SANTO: Social Aerial NavigaTion in Outdoors

A platform for faster AI prototyping in drones

Author: Joaquin Terrasa Moya
Director: Anais Garrell Zulueta

October 2019

Contents

0.1 Acknowledgements

To my family, who have been very supportive throughout the bachelor's degree and specially through the ups and downs of this project. To the researchers, including the director of this project, of the department I have been able to work close with, who have invested their time and dedication in helping me to understand the concepts and apply them properly.

0.2 Summary

In recent years, the advances in remote connectivity, miniaturization of electronic components and computing power has led to the integration of these technologies in daily devices like cars or aerial vehicles. From these, a consumer-grade option that has gained popularity are the drones or unmanned aerial vehicles, namely quadrotors. Although until recently they have not been used for commercial applications, their inherent potential for a number of tasks where small and intelligent devices are needed is huge.

However, although the integrated hardware has advanced exponentially, the refinement of software used for these applications has not beet yet exploited enough. Recently, this shift is visible in the improvement of common tasks in the field of robotics, such as object tracking or autonomous navigation. Moreover, these challenges can become bigger when taking into account the dynamic nature of the real world, where the insight about the current environment is constantly changing.

These settings are considered in the improvement of robot-human interaction, where the potential use of these devices is clear, and algorithms are being developed to improve this situation. By the use of the latest advances in artificial intelligence, the human brain behavior is simulated by the so-called neural networks, in such a way that computing system performs as similar as possible as the human behavior.

To this end, the system does learn by error which, in an akin way to the human learning, requires a set of previous experiences quite considerable, in order for the algorithm to retain the manners. Applying these technologies to robot-human interaction do narrow the gap.

Even so, from a bird's eye, a noticeable time slot used for the application of these technologies is required for the curation of a high-quality dataset, in order to ensure that the learning process is optimal and no wrong actions are retained. Therefore, it is essential to have a development platform in place to ensure these principles are enforced throughout the whole process of creation and optimization of the algorithm.

In this work, multiple already-existing handicaps found in pipelines of this computational gauge are exposed, approaching each of them in a independent and simple manner, in such a way that the solutions proposed can be leveraged by the maximum number of workflows.

On one side, this project concentrates on reducing the number of bugs introduced by flawed data, as to help the researchers to focus on developing more sophisticated models. On the other side, the shortage of integrated development systems for this kind of pipelines is envisaged, and with special care those using simulated or controlled environments, with the goal of easing the continuous iteration of these pipelines.

0.3 Resumen

En los últimos años, los advances en telecomunicaciones, miniaturización de componentes hardware y el incremento del poder computacional han permitido la integración de estas tecnologías en dispositivos comunes, como coches o vehículos aéreos. Entre estos últimos, una opción popular al alcance del consumidor son los drones o Vehículos Aéreos no Tripulados (VAT). Aunque, hasta hace poco, el interés para el uso profesional de estos dispositivos ha sido bajo, su valor como herramientas inteligentes y de dimensiones reducidas es único.

Aún así, aunque la complejidad del hardware haya avanzado exponencialmente, las posibilidades de los algoritmos que puedan aprovechar estas propiedades es limitada. Recientemente, se ha podido ver esta tendencia en la mejora de tareas comunes en el mundo de la robótica, como pueden ser la búsqueda de objetos en el entorno o la navegación autónoma. Además, estos retos se acrecientan cuando tenemos en cuenta la naturaleza dinámica de la realidad, en la que la percepción del entorno es variable en todo momento.

Estos contextos son enfatizados en la mejora de la interacción robot-humano, donde el potencial de uso de estos dispositivos es claro, y se trabaja en algoritmos que mejoren esta interacción. Mediante el uso de los últimos avances en intelligencia artificial, se trata de emular al cerebro humano, en las llamadas redes neuronales, de forma que el sistema computacional logre asemejarse lo máximo al comportamiento humano.

Para ello, el sistema se basa en el aprendizaje por error que, de forma similar al aprendizaje humano, requiere un conjunto de experiencias de aprendizaje notable, para que el algoritmo memorice adecuadamente el comportamiento. Aplicando estas tecnologías a la interacción robot-humano, se consigue que la brecha se reduzca.

Aún así, a gran escala, una notable parte del tiempo dedicado a aplicar estas tecnologías se concentran en generar un conjunto de datos de calidad, de modo que el aprendizaje sea óptimo y no memorice las acciones equivocadas. Por ello, es fundamental disponer de una plataforma de desarrollo para que estos principios prevalezcan a lo largo de la creación y optimización del algoritmo.

En este trabajo, se exponen varios impedimentos encontrados durante el desarrollo de estos modelos computacionales, tratando de solucionarlos de manera independiente y simple, de forma que pueda ser adaptado al mayor número de rutinas de trabajo.

Por un lado, se busca reducir el número de errores introducidos por el uso de datos erróneos, de forma que los investigadores centren sus esfuerzos en el desarrollo de modelos más sofisticados. Por otro lado, se ataca la carencia de sistemas de desarrollo integrados, en específico, para el desarrollo en entornos simulados o controlados, con el fin de facilitar el prototipado de estos modelos.

0.4 Resum

En els darrers anys, els avanços en telecomunicacions, miniaturització de components hardware i l'increment del poder computacional han permès l'integració de aquestes tecnologias en dispositius diaris, com poden ser els cotxes o els vehicles aeris no tripulats. Malgrat que fa poc que l'interès per a l'ús professional d'aquestes dispositius ha estat sota, el seu valor com a eines intel · ligents i de dimensions reduïdes és destacable.

Tot i així, encara que la complexitat del hardware hagi evolucionat exponencialment, les possibilitats dels algoritmes que puguin aprofitar aquestes característiques és limitada. A més, cada vegada amb més freqüència es pot veure aquesta tendència a la millora de tasques comunes en el món de la robòtica, com poden ser la recerca d'objectes en l'entorn o la navegació autònoma. Si també tenim en compta que aquests reptes augmenten quan tenim en compte la naturalesa dinàmica de la realitat, en la qual la percepció de l'entorn és variable en tot moment, la complexitat es pot veure incrementada.

Aquests contextos són emfatitzats en la millora de la interacció robot-humà, on el potencial d'ús d'aquests dispositius és clar, i es treballa en algorismes que milloren aquesta interacció. Mitjançant l'ús dels últims avenços en intel · ligència artificial, es tracta d'emular al cervell humà, en les anomenades xarxes neuronals, de manera que el sistema computacional aconsegueixi assemblar-se el màxim al comportament humà.

Amb aquesta finalitat, el sistema es basa en l'aprenentatge per error que, de manera semblant a l'aprenentatge humà, requereix d'un conjunt d'experiències d'aprenentatge significatiu perquè l'algoritme memoritzi adequadament el comportament. Aplicant aquestes tecnologies a la interacció robot-humà, s'aconsegueix que la bretxa es redueixi.

Així, a vista d'ocell, una notable part del temps dedicat a aplicar aquestes tecnologies es concentren en generar un conjunt de dades de qualitat, de manera que l'aprenentatge sigui òptim i no memoritzi les accions equivocades. Per això, és fonamental disposar d'una plataforma de desenvolupament perquè aquests principis prevalguin al llarg de la creació i optimització de l'algorisme.

En aquest projecte, s'exposen diversos impediments trobats durant el desenvolupament d'aquests models computacionals, tractant de solucionar-los de manera independent i senzilla, de manera que pugui ser reutilitzat per al major nombre de rutines de aprenentatge.

D'una banda, es busca reduir el nombre d'errors introduïts per l'ús de dades incorrectes, de manera que els investigadors centrin els seus esforços en el desenvolupament de models més sofisticats. D'altra banda, es treballa la manca de sistemes de desenvolupament integrats, en específic, per al desenvolupament en entorns simulats o controlats, per tal de facilitar el prototipat d'aquests models.

0.5 Glossary

IDE: Integrated Development Environment

API: Application Programming Interface

SDK: Software Development Kit

SITL: Software-In-The-Loop

HITL: Hardware-In-The-Loop

UAV: Unmanned Aerial Vehicle

ML: Machine Learning

AI: Artificial Intelligence

NN: Neural Network

DNN: Deep Neural Network

DL: Deep Learning

RNN: Recurrent Neural Network

1 Context and scope of the project

In this section, the project's goal is introduced by providing context and previous working solutions to the reader in order to understand it and all the implication it has for the project's scope. In addition, the possible obstacles and risks to be found are detailed, as well as the proposed ways to prevent or minimize them.

1.1 Context and problem formulation

The present work introduces a novel platform for aerial mobile robotics, namely quadrotor drones, that focuses on improving the capabilities of existing ones in machine-learning pipelines. This application emerges from the need of a software overlay identified during the research of autonomous systems for drones by applying deep-learning techniques. But before diving into the details of the proposed solution, it is required to give a short introduction about the core ideas and the context of this work.

Since the middle of the XIX century, continuous advances in computing, electronic and mechanical engineering have sought the dream of the modern humanity: the automation of repetitive or error-prone tasks, like dish-washing, cooking or driving. For specific roles, such as plane steering or bank transactions, sophisticated algorithmic systems have been designed in such a way that little or no help is needed to complete the task without errors.

Nowadays, the huge advance that this field has experienced, paired with equally-sized progress in computer science and electronics, has allowed us to build and use the smallest intelligent devices ever done. From this wide spectrum, the work undertaken here explores the possibilities of *Unmanned Aerial Vehicles* (UAV), in particular *quadrotors*. The popularity and use of these devices have taken off recently, thanks to its small size, on-board computing power, available sensors and battery life. However, one ability that still is far from desirable is *autonomous navigation*.

Furthermore, UAV navigation in daily outdoors environments, such as a city neighborhood or a crowded event, is even more challenging. Hence, there is a strong need to enhance human-drone interaction, in a way that humans feel more comfortable when pairing with drones. However, these systems rely on complex systems that are created by processing big quantities of data, which certainly are not reachable for certain use cases. In the present work, an approach to tackle this need is developed, exploring the resources of previous works in the field, and proving its profits by means of the same techniques.

In recent times, *Machine-Learning* (ML) algorithms have been proved to enhance existing solutions in a very broad range of applications. This field has benefit from extensive research in the last decade, successively applying ideas from biology and psychology to develop algorithmic frameworks that behave like humans. One of these frameworks, called *Neural Networks* (NN), mimics the neuron behavior from the brain and tailors it to build more accurate algorithms. This way, by sequentially connecting layers of neurons to some input, like a photography, an application can detect if there is a cat or not in the picture (Le 2013).

A key property of these structures is that they can keep the problem definition from the user side simple, at the expense of requiring way bigger databases of environment data to learn the complex parts blocks of the problem. This characteristic has allowed to speed-up the research and application of these model architectures to consumer-level products, however the process of obtaining these databases is still challenging.

Multiple companies(Palossi et al. 2018) (Kendall et al. 2017) have successfully launched drones that seize the power of neural networks on-board. However, the uses for these drones do not consider crowded environments. To this end, this project proposes a system to tackle human interactions based on the *Social Force Model* (SFM) (Helbing and Molnár 1995), which takes human interaction into account. This work was further developed for aerial vehicles (ASFM)

Figure 1: Parrot Anafi 4K drone

(Garrell Zulueta and Al 2017) and lately it has been explored with neural networks (NASFM) (Coll Gomila 2018).

The scope of this project is to further enhance the latter model, by focusing on the need of a robust quality for training data, which is essential in critical applications like human-drone interaction, and leveraging the sensory palette offered by high-end drones. This task, although seemingly hard at first, is more feasible to develop thanks to the use of middleware software that ensures the platform's development and deployment faster.

1.2 Stakeholders

Stakeholders are any person, institution or organization interested in the project, as it will report some kind of profit to them. These groups are fundamental to the project as they will be the recipients of the final state of the project.

1.2.1 Consumers

The project itself is focused on developing and improving on an existing development platform. Therefore, its main consumer group are researchers and developers who wish to improve its current implementation, or develop novel ones, by using this solution.

Moreover, these stakeholders will also be able to benefit from the possibility of expansion of these modules, making easier the configuration and refinement of the computing models.

1.2.2 End Users

Individuals, businesses and other institutions alike will be able to benefit from the scheme presented in this project, using the research material given or porting it to new devices. Moreover, this also includes Parrot("Parrot," n.d.), the business that sells the drone model used in this project, as it can be used as a plugin.

1.2.3 Beneficiaries

Either individuals or corporations that use UAVs for specific tasks, like video shooting or security. Ultimately, the whole society could benefit indirectly from this advances, as trained models using the scheme should prove to have more resilient behaviours.

1.3 State-of-the-art

To further realize how far the previous developments in the topic have gone and their how can this project fit in the timeline, it is essential to survey the most remarkable works. Nowadays, human-robot interaction is a very active research field, however, the research on aerial motion planning in the presence of humans is still new (H. Y. Kim, Kim, and Kim 2016) compared to other robotics research topics.

In recent years, the research community has gained interest in aerial robots, due to recent advances in *Unmanned Aerial Vehicles* and quadrotor technology, and also due to the huge potential of new applications that could derive from them. Most of the effort has been dedicated to the development of new navigation techniques for aerial robots. However, due to the payload limitations of aerial robots, the most recent navigation solutions are based on control models learned from monocular camera input.

The authors of (Giusti and Al 2016) successfully taught a quadrotor micro aerial vehicle equipped with just a monocular camera to follow forest trails, using a deep CNN. This work was extended in (Smolyanskiy and Al 2017) by augmenting the original dataset, incorporating 3 additional output categories and using an improved deep CNN, which provided increased accuracy and computational efficiency.

Other researchers (D. K. Kim and Chen 2015) have used a deep CNN to teach a Micro Aerial Vehicle a controller strategy, which mimics an expert pilot choice of action so that the quadcopter can autonomously navigate indoors and can find a specific target. A similar work can be found in (Zhang and Al 2016), where a simulated quadcopter successfully learns obstacle avoidance policies by training a Deep Neural Network with simulated raw sensor inputs.

Likewise, the authors of (Loquercio and Al 2018) successfully taught an UAV to safely navigate in the streets of a city by training a deep CNN with self-driving cars data. The CNN was able to accurately predict steering angle as well as probability of collision, and the learned policy generalized to indoor environments.

Differing from the previous approaches, a team of researchers (Martinez, Black, and Romero 2017) developed a scalable RNN architecture that can predict future human motion. Moreover, at (Lee et al. 2017) it is proven that RNN architectures are also valid for distant motion prediction, and even (Altche and De La Fortelle 2018) for frenetic, crowded paths.

In a similar fashion, (Alahi et al. 2016) poses a RNN model, namely a LSTM model, that predicts motion dynamics for humans in crowded spaces. This is further extended by the works of (Robicquet et al. 2016), which explores different human-human interactions, and (Sadeghian et al. 2018), whose model successfully predicts human paths compliant to social constraints.

To this day, the most similar approach that tackles the problem of an aerial robot that accompanies a human in populated environments is a previous work of the *Institute of Robotics and*

Informatics at UPC(Coll Gomila 2018) , where the authors apply a NN-based workflow that allows the drone to learn from expert trajectories, in order to safely fly in indoor spaces. The approach proposed in this project learns from the drawbacks exposed in the latter work and tries to fill the gaps towards a better model training.

1.4 Scope

This work draws from the latter project and essentially is focused on solving some of its weaknesses. Therefore, the overall goal is to provide an interface that is closer to the researcher in terms of data quality and faster prototyping. Hence, the project objectives should attain:

- To build a platform from scratch that relies on well-tested architectures but is able to be extended, and provides means to the generation of robust synthetic databases and the abstraction of common routines.
- To understand how to leverage applications, suited for limited computing power and battery life, that are truly autonomous.
- To validate the system usefulness for real-world applications, by conducting experiments to prove the framework for the given use cases.

1.5 Possible obstacles

1.5.1 Learning curve of Machine-Learning methodologies

Even though this project is developed with previous foundations on Machine Learning and Neural Networks, these fields are so new and innovative that it is easy to get lost at some points.

To overcome this, advances courses and researches about these topics will be thoroughly studied prior and during the project development, as a way to ensure that recurrent errors are discarded.

1.5.2 Framework incompatibilities

Due to the use of multiple frameworks or software applications to do the project, relying these on different technologies, the mixture of them can represent a bottleneck in the development stage. Moreover, the global configuration of the underlying system can also suppose an obstacle to make these software blocks work together.

To deal with this, a *plugin* can be bootstrapped in a way that it serves as a workaround to overcome the situation, ensuring the correct functioning of the join. The improvement of this module is later added to the development roadmap as means of reduce the possible system overload.

1.5.3 Data generation inconsistency

Due to the inherent multivariate nature of the data collected, which is coming from different sources at diverse rates, the gathering and processing of them in a real-time-fashion can be challenging at some points of the development, where high-consuming processes are required to be ran.

This handicap can be surpassed with a thoughtful implementation of the data logging application in a way that it collects data given its timestamp, and mapping each source to a processing services.

1.5.4 Error-prone machine learning procedures

The importance of training and testing a machine-learning-based autonomous application with the abovementioned platform is key to prove its usefulness. However, some common pitfalls may be found while developing it, such as wrong learning from the search space or overfitting to the training data.

This situation shall be solved by following the techniques and guides stated by previous efficient models or the practice guides given by the development framework.

1.6 Risks

1.6.1 Scheduling

The original project timetable covered 9 months, but due to external restrictions, it had to be shrunk into 5 months. This reduces the available time to barely a half of it.

To overcome this problem, a tighter but manageable schedule is set, with weekly meetings with the project tutor, following agile methodologies, to both tackle delays, reschedules and results.

1.6.2 Implementation errors

As this project is novel in the way that it explores new solutions to a given problem, the software re-use and guides applied from existing projects in the same topic might introduce present errors or incompatibilities. This will be solved by focusing on current solutions, backed by the community, that are open to extensions and rely on fault-tolerant patterns.

Moreover, code errors (*bugs*) may be introduced in the software structure, as this is common in any software project. In order to improve the code quality and the error resilience, the project is built on industry-standard guides for machine learning and testing is implemented for each module to verify that the application is executed properly.

1.6.3 Development Platform incompatibilities

Even though the *Development Platform* works with the used drone model, it fully relies on open-source software, which is bug-prone. Therefore, at any point on the platform configuration and model development, we may get errors from either the hardware used or the base software stack, including the operating system, that will cause a configuration overload.

To reduce the impact of this problem, a minimal configuration will be used in order to reduce possible errors, even falling back to legacy or well-proven modules. If this is not possible, one (or more, if needed) scripting modules will be added to the setup, thanks to the ease of patching the aforementioned applications with Python.

1.7 Methodology

Given the rigid schedule this project holds, an agile methodology will be used, as it provides flexibility, rapid development and robust project guidelines. This eases the iterative, incremental development of the project by also taking into account the advices from the project manager. To track the project in an easier way, a set of milestones will be used:

1.7.1 Initial project integration

This milestone tracks if the initial project setup has been made; this is, all the required systems for the project has been configured, a thorough review of ML concepts has been done, and the drone device is ready to use.

1.7.2 Design of the platform is ready

This milestone checks if the study and design of the to-be-built platform has been done in a proper way, considering all the pros and cons. This allows us to compare it to similar solutions and provide a greater insight about the work, giving us a broader learning space.

1.7.3 Iterative project development

This milestone proves that the platform correctly covers the chosen properties and improves the base requirements, done progressively throughout stacking iterations. There will be an special focus on the pitfalls of the foundational pipeline.

1.7.4 Platform testing

This milestone checks if, when running a trained model against a test set, it can ensure that the trained model is performant to new situations.

1.8 Version tracking

The platform implementation follows a roadmap divided by milestones, allowing us to keep track of key features and to prioritize over other pipeline implementations. This also helps to maintain short development cycles, which ensures a more solid building of model foundations.

Git("Git," n.d.) and Github("Github," n.d.) are popular tools for version control. These tools, combined with a industry-standard branching model and best practices allow to ensure that milestone-guided development is correctly done.

1.9 Validation

Each phase will have a set of checks to pass in order to finish it. This checking will be done along with the project manager. Given the case that a check round is not fulfilled, the necessary restructuring will be made at the timetable with the aim of completing it. If it cannot be rescheduled due to time or resource constraints, it will be discarded.

2 Project planning

In order to reach the maximum spectrum of the proposed scope of the project, a task and time scheduling is undertaken. This will allow us to prevent possible downfalls beforehand, and take action according to them.

2.1 Planning and Scheduling

The estimated project duration is of about 5 months, namely since 1^{st} February 2019 until 23^{th} July 2019, being the required day for the project's submission. Moreover, we have to take into account that the initial project planning was set to 9 months; this time constraint binds the project to a tighter timetable. Finally, due to the untracked *IDE* situation, the project planning had to be changed in order to attain certain objectives within the given timespan.

As a way to break down the time constraint into simpler pieces, this project is developed as a bachelor course(Curs 2013), binded to college credits, namely 18, where each one is redeemed with 30 hours("UPC," n.d.). This leaves us with a total of 540 hours. However, the initial 90 hours are devoted to an initial course to prepare the documentation, leaving only 450 hours left. Moreover, as detailed before, an untracked risk during the progression of the project forced to rearrange the tasks set, having an extra time constraint of 120 hours lost. To tackle this lack, an extra bag of 100 hours was added.

Therefore, a total of 640 hours distributed over 127 days gives us an average of 5 hours per day.

2.2 Resources used

This project is developed with a specific platform in mind. As a result, all the tasks included in this project will use a set of shared resources.

2.2.1 Hardware resources

- Parrot Anafi drone.
- OptiTrack motion-capture system.
- Lab's tower PC

Intel® Core™ i5-4440 8 GB RAM DDR4 NVIDIA GeForce GTX 660

2.2.2 Software resources

- Ubuntu 16.04: used in all tasks.
- Python 2.7 and Python 3.5
- Parrot Sphinx, that is built on Gazebo simulator.
- Parrot Olympe, that uses Python 3.5 underneath.
- Tensorflow 1 with Keras high-level interface.
- ROS 1 and ROS 2, which provide a ROS interface for both Python 2 and Python 3, respectively.
- Pandoc: used at all stages for writing documentation in *Markdown* and publishing with *LaTeX*.
- Zotero: used to collect and feed the scientific citations to Pandoc.

2.2.3 Human resources

- Project manager: The project tutor will take on the role of project manager, offering continuous counsel and feedback for the project development and schedule.
- Software developer: The main contributor of this project itself can be disguised as a software developer, taking part in the core development of the project objectives, conducting quality testing and readjusting the development schedule if agreed with the project manager.

2.3 Task descriptions

Given the time and resources constraint, the tasks have been grouped into three (3) sections: *Context Analysis*, *Development* and *Validation*. Aside from these main blocks, a continuous task shared by all the three sections will be documenting.

2.3.1 Documentation

Included at all stages, the project's documentation is one of the most fundamental blocks, that allows to understand the overall research, to keep a consistent objective tracking, and to ease the learning curve of the advances made here, to further develop or implement them.

Namely, the documentation has two major stages: the documentation *skeleton* building, done at the *Preamble* stage, and the iterative documentation updating, done at the *Development* and *Optimization* stages. This task is developed throughout the whole project, nonetheless it does only require a few hours. This task also demands the lab computer, Pandoc, Zotero and Python as material resources. An amount of human resources to write and update the documentation files is also required.

2.3.2 Context Analysis

2.3.2.1 GEP course fulfillment

As a mandatory prerequisite for doing this project, the *GEP (Gestión de Proyectos)* course consists of building a solid base for the project's documentation. This task takes 6 weeks to be completed, and relies on human resources to write, review and prepare the project's documentation. Moreover, this task also uses Pandoc and Zotero as material resources.

2.3.2.2 Acquire background of Machine Learning pipelines

In order to get to know how to apply a successful machine learning workflow, a notion of artificial intelligence techniques is required. Following the project tutor recommendation, the *Coursera Deep Learning Specialization* (Andrew Ng 2017), imparted by Andrew Ng, a prominent figure in this field, provided us a comprehensive theoretical and practical background in *NN* and *DL*. This task does not use a material source, and it required several weeks to be completed, as it had been done in parallel with the other *Preamble* tasks. Human resources are needed to complete the online course.

2.3.2.3 Analysis of available development environments

To address the problematic of the lack of a development environment for applying an autonomous model for the drone, a thorough study of the available solutions and their costs had to be done. By documenting ourselves and comparing the platforms and their features, we were to determine if any of them fitted our requirements. This task does not use a material source, and it required several weeks to be completed, as it had been done in parallel with the other *Preamble* tasks.

Human resources were needed to undertake the search, comparison and selection of the chosen toolset.

2.3.2.4 System setup

Knowing the software and hardware stack to be used, a simple setup was essential to prove their usefulness. This setup consisted of easy tasks with the drone and the simulator, such as human control, raw data logging or environment configuration. This task required the drone, OptiTrack and the simulator as material resources, requiring a week to be completed, being done in parallel with the other *Preamble* tasks. Moreover, it also requires human resources to set up the latter tools.

2.3.3 Development

2.3.3.1 Design of the platform architecture

A thorough study and design of the platform scheme must be previously done to the development phase, as to make sure that the modules can co-operate between them and the risk of failure is minimized. Note that this task was undertaken in parallel with other *Development* tasks. This task took several weeks to be completed, as well as the simulator and Python as material resources. Moreover, human resources were needed to write down the application.

2.3.3.2 Development of the data processing module

To ensure that the data cleaning step is minimized from the beginning, as well as an eventual data corruption, a data processing application have been built to apply any modifications directly when the source is fed. Tightly related with this process is the real-time analysis of the platform's data streams, that can be used to keep watch of how the pipeline is evolving. This task took several weeks to be completed, as well as the simulator and Python as material resources. Moreover, human resources were needed to write down the application.

2.3.3.3 Development of the random world generator

To assess an unsuccessful learning due to data corruption from the source, a random world generator was built to ensure sample distributions from the used data can be set up and even replicated if the seed is kept. This task took several weeks to be completed, as well as the simulator and Python as material resources. Moreover, human resources were needed to write down the application.

2.3.3.4 Development of the control module

This straightforward task required to create an application to send takeoff, landing or move commands to the drone. An special emphasis was put on controlling exceptional cases, as this controller also takes care of the real model in case of an accident. This task took multiple week, and uses the simulator and Python as material resources. Human resources devoted to this task are used to develop and test the correct fulfillment of the requirements.

2.3.4 Validation

2.3.4.1 Platform validation

To ensure that the platform can help the iterative development and testing of a machine learning pipeline applied on drones, a straightforward example is built. This includes the setup of a synthetic database and the training of a simple neural-network-based model. This task requires

a week to be fulfilled. Human resources are needed to explore, tune and run the chosen options. The material resources needed are the the simulator and the platform.

2.4 Time table

Task	Estimated duration (h)
GEP course fulfillment	90
Uncommittable time due to project re-planning	**120**
Acquire background of Machine Learning pipelines	75
Analysis of available development environments	86
System setup	19
Design of the platform architecture	40
Development of the data processing module	40
Development of the random world generator	60
Development of the control module	40
Platform validation	10
Documentation	60
Total	**640**

2.5 Gantt chart

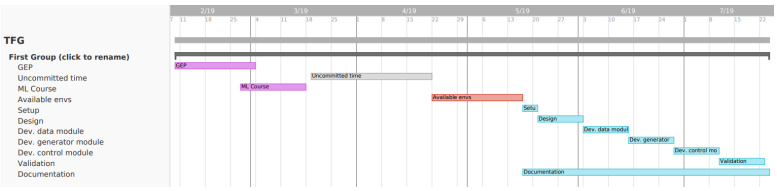

Figure 2: The gantt chart

2.6 Alternatives and action plan

Given the time constraints that the project holds from the very beginning, an action plan must be created as a way to react ahead of time against prevent possible delays or failures. Therefore, we must assess the possible setbacks taking advantage of agile methodologies, which will allow us to response quickly.

2.6.1 Database generation

Even though the dataset generation shall be easier than in the foundational paper, there could be errors out there that lead to a flawed database, probably coming from the use of independent software blocks. This will be tackled by hand-picking the parts which will be used in the data feed, and elaborating a tool that can check that the quality of this data gathering has been correctly done.

2.6.2 Platform not working

Given that a shift has been needed to switch from the original platform to a new one, the risk still needing another switch is present. To tackle this, a ranking of the most suitable alternatives was done while evaluating the current platform used.

2.6.3 Untracked setbacks

Even though we can subdue possible failures during the development of the project, unexpected problems must always be expected. Although these are not traceable, in the event of one taking place, a meeting with the project manager will be arranged to measure it and take an action that has the least possible impact.

2.7 Deviations from original planning

In this section, it will be explained, as long as the project evolves, the multiple, if any, drawbacks that have been found during the development of the project, pointing the source, impact and solution given to them.

2.7.1 Deviation 1: Longer term on learning ML pipelines

Due to the detailed information on how advance ML workflows work, namely those using NN, and due to the fact that the providing drone company was delaying the access to their development platform, only providing a test access, a longer time was spent on studying and playing around with these pipelines.

This actually helped the author to identify actual simplicity of the previous procedures and the bottlenecks found mainly at the creation of the training database, as well as the possibilities of the drone development platform.

2.7.2 Deviation 2: Denial to the access to the development platform

During the initial phase of the project, an untracked setback happens: the drone provider, which initially is offering a full-blown development platform, denies the access to it. To this day, the platform no longer exists and has already been wiped out ("Skydio SDK," n.d.) from the provider's services.

Therefore, this handicap forced the research team to do a complete volte-face in the development plans. The need for a new drone platform similar to the previous one yet available was the primary concern to find out. Therefore, a detailed comparison among the current service providers was conducted, and the Parrot development toolkit was chosen.

Due to the inability to rearrange the already-invested hours and the need for extra tasks, the project planning was redone, reallocating the budget as well to extend the project duration 1 month more. This was ultimately done to ensure that a minimal set of objectives could be attained without excessively compromising the global goal.

3 Budget and sustainability

Despite sustainability is not a core subject in the Computer Science bachelor, it is a matter that must be carefully taken into account. Being one of the major topics in current society, its real environmental *impact* in software projects is not as traceable as the environmental impact it has over other disciplines, perhaps because of the intangible nature of the field. However, unlike many other subject areas, it usually has a profound impact in society, and consequently, economy.

Throughout creativity and innovation, software projects can take advantage of common devices and extend its capabilities. This is proven in authentication systems, in driving systems and even healthcare systems. The increasing presence of classy software products nowadays has risen both the necessity and awareness to deliver efficient, mindful products that purposefully give value to the economy or the environment, and ultimately, the society.

Hence, by means of auto-evaluation and critical thought, the author of this project has had to learn from accessibility, security, resource sustainability and equality, among other insights, to assess, design and deploy in a thoughtful way software systems where the positive impact on society, economy and environment is maximized.

3.1 Project Budget

Taking into account the outlined resource breakdown, in the next pages an estimation of the cost of the project, tightly related to these resources, is presented. Moreover, the indirect costs stemmed from the project risks is also detailed.

3.1.1 Hardware budget

The hardware budget takes into account the useful life of every component, as well as the amortization for each one. In the following table, the used resources are listed.

Product	price	units	useful life	amortization
Parrot Anafi	599	1	2 years	104
Intel Core i5-4440	339	1	4 years	23
NVIDIA GeForce GTX 660	199	1	3 years	21
Crucial 2x4 GB RAM DDR4	89	1	4 years	7
TOTAL	1226			155

3.1.2 Software budget

The following table presents the absolute costs related to the software tools used in this project. Note that all of them are for *free*, because its open-source nature or because it is already included with the Parrot drone.

product	useful life	price (€)	amortization (€)
Ubuntu 16.04	-	0	0
Python environment	-	0	0
Pandoc	-	0	0
Zotero	-	0	0
Parrot tools (Olympe + Sphinx)	-	0	0
TOTAL	-	0	0

3.1.3 Human resources budget

Albeit this project is mainly developed by a single person, with blinking help from the project tutor, the former will have to fulfill multiple positions, as a manner to be able to modify, execute and validate the project on-going. In the next table, an cost estimation of human resources can be found. These wage estimations has been estimated following the indications for these jobs according to Spain ("PayScale - Salary Comparison, Salary Survey, Search Wages," n.d.) ("Tendencias del Mercado Laboral," n.d.).

Also note that, as a way to keep the costs more realistic, the software developer profile has been adjusted to meet the backgrounds needed to carry out this project.

Role	Hours	€ / Hour	Wage
Project manager	80	27	2160
Software developer	560	11	6160
TOTAL	640		8320

3.1.4 Indirect costs

Electricity or Internet connection are usually the most used resources when doing a research. Therefore, in order to keep the project costs real, they are also tracked. These costs have been estimated following the European Union's digital market site ("EU's broadband price, 2015," n.d.) ("EU Energy Rates," n.d.).

Concept	Cost per month (in €)	Months	Total Cost
Internet connection	50	6	300
Energy costs	20	6	120
TOTAL			420

3.1.5 Unexpected costs

In the event that some delays or problems arise, a budget for unexpected situations is detailed hereunder, as a way to ensure a more comprehensive estimate.

- If any task incurs in a time delay, an extra of 120 hours is kept as it has been previously explained. Therefore, a buffer of 2500€ is retained, guaranteeing up to 5 more weeks. This price is estimated from the wage of the software developer. This event has 40% chance of occurring.
- If any material, namely the drone or the laboratory computer, gets damaged, a buffer of 615€ is kept, as both devices can be repaired with it, or switched if it is too affected. This risk is, however, quite unlikely to happen (5%).

Concept	Estimated budget (in €)
Project delay	1512
Damaged materials	615
TOTAL	2127

3.1.6 Contingency buffer

10% of the overall budget will be kept as contingency buffer, to cover missing costs and over - or under - adjusted costs. This results in 960€.

3.1.7 Total budget

Cost type	Allocated budget (in €)
Hardware budget	155
Software budget	0
HR budget	8320
Indirect costs	420
Unexpected costs	2127
Contingency budget	1102
TOTAL	12124

3.2 Budget monitoring

With the aim of adjusting our real budget, a tracking of the real expenses at the end of each task will be made. This record will help us spot if our estimations match with the actual use of hours, the cost of the resources used and the expenses of any unexpected even that may have happened. To this purpose, the following formulas will be computed:

$$Cost\ deviation = (EC - RC) \times RH$$

$$Consumption\ deviation = (EH - RH) \times EC$$

being

$EH = estimated\ hours$	$RH = real\ hours$
$EC = estimated\ cost$	$RC = real\ cost$

This task, however, will be developed as the project progresses, as RH and RC cannot be computed currently.

3.3 Sustainability and social commitment

In order to evaluate and analyze the sustainability of the project, the environmental impact will be studied over three dimensions: environmental, economical and social. In the following table, usually called *Sustainability Matrix*, an overview of the sustainability score can be seen for this project.

	Project already running	Useful Life	Risks
Environmental	*Design consumption*	*Ecological footprint*	*Environmental risks*
	8/10	16/20	-5 / -20
Economical	*Invoice*	*Viability plan*	*Economical risks*
	6/10	15/20	-4 / -20
Social	*Personal impact*	*Social impact*	*Social risks*
	9/10	18/20	-10/ -20
Sustainability Range	**23/30**	**49 / 60**	**-19 / -60**
		53 / 90	

3.3.1 Environmental dimension

3.3.1.1 Design consumption

The project itself only has a major dependency on electricity and internet connection, as they are used to power both the lab PC and the drone.

3.3.1.2 Ecological footprint

The proposed scheme seeks to improve the task of drone AI development by directly improving data quality and code redundancy, therefore reducing the energy consumed in low-performing processes and more complex driving systems. Moreover, the energy used over the project can be efficiently distributed, as it does not use a significant quantity of energy throughput in any task.

3.3.1.3 Environmental risks

It is possible that the current project scheduling gets extended, thus incurring in the use of more material resources.

3.3.2 Economical dimension

3.3.2.1 Invoice

A thorough cost estimation has been made in the previous section, both at the human and material dimensions. Bear in mind that this project uses a set of specific hardware and software technologies that have been developed in the last 10 years. Therefore, both the material, time and intellectual cost required to undertake a project of this magnitude is surely fairly expensive.

3.3.2.2 Viability plan

The project's goal tackles the adaptation and improvement of development environments for drone machine learning pipelines. This advance would simplify the software and hardware requirements to let a drone learn by itself to behave correctly in diverse use cases, thus reducing the costs introduced with older systems. Note that, although this is a key concern in the project, the intertwining of multiple software frameworks could derive into untracked risks in the future, therefore leading to a higher expense.

3.3.2.3 Economical risks

As autonomous driving in drones is a hot topic nowadays, one of the biggest economical drawbacks to suffer is the release of a more efficient platform. Despite the scheme developed here is oriented in practice to one proprietary device in the current market, it can be exported to other devices.

3.3.3 Social dimension

3.3.3.1 Personal impact

The execution of this project has implied significant considerations in a professional level, as it ultimately tackles the use of machine learning in embedded, daily devices as a mean to improve life quality. It has meant a considerable shift of the reflections on robotics and machine learning and the possible uses for society to the author of this project.

3.3.3.2 Social impact

As explained before, this is currently a hot topic in the research community, therefore researchers could benefit from this work. Moreover, this scheme should also facilitate both new researchers and hobbyists to engage with the development of models in a faster and more interactive way.

3.3.3.3 Social risks

Due to the nature of this application, which simply involves the improvement of previous development systems, it does not have a direct impact in society. However, as latest AI researches have shown, the ethical problems that underlies in the technologies this software is focused on, can amplify the problem about personal privacy and crowd manipulation.

4 Design of the Development Platform

In this section it is discussed the set of available choices taken into account as a result of the platform access denial.

4.1 Platform features

As tested out by the author, the development platform offered by the drone provider, namely Skydio, at the early project trials provided a seamlessly interface to the following tasks:

Figure 3: Skydio Simulator

- Remotely operate the drone, both in real life and in a simulated environment.
- Collect data from the environment, using global information given by the simulator or local drone information, accessible from the built-in sensors, both in the simulator and in real life. This data is offered through a reduced *API*. This allowed to define and limit the quantity and quality of the data to be used later in the training and testing of the ML model.
- Leverage the use of the *skills* already available in the drone, such as obstacle avoidance or human tracking, to extend their features or implement new ones.
- Incrementally train and test ML algorithms in a simulator, where multiple environments are offered, before trying in a real-life device.

All of these tasks are accessible through a SDK shipped with the drone. However, as explained before, this software toolkit was not made available to the research team. Therefore, in the next step a comprehensive survey was conducted to analyze the available drone SDKs.

4.1.1 Survey of available solutions

To this end, a comparison of commercial-grade drone devices was done, taking into account the factors exposed above, as well as their compatible development applications. Industrial-grade drones were discarded from the beginning because their starting price did not fit the project budged and required specialized knowledge on electronics, which the research team did not have. Given the software requirements, the survey firstly explored the simulator options and from there, a mapping to the compatible drone devices was done.

By looking up at previous papers on the field, a significant number of them used the open-source tools ROS1, which acted as middleware for software modules, and Gazebo, which acted as a simulator engine. This was the first option to consider, as it would allow to reuse much of the work done. However, the compatible drone devices were on the market since up to 8 years, such as the Parrot ARDrone 1, and either a lot of them were not available to buy, or their control software was not officially supported due to its age. Other community platforms like PixHawk does use much modern devices and are compatible with these tools, although they are oriented towards low-level device features and require circuitry knowledge.

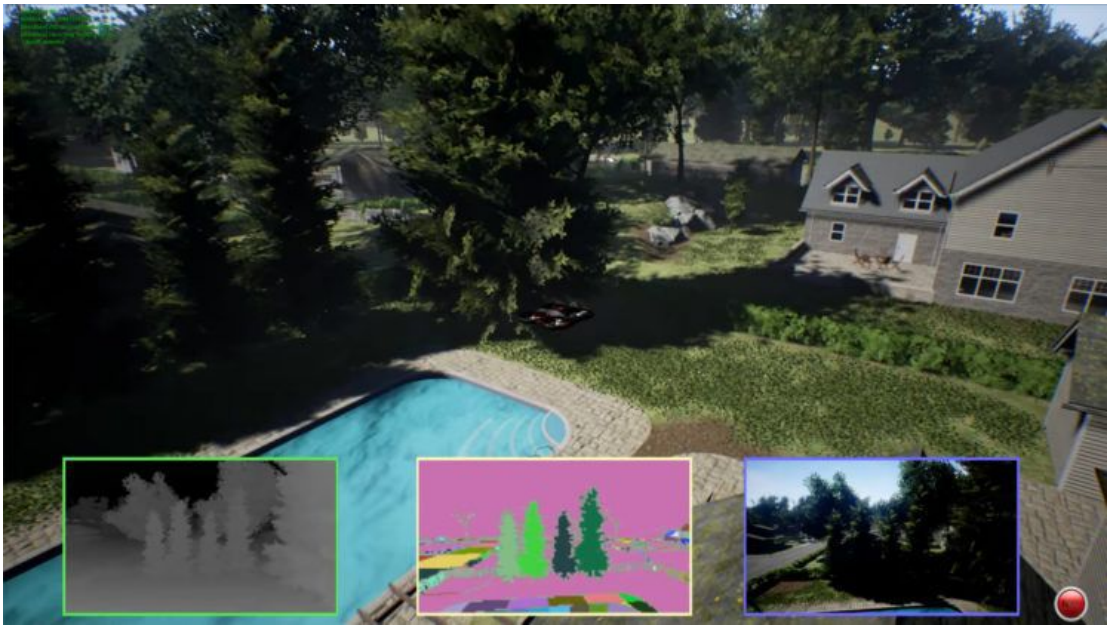

Figure 4: AirSim simulator environment

In addition, there are also more complex, realistic simulator engines, such as AirSim or CARLA, that includes off-the-box setup for controlling, collecting and processing the environment data thanks to the use of up-to-date game engines. However, due to these feature set, the applications required a high-performing computer to run the simulations and the documentation about the setup was lacking.

Other development platforms for commercial-grade devices such as DJI were incomplete, either because of the lack of a simulation environment or software SDK.

4.1.2 Platform choice

Finally, after comparing the myriad of development platforms, the Parrot Anafi drone was chosen as hardware device, mainly because the community behind its middleware and the compatibility with Gazebo. Their middleware provides a drone and Gazebo interface using Python 3.5, and allows to process data in a minimal way. In contrast, it did not offered such a refined sensor palette as the Skydio, using just a front and down monocular cameras, compared to the set of 12 stereo ones used in the Skydio device. Moreover, as it is the same provider that created some of the drone devices used in previous works, it retained retro-compatibility with ROS, though implementing version 2 instead of 1, which requires rewriting of ROS 1 modules if you are willing to use them.

Thereupon, the goal was to improve the existing features of the Parrot middleware in order to obtain a basic development platform adapted to the machine learning pipelines of the research team.

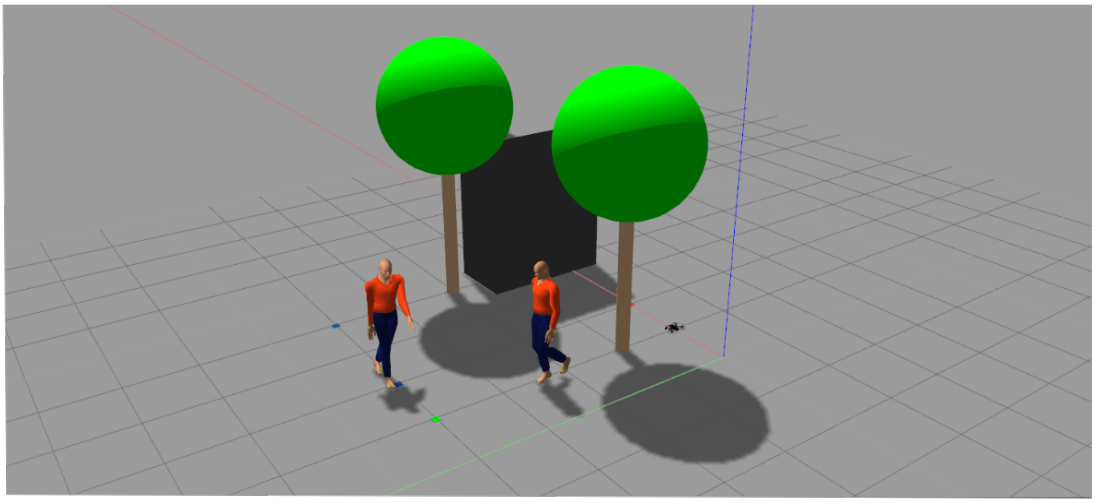

Figure 5: A generated environment in Gazebo

4.2 Specifications of the application

In order to correctly attain the requirements needed to use the platform, a set of features have to be defined in such a way that they cover them. On the basis of the available features described above, here it is detailed the essential specification that the application should have to supersede the toolkit-to-be:

1. Firstly , it must offer methods to teleoperate the device, this is, to control its movement and actions, such as takeoff, landing or move.
2. Secondly, it must be open to the addition of new *smart* actions, such as obstacle avoidance or or object tracking. These intelligent actions should be build on top of the built-in ones, and they could be stackable as the included ones, meaning that they can work at the same time.
3. Thirdly, to minimize the distribution difference of simulated and real-life inputs, it must include modules to ensure the robustness of the data generation, being able to generate different environments given a list of configuration parameters, and the collection of the data produced by them within a certain delay range. An additional feature would be to benefit from this delay to produce streaming analytics.
4. Fourthly, it must be open to implement new sensors for the devices used, as well as adding new devices, given that these models already have a module that can be plugged into the system. These modules could be linked to environment sensors, for recording global features like gravity or humidity, or to device sensors, for recording battery levels, camera or gyroscope output.
5. Finally, the overlay must support the iterative design usually found in ML pipelines.

[imagen to wapa de una figura enseñando el funcionamiento del overlay y su similitud con el visto en otros sistemas].

5 Development Platform Implementation

In this section, a detailed vision of the platform modules will be shown, emphasizing in the features, improvements and issues found in the implementation of each one. This explanation will be reinforced by the addition of metrics to demonstrate the efficiency on some aspects. Likewise, a glimpse of how they integrate into the final system is specified in the end of this section.

5.1 ROS retrocompatibility

The initial idea for the development of the scheme was to enable its fully intregation within the ROS ecosystem. However, a framework incompatibility due to different versioning was found. Mainly, the open-source implementations of well-known services in the ROS ecosystem, like having access to tons of data sources or plug-and-play ML algorithms, were implemented using ROS 1.

Knowing that current Parrot middleware uses Python 3.* and ROS 1 is only compatible with Python 2.*, a working solution could only be found for ROS1-to-ROS2 connections, meaning that data could be read in but not write out. Bear in mind that the official compatibility between both versions is at the early stages. Moreover, both Python 2.* and ROS 1 versions are being dropped out by 2020. Therefore, there is an extra incentive to implement the project in supported versions of the software.

Therefore, the current implementation of the proposed platform has been done using Python 3 and ROS 2.

5.2 Remote control module

The adoption of this module is relatively simple compared to others, as usually this toolset is already offered by the drone providers along with the device, if they have a linked simulator. Even though, three interfaces have been described and added, in order to enable the device control in any possible case:

- By means of a physical controller, this is, an input device like a gamepad or a keyboard, in such a way that it can be teleoperated by an expert pilot.
- By means of an application that, given a pair of start and goal points, the device travels along the path. This path can be manually entered or it is computed by a tweaked version of the A* algorithm.
- By means of a virtual controller, in such a way that it data is automatically inputted and the controller applies the piloting commands as if it were a physical controller. This can be useful for ML algorithms, whose output can be a velocity command.

All of this three cases have been implemented for the Parrot Anafi by making use of the Parrot Olympe library. This service runs a TCP/IP server that communicates with the drone through a network, usually generated by the drone, and send MAVlink commands, which commands the drone move, speed up or down, take off or land.

5.3 Data generation module

A special focus is dedicated to this module, because, as it has been detailed before, to ensure that the trained model learns to apply the correct actions to unseen data when tested, the data quality of the training set has to remain optimal. To satisfy this optimality, the data distribution generated for both a simulated and a real-life environment cannot differ that much from one another.

This is tackled on one way by providing a satisfaction metric. Let's assume that the limits of an environment e_1 are within a grid G_{MxN}, where G_{ij} points to the element k stored in the cell (i, j). In a similar fashion, there exists another environment e_2 whose limits are within another grid G'_{MxN}.

Moreover, each environment holds an associated function $f_{ij} : H^1_{MxN} \times H^2_{MxN} \rightarrow \{0, 1\}$ so that

- $f_{ij} = 1$ if $H^1_{ij} \neq H^2_{ij}$, this is, if $H^1_{ij} = k \neq k' = H^2_{ij}$.
- $f_{ij} = 0$ if $H^1_{ij} = H^2_{ij}$.

Therefore, if we want to replicate the conditions found in a real environment to a simulated environment, we ought to minify the overall distance between two given grids, this is,

$\sum_i^M \sum_j^M f_{ij}$ for the grids H^1, H^2.

Likewise, if we want to keep the major difference among generated simulated environments, we ought to maximize the metric above.

In addition, once the world is generated and the static objects are placed, there is still the need to configure the pedestrian paths. In order to ensure that these paths also differ from one simulation to the other, a tweaked, constrained version of the A* algorithm is used for its generation. These constraints can be specified at runtime in order to keep the similarity to real-world environments.

Moreover, with the proposed scheme, we can still provide reproducible examples by previously setting a fixed seed.

5.4 Data gathering module

To ensure that all available data sources can be connected to the system, the data collection module allows to connect sources coming from ROS 1 modules by bypassing it with ROS 2. This way, data from ROS 1 systems can be exploited. Moreover, existing Python 3 APIs can be easily integrated.

The data gathering process is intended to be done in an uniform way for both the real-life and simulated environments, so that the data distributions are not so different from one another. To this end, a database specifying the data schemes is used for both scenarios. This data schemes hold their own description but similar data properties and relations. This way, this can also be leveraged for a high-level use in ML pipelines.

These guidelines have been applied to the module implementation, which contains an OptiTrack ("Optitrack," n.d.) Client, which is the data source used in real-life environments, and a Parrot Telemetry Client, which is the data source used in simulated environments.

[una figura de la BD usada: tiene un objeto para el Drone, otro para el Sujeto, otro para Viandantes, asi como objetos relativos a los elementos estaticos del sistema: Arbol, Muro y Puerta]

5.5 Integration of ML pipelines

Merging the work that has been developed and detailed abovehead, we shall be able to create a ML-oriented pipeline. Thus, let's describe the needed phases to do it:

1. Set the goal to achieve, the problem that is trying to be solved and the elements and data, a priori, needed to complete the task.
2. Learn about the context: explore previous works from the same topic or similar ones, and check where the pitfalls can be found; they can be at the available toolset, at the data generation, at the model structure or when training the model.

3. Do an initial setup of the devices to be used and play around with the development platform.
4. A training database is defined, synthetic or not, in order for the machine learning algorithm to *learn* which is the appropriate action given an input. Likewise, a validation and set databases have to be constructed as well, although the latter is not used until the end.
5. Train the model in an iterative and incremental way, this is, by using both the training and validation sets to respectively train and test if the model does not overfit. At this stage, both the databases and the model structure can be tweaked to adjust it better to the end goal.
6. Finally, the model shall be tested with unseen data included in the test database. This test helps us determine if the model has learned enough, if some parts have to be corrected, or if the model scheme is not the optimal one.

5.5.1 Proof of work

5.5.1.1 Preparation

Finally, to prove that the system is capable of handling a machine learning routine, a basic pipeline will be created. Here, the problem to solve is the next one: given an environment with variable quantity of static elements, such as trees or walls, and a variable quantity of dynamic elements, such as cars or pedestrians, a drone is given the task of following a chosen pedestrian while avoiding the rest of the elements.

This task can be seen as a combination of obstacle avoidance and human tracking. Moreover, to make the drone *behave* like another human, a certain distance range is assigned to the chosen pedestrian, making sure that the drone stays within it.

To tackle this goal, four set of use cases are depicted:

1. Environments with just the drone and the subject.
2. Environments including the elements in (1) plus static elements.
3. Environments including the elements in (1) plus dynamic elements.
4. Environments that combine the previous ones.

To this end, the drone's navigation system has to know the following data:

- Distance from the drone to the subject.
- Distance from the drone to static elements.
- Distance from the drone to dynamic elements.

5.5.1.2 Data generation

Given the problem constraints, a train, validation and test datasets will be generated. These datasets will be composed by position, velocity and acceleration data from each of the present elements in each environment. The data will be collected from the simulator's global logger, which can provide the needed data from all the elements within a given simulation. To keep this data structure similar for the real-life tests, the data in these cases will be collected by using an OptiTrack motion capture system.

The dataset is composed of runs each one holding 7 to 8 seconds per journey. For each second, the position, velocity and acceleration from all the elements in the environment will be available, having zero velocity and acceleration and a fixed position the static ones. Moreover, the volumes of all the present elements are known, in order to compute the distances.

These runs will be conducted by an expert pilot by using the control module.

5.5.1.3 Model training and testing

With all the previous steps completed, I trivial model is implemented. This model simply takes action within a continuous range of movement of $[-1..1]$ for each dimension randomly. The collected data is logged into the system, but it does not provide relevant information to the model. The MSE value is used as estimator, although as it may be disguised, its precision is not optimal.

Moreover, a lineal regression algorithm is also implemented but not tested. This has been due to time constraints at this final stage.

6 Closure

In this section, the overall result of the project is detailed, as well as the future lines of action. A personal review from the author of the paper is also provided.

6.1 Results

As means of reviewing the project from a high-level point of view, this project can be seen as the sum of two blocks. This way, it can be seen as the sequential evolution of the milestones attained based on the exploration of possibilities in this novel field.

In the first block, we seek to overcome the pitfall of not having an integrated development environment for UAV-based machine learning, which was at first given by the drone provider but later denied. Therefore, a set of existing tools are compared in terms of data collection, processing and machine learning model generation. Multiple options are explored and a modern framework from Parrot drones is chosen, which allows us to use the existing structures and extend them or create new ones. The properties of a machine-learning workflow are used as a reference of the ongoing work.

Thereupon, a minimal set of tools is iteratively designed and implemented to match the needs specified before. Namely, the platform is build on connected, independent modules following a common design pattern in this field, but it is accommodated to the requirements specified. Moreover, it relies on functional techniques that eases the iterative development process, as wells as helping analytics to get richer insights about it. At this stage, these tools provide the essential functionalities to build a pipeline, but these can be extended within future lines.

6.1.1 Personal insight

The work documented here has allowed me to get to know the possibilities and pitfalls of machine learning for robotics and mainly for drones. Throughout the project, I have consolidated my ability to learn and apply abstract concepts in the fields of data analysis, machine learning and data generation.

To be able to fulfill the project, I had to put an special emphasis on learning and mastering diverse technologies, such as ROS, Gazebo and Python, and apply the knowledges acquired in the degree to apply data-processing patterns and design bundled systems in order to match the objectives. Although the project requirements have been changing throughout the process of building it, I have been able to adapt the time and reach constraints to accomplish a milestone ahead of previous projects.

Overall, I believe that the global objective of learning and testing my understanding of the topics addressed has been achieved.

6.2 Goal fulfillment

The overall result of this project has to comply with a set of requirements settled by the author and the director. These requirements can change on behalf of the global goal that the project has, but nevertheless a solid knowledge of the related fields and technologies is needed for the successful completion of the requirements.

As of this project, a sound foundation in a wide set of topics has been required to fulfill the diverse objectives and requirements, as well as the cross-topic know-how to keep a tight project structure.

6.2.1 CCO1.1: Evaluate the computational complexity of a problem and be able to recommend, develop and apply the algorithmic approach that best performs given the requirements.

The cost of exploring the space of possible actions in a search problem may be key given the problem constraints, as this one can be embedded in a platform whose performance cannot overcome a limit. In this project, an iterative analyze and design of algorithmic approaches to solve these bottlenecks, such as in the random-world path-finding problem, has been an important procedure to address the weaknesses of the platform and implement an approach to guarantee its performance.

6.2.2 CCO1.2: Prove knowledge of programming language foundations and the techniques for lexical, syntactical and semantical language processing, and prove to know how to apply them for the creation, design and processing of languages.

In the context of sensor data gathering and classification, a filtering capability was required to discard the given data not wanted. Therefore, a set of core language processing techniques have been key to gain performance at this stage. A similar process has been also applied to the input of the learned data into the simulation platform.

6.2.3 CCO1.3: Define, evaluate and choose hardware and software platforms for the development and production of applications and computer services of mixed complexity

One of the main driving forces of the project has been the selection of a reliable pair of software and hardware platforms to ensure the successful implementation and testing of this project. Although it has not been easy, a rightful choice, taking into account the sophistication the modules needed, allows to implement certain services that can be extended afterwards to other platforms.

6.2.4 CCO2.1: Prove knowledge of the foundations, paradigms and techniques unique to intelligent systems, and be able to analyze, design and build computer systems, services and applications that can leverage these in any field

In order to generate an autonomous application that is able to provide a service in its own, artificial-intelligence techniques have to be used. Therefore, in multiple sections of this project, such as in the random world generation or path-finding module, a computer system has to be designed and implemented to take care of these needs.

6.2.5 CCO2.2: Ability to acquire, formalize and represent human knowledge as to be able to apply it for problem resolution via a computer system, namely for those in which a computing, perception or environment-aware activity context are found.

A recurring feature of the system is to be able to, either implicitly or explicitly, apply human knowledge to the learning process. Therefore, an accent has been put on specifying the machine's capabilities to extract data and apply human-like assumptions for interactive, dynamic environments in which they take actions.

6.2.6 CCO2.4: Prove knowledge of machine learning techniques, as well as apply it to design, develop and implement applications and systems that use them, including those which can generate knowledge from big data sources.

The design, iterative development and testing of machine-learning-based systems is one of the main reasons to develop this project. Therefore, this requirement have been explicitly fulfilled by creating a machine-learning-based position prediction application, successfully build by ensuring the completion of all the learning steps in the system.

6.2.7 CCO3.1: Implement critical code following efficiency, security and runtime criteria

This requirement has been fulfilled due to the careful integration of the different modules in a way to keep the system modular and maintainable. Moreover, dealing with complex platforms such as ROS or Olympe has helped to keep track of critical blocks in which the cohesion of the system depends.

6.3 Skill set

Closely related to the previous section, highlighting the set of qualifications required for the design, development and implementation of this or similar systems can be helpful if pursuing a similar goal.

6.3.1 Analysis and Design of Algorithms

In order to automatize and self-learn several tasks of the system, the following core concepts has been used.

6.3.1.1 Artificial Intelligence

Given the limitations of multiple challenges addressed in the project, the implementation of automatized processes has been fundamental to build a performant system. The following concepts can be highlighted:

- Dynamic programming
- Computational logic
- Randomized, algorithmic generators

6.3.1.2 Data Structures

To correctly handle multivariate, complex data sources and link them with structured databases, the understanding of the different available data structures is key to match the efficiency requirements. The following concepts can be highlighted:

- Time complexity
- Space complexity
- Sorting algorithms

6.3.1.3 Big Data Processing

A functional understanding of the analysis, cleaning and mapping for data in huge quantities is essential to speed-up the management of multivariate data. The following concepts can be highlighted:

- Data cleaning
- Feature extraction
- Map-Reduce paradigm

6.3.1.4 Language Processing

A notion of how language semantics and syntax can be used to define ideas, and hence data, is important to functionally manage the streams of the platform. The following concepts can be highlighted:

- Data extraction
- Language processing
- Domain-specific languages

6.3.2 Computer programming

A great asset to help build up complex systems that mix multiple technology patterns is to manage a scalable and maintainable codebase in the most cost-effective way.

6.3.2.1 Concurrency

The know-how of task scheduling and management of storage resources can keep a system hassle-free. The following concepts can be highlighted:

- Threading
- Subprocesses
- Data locks

6.3.2.2 Network programming

This project is based on the communication of multiple, independent modules that use a common network interface. Therefore, to correctly serve the system, the following concepts shall be known:

- TCP/IP socket programming
- Publish-subscribe pattern
- Master-slave architectures

6.3.2.3 Databases

The need for a concurrent, constant data processing feature in the platform forces to use high-performing databases, either temporal or static, that can handle real-time data feeds in an efficient way. The following concepts can be highlighted:

- Database models
- In-memory databases
- Data locks

6.3.3 Robotics

The knowledge of foundational robotics concepts is critical to the development of this project, as its core objective is to provide easier means for robotics simulation.

6.3.3.1 Robotics

Without the understanding of the courses of action, environment perception and the learning capacity of robots is quite hard to address the main objective stated here. The following concepts can be highlighted:

- Control optimization
- Sensing systems
- Human-robot interaction

6.3.4 Project Management

To correctly organize and plan the ongoing activities throughout the project, a notion of project management is needed.

6.3.4.1 Project Management

Due to the complex dependencies and lifespan of hierarchical tasks, the know-how of how to administrate them is a key asset for the correct achievement of the goals. The following concepts can be highlighted:

- Time planning
- Task definition and allocation
- Incremental stages

6.4 Future work

Given the time and expertise constraints, the extent of this project could ensure a core set of features to prove the system feasibility. However, in the event of extending this project, a few remarks on the system bottlenecks should be helpful for the development of this platform.

6.4.1 Extend the robot model coverage

The provided platform is currently oriented to Parrot drones, as the base framework is the one available to them. However, the project guidelines should allow other drones to be seamlessly connected to the platform.

6.4.2 Upgrade the random world generation

The world generation and path-planning algorithms are far from an production-ready scheme. Moreover, it only takes a few parameters into account and does not expand over certain limits to ensure efficiency.

6.4.3 Improve the data gathering quality

Currently, the management and processing of sensor data is adapted to the needs of the project. For this reason, it may need some rework to be fully compatible with other available schemes.

6.4.4 Upgrade the simulation engine

At the time of delivering this report, the research team has noticed that the previously-detailed AirSim simulation engine has been substantially improved over the course of the project, now providing a full-blown environment for end-to-end deep learning. This attains the project's goal from a different perspective, and considering the fact that the overlay implemented here is largely suitable for this engine, it can also be considered as a future development line.

6.4.5 Enhance the automated control models

In this project, a simple trained model is provided to control the drone. However, this one is far from state-of-the-art models due to underfitting and insufficient model exploration.

6.5 Legal information about the project

In this section, all aspects related to the legal part of the project are contemplated.

6.5.0.1 Regulation on drones in Spain

The normative about drones in Spain ("BOE 29/12/2017," n.d.) forbids to fly in urban and crowded spaces. To overcome this limitation, the project has been developed in a controlled environment within the university grounds, in such a way that this constraint could be minimized as much as possible.

6.5.0.2 Regulation on data protection for autonomous sensor tracking

As of 2019 in Europe with the GPDR law (InterSoft Consulting 2018), if someone is being recorded, it is mandatory to hold a signed document from him/her agreeing to be recorded, with no legal constraints whatsoever.

Therefore, the project has been developed in a lab environment where all the people who participated in gave its consent about being recorded. # Annexes

References

Alahi, Alexandre, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. 2016. "Social LSTM: Human trajectory prediction in crowded spaces." In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-Decem:961–71. doi:10.1109/CVPR.2016.110.

Altche, Florent, and Arnaud De La Fortelle. 2018. "An LSTM network for highway trajectory prediction." In *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2018-March:353–59. IEEE. doi:10.1109/ITSC.2017.8317913.

Andrew Ng. 2017. "Deep Learning | Coursera." https://www.coursera.org/specializations/deep-learning.

"BOE 29/12/2017." n.d. https://www.boe.es/boe/dias/2017/12/29/pdfs/BOE-A-2017-15721.pdf.

Coll Gomila, Carles. 2018. "Learning aerial social force model for drone navigation," October. Universitat Politècnica de Catalunya. https://upcommons.upc.edu/handle/2117/127536.

Curs, Grau. 2013. "Normativa del Treball Final de Grau," 1–12. https://www.fib.upc.edu/sites/fib/files/documents/estudis/normativa-tfg-gei-final.pdf.

"EU Energy Rates." n.d. https://1-stromvergleich.com/electricity-prices-europe/.

"EU's broadband price, 2015." n.d. https://digital-agenda-data.eu/.

Garrell Zulueta, Anais, and Et Al. 2017. "Aerial social force model: A new framework to accompany people using autonomous flying robots." *IEEE/RSJ I*, no. International Conference on Intelligent Robots and Systems (IROS).

"Git." n.d. https://git-scm.com/.

"Github." n.d. https://github.com/about.

Giusti, Alessandro, and Et Al. 2016. "A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots." *IEEE Robotics and Automation Letters1.2*, 661–67.

Helbing, Dirk, and Péter Molnár. 1995. "Social force model for pedestrian dynamics." *Physical Review E* 51 (5). American Physical Society: 4282–6. doi:10.1103/PhysRevE.51.4282.

InterSoft Consulting. 2018. "General Data Protection Regulation (GDPR) – Final text neatly arranged." https://gdpr-info.eu/.

Kendall, Alex, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. 2017. "End-to-End Learning of Geometry and Context for Deep Stereo Regression." In *Proceedings of the IEEE International Conference on Computer Vision*, 2017-Octob:66–75. doi:10.1109/ICCV.2017.17.

Kim, Dong Ki, and Tsuhan Chen. 2015. "Deep neural network for real-time autonomous indoor navigation." *ArXiv Preprint ArXiv:1511.04668*, no. 1511.04668.

Kim, Hyun Young, Bomyeong Kim, and Jinwoo Kim. 2016. "The Naughty Drone." In *Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication - IMCOM '16*, 1–6. New York, New York, USA: ACM Press. doi:10.1145/2857546.2857639.

Le, Quoc V. 2013. "Building high-level features using large scale unsupervised learning." In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 8595–8. doi:10.1109/ICASSP.2013.6639343.

Lee, Namhoon, Wongun Choi, Paul Vernaza, Christopher B. Choy, Philip H.S. Torr, and Manmohan Chandraker. 2017. "DESIRE: Distant future prediction in dynamic scenes with in-

teracting agents." In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua:2165–74. doi:10.1109/CVPR.2017.233.

Loquercio, Antonio, and Et Al. 2018. "Dronet: Learning to fly by driving." *IEEE* Robotics a: 1088–95.

Martinez, Julieta, Michael J. Black, and Javier Romero. 2017. "On human motion prediction using recurrent neural networks." In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua:4674–83. doi:10.1109/CVPR.2017.497.

"Optitrack." n.d. https://www.optitrack.com.

Palossi, Daniele, Antonio Loquercio, Francesco Conti, Eric Flamand, Davide Scaramuzza, and Luca Benini. 2018. "A 64mW DNN-based Visual Navigation Engine for Autonomous Nano-Drones," May. http://arxiv.org/abs/1805.01831.

"Parrot." n.d. https://developer.parrot.com/.

"PayScale - Salary Comparison, Salary Survey, Search Wages." n.d. https://www.payscale.com/.

Robicquet, Alexandre, Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. 2016. "Learning social etiquette: Human trajectory understanding in crowded scenes." In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9912 LNCS:549–65. Springer, Cham. doi:10.1007/978-3-319-46484-8_33.

Sadeghian, Amir, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, S. Hamid Rezatofighi, and Silvio Savarese. 2018. "SoPhie: An Attentive GAN for Predicting Paths Compliant to Social and Physical Constraints," June. http://arxiv.org/abs/1806.01482.

"Skydio SDK." n.d. https://www.skydio.com/developer/.

Smolyanskiy, Nikolai, and Et Al. 2017. "Toward low-flying autonomous MAV trail navigation using deep neural networks for environmental awareness." *ArXiv Preprint ArXiv:1705.02550*.

"Tendencias del Mercado Laboral." n.d. https://www.michaelpage.es/.

"UPC." n.d. https://alumni.upc.edu/es/carreras-profesionales/ects.

Zhang, Tianhao, and Et Al. 2016. "Learning deep control policies for autonomous aerial vehicles with mpc-guided policy search." *IEEE* IEEE inter.