# Borrador Memoria TFG

***Entorno de simulación para el entrenamiento mediante Reinforcement Learning del vuelo autónomo de un cuadricóptero***

**Índice**

1. *Introducción*
2. *Objetivos*
3. *Resultados*
4. *Conclusiones*
5. *Bibliografía*

*Anexos:*

* *Estado del Arte*

## INTRODUCCION

Una de las tecnologías que más destacan y llaman la atención en estos últimos años son los llamados drones, definidos formalmente como *aeronaves no tripuladas* por la RAE [http://dle.rae.es/?id=ED2QqnQ]. Estos aparatos se han convertido en una atracción para casi cualquier ámbito, comenzando en usos militares y llegando a ser juguetes de ocio, no sin pasar por usos comerciales como el transporte o relacionados con seguridad y defensa.

Su bajo coste, reducido tamaño y la posibilidad de manejarlos remotamente les da un potencial inmenso para una infinidad de usos, entre otras cosas, en tareas que debería hacer una persona pudiendo tener que asumir riesgos o peligros, como puede ser la vigilancia desde el aire en zonas de conflicto, la búsqueda de supervivientes tras una catástrofe natural o la actuación en incendios forestales.

Sin embargo, para algunas tareas los recursos humanos son limitados y no es viable ocupar a una persona pilotando un dron, creándose de esta forma la necesidad de construir aeronaves que puedan llevar a cabo su cometido de forma autónoma.

Volar de forma autónoma es una tarea muy compleja en la que pueden suceder innumerables situaciones diferentes e imprevisibles, a las que se debe dar respuesta de la forma más rápida y acertada posible. Es necesario dotar a los drones de inteligencia para que puedan desenvolverse por sí mismos y aprender de su propia experiencia.

En este trabajo fin de grado se pretende experimentar con el uso de un entorno de simulación [ Referencia a Anexo Simuladores ] para entrenar un cuadricóptero con técnicas basadas en *Reinforcement Learning* [ Referencia a Anexo RL ] y conseguir que este sea capaz de volar de forma autónoma en la realidad.

## OBJETIVOS

Los objetivos son probar el simulador con la configuración de PX4 y ROS, y hacer un experimento con él para adentrarnos en el mundo de los vehículos autónomos.

Lo que haremos será bla bla bla…

1. Montar el simulador
2. Crear el software de RL
3. Entrenar el dron en el simulador
4. Probar el dron en vuelo real

## DESARROLLO DEL PROYECTO

Este proyecto se ha desarrollado en dos fases bien diferenciadas:

La primera, montar y poner a funcionar el simulador:

* Instalar AirSim, PX4 y ROS, y hacer que funcionen juntos
* Probar un software de terceros para probar el funcionamiento del simulador

La segunda, crear el sistema autónomo y llevarlo a la practica

* Diseñar el software de vuelo autónomo
* Entrenar el dron en el simulador
* Migrar el sistema a un dron real para hacer pruebas en vuelo real

## RESULTADOS

Comentar cómo ha finalizado el proyecto, qué nos ha llevado más trabajo, que complicaciones han aparecido y como las hemos solucionado.

Voló o no voló?

## CONCLUSIONES

Se han cumplidos los objetivos?

Conclusión sobre el entorno de simulación: esta chulo el AirSim con PX4 y ROS?

Conclusión sobre el sistema autónomo de RL: pues al final el dron acabó en un árbol jeje :P

## BIBLIOGRAFIA

## ANEXO I: ESTADO DEL ARTE

Reescribir intro al estado del arte:

En el pasado como se hacían los drones autónomos? Fue la aparición de los simuladores la que permitio empezar a experimentar con estas técnicas?

### Simuladores

Hablar de HITL y SITL – donde meterlo?

Desarrollar y experimentar con algoritmos para vehículos autónomos es un proceso muy complejo que requiere tiempo y muchas pruebas que no siempre salen bien. Además, para desarrollar sistemas inteligentes a menudo es necesario recopilar una gran cantidad de datos de entrenamiento en diferentes condiciones y entornos. Realizar este proceso en el mundo real no es viable, por ello para llevarlo a cabo se utilizan entornos de simulación, sistemas informáticos que procuran imitar lo más fielmente posible el comportamiento que tendría el vehículo en un entorno real. Este tipo de software nos permite experimentar, analizar y entrenar a nuestros aparatos evitando los importantes inconvenientes que implican hacer lo propio en la realidad: accidentes, golpes, costosas roturas, desajustes, etc.

*Datos comparativos entre simuladores sacados del paper de AirSim* ***[*** <https://www.microsoft.com/en-us/research/wp-content/uploads/2017/07/1705.05065.pdf> ***]***

A día de hoy existen variedad de entornos de simulación para drones, entre ellos, uno de los más utilizados es Gazebo [WEB <http://gazebosim.org/tutorials?tut=guided_b1&cat=#WhatisGazebo?> ] un simulador de código abierto para todo tipo de robots, capaz de simular ambientes interiores y exteriores complejos con alto grado de fidelidad. Provee una amplia biblioteca de robots y entornos, gran cantidad de sensores y plugins para integrar otras tecnologías. Una de las grandes ventajas que tiene utilizar Gazebo es la amplia comunidad de usuarios que participan activamente y la gran cantidad de documentos académicos basados en la experimentación con este simulador.

Las principales desventajas de Gazebo son su orientación a robots genéricos y la complejidad para crear entornos realistas a gran escala y con gran cantidad de detalles que simulen mejor el mundo real.

Una alternativa que pretende solventar la falta de realismo es Hector [PAPER <https://pdfs.semanticscholar.org/3ed3/948827e0949770e8583b51bd0fedf4fd73fe.pdf>], un trabajo que integra el middleware ROS [ Referencia (anexo? Web?) ] con el simulador Gazebo orientado a cuadcópteros. Lo que ofrece Hector [ fuente <http://wiki.ros.org/hector_quadrotor> ] es una simulación más realista del comportamiento del aparato, pero por el contrario carece de soporte para plataformas de autopilot como PX4 y protocolos de comunicación estándares como MavLink [ WEB <http://qgroundcontrol.org/mavlink/start> ].

De forma similar, RotorS [ PAPER <https://link.springer.com/chapter/10.1007/978-3-319-26054-9_23> ] es un framework modular de simulación que permite diseñar drones y desarrollar algoritmos para controlar su comportamiento y estimar su estado de forma más precisa. Al igual que en el caso anterior, RotorS [ fuente <http://wiki.ros.org/rotors_simulator> ] se basa en dos módulos: ROS y Gazebo (como entorno gráfico el segundo, con lo que ello implica), pero en este caso sí se soporta la integración de Pixhawk en modo HITL.

Ya lejos de Gazebo existe jMAVSim [ WEB <https://pixhawk.org/dev/hil/jmavsim> ], un sencillo entorno de simulación diseñado con el objetivo de experimentar con el firmware de la plataforma de autopilot Pixhawk. Se trata de un entorno muy ligero y fácil de usar que consta de un motor de renderizado muy simple, imposibilitando la generación de objetos en la escena.

Otros entornos de simulación como son DroneSimLab [WEB <https://ir.canterbury.ac.nz/bitstream/handle/10092/14677/WSCG2017.pdf?sequence=2&isAllowed=y> ], Sim4CV [WEB <https://arxiv.org/pdf/1708.05869.pdf> ] o Microsoft AirSim [ WEB <https://www.microsoft.com/en-us/research/wp-content/uploads/2017/07/1705.05065.pdf> ] son entornos orientados a drones y otros vehículos principlamente, y utilizan el motor de videojuegos Unreal Engine como base, incorporando así entre sus características más notables el realismo en los gráficos y en la física del comportamiento de los vehículos, gracias a las avanzadas tecnologías desarrolladas por Unreal.

Microsoft AirSim [ WEB <https://www.microsoft.com/en-us/research/wp-content/uploads/2017/07/1705.05065.pdf> ] forma parte del proyecto *Aerial Informatics and Robotics Platform* [ WEB <https://www.microsoft.com/en-us/research/project/aerial-informatics-robotics-platform/> ] desarrollado por Microsoft, y consiste en un entorno de simulación basado en el motor Unreal Engine [ WEB <https://www.unrealengine.com/en-US/what-is-unreal-engine-4> ] y un conjunto de herramientas para el desarrollo de sistemas autónomos. Se trata de una plataforma que cuenta con las tecnologías más avanzadas de simulación en cuanto a física de los vehículos y visualización del entorno, para reducir al máximo la diferencia entre realidad y simulación, ayudando así a crear sistemas autónomos más avanzados, seguros y eficaces.

AirSim es una plataforma modular con énfasis en la escalabilidad, dando la posibilidad de integrar nuevos tipos de vehículos, plataformas hardware y protocolos software. Proporciona soporte para MavLink, simulación SITL y HITL, integración con ROS y autopilots como PX4.

La reciente novedad de AirSim y su apuesta por la muy alta fidelidad de la simulación, además de la menor presencia de trabajos realizados con esta plataforma y el prestigio de la propia compañía, han sido los factores que nos han impulsado a elegir este entorno de simulación para realizar este trabajo de fin de grado.

Existen muchos más entornos de simulación, pero la mayoría están orientados a robots de tipo genérico (como ARGoS [WEB <http://www.argos-sim.info/index.php> ] o V-REP [WEB <http://www.coppeliarobotics.com/index.html> ]), por lo que no incorporan una simulación realista del comportamiento de un dron, o bien no son de código abierto (como XPLANE [WEB <http://www.x-plane.com/> ] o RealFlight [WEB <https://www.realflight.com/> ]).

### Algoritmos de RL

Hablar de 3 puntos esenciales:

* Que es el RL y que utilidad general tiene (brevemente, solo como intro)
* Que algoritmos hay en RL y para que se utiliza cada uno
* Aplicación del RL a los vehículos autónomos + concluir con que algoritmo vamos a utilizar

El *Reinforcement Learning (RL)* o Aprendizaje por Refuerzo es un área del *Machine Learning* que permite a un agente descubrir un comportamiento óptimo de forma autónoma a través de interacciones de prueba y error con el entorno que lo rodea. En lugar de proporcionarle al agente la solución a un problema, en el RL el entorno le devolverá una recompensa positiva (refuerzo) o negativa (castigo) al agente según la acción que haya llevado a cabo. Con esto, el agente aprenderá qué tipo de conductas son las que debe mantener para obtener recompensas cada vez más positivas.

La razón de utilizar este tipo de aprendizaje para desarrollar sistemas autónomos reside en la inmensa complejidad que supone programar las infinitas situaciones que se pueden dar para un vehículo y que debe saber resolver. Ante esta complejidad, resulta muy atractivo el proceso de crear un sistema que pueda generar ese conocimiento por sí mismo a través de la asociación de relaciones causa-efecto de su propia experiencia. Por ende, en este trabajo fin de grado se aplicarán técnicas *Reinforcement Learning* para diseñar el sistema de navegación autónoma.

[ /Î\ Referencias:

Paper <http://www.ias.tu-darmstadt.de/uploads/Publications/Kober_IJRR_2013.pdf>

Paper <https://arxiv.org/pdf/cs/9605103.pdf>

Libro: Artificial Intelligence, A Modern Approach

Libro: Reinforcement Learning, An Introduction]

### Experimentos de vuelo autónomo

Otros experimentos relacionados que ya se hayan hecho, sean más o menos avanzados.

* El experimento de Microsoft con redes neuronales
* La tesis de keras-rl y airgym
* Paper: Autonomous Navigation of UAV by Using Real-Time Model-Based Reinforcement Learning (TEXPLORE)
* Paper: Autonomous UAV Navigation Using Reinforcement Learning

### Conclusión

Sabiendo todo esto, lo que nosotros pretendemos es aprender y adentrarnos en este mundo a través de un experimento que de entrada sabemos que es factible hacer, y que se basa en bla bla bla…

## / / / / / / / / / / / /

## INTRODUCCION

Presentar las aplicaciones del proyecto: ¿para que va a servir?

------

Hoy en día la experimentación con drones y la implantación de inteligencia en estos vehículos está en auge. Sin embargo, por múltiples motivos (riesgo de posibles accidentes debidos a fallos o pérdidas de control, gasto de energía, tiempo necesario para llevar a cabo el entrenamiento, etc) la experimentación directa sobre estos aparatos no es la opción más viable para desarrollar prototipos experimentales.

En respuesta a esta necesidad, este proyecto pretende poner a funcionar un entorno de simulación realista y gratuito que, no solo sirva para experimentar con prototipos, sino que además facilite la migración del sistema creado a un dron real para poder probarlo.

## OBJETIVOS

El objetivo de este proyecto es construir un entorno de simulación a partir de la integración de herramientas de código abierto ya existentes. A continuación, se realizará una experimentación sobre este entorno para probar su utilidad, entrenando el comportamiento de un dron para que aprenda a volar de forma autónoma. Se utilizarán técnicas de **aprendizaje por refuerzo**.

El motor de la simulación en sí constará de dos partes: en primer lugar el entorno 3D, para el cual utilizaremos el motor de videojuegos Unreal Engine 4; y en segundo lugar, el software de simulación de comportamiento del cuadricóptero, para el cual utilizaremos Microsoft AirSim.

Para interactuar con el vehículo y poder entrenarlo utilizaremos dos herramientas que se integrarán entre ellas: por un lado, un *autopilot* que controlará el comportamiento básico del vehículo y que será el *Pixhawk* (PX4); y por otro lado, también utilizaremos ROS (Robot Operating System), un sistema operativo que nos facilitará el manejo del aparato y de sus componentes añadidos (como una cámara, para emplear visión artificial).

El potencial de este método reside en que el software generado, se integra de la misma forma en el entorno de simulación y en el entorno real del dron. Esta característica es la que nos permite migrar el contenido de un punto al otro sin apenas modificar nada.

## RESULTADOS

## CONCLUSIONES

## BIBLIOGRAFIA

## ANEXO I: ESTADO DEL ARTE

Por que se utilizan entornos de simulación para drones? Porque entrenar un dron exige muchas pruebas y errores, lo cual conlleva golpes y roturas que cuestan dinero y desajustan los aparatos. Además, entrenar a un dron exige hacer muchas muchas pruebas, que en vuelo real llevarían mucho más tiempo que en un entorno simulado.

Que simuladores existen? Gazebo, ARGoS, V-REP y el reciente AirSim.

Pegas de los tres primeros? Los gráficos y el comportamiento de los aparatos es menos realista. Además, están orientados a cualquier tipo de robot.

Opción escogida: Microsoft AirSim.

Por que utilizar AirSim? Por que es una plataforma para desarrollo e investigación en IA.

Además, está orientado a drones y tiene un comportamiento muy realista.

Es la novedad, se hizo open source a principios de febrero del año pasado.

----

Hablar de proyectos en los que se utilice el entrenamiento de drones con aprendizaje automatico???

Hablar de Reinforcement Learning