

UNIVERSIDAD POLITÉCNICA DE MADRID

Thesis Title

by

Guillermo Echegoyen Blanco

A master thesis submitted in partial fulfillment for the
masters degree of Artificial Intelligence

in the

Escuela Técnica Superior de Ingenieros Informáticos
Departamento de Inteligencia Artificial

June 2018

Declaration of Authorship

I, AUTHOR NAME, declare that this thesis titled, 'THESIS TITLE' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

“Write a funny quote here.”

If the quote is taken from someone, their name goes here

UNIVERSIDAD POLITÉCNICA DE MADRID

Abstract

Escuela Técnica Superior de Ingenieros Informáticos

Departamento de Inteligencia Artificial

Master Degree in Artificial Intelligence

by [Guillermo Echegoyen Blanco](#)

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too...

Acknowledgements

The acknowledgements and the people to thank go here, don't forget to include your project advisor...

Contents

Declaration of Authorship	i
Abstract	iii
Acknowledgements	iv
List of Figures	vii
List of Tables	viii
1 Introduction	1
1.1 Context	1
1.2 Motivation	1
1.3 General Objective	2
1.4 Specific Objectives	2
1.5 Overview	2
2 Problem Description	3
2.1 The Aerostack Framework	3
2.2 Requirements	3
2.3 Details of the New Features	4
2.4 Previous Work	5
3 State of the Art	6
3.1 Localization	7
3.1.1 Outdoor localization	7
3.1.2 Indoor localization	8
3.1.3 SLAM	8
3.1.3.1 Extended Kalman Filters SLAM	9
3.1.3.2 Particle Filters	9
3.1.3.3 Graph Optimization SLAM	10
3.1.3.4 Lidar SLAM	10
3.1.3.5 Visual SLAM	10
4 Modules	12

4.1	Technical Goals	12
4.2	Specification	12
4.3	Integration	12
 A Appendix A		 13
 Bibliography		 14

List of Figures

2.1 The Aerostack architecture	4
--	---

List of Tables

For/Dedicated to/To my...

Chapter 1

Introduction

[ToDo := Refs & Links]

In the following work an integration of a navigation system onto the Aerostack platform is presented.

1.1 Context

Aerostack is a framework for aerial robots, aimed at giving flight autonomy to some extent. It features a modular approach for the construction of behaviors that can be used to develop complex flights and automatic handling for certain situations such battery level or hardware conditions. It is the frame for the following work, which adds more autonomy through the integration of a octomap-based navigation system and a global planner. This provides a novel localization technique for the framework.

1.2 Motivation

So far, there exists only one simple geometry planner and an Aruco-based localization technique. In indoor environments, this system compels the need for environment preparation, the Arucos must be placed beforehand in well known localizations that must be hardcoded in the robot map. In this sense, there exists a need for a more robust, preparation-free localization system and accompanying planner. This work provides such an improvement with the introduction of a octomap-based navigation system and a global planner.

1.3 General Objective

This work aims at adding a novel navigation system to the Aerostack framework.

1.4 Specific Objectives

This section enumerates a comprehensive list of objectives.

1. Enrich the current navigation and localization systems.
2. Test and validate the new navigation and localization systems through simulated environments.

To achieve the first objective, the following additions to the framework are proposed:

- Add a robust planner based on a new navigation technique.
- Add a robust navigation technique through the use of octomaps.
- Add a robust localization technique based on octomaps.
- Add octomaps construction support through lidar.

1.5 Overview

[ToDo := Review when everything is finished]

This dissertation is organized as follows: ...

Chapter 2

Problem Description

In the present chapter the problem is presented, along with an introduction to the previous work. It is structured as follows: Section 2.1 introduces the Aerostack framework, Section 2.2 presents the context of the problem and the requirements a replacement should have. Section 2.3 describes deeply the improvements presented and the decisions taken to end in Section 2.4 with the description of the previous system.

2.1 The Aerostack Framework

Aerostack is an agnostic framework to build and design control architectures for aerial robotic systems. It provides some low level components as well as coordination processes and some planners. Figure 2.1 shows the general architecture of the framework.

2.2 Requirements

As of the second version of Aerostack, the only localization technique available is based on the recognition of a special type of marker called Aruco, first used for augmented reality applications. It is a fast and reliable technique to estimate the pose of the camera capturing the image. Although this system works fine for many applications, it imposes the need of preparing the environment, placing this markers in a very precise way and annotating it's exact position before the experiments. While this might not be a problem in an augmented reality like scenario, when it comes to live localization in unknown environment it becomes useless. Hence, a new system for localization is required.

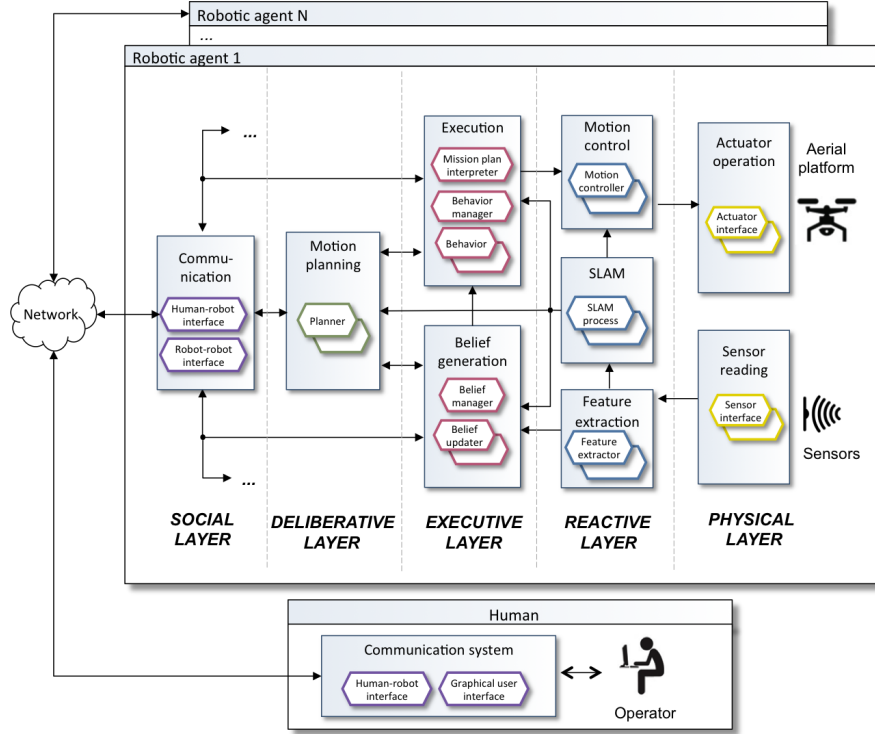


FIGURE 2.1: The Aerostack architecture

Along with the aforementioned localization technique comes the navigator which coordinates with a 2D geometric planner to accomplish the mission at hand. As the localization technique is to be changed, leading to a new way to perceive the environment, a new navigator and planner will be necessary.

2.3 Details of the New Features

A lidar is going to be used as the main source for localization. The module in charge of this part is already present in the Aerostack framework, but it is not being used. Based on lidar input an octomap is built. This octomap is then used by the navigator, along with the planner instructions to build a motion plan to execute.

The lidar + octomap functionality is provided by a module called `hector_slam`, developed at the Darmstadt University [**hector'slam**]. Taking the lidar's cloud of points it is able to reconstruct an octomap and then localize inside it (SLAM). The output of this module can be used to create a plan avoiding obstacles to reach a target point.

[ToDo := Review, NavStack & Global Planner]

2.4 Previous Work

By now, there is a process responsible for doing recognition of Aruco markers, it is connected to the robot camera and fetches data every n milliseconds, where n is a user defined constant. When an aruco is recognized, the pose of the robot can be estimated. Knowing the exact position of the marker enables the process to estimate the absolute position of the robot, leading to a high precision coordinates localization. While this approach has many advantages, it fails when the environment is not prepared beforehand.

To create a plan, a target position and the used Aruco markers are specified, then the motion planner creates a 2D plan to follow, using the Arucos to localize in the process.

Chapter 3

State of the Art

This chapter will review the classic methods used for localization and mapping, which is the core for any navigation technique, it starts describing the problems that arise both in localization and mapping. Continue with some of the most prolific solutions found to these problems to finish exposing the techniques and algorithms used by the Aerostack framework.

Localization is referred to all the techniques used to find the position in coordinates of an agent or object inside the world, relative to a reference frame. As far as it's absolute coordinates are known, anything can be used as reference frame, it is used as the coordinates system's centre. In an outdoors environment, the Earth could be the reference frame and the robot's coordinates can be acquired with Satellite systems, giving an absolute point inside the three dimensional space. It is desirable for these coordinates to be in a way that a computer can handle efficiently, typically as two or three floating point numbers (although integer numbers are used sometimes too), depending on the number of dimensions used to represent the space. To save computational effort, the y axis could be unused in a wheeled robot.

Moving a robot avoiding possible obstacles through the space is tricky in itself, obstacles must be detected and handled correctly, moving objects can appear in the way, and so on. This alone does not provide any intelligence nor it helps planning, to aid in planning and moving smartly in the space, a map can be constructed while the localization is happening. The term mapping covers all the algorithms used to construct a map combining the data acquired from the many input sources a robot can have. Mapping opens the door for smart planning, along with many more advantages. A classic example is finding cycles in planned paths.

3.1 Localization

Localization techniques are divided into two groups: Outdoor and indoor techniques. The distinction comes from the fact that Satellite systems signals cannot go through walls. This fact has led to a whole new set of technologies and techniques that are able to localize in environments without an absolute reference of the world.

This section is organized as follows: First outdoor localization will be analyzed, with a brief review of its core components, then various indoor localization techniques will be exposed, focusing on lidar based techniques.

3.1.1 Outdoor localization

By now, the most robust, reliable solution for outdoor localization is based on combining different sources of data. The mobile platform has drawn great attention over the past few years, pushing some of the largest companies in a shared effort to improve localization services while minimizing the impact over the battery's performance.

Satellite systems localization in mobile applications have certain drawbacks: The most obvious one is that acquiring and processing the signal wastes power, but also that in the case of civilian systems (such as GPS) the localization has precision of 10 meters for security reasons.

To aid both problems two new localization techniques were implemented:

1. GSM Localization. As every GSM antenna has a well known location stored in a database, one can localize trilaterating the near GSM antennas. Obviously, this method is subject to GSM signal coverage.
2. WiFi Localization. This powerful method can serve both as an indoor and outdoor localization technique (see sect. 3.1.2). When a smartphone detects a WiFi hotspot, it sends its BSSID along with the GPS coordinates if available to a centralized server. As this database grows the localization precision improves and every user can take advantage from it. This is specially useful in urban areas where stallite signal is poor or intermitent and improves as more and more users log information.

The best localization services can be provided with the conjunction of these three main techniques, which can be easily merged with an *Extended Kalman Filter*.

Although these techniques are widely used they are limited to mobile platforms, which are restricted both in sensorization and in processing capabilities. For robotic applications more sophisticated sensors are used, this includes depth cameras and lidars for the most part. Again, all the sensors output is merged to get the best estimates.

In robotics, it is usual that the localization we are interested in is relative instead of absolute. This is done to aid in locating near objects in the space relative to the robot but also to construct a map, the process of localization and mapping simultaneously is called *SLAM - Simultaneous Localization And Mapping* (see sect. [3.1.3](#)).

3.1.2 Indoor localization

To localize in indoor environments many strategies can be followed, the general trend is to place different markers (active or passive) beforehand in well known locations. These markers are then recognized by the localizing device to know its location. These *recognizable markers* can be anything, a Bluetooth beacon, a WiFi hotspot, etc.

Bluetooth beacons are specially crafted for this purpose as they can provide much more information. It has been extensively used in congresses and hotels to provide hosts with more information beyond localization, as services and timetables based on location.

In the case of WiFi hotspots localization is usually done by analyzing the signal strength and incoming angle. This method only works when the hotspots' location is known beforehand and is very prone to errors because the device must remain static in a certain angle.

From the computer vision perspective, visual markers can be placed too and processed by the device localizing and again, this requires preparation beforehand. One example of this setup are the well known [Aruco](#)

In many robotic applications like swarm robotics there is a necessity to track each member of the swarm in a closed, contained environment for this purpose an [OptiTrack](#) system can be used, it is a highly precise camera set that can track various markers (marked swarm members) and serve through the network. This setup is specially useful to monitor the swarm, enabling each member to access its location.

3.1.3 SLAM

SLAM is the process of mapping and, at the same time localizing inside that map. This is specially useful in environments that are not prepared like the ones exposed previously,

enabling the robot to work on an unknown place without getting lost nor entering cyclic paths. SLAM techniques are crucial in any robot with some degree of autonomy, it makes the navigation possible.

There are three main variants here, the ones based on Extended Kalman Filters, the probabilistic ones and the ones based on Graph Optimization.

3.1.3.1 Extended Kalman Filters SLAM

Extended Kalman Filters (EKF) SLAM is the earliest technique developed for SLAM. EKF is a general technique to find the best estimate for the measuring variable based on the mean and covariance. In this case the inputs are the odometry used to estimate the robot position, features of the environment anchor the odometry measures and the robot motor system sensors (wheel decoders, etc) to estimate the change on the position. Then the objective is to find the best estimate for the current robot's position.

At the start of the process, the system's $(0, 0)$ coordinate is established where the robot is, this is the most confident measure about the robot position. As the robot navigates the environment, successive measures are taken and paired with the known landmarks, when a previously seen landmark is witnessed again, the position estimate is corrected with the covariance matrix, and the error correction is propagated along the previous estimates. In this way as long as the robot is navigating and sensing the landmarks, the position estimate improves.

Although this technique works well on small maps, it renders unusable for large maps, this happens because the covariance matrix used to correlate the position estimates grows quadratically with the measures, making the memory footprint wildly large and the overall processing time very high. Some researchers have proposed an improvement over the EKF SLAM algorithm through submap decomposition [2, 4].

3.1.3.2 Particle Filters

Particle Filters are a probabilistic approach to position estimation, usually called Fast SLAM [5]. It uses various *particles* that represent the posterior probability of the true distribution of maps and possible paths. To do so it stands over a method called Rao-Blackwellization, that aids in dimensioning the number of particles needed to represent the map. Also, as conditional independence is assumed between the observed landmarks, every landmark can be represented as N small Gaussians, which is linear, instead of exponential on the number of landmarks.

Although this method is easy to implement, is fast enough for real time applications with not very high demanding software and yields good results on small to medium maps, it suffers from the fact that lots of particles are needed to represent big maps, specially with multiple nested loops. Therefore, many improvements have been proposed, [1] for example uses occupancy grids instead of Gaussians.

3.1.3.3 Graph Optimization SLAM

The Graph Optimization SLAM techniques try to optimize a graphical model representing the landmark and robots locations. In this representation, each location is viewed a node in the graph, and the edge (called soft constraint) between two consecutive nodes is the captured odometry. The key intuition behind these methods is that at the end, the graph is sparse, because, each node will have just a few connections to other nodes. Also, at worst, the number of entries in the graph is linear in the time elapsed and in the number of nodes.

This is the most widely used approach because sparse linear optimization is in a very advanced stage, allowing for scalable, yet efficient implementations of the algorithm.

3.1.3.4 Lidar SLAM

Lidar is a well established laser range sensor that can be used for depth estimation. By doing fast sweeps in 360 degrees it can compose a depth map which can be used for SLAM.

Hector SLAM [3] is a technique developed in the Darmstadt University. It uses a lidar sensor to do a fast SLAM by matching rays along sweeps.

Along this work, the *hector_slam* ROS module will be integrated into the Aerostack framework, providing a robust SLAM technique ready to use for the drones equipped with lidar.

3.1.3.5 Visual SLAM

Using computer vision for SLAM have been a challenge since it's conception, it raises the difficulty, specially for monocular cameras. Although many features can be extracted from images, it is not clear how to process nor store the data taking into account the full 6 degrees of freedom in a camera. All the parameters of the camera must be known

beforehand, depth cameras include a lot of noise and monocular cameras do not have scale.

In [6], ORB-SLAM is proposed. Intuitively, it creates ORB features from a visual input and stores it in a sparse matrix, then a matching process is launched to localize every feature, improving the localization along the way. It can work on Monocular (no scale), Stereo and Depth Cameras, giving extraordinary results.

Chapter 4

Modules

In this chapter, the technical goals of the project are introduced. Namely, each developed module will be explained as well as its requirements and its specification details. For each module, it will be explored:

- The technical goals
- The problem to which the module provides a solution
- Some of its properties (reusability, scalability ...)
- Integration with the current version of Aerostack

4.1 Technical Goals

4.2 Specification

4.3 Integration

Appendix A

Appendix A

Bibliography

- [1] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. “Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters”. In: *IEEE Transactions on Robotics* 23.1 (Feb. 2007), pp. 34–46. ISSN: 1552-3098. DOI: [10.1109/TR0.2006.889486](https://doi.org/10.1109/TR0.2006.889486). URL: <http://ieeexplore.ieee.org/document/4084563/>.
- [2] J.E. Guivant and E.M. Nebot. “Optimization of the simultaneous localization and map-building algorithm for real-time implementation”. In: *IEEE Transactions on Robotics and Automation* 17.3 (June 2001), pp. 242–257. ISSN: 1042296X. DOI: [10.1109/70.938382](https://doi.org/10.1109/70.938382). URL: <http://ieeexplore.ieee.org/document/938382/>.
- [3] S Kohlbrecher et al. “A Flexible and Scalable SLAM System with Full 3D Motion Estimation”. In: *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. IEEE. Nov. 2011.
- [4] John J. Leonard, John J. Leonard, and Hans Jacob S. Feder. “A computationally efficient method for large-scale concurrent mapping and localization”. In: (2000), pp. 169–176. URL: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.122.4705>.
- [5] Michael Montemerlo et al. “FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem”. In: *IN PROCEEDINGS OF THE AAAI NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE* (2002), pp. 593–598. URL: <http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.16.2153>.
- [6] Montiel J M M Mur-Artal Raúl and Juan D Tardós. “ORB-SLAM: a Versatile and Accurate Monocular SLAM System”. In: *IEEE Transactions on Robotics* 31.5 (2015), pp. 1147–1163. DOI: [10.1109/TR0.2015.2463671](https://doi.org/10.1109/TR0.2015.2463671).