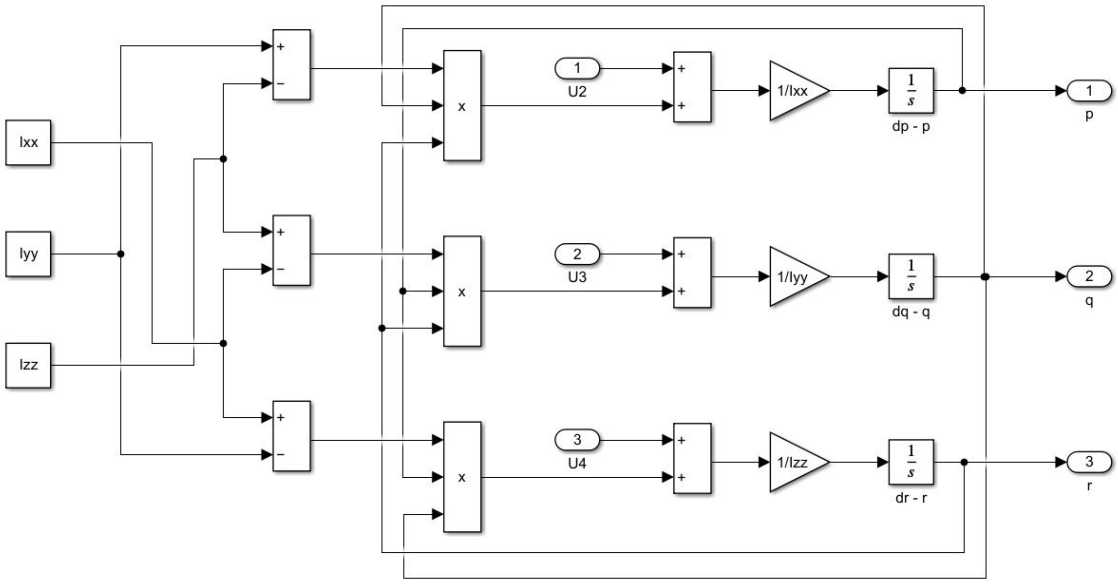


Figura 6.5: Bloque de la dinámica rotacional.

De este, se pasa al bloque donde se realiza el cambio de ejes cuerpo a ejes tierra de la velocidad angular, representado en la Figura 6.6, donde entra $[p\ q\ r]$ para rotarlo a $[\dot{\phi}\ \dot{\theta}\ \dot{\psi}]$. También son necesarios $[\phi\ \theta]$ para realizar el cambio de ejes. Dentro de él se encuentra la matriz (5.8), cuya construcción es la de la Figura 6.7.

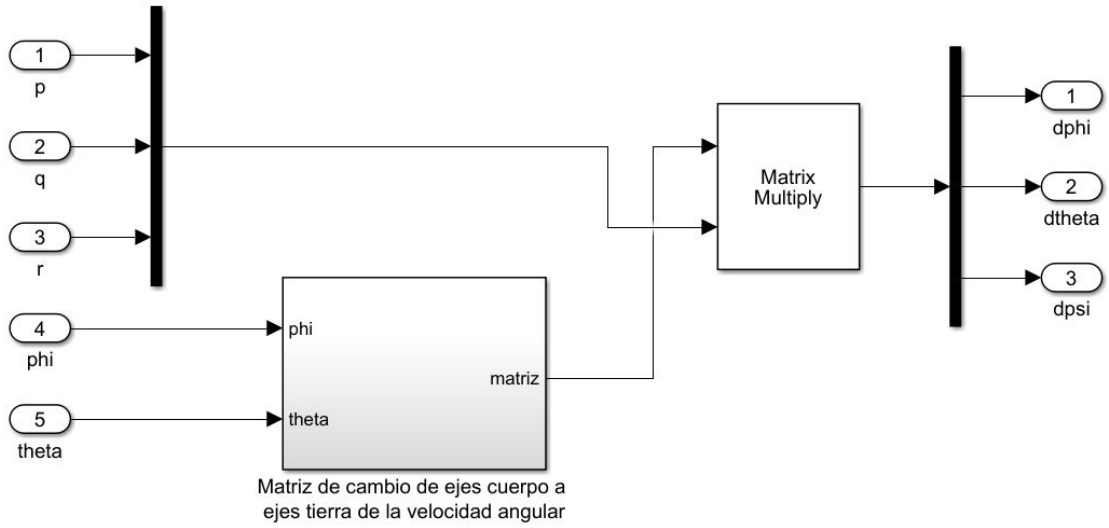


Figura 6.6: Bloque de cambio de B a E de la velocidad angular.

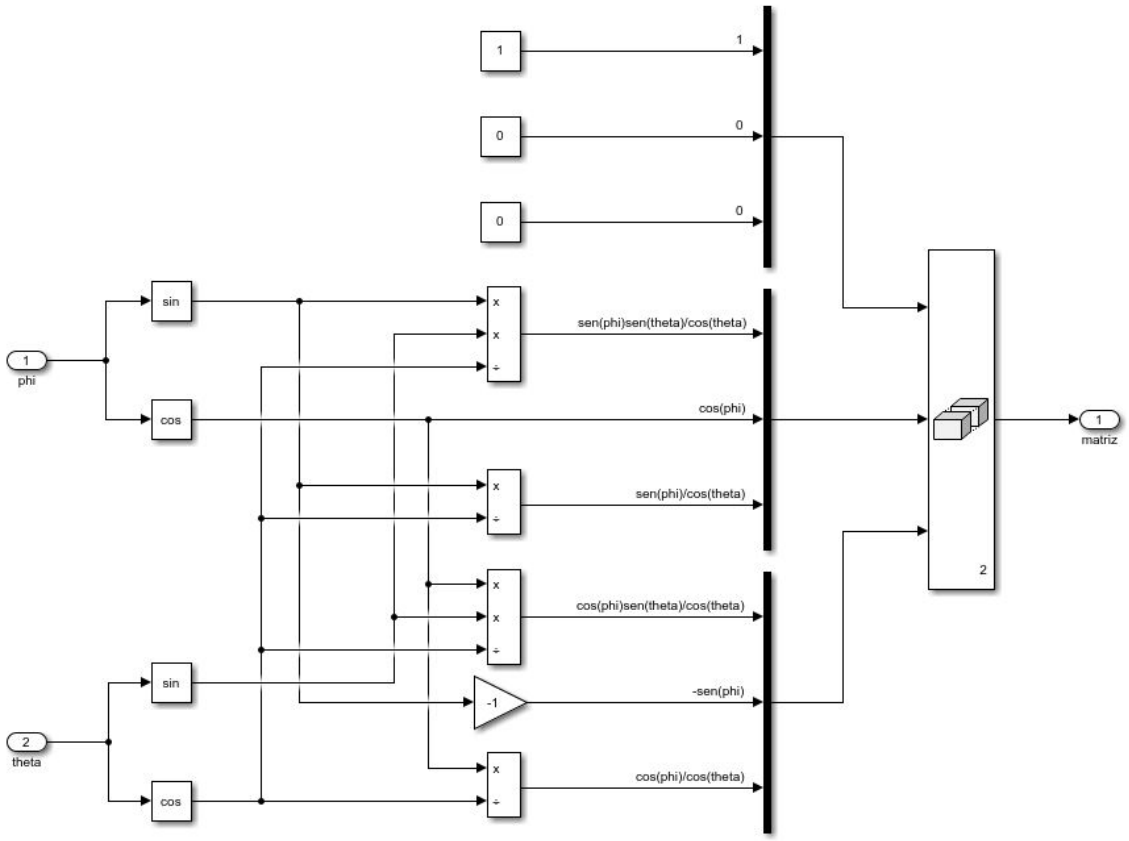


Figura 6.7: Bloque de la matriz cambio de B a E de la velocidad angular.

Por último, se tiene el bloque de la Figura 6.8 que incluye la dinámica traslacional, que se nutre de $[U_1\ \dot{\phi}\ \dot{\theta}\ \dot{\psi}]$ y resulta $[x\ y\ z\ \phi\ \theta\ \psi]$. En él, primero, se encuentra otro bloque (Figura 6.9) que sirve para cambiar la referencia del empuje de ejes cuerpo a ejes tierra mediante la matriz (5.6) de la Figura 6.10. Después con las ecuaciones (5.19), (5.20) y (5.21) y varias integraciones, se obtiene lo que se busca.

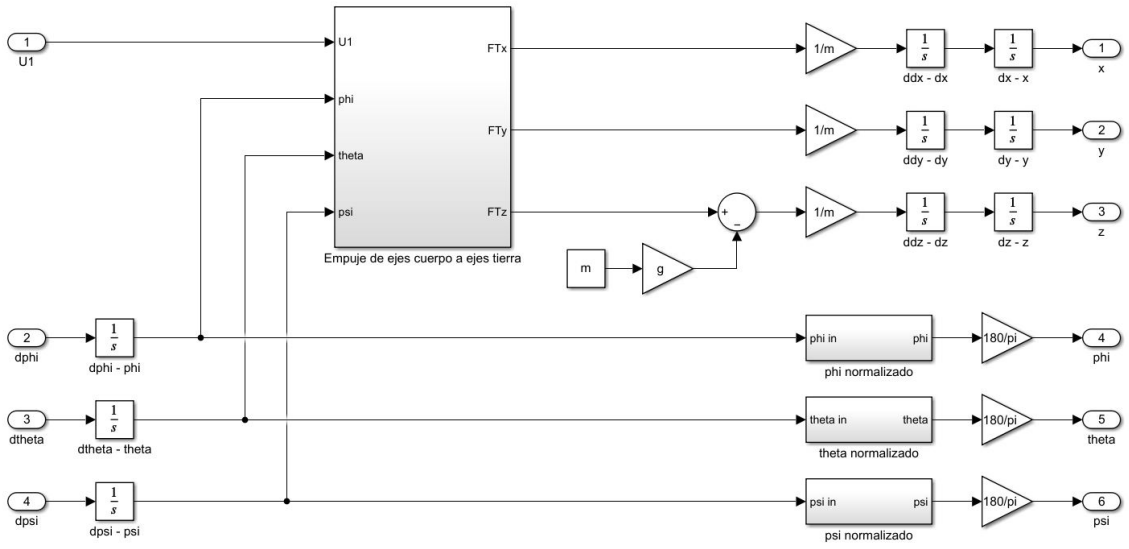


Figura 6.8: Bloque dinámica traslacional.

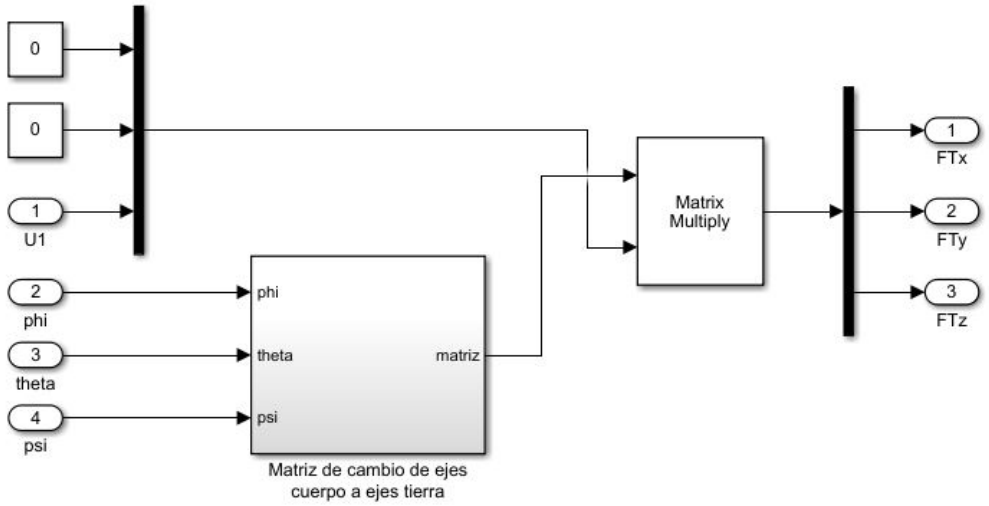


Figura 6.9: Bloque cambio de B a E del empuje.

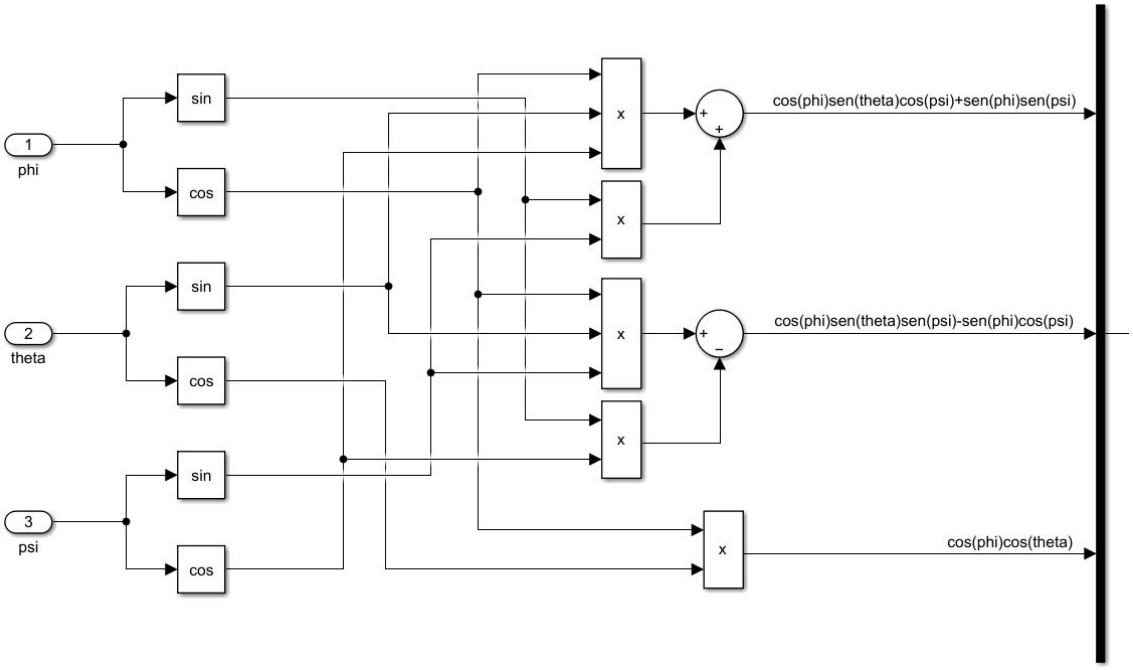


Figura 6.10: Bloque matriz cambio de B a E del empuje.[21]

6.3.2. Código de Matlab

Ahora, se presentan las distintas partes por las que está formado el código que se ha elaborado en Matlab.

Lo primero que se ha hecho es definir una serie de parámetros y constantes. El primer bloque que se diferencia está compuesto por las características del cuadricóp-

[21] Sólo se muestran las posiciones R_{31}, R_{32} y R_{33}, ya que son los únicos términos que intervienen en esta rotación y los demás se han representado nulos.

tero [93]. Destacar aquí el parámetro del empuje máximo que puede aportar cada hélice (T_{max}) que se ha tomado como 3 veces el peso dividido entre las 4 hélices y se ha redondeado. En el segundo bloque se definen unos parámetros útiles para la simulación, donde lo más destacable son los límites máximos y mínimos de la acción que se va a controlar, en este caso el empuje. Estos límites[22] se han tomado 0 y 1, que luego se escalarán.

```
%% PARAMETROS Y CONSTANTES.

m = 0.468;                      % masa (kg)
Ixx = 4.856e-3;                 % momento de inercia respecto al eje x (kg m^2)
Iyy = 4.856e-3;                 % momento de inercia respecto al eje y (kg m^2)
Izz = 8.801e-3;                 % momento de inercia respecto al eje z (kg m^2)
L = 0.225;                      % distancia desde cada rotor al cm del dron (m)
n = 4;                          % numero rotores
g = 9.81;                       % gravedad (m/s)
kt = 2.980e-6;                  % coeficiente de empuje (N s^2)
kd = 1.140e-7;                  % coeficiente de arrastre (N m s^2)
Tmax = ceil((3*m*g/n)*2)/2;     % estimacion del empuje maximo de cada rotor

uc_min = 0;                     % valor minimo que toma la accion
uc_max = 1;                     % valor maximo que toma la accion
Ts = 0.025;                     % tiempo de paso (s)
Tf = 10;                        % tiempo de episodio (s)
zf = 1;                         % posicion de referencia (m)
z0 = 1;                         % posicion inicial (m)
```

A continuación, se modela el entorno de actuación del agente. Tanto las observaciones como las acciones se definen como valores continuos. Primero, las observaciones que recibe el agente serán 2: el error de posición y la velocidad en el eje z en cada momento. Y segundo, está el empuje como única acción con sus límites definidos. Lo siguiente es crear el entorno llamando al archivo de Simulink donde se moverá el agente. Finalmente, se llama a la función de reinicio para cada episodio.

```
%% ENTORNO.

% Observaciones.
numObs = 2;  % error, dz
observationInfo = rlNumericSpec([numObs 1]);
observationInfo.Name = 'observaciones';

% Acciones.
numAct = 1;  % T
actionInfo = rlNumericSpec([numAct 1]);
actionInfo.LowerLimit = uc_min;
actionInfo.UpperLimit = uc_max;
actionInfo.Name = 'control';

% Crear entorno.
mdl = 'drone3d_Simulink_zfijo';
open_system(mdl)
blk = [mdl,'/Agente RL'];
env = rlSimulinkEnv(mdl,blk,observationInfo,actionInfo);

% Funcion reset.
env.ResetFcn = @(in)localResetFcn(in);
rng(0);
```

[22] Se considera que las hélices sólo generan empuje en una dirección por tanto, no habrá empuje negativo.

La función de reinicio establece la posición de referencia y la inicial como aleatorias para cada episodio. La posición de referencia estará ubicada entre 5m y 20m, mientras que la posición inicial va desde 0 hasta el doble de esta posición de referencia.

```
%% FUNCION DE REINICIO.

function in = localResetFcn(in)

zf = round(15*rand(),0)+5;          % posicion de referencia entre 5 y 20
z0 = round(2*zf*rand(),1);          % posicion inicial entre 0 y el doble de zf

in = setVariable(in,'zf', zf);      % asignar valor a zf definido al inicio
in = setVariable(in,'z0', z0);      % asignar valor a z0 definido al inicio
end
```

Lo siguiente es crear el agente que va a someterse a entrenamiento. Este agente tiene estructura de actor-crítico, para lo que hay que crear una red neuronal para el crítico y otra para el actor. El diseño de estas redes se basan en las referencias que se comentaron antes: [89] y [92]. Estas redes, básicamente, están formadas por una capa de entrada (*featureInputLayer*) con tantas neuronas como entradas se den, unas capas intermedias con un número variable de neuronas (*fullyConnectedLayer*), entre las que hay unas capas (*reluLayer*) que definen las conexiones entre dichas capas intermedias, y una capa de salida con el número de neuronas correspondiente.

En primer lugar, se crea la red neuronal del crítico, que se muestra en el código a continuación. Esta consta de 2 ramas de entrada: observaciones y acciones, que se unen para dar como salida un aestimación del valor de la recompensa a largo plazo. Finalmente, se configuran las opciones del crítico y se crea este. El esquema de la red del crítico se puede ver en la Figura 6.11(a).

```
%% RL - AGENTE DDPG.

neuronas1 = 100;
neuronas2 = 70;

% CRITICO.

% Rama observaciones.
obs_inputlayer = featureInputLayer(numObs,'Name','observaciones');
obs_firstfc = fullyConnectedLayer(neuronas1,'Name','obsfc1');
obs_firstrelu = reluLayer('Name','obsrelu1');
obs_secondfc = fullyConnectedLayer(neuronas2,'Name','obsfc2');
obs_path = [obs_inputlayer obs_firstfc obs_firstrelu obs_secondfc];

% Rama acciones.
act_inputlayer = featureInputLayer(numAct,'Name','acciones');
act_firstfc = fullyConnectedLayer(neuronas2,'Name','accfc1');
act_path = [act_inputlayer act_firstfc];

% Rama comun.
addlayer = additionLayer(2,'Name','comun');
relu = reluLayer('Name','comunrelu');
outputlayer = fullyConnectedLayer(1,'Name','valor');
join_path = [addlayer relu outputlayer];

% Crear la red neuronal.
critic_red = layerGraph(obs_path);
critic_red = addLayers(critic_red,act_path);
critic_red = addLayers(critic_red,join_path);
critic_red = connectLayers(critic_red,'obsfc2','comun/in1');
```

```
critic_red = connectLayers(critic_red,'accfc1','comun/in2');

% Visualizar la red neuronal.
figure;
plot(critic_red)

% Opciones.
critic_opts = rlRepresentationOptions('LearnRate',0.0001,'GradientThreshold',1);

% Crear critico.
critic = rlQValueRepresentation(critic_red,observationInfo,actionInfo,'↩
    Observation','observaciones','Action','acciones',critic_opts);
```

De forma similar al crítico, se crea la red neuronal del actor, como se expone en el siguiente código. Esta red está formada por una sola línea con las observaciones por entradas y las acciones a llevar a cabo como salida. Luego, se configuran las opciones del actor y se crea este. El esquema de esta red se muestra en la Figura 6.11(b).

```
%% RL - AGENTE DDPG.

neuronas1 = 100;
neuronas2 = 70;

...

% ACTOR.

% Rama unica.
inputlayer = featureInputLayer(numObs,'Name','observaciones');
firstfc = fullyConnectedLayer(neuronas1,'Name','fc1');
firstrelu = reluLayer('Name',"relu1");
secondfc = fullyConnectedLayer(neuronas2,'Name','fc2');
secondrelu = reluLayer('Name','relu2');
outputlayer = fullyConnectedLayer(numAct,'Name','accion');
tahnlayer = tanhLayer('Name','tahn');
actor_path = [inputlayer firstfc firstrelu secondfc secondrelu outputlayer ↩
    tahnlayer];

% Crear la red neuronal.
actor_red = layerGraph(actor_path);

% Visualizar la red neuronal.
figure;
plot(actor_red)

% Opciones.
actor_opts = rlRepresentationOptions('LearnRate',0.00001,'GradientThreshold',1);

% Crear actor.
actor = rlDeterministicActorRepresentation(actor_red,observationInfo,actionInfo,'↩
    Observation','observaciones','Action','tahn',actor_opts);
```

Hecho esto, se configura y se crea el agente. De estas opciones, destacar los valores establecidos para la exploración (las líneas de *NoiseOptions*), muy importante en los problemas de RL, que se han elegido según [94].

```
%% RL - AGENTE DDPG.

neuronas1 = 100;
neuronas2 = 70;

...

% AGENTE.
```

```
% Opciones.
agent_opts = rlDDPGAgentOptions('SampleTime',Ts,'ExperienceBufferLength',1e6,'↩
    MiniBatchSize',32,'TargetSmoothFactor',1e-3,'DiscountFactor',0.99);
agent_opts.NoiseOptions.Variance = 0.6;
agent_opts.NoiseOptions.VarianceDecayRate = 1e-5;

% Crear agente.
agent = rlDDPGAgent(actor,critic,agent_opts);
```

(a) Red neuronal del crítico. **(b)** Red neuronal del actor.

Figura 6.11: Redes neuronales.

Finalmente, se realiza el entrenamiento, lo cual puede llevar horas, y se simula.

```
%% ENTRENAMIENTO.

maxepisodes = 5000;
maxsteps = ceil(Tf/Ts);
training_opts = rlTrainingOptions(...
    'MaxEpisodes',maxepisodes,...
    'MaxStepsPerEpisode',maxsteps,...
    'StopTrainingCriteria','AverageReward',...
    'StopTrainingValue',1750,...
    'ScoreAveragingWindowLength',20,...
    'SaveAgentCriteria','EpisodeReward',...
    'SaveAgentValue',1750);

doTraining = true;
```

```
if doTraining
    % Entrenar agente.
    training = train(agent,env,training_opts);
    % Guardar agente final.
    save('Agente_drone3d_zfijo_4.mat','agent');
else
    % Cargar un agente previamente entrenado.
    load('Agente_drone3d_zfijo_1.mat','agent');
end

%% SIMULACION.

sim_opts = rlSimulationOptions('MaxSteps',400,'NumSimulations',1);
experience = sim(env,agent,sim_opts);
```

6.3.3. Modelo de Simulink

Ahora, se construye el modelo que representa el entorno del problema. Su estructura se muestra en la Figura 6.12, donde se distinguen varios bloques: agente RL, ecuaciones, observaciones, recompensas y detener simulación. A continuación, se explicarán uno a uno. Como se ve, el único parámetro que tiene salida no nula es la coordenada z, ya que es lo que se está controlando, como se comentó al inicio del capítulo.

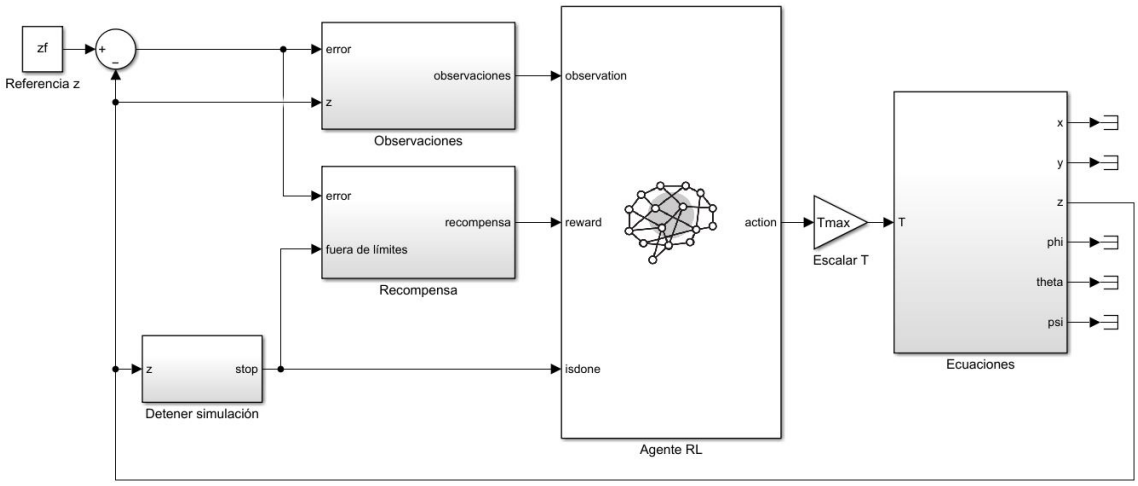
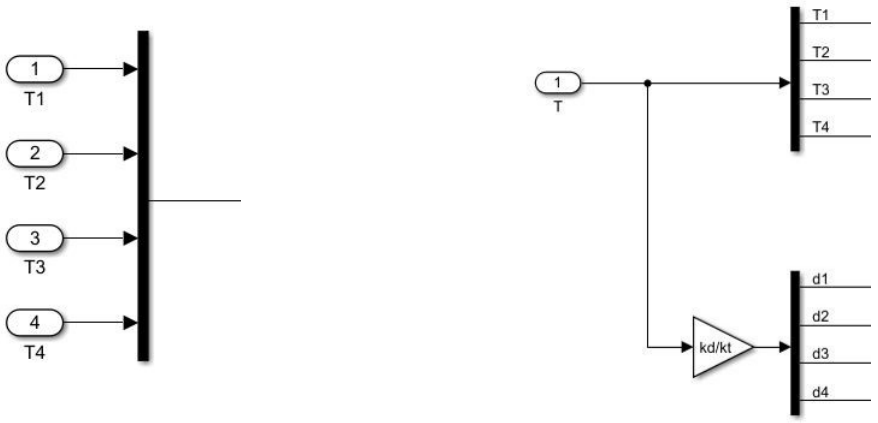


Figura 6.12: Esquema general del problema en Simulink.

El bloque del agente es un bloque proporcionado por la biblioteca de Simulink que representa al agente que se ha diseñado en el script de Matlab que se ha visto. Su función es relacionar al agente con el entorno.

El bloque de las ecuaciones es prácticamente el mismo que se ha presentado en el Apartado 6.3.1, aunque tiene algunas modificaciones. Antes se dijo que las entradas eran las velocidades de giro de los rotores, sin embargo, aquí se va a tratar directamente con los empujes. De esta forma, el bloque de control que se vio en la Figura 6.3 quedaría según la Figura 6.13.



(a) Modificación de la Figura 6.3. **(b)** Modificación de la Figura 6.4.

Figura 6.13: Modificaciones del bloque de control.

En el bloque de observaciones (Figura 6.14) corresponde a la información del estado que se le comunica al agente, como ya se dijo, el error de posición y la velocidad. Se ha hecho a criterio propio, aunque la construcción se ha basado en [90], [91] y [92].

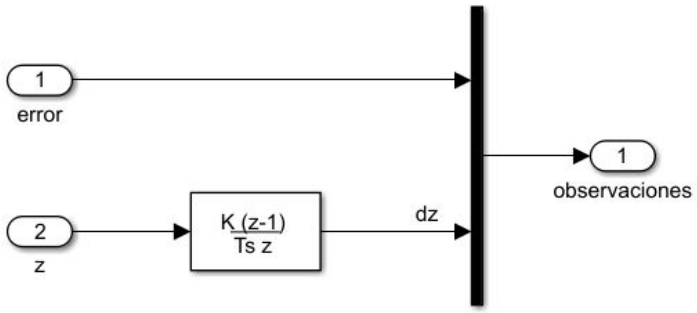


Figura 6.14: Bloque de observaciones.

El siguiente es el bloque (Figura 6.15) donde se indican las causas de la interrupción temprana del episodio, que son que la altura de vuelo alcance el suelo o que se vaya demasiado lejos de la altura de referencia.

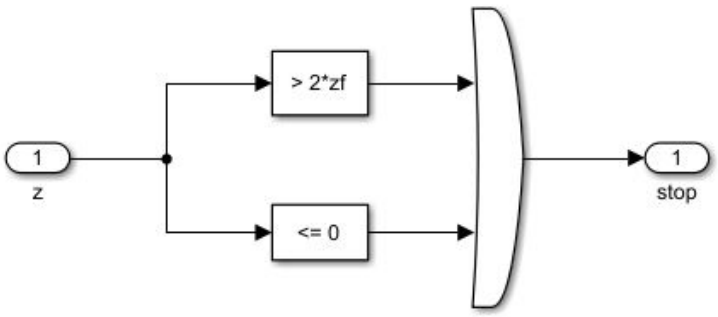


Figura 6.15: Bloque de detención de la simulación.

Finalmente, está el bloque de recompensa que, probablemente, sea el bloque más importante, ya que su función es servir de guía al aprendizaje del agente. Su diseño consta de varias partes: dar una recompensa muy mala cuando se alcance el fuera de límites; dar una buena recompensa de magnitud considerable cuando el error absoluto sea mínimo, es decir, se esté muy cerca del objetivo; y una recompensa negativa de valor unidad, como máximo, según sea el porcentaje de error cometido, para el resto de estados. Todo esto, se ve en la Figura 6.17.

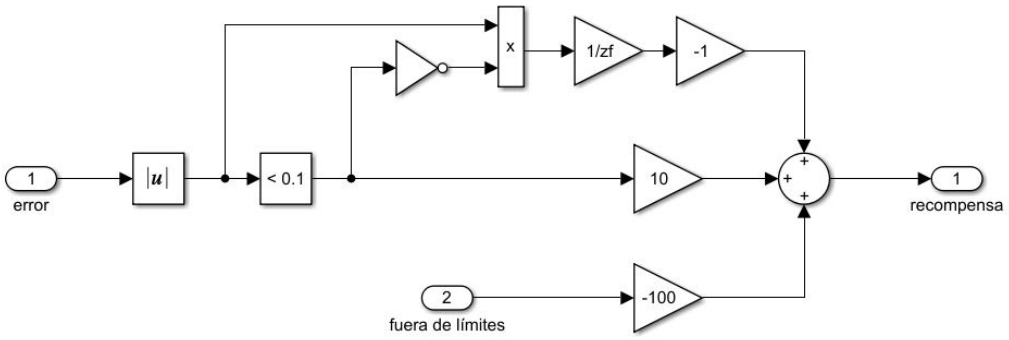


Figura 6.16: Bloque de recompensa.

6.4. Resultados

Se ha realizado el entrenamiento al modelo que se ha presentado anteriormente y se ha obtenido la siguiente curva de aprendizaje en la ventana *Reinforcement Learning Episode Manager*. En ella se representa la recompensa que se ha ido obteniendo en cada episodio. Como se puede ver, al principio la recompensa es muy baja porque siempre se salía de los límites establecidos, pero con el paso de los episodios, esta curva de la recompensa asciende mucho, lo que indica aprendizaje.

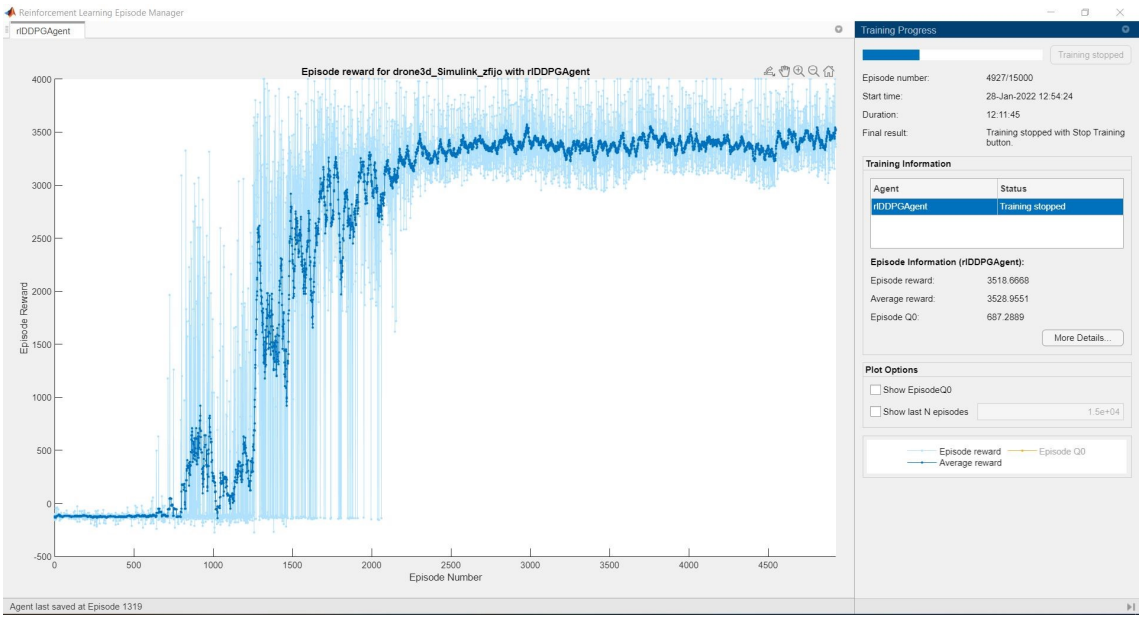


Figura 6.17: Curva de aprendizaje en *Reinforcement Learning Episode Manager*.

Tras realizar el entrenamiento, se observan los resultados representados en gráficas. En primera instancia se representa el cambio de altitud del dron y cómo alcanza la altura de referencia en poco tiempo y a partir de ahí se mantiene sin titubear. Lo segundo es la gráfica de la recompensa que va obteniendo en cada instante. Por último, se observa la variación de empuje a lo largo del recorrido.

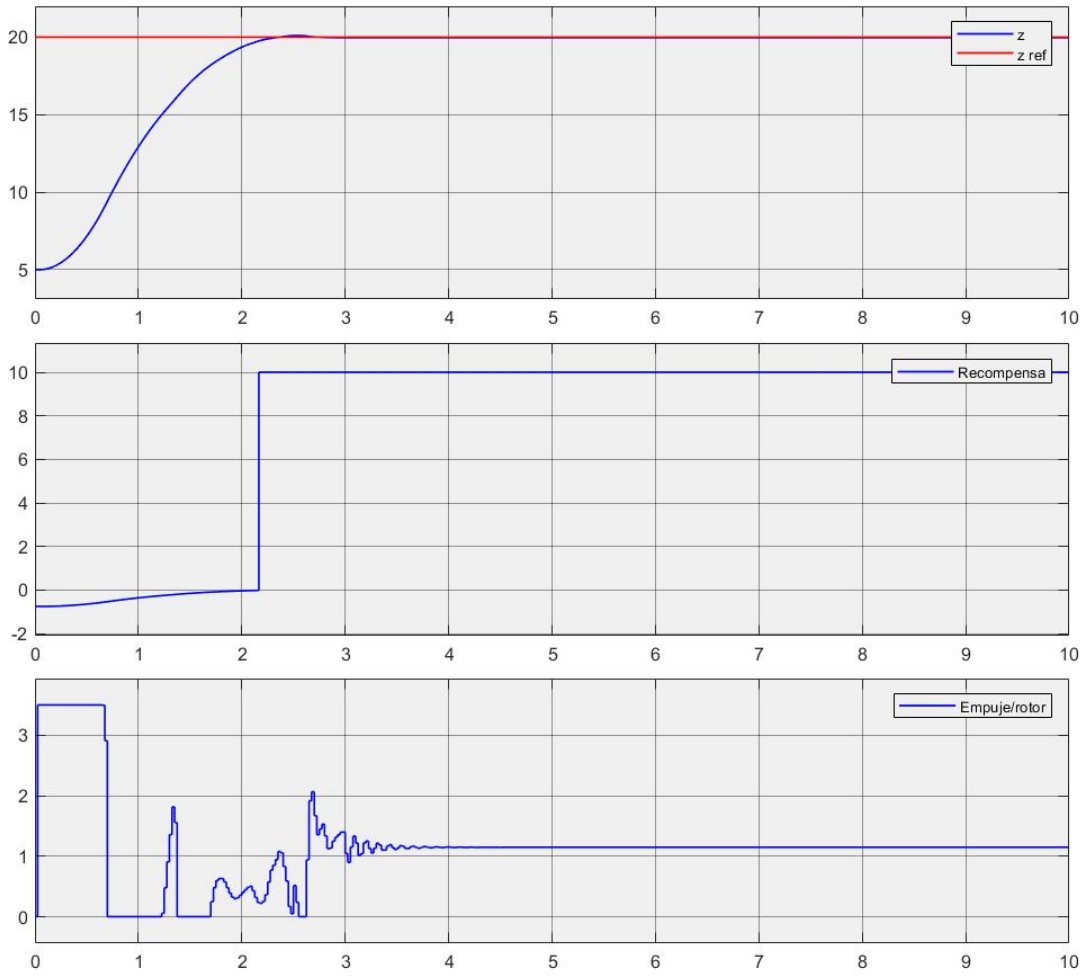
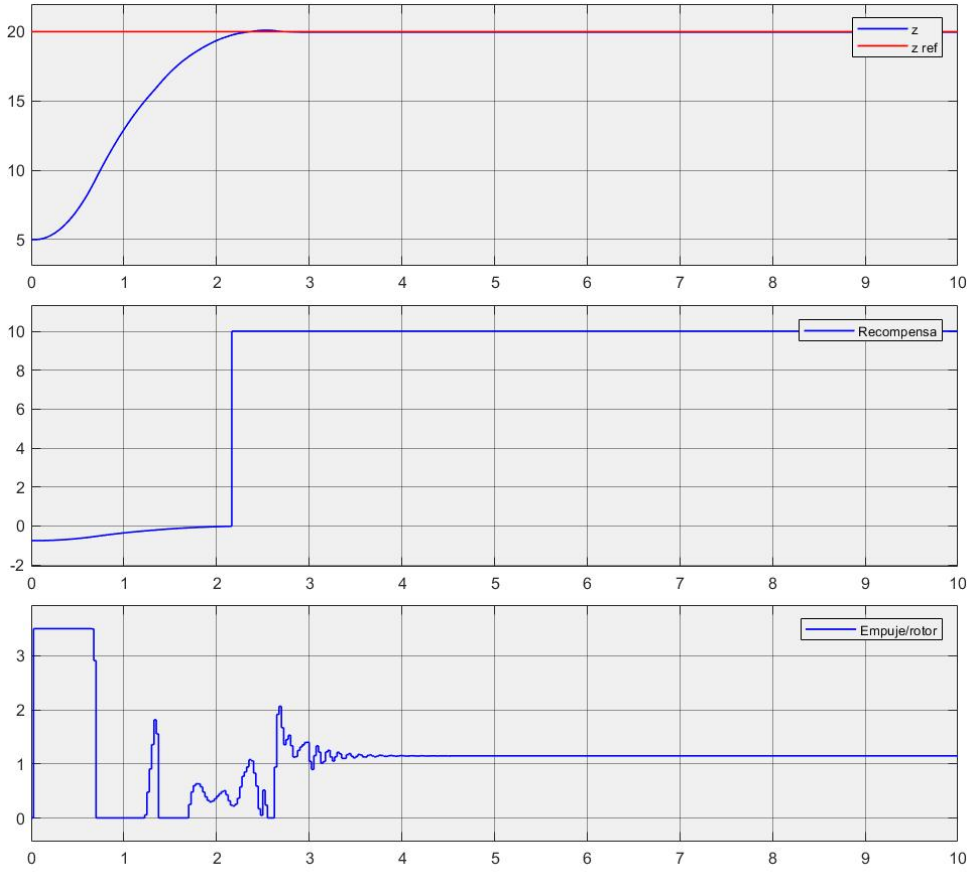


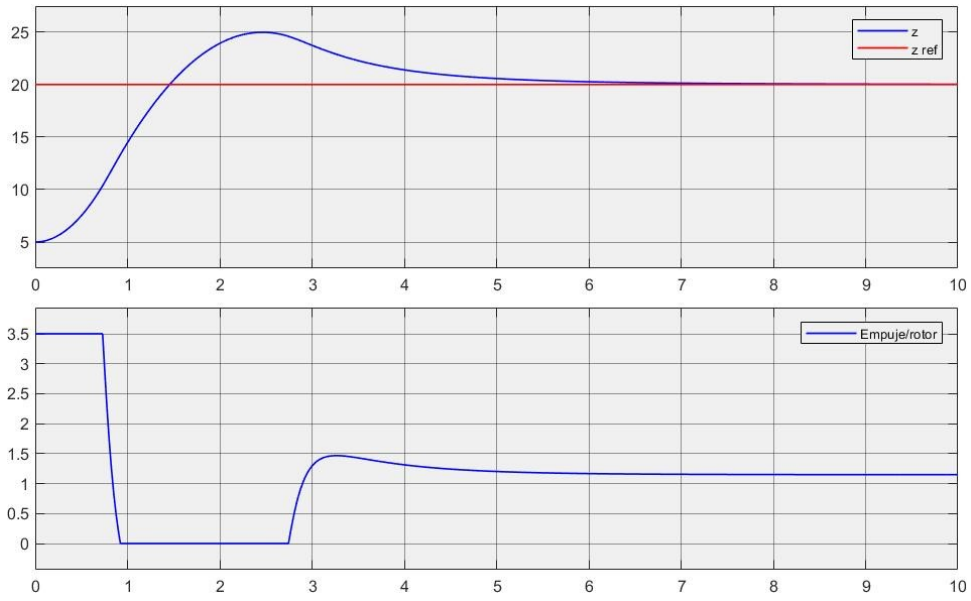
Figura 6.18: Muestra de resultados.

6.4.1. Comparación con un PID

Obtenidos los resultados, se ha pensado en compararlos con los que daría un controlador PID, que es lo que es normal usar en estos casos. Esta comparativa se muestra en las Figuras 6.19 y 6.20. Viendo los resultados, se puede decir que la actuación del RL es mejor, ya que alcanza la referencia en menos tiempo. Además, alcanza la referencia y no la deja en ningún momento tras ello, al contrario que la gráfica del PID. Esto se debe a la rapidez de reacción en la variación del empuje que se adquiere con el entrenamiento, mientras que el cambio de empuje con el PID es más suave, como se puede ver comparando las dos gráficas.

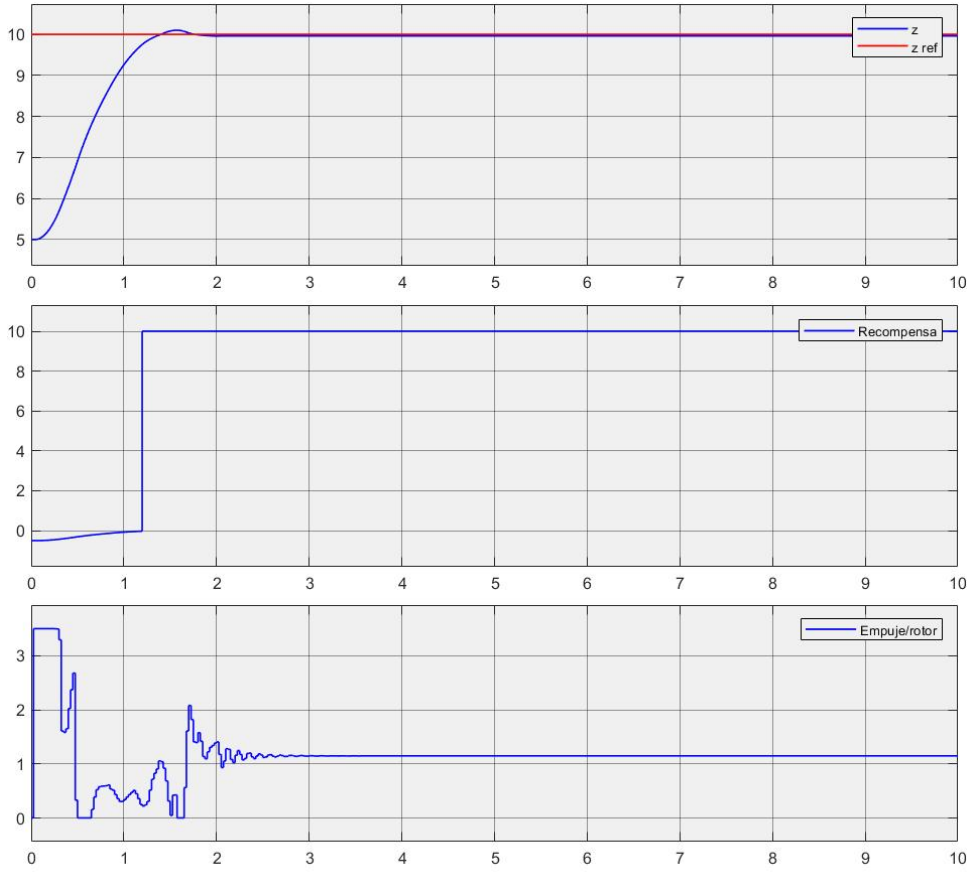


(a) Resultados del entrenamiento por RL.

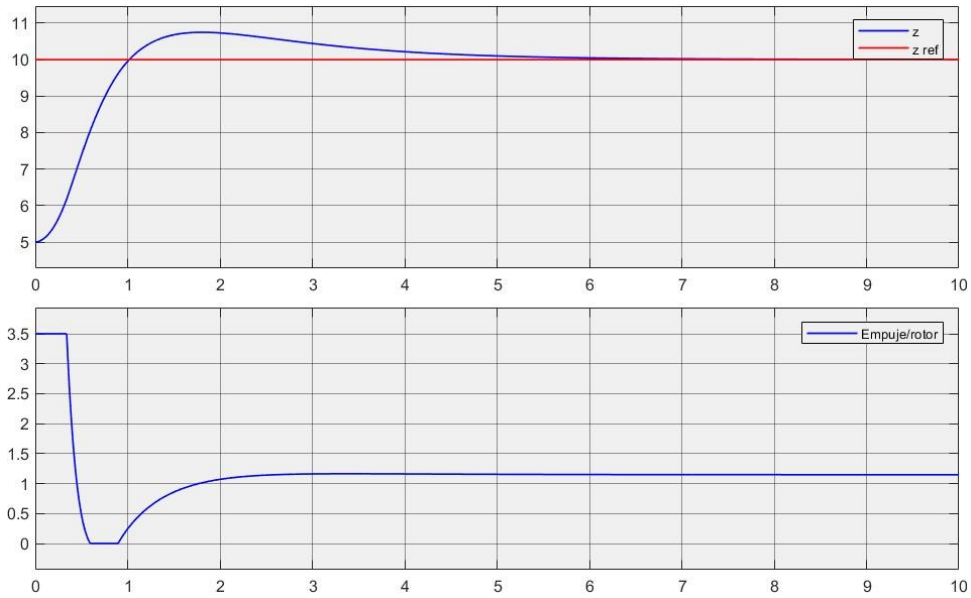


(b) Resultados de un PID

Figura 6.19: Comparación de resultados 1.



(a) Resultados del entrenamiento por RL.



(b) Resultados de un PID

Figura 6.20: Comparación de resultados 2.

6.4.2. Conclusiones

No hay duda de que este método de entrenamiento mediante RL presenta grandes puntos a favor, como es la rápida y precisa actuación. Sin embargo, esto se costea mediante un proceso caro en cuanto a tiempo de entrenamiento, ya que puede durar horas cuyo número incrementa con la complejidad del problema. A pesar de ello, una vez que se tiene un agente entrenado, este se puede entrenar ampliando su rango de actuación sin tener que comenzar todo el proceso desde el principio. Sería un entrenamiento de lo entrenado, lo que reduciría los tiempos.

Por otra parte, aunque la aplicación del PID no requiera tanta inversión de tiempo, también necesita implicación para obtener una buena configuración. Además, el rango de actuación está más limitado y habría que andar buscando una nueva configuración para cada caso.

Por tanto, como conclusión, la elección de aplicación de uno u otro método quedaría marcado por los requisitos de actuación que se deban cumplir y la limitación de recursos. Es decir, si se necesita un posicionamiento muy preciso desde un primer momento, sería conveniente realizar un entrenamiento que permita obtenerlos; mientras que si no es necesaria tanta exactitud, bastaría con seguir usando los controladores que se han venido usando hasta ahora.

6.4.3. Trabajo futuro

Este trabajo se ha limitado a la traslación de un cuadricóptero en el eje vertical y se ha utilizado una referencia fija. De forma que esto ha sido sólo un primer paso en la implementación del RL en el control de un cuadricóptero y aún quedarían puertas abiertas por las que se podría ampliar el proyecto jugando con la combinación de varios aspectos:

- Movimiento en más direcciones y no limitarse sólo al cambio de altura.
- Punto de referencia móvil.

Referencias

[1] F. Sancho Caparrini, "Breve historia de la Inteligencia Artificial", *Fernando Sancho Caparrini*, 2020. [En línea]. Disponible en: `https://bit.ly/34CExIC`.

[2] H. A. Banda Gamboa, "Fundamentos de la Inteligencia Artificial", en *Inteligencia Artificial: principios y aplicaciones*, 1ª ed. Quito: Escuela Politecnica Nacional, 2014, pp. 1-19.

[3] S. Jaiswal, "History of Artificial Intelligence", *JavaTpoint*, 2019. [En línea]. Disponible en: `https://bit.ly/3osmtaU`.

[4] R. E. López Briega, *Libro online de IAAR*, 2017. [En línea]. Disponible en: `https://iaarbook.github.io/`.

[5] "La Cuarta Revolución Industrial", *Iberdrola*, 2020. [En línea]. Disponible en: `https://bit.ly/3HzkGsm`.

[6] R. Perrault *et al.*, "The AI Index 2019 Annual Report", AI Index Steering Committee, Human-Centered AI Institute, Stanford University, Stanford, CA, Estados Unidos, dic. 2019. [En línea]. Disponible en: `https://hai.stanford.edu/`.

[7] C. Ferrer Caballero, "Las 6 leyes de la robótica de la Unión Europea", *Blogthinkbig.com*, 2017. [En línea]. Disponible en: `https://bit.ly/33131uu`.

[8] "¿Qué es la Inteligencia Artificial?", *Iberdrola*, 2020. [En línea]. Disponible en: `https://bit.ly/3skgINQ`.

[9] "Qué es el Machine Learning", *Iberdrola*, 2020. [En línea]. Disponible en: `https://bit.ly/3otYecz`.

[10] J. I. Bagnato, "Aplicaciones del Machine Learning", *Aprende Machine Learning antes de que sea demasiado tarde*, 2020. [En línea]. Disponible en: `https://bit.ly/3uvx60w`.

[11] M. Carbo, "Qué es la Inteligencia Artificial", *AuraQuantic*, 2019. [En línea]. Disponible en: `https://bit.ly/3B2WLiu`.

[12] Real Academia Española, "Diccionario de la lengua española", 2020. [En línea]. Disponible en: `https://dle.rae.es/`.

[13] "Informática", *Wikipedia. La Enciclopedia Libre*, 2020. [En línea]. Disponible en: `https://bit.ly/3gtZwji`.

[14] "Artificial Intelligence - Quick Guide", *Tutorials Point*, 2020. [En línea]. Disponible en: `https://bit.ly/3B5Ua7l`.

[15] S. J. Russell y P. Norvig, "Introducción", en *Inteligencia Artificial. Un enfoque moderno*, 2ª ed. Madrid: Prentice-Hall, 2004, pp. 1-36.

[16] J. Pérez Porto y A. Gardey, "Definición de inteligencia", *Definición.de*, 2012. [En línea]. Disponible en: `https://definicion.de/inteligencia/`.

[17] J. Pérez Porto y M. Merino, "Definición de artificial", *Definición.de*, 2017. [En línea]. Disponible en: `https://definicion.de/artificial/`.

[18] F. Escolano Ruiz, M. A. Cazorla Quevedo, M. I. Alfonso Galipienso, O. Colomina Pardo y M. A. Lozano Ortega, "Introducción a la Inteligencia Artificial", en *Inteligencia Artificial. Modelos, técnicas y áreas de aplicación*. 1ª ed. Madrid: Thompson, 2003, pp. 3-8.

[19] L. González, "Tipos de Inteligencia Artificial", *Aprende IA*, 2019. [En línea]. Disponible en: `https://bit.ly/3uvrROD`.

[20] A. Pérez Barreiro, "Tipos de Inteligencia Artificial. Débil, general y súperinteligencia", *Futuro Eléctrico*, 2020. [En línea]. Disponible en: `https://bit.ly/3LgV8m4`.

[21] R. Riquelme, "4 tipos de Inteligencia Artificial que debes conocer", *El Economista*, 2016. [En línea]. Disponible en: `https://bit.ly/31Xv3m6`.

[22] M. Carbo, "Tecnologías de Inteligencia Artificial y sus categorías", *AuraQuantic*, 2019. [En línea]. Disponible en: `https://bit.ly/34Gd1tq`.

[23] S. Jaiswal, "Subsets of Artificial Intelligence", *JavaTpoint*, 2019. [En línea]. Disponible en: `https://bit.ly/3Gu9BY7`.

[24] "Automated planning and scheduling", *Wikipedia. The Free Encyclopedia*, 2020. [En línea]. Disponible en: `https://bit.ly/3owQCpF`.

[25] C. Nicholson, "A.I. Wiki. A Beginner's guide to important topics in AI, Machine Learning, and Deep Learning - Definition Machine Learning", *Pathmind*, 2020. [En línea]. Disponible en: `https://bit.ly/3slm5MF`.

[26] "Machine Learning tutorial for beginners ", *Guru99*, 2020. [En línea]. Disponible en: `https://bit.ly/3J53lYw`.

[27] A. Géron, "The Machine Learning landscape", en *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow. Concepts, tools, and techniques to build intelligent systems*, 2ª ed. Sebastopol: O'Reilly Media, 2019, pp. 9-40.

[28] J. Hurwitz y D. Kirsch, "Looking inside Machine Learning", en *Machine Learning for dummies*, edición limitada IBM. Hoboken: John Wiley & Sons, 2018, pp. 27-38.

[29] J. Martínez Heras, "Machine Learning e Inteligencia Artificial", *IArtificial.net*, 2019. [En línea]. Disponible en: `https://bit.ly/3uw0EeG`.

[30] K. P. Murphy, "Introduction", en *Machine Learning: a probabilistic perspective*, 1ª ed. Cambridge: The MIT Press, 2012, pp. 1-26.

[31] L. González, "Clasificación de Machine Learning", *Aprende IA*, 2018. [En línea]. Disponible en: `https://bit.ly/3oqgCTO`.

[32] F. Sancho Caparrini, "Aprendizaje por refuerzo: algoritmo Q Learning", *Fernando Sancho Caparrini*, 2019. [En línea]. Disponible en: `https://bit.ly/3Lau73L`.

[33] J. I. Bagnato, "Aprendizaje por Refuerzo", *Aprende Machine Learning antes de que sea demasiado tarde*, 2020. [En línea]. Disponible en: `https://bit.ly/3HzpFcm`.

[34] E. Alpaydın, "Introduction", en *Introduction to Machine Learning*, 2ª ed. Cambridge, MA: The MIT Press, 2010, pp. 1-20.

[35] R. S. Sutton y A. G. Barto, "Introduction", en *Reinforcement Learning: an introduction*, 2ª ed. Cambridge, MA: The MIT Press, 2018, pp. 1-22.

[36] J. Martínez Heras, "Redes neuronales desde cero (I) – Introducción", *IArtificial.net*, 2020. [En línea]. Disponible en: `https://bit.ly/35NldIZ`.

[37] "Inteligencia Artificial", *Numerentur.org*, 2021. [En línea]. Disponible en: `https://bit.ly/3rtk023`.

[38] L. González, "¿Qué es Deep Learning?", *Aprende IA*, 2018. [En línea]. Disponible en: `https://bit.ly/3Hy6GPw`.

[39] C. Nicholson, "A.I. Wiki. A beginner's guide to important topics in AI, Machine Learning, and Deep Learning - A beginner's guide to Deep Reinforcement Learning", *Pathmind*, 2020. [En línea]. Disponible en: `https://bit.ly/34IO3Kk`.

[40] S. Jaiswal, "Reinforcement Learning Tutorial", *JavaTpoint*, 2019. [En línea]. Disponible en: `https://bit.ly/3urVSic`.

[41] D. Silver, "Lecture 1: Introduction to Reinforcement Learning", *UCL Course on RL*, 2015. [Presentación de diapositivas]. Disponible en: `https://www.davidsilver.uk/teaching/`.

[42] R. S. Sutton y A. G. Barto, "Finite Markov Decision Processes", en *Reinforcement Learning: an introduction*, 2ª ed. Cambridge, MA: The MIT Press, 2018, pp. 47-72.

[43] M. Wiering y M. van Otterlo, "Reinforcement Learning and Markov Decision Processes", en *Reinforcement Learning: state-of-the-art*, 1ª ed. Berlin, Heidelberg: Springer, 2012, pp. 3-42.

[44] D. L. Poole y A. K. Mackworth, "Planning under uncertainty", en *Artificial Intelligence: foundations of computational agents*, 1ª ed. Cambridge: Cambridge University Press, 2010, pp. 371-422.

[45] D. L. Poole y A. K. Mackworth, "Reasoning under uncertainty", en *Artificial Intelligence: foundations of computational agents*, 1ª ed. Cambridge: Cambridge University Press, 2010, pp. 219-280.

[46] D. Silver, "Lecture 2: Markov Decision Processes", *UCL Course on RL*, 2015. [Presentación de diapositivas]. Disponible en: `https://www.davidsilver.uk/teaching/`.

[47] J. L. Ruíz Reina y F. J. Martín Mateos, "Procesos de Markov", Ampliación de Inteligencia Artificial, Dpto. Ciencias de la Computación e Inteligencia Artificial, Universidad de Sevilla, 2015-2016. [Presentación de diapositivas]. Disponible en: `https://bit.ly/3Hro7Bf`.

[48] F. Fernández Rebollo, "Parte I: Introducción al Aprendizaje por Refuerzo", Grupo de Planificación y Aprendizaje (PLG), Dpto. Informática, Escuela Politécnica Superior, Universidad Carlos III de Madrid, 2013. [Presentación de diapositivas]. Disponible en: `https://bit.ly/3grfxGH`.

[49] D. L. Poole y A. K. Mackworth, "Beyond Supervised Learning", en *Artificial Intelligence: foundations of computational agents*, 1ª ed. Cambridge: Cambridge University Press, 2010, pp. 451-488.

[50] R. S. Sutton y A. G. Barto, "Monte Carlo Methods", en *Reinforcement Learning: an introduction*, 2ª ed. Cambridge, MA: The MIT Press, 2018, pp. 91-118.

[51] R. S. Sutton y A. G. Barto, "Dynamic Programming", en *Reinforcement Learning: an introduction*, 2ª ed. Cambridge, MA: The MIT Press, 2018, pp. 73-90.

[52] R. S. Sutton y A. G. Barto, "Temporal-Difference Learning", en *Reinforcement Learning: an introduction*, 2ª ed. Cambridge, MA: The MIT Press, 2018, pp. 119-140.

[53] G. Urcera Martín, "Generación de trayectorias robóticas mediante aprendizaje profundo por refuerzo", Proyecto final de máster, Dpto. de Ingeniería de Sistemas y Automática, Universitat Politècnica de Catalunya, Barcelona, España, 2018. [En línea]. Disponible en: `http://hdl.handle.net/2117/117602`.

[54] J. Achiam, *OpenAI spinning up*, 2018. [En línea]. Disponible en: `https://spinningup.openai.com`.

[55] "Help Center. Deep Deterministic Policy Gradient Agents", *MathWorks*, 2022. [En línea]. Disponible en: `https://bit.ly/3GxQ9d9`.

[56] "¿UAV, UAS, RPAS o drones?", *Instituto Nacional de Técnica Aeroespacial (INTA)*, 2019. [En línea]. Disponible en: `https://bit.ly/3utSsLP`.

[57] K. Dalamagkidis, "Definitions and terminology", en K. P. Valavanis y G. J. Vachtsevanos (eds.), *Handbook of Unmanned Aerial Vehicles*, 1ª ed. Dordrecht: Springer, 2015, pp. 43-55.

[58] C. Cuerno Rejado, L. García Hernández, A. Sánchez Carmona, A. Carrió Fernández, J. L. Sánchez López y P. Campoy Cervera, "Evolución histórica de los vehículos aéreos no tripulados hasta la actualidad", *Dyna*, vol. 91, nº 3, pp. 282-288, may. 2016. [En línea]. Disponible en: `https://doi.org/10.6036/7781`.

[59] C. Calvo González-Regueral, F. Herranz y P. Calvo Aguilar, "De los UAV a los RPAS", *Perfiles IDS*, feb. 2014. [En línea]. Disponible en: https://bit.ly/3slr5kn.

[60] "Partes de un sistema aéreo no tripulado (UAS)", *HispaDrones*, 2019. [En línea]. Disponible en: https://bit.ly/34hflaO.

[61] K. Dalamagkidis, "Classification of UAVs", en K. P. Valavanis y G. J. Vachtsevanos (eds.), *Handbook of Unmanned Aerial Vehicles*, 1ª ed. Dordrecht: Springer, 2015, pp. 83-91.

[62] D. Sukman, "The institutional level of war", *The Strategy Bridge*, 2016. [En línea]. Disponible en: https://bit.ly/34na1Td.

[63] G. Singhal, B. Bansod y L. Mathew, "Unmanned Aerial Vehicle classification, applications and challenges: A review", *Preprints*, 2018. [En línea]. Disponible en: doi:10.20944/preprints201811.0601.v1.

[64] "Tipos de drones", *HispaDrones*, 2019. [En línea]. Disponible en: https://bit.ly/3HwufYN.

[65] "UAVOS fixed-wing UAV Sitaria completed flight tests", *UAVOS*, 2018. [En línea]. Disponible en: https://bit.ly/335dHbC.

[66] "Northrop Grumman MQ-4C Triton", *Wikipedia. The Free Encyclopedia*, 2021. [En línea]. Disponible en: https://bit.ly/3ow5R2h.

[67] "Northrop Grumman MQ-8 Fire Scout", *Wikipedia. The Free Encyclopedia*, 2021. [En línea]. Disponible en: https://bit.ly/3snePQh.

[68] J. Brown, "Drones reviews. Yuneec Tornado H920: at the cusp of copter technology", *My Drone Lab*, 2017. [En línea]. Disponible en: https://bit.ly/3B1clLv.

[69] M. Hassanalian y A. Abdelkefi, "Classifications, applications, and design challenges of drones: a review", *Progress in Aerospace Sciences*, vol. 91, pp. 99-131, may. 2017. [En línea]. Disponible en: https://doi.org/10.1016/j.paerosci.2017.04.003.

[70] "Drone technology", *Builtin*, 2022. [En línea]. Disponible en: https://builtin.com/drones.

[71] "15 aplicaciones de los drones en nuestra sociedad", *Pymealdía*, 2016. [En línea]. Disponible en: https://bit.ly/3osoKCY.

[72] R. Ragel de la Torre, "Modelado, control y simulación de un quadrotor equipado con un brazo manipulador robótico", Proyecto fin de carrera, Dpto. de Ingeniería de Sistemas y Automática, Universidad de Sevilla, Sevilla, España, 2012. [En línea]. Disponible en: https://bit.ly/3urReR9.

[73] "Mavic Air 2", *DJI Tienda*, 2021. [En línea]. Disponible en: https://bit.ly/3HzxVJk.

[74] E. D. Nieto Guerrero y F. A. Vaca De La Torre, "Desarrollo de un modelo matemático, cinemático y dinámico con la aplicación de software, para modificar el funcionamiento de un dron, para que este realice monitoreo automático", *Recimundo*, vol. 4, pp. 332-343, mar. 2020. [En línea]. Disponible en: `https://bit.ly/3ovXWBQ`.

[75] M. A. Gómez Tierno, M. Pérez Cortés y C. Puentes Márquez, "Ecuaciones generales del movimiento", en *Mecánica del vuelo*, 2ª ed. Madrid: Garceta, 2012, pp. 17-32.

[76] J. Estévez Sanz, "Quadrotor team modeling and control for DLO transportation", Tesis doctoral, Dpto. de Ciencias de la Computación e Inteligencia Artificial, Universidad del País Vasco, Donostia - San Sebastián, España, 2016. [En línea]. Disponible en: `http://hdl.handle.net/10810/21886`.

[77] "Yaw, pitch, roll and omega, phi, kappa angles", *PIX4D*, 2021. [En línea]. Disponible en: `https://bit.ly/3uw35Om`.

[78] C. Santoro, "How does a quadrotor fly? A journey from physics, mathematics, control systems and computer science towards a «Controllable flying object»", Autonomous and Robotic Systems Laboratory (ARSLAB), Dpto. de Matemáticas e Informática, Universidad de Catania, Italia, 2014. [Presentación de diapositivas]. Disponible en: `https://bit.ly/3Hzsj1M`.

[79] M. A. Gómez Tierno, M. Pérez Cortés y C. Puentes Márquez, "Sistemas de referencia", en *Mecánica del vuelo*, 2ª ed. Madrid: Garceta, 2012, pp. 1-16.

[80] J. L. Fernández, "Movimiento y sistemas de referencia", *FisicaLab*, 2013. [En línea]. Disponible en: `https://bit.ly/3J8GATz`.

[81] C. Tytler, "Modeling vehicle dynamics – Euler Angles", *Autonomy in motion*, 2017. [En línea]. Disponible en: `https://bit.ly/3GxoIjJ`.

[82] C. Tytler, "Modeling Vehicle Dynamics – Quadcopter Equations of Motion", *Autonomy in motion*, 2017. [En línea]. Disponible en: `https://bit.ly/3HJDS6w`.

[83] R. Guardeño Ramírez, "Modelado, simulación, diseño y construcción del sistema de control de un quadrotor", Trabajo de Fin de Grado, Grado en Ingeniería Electrónica Industrial, Universidad de Cádiz, 2016. [En línea]. Disponible en: `http://hdl.handle.net/10498/18548`.

[84] R. Rico, P. Maisterra, M. Gil-Martínez, J. Rico-Azagra y S. Nájera, "Identificación experimental de los parámetros de un cuatrirrotor", *Actas de las XXXVI Jornadas de Automática*, Bilbao, pp. 973-982, sep. 2015.

[85] *MathWorks*, 2022. [En línea]. Disponible en: `https://es.mathworks.com/`.

[86] "MATLAB", *Wikipedia. La Enciclopedia Libre*, 2021. [En línea]. Disponible en: `https://bit.ly/35NqccF`.

[87] "Simulink", *Wikipedia. The Free Encyclopedia*, 2021. [En línea]. Disponible en: `https://bit.ly/3JaKPxM`.

[88] "Reinforcement Learning Toolbox", *MathWorks*, 2022. [En línea]. Disponible en: https://bit.ly/3ovYtnk.

[89] "Help Center. Training and Validation — Examples", *MathWorks*, 2022. [En línea]. Disponible en: https://bit.ly/3rtur5M.

[90] "Formación en MATLAB y Simulink. Curso introductorio Reinforcement Learning Onramp", *MathWorks*, 2022. [En línea]. Disponible en: https://bit.ly/35NbfHD.

[91] "Help Center. Create Simulink environment and train agent", *MathWorks*, 2022. [En línea]. Disponible en: https://bit.ly/3spPuWd.

[92] K. Chun-Wei, "Neural Newtork Controller", 2022. [En línea]. Disponible en: https://bit.ly/3rvn8dK.

[93] T. Luukkonen, "Modelling and control of quadcopter", Proyecto independiente en matemáticas aplicadas, Aalto University School of Science, Espoo, 2011.

[94] D. Davis, "MATLAB Answers. Noise parameters in Reinforcement learning DDPG", *MathWorks*, 2019. [En línea]. Disponible en: https://bit.ly/3B2imHO.