INTELLIGENT AUTONOMOUS INSPECTIONS USING DEEP LEARNING AND DETECTION MARKERS

ALEJANDRO MARTINEZ ACOSTA

Doctoral Program in Mechanical Engineering

APPROVED:

Angel Flores-Abad, Chair, Ph.D.

Ahsan R. Choudury, Co-Chair, Ph.D.

Joel Quintana, Ph.D

David Murakami, Ph.D.

Stephen Crites, Ph.D.
Dean of the Graduate School

To my

MOTHER, FATHER and WIFE

with all my love

INTELLIGENT AUTONOMOUS INSPECTIONS USING DEEP LEARNING AND DETECTION MARKERS

by

ALEJANDRO MARTINEZ ACOSTA

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fullfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

Department of Aerospace and Mechanical Engineering

THE UNIVERSITY OF TEXAS AT EL PASO

December 2022

Acknowledgements

I would like to express my gratitude to my PhD. advisor Dr. Flores-Abad, who gave me the opportunity to work under his supervision on this dissertation project. I would like to thank him for the knowledge and guidance throughout my doctoral career.

Additionally, I would like to thank David Murakami PhD. for his guidance and feedback on this project as well as the opportunity to complete three internships at NASA AMES. Dr. Murakami was a crucial part of this project as he helped me focus on specific portions of this project. Our weekly meetings helped keep track of my progress and prioritize tasks.

I would also like to thank the Center for Space Exploration Technology Research (cSTER), for the further developing my engineering skills. Many important engineering and life lessons were taught to me by brilliant staff and faculty members of the glscster. Many thanks to all the people that gave me the chance to prove myself and gain confidence at the cSTER labs.

I want to also thank my brother Rodrigo Martinez and my sister Gabriela Martinez, that through their exemplar achievements and careers, have taught me to always pursue for greatness, devotion and passion for your career. I am deeply in debt for their support and love throughout our years growing up, an above all, for pushing me to always become a better person at all stages in my life.

I also owe this achievement to both of my parents Miguel Martinez and Yolanda Acosta. Without their unconditional love and support, this would not have been possible. To my father, I would like to express my eternal gratitude for the many life lessons that you taught me. For raising me to become the man I am right now. To my mother, thank you for your unconditional love, devotion, dedication and sacrifice. For all of the sacrifices you made to ensure that I will have a suitable and everlasting education, I cannot thank you enough.

Lastly, I would like to thank my wife and love of my life, Mireya Jimenez. Thank you for sticking by my side no matter how difficult this journey was. Your presence and love made all of this much more endurable and sustainable. Thank you for pushing me to always overcome difficulties with grace and gratitude. For your unconditional love and support on doing what I love. This is also for you!

Abstract

Inspection of industrial and scientific facilities is a crucial task that must be performed regularly. These inspections tasks ensure that the facility's structure is in safe operational conditions for humans. Furthermore, the safe operation of industrial machinery, is dependent on the conditions of the environment. For safety reasons, inspections for both structural integrity and equipment is often manually performed by operators or technicians. Naturally, this is often a tedious and laborious task. Additionally, buildings and structures frequently contain hard to reach or dangerous areas, which leads to the harm, injury or death of humans.

Autonomous robotic systems offer an attractive solution to the automation of inspections. This is due to their ability to reach remote and dangerous areas. Consequently, significant research efforts have been made for the use Unmanned Aerial Vehicle (UAV) and drones as flight inspection units. Compared to Unmanned Ground Vehicle (UGV), drones offer significantly better acrobatic and agility performance.

Needless to say, an autonomous drone must posses certain abilities to carry out inspections with little or no human intervention. Foremost, the drone must be able to position and orient itself towards inspection waypoints. Moreover, the drone must be able to use a map of the environment to identify and navigate to areas of interest on an inspection mission.

These abilities alone do not provide the drone the autonomy and efficiency required to perform quality inspections within required time frames. The inspection problem is often framed as a way to maximize the amount of volume a drone can cover through the use of coverage path planning. Therefore, drone systems spend most of the time inspecting areas or objects that produce little to no value for identifying defects. In addition, more time and drone flights must be performed to cover an entire area. Thus, this thesis proposes a novel approach to perform intelligent inspections using deep learning and Quick Response (QR) codes to identify, prioritize and segment objects of interest in the context of an inspection. Additionally, a novel navigation architecture that can find and prioritize collision-free trajectories is proposed. This architecture proposes the use of hierarchical state machines to capture complex reactive behavior found on a task driven robot. In conjunction, the proposed systems solve the inspection problem by identifying, prioritizing, labelling and navigating towards areas or objects of interest.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

Mobile robots have been used for the inspection of industrial and scientific environments. Robots can potentially improve inspection tasks by making them safer and faster on tasks such as welding, painting and packaging [1]. Additionally, a robot can utilize a multitude of sensors such as cameras, range detectors, radar and depth sensors that yield a greater set of data, compared to just visual human inspection. The data gathered from these sensors provides accurate measurements for a multitude of environment variables that can assist in the process of automating human related tasks. Therefore, aerial mobile robots or drones, have recently gained popularity in the field of automated inspections due to their low-cost, speed, agility and inspection data output. In contrast, human inspection requires qualified personnel to visually inspect structural and/or equipment, which is time consuming and in some cases, even dangerous. Thus, the objective of an automated inspection task, is to detect structural or equipment damage or defects[1].

However, drone robots still lack the autonomy to determine whether there are defects or not while performing a flight mission. Current approaches heavily rely in the use of Ground Control Station (GCS), to plan, execute and monitor drone flight missions. Other approaches simply require an operator to conduct the mission manually through the use of a remote controller. Consequently, drone inspections are manual, slow and energy inefficient, as several fly missions are required to fully inspect an area.

This research aims to add a layer of automation and intelligence using deep learning neural networks and Quick Response (QR) codes as navigation markers. In addition, a novel drone navigation architecture with priority oriented navigation goals is presented in this thesis. This navigation architecture has the autonomy to choose between a low priority inspection plan, and high priority object or QR detection marker by interrupting and re-planning the original navigation trajectory, with a new higher priority trajectory.

This chapter will introduce the research problem, followed by aims, objectives and research significance. Additionally, several background research sections are presented at the beginning of this chapter. These sections provide the background study and concepts used to develop the ideas presented in this thesis. Finally, a section is dedicated to outline the chapters included in this thesis document.

[1] A damaged or defect is defined as visual signs of wear, tear or environment exposure that could potentially compromise the functionality and safety of the inspected item

1.1 Research Problem

Performing inspections using drones is certainly not a new research topic. Drone systems have been proposed for the inspection of public and private infrastructure such as bridges, power plants, sewers and wind turbines [2]. Inspection drones have also been used in areas such as cartography, agriculture, volcanology and photogrammetry [3]. Nonetheless, there is still a great interest on increasing the autonomy of drones, specifically on running Artifical Intelligence (AI) algorithms directly on the drone flight control computer. Increasing the level of autonomy on a drone would reduce human intervention, while at the same time increasing the decision making process of the drone. Moreover, increasing the level of autonomy would result in energy efficient inspections, as the number of flights will be reduced by limiting the inspection problem to observing and prioritizing detected areas, objects or markers.

Several efforts have been made to include deep learning to drones, with the aim to perform image recognition and segmentation [4, 5]. Traffic [6], agriculture [7] and natural disaster [8] surveillance, landing marker detection [9] and object marker detection [10] are some of the applications that make use of deep learning on drones.

Nevertheless, these applications are fairly limited on what they can accomplish. As an example, deep learning for object recognition or segmentation are only used as defect markers for post-processing tasks, as it is the case for agriculture surveillance. Moreover, these defect markers do not affect the flight mission of the drone, as they are simply ignored for navigation purposes. This is troublesome for applications such as natural disaster surveillance or automated inspections, where further actions need to be taken after a detection has been confirmed. For instance, in the application of natural disaster surveillance, assistance should be provided to human subjects in distress or in a dangerous situations by performing a set of actions or relaying information about the status of the situation.

Ultimately, this leads to inaction and a lack of autonomy, as the drone will not be capable of making smart and reactive decisions based on the output of its sensors and AI algorithms. This is a problem because some cases require extra actions to be taken. In the context of inspections, detecting defects on a building or equipment might require further actions such as navigating to the detected anomaly, recording a video, obtaining images, or capturing point-cloud data. This is especially important since drone flight time is limited, so capturing and detecting data becomes a crucial task.

Therefore, in this work, a concrete scenario is presented in section 7.6, where an initial inspection plan is generated on one of the walls of the National Full-Scale Aerodynamics Complex (NFAC) facility located at NASA AMES. While performing the initial inspection mission, a QR code marker is detected, which signals the drone to interrupt the initial inspection trajectory for a new higher priority inspection trajectory task. The drone then navigates to this new location where the QR code marker was detected. The drone performs

actions such as recording images, video and point-cloud measurements at the marker's location. Lastly, the drone resumes its initial inspection mission and waits for another marker detection.

1.2 Research Aims and Objectives

Given the lack of research and development of drone autonomy in the context of inspections, this study aims to build and study an autonomous drone that can perform actions based on marker detections using deep learning and other image based codes. To complete this study, the following items must be accomplished:

1. Study and develop a drone navigation stack that can perform path planning using a map[2] (Chapter 4)
2. Identify a suitable deep learning neural network model and a QR system for automated detections (Chapter 5)
3. Construct and study the framework needed to bridge the gap between object recognition and drone navigation, which will add a new level of autonomy to the proposed inspection drone system. (Chapter 3)
4. Integrate actions that will improve and enhance the inspection quality (Chapter 6)

Ultimately, this thesis seeks to find an answer to the following specific question:

How can deep learning in conjunction with path planning be used to increase the level of autonomy of a drone for an inspection task?

1.3 Research Significance

This study will bring in new ideas on how to enhance the autonomy of a drone using path planning and deep learning neural networks by constructing a new framework that uses image recognition as markers for path planning. Additionally, this study also contributes to the field of autonomous inspection using drones by presenting a study case on a scientific environment.

This will address the lack of autonomy found on both commercial and research drones for inspection tasks. Furthermore, the presented work will fill the technical gap needed to achieve this autonomy by providing the software tools needed to complete this study. This will result in a reduction of human personnel needed to

[2]For a definition of a map in the context of this work see section 1.4.1

inspect structures, reduced facility downtime, an increase of periodic and reliable inspections, and a greater volume of data such as video, images and point-cloud that will assist in the early detection of damages.

1.4 Mobile Robot Navigation Architecture

An autonomous mobile robot must be able to navigate its environment on both static and dynamic objects[3]. Furthermore, it must obtain data from sensors and interpret it in a way that can be used to both localize and position the robot. Thus, a navigation architecture integrates components that can accomplish these tasks. For the majority of mobile robots, a navigation architecture requires two main components: Path planning and collision avoidance. Path planning tries to find a trajectory given a map. Its main objective is to change the robot from an initial state $p_{initial}$ to a goal state p_{goal}. In contrast, collision avoidance uses real-time sensor information to perform fast and dynamic avoidance of obstacles.

Another important aspect of navigation is the discretization of a continuous environment. This must always be the case before performing path planning, and it can have a great impact on the performance of the navigation system. For example, high resolution maps greatly increase the memory and storage requirements, which subsequently affects the computational complexity required for navigation, and the real-time response of the robotic system.

The following sections present the background study for mobile robot navigation architectures, with a focus on drone systems. On the first section, environment discretization with octrees and octomap is presented. This is followed by a brief introduction to path planning, along with the key research and techniques relevant to this thesis.

1.4.1 Octrees and Octomap

The first step in any navigation task is to transform a continuous environment into its discrete counterpart [11]. This can be accomplished using ranging sensor data and storing it into a spatial data structure. Some examples of spatial data structures are grid-maps, quad-trees, oc-trees and k-d-trees.

Although grid-maps are a popular choice in ground mobile robots, for drone applications this approach is impractical due to the memory size needed to store medium to large maps in $\mathbb{R}^3$. This fact is especially severe for embedded computers with constrained memory resources. For this reason, tree data structures are preferred over grid-maps, specifically for mobile robots that are not constrained in $\mathbb{R}^2$. Octrees can consol-

[3]This study defines static objects as those that cannot or are infrequently positioned into a different location and dynamic objects those that frequently change positions

idate large volumes of occupied or unoccupied space into a single voxel. Therefore, for drone applications, using octrees present an efficient way of creating and storing maps.

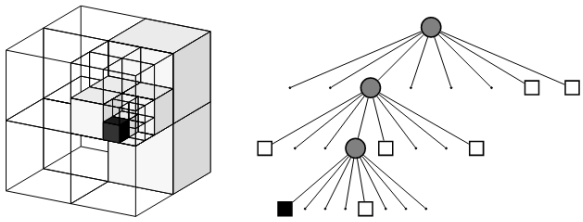


Figure 1.1: Example of a octree data structure and its tree representation [12]

An octree is a tree data structure that has exactly 8 node children. An octree is used along with sensor readings to discretize an environment in $\mathbb{R}^3$, by recursively partitioning space into 8 octants. Figure 1.1 shows an example of the tree structure used to partition space efficiently. Notice how some partitions are larger than others, this fact is the reason octrees are more efficient at discretizing environments for robotic applications in $\mathbb{R}^3$, since an octree can cluster portions of a map into bigger voxels.

Octomap is an efficient probabilistic 3D mapping software framework that uses an octree data structure to generate 3-dimensional maps [12]. The octomap framework is capable of representing occupied space through the following probabilistic model:

$$P(n|z_{1:t}) = [1 + \frac{1 - P(n|z_t)}{P(n|z_t)} \frac{1 - P(n|z_{1:t-1}}{P(n|z_{1:t-1})} \frac{P(n)}{1 - P(n)}]^{-1} \tag{1.1}$$

Where, z_t is a current measurement and $P(n)$, $P(n|z_{1:t-1})$ is a prior and previous estimate respectively. Finally, $P(n|z_t)$ represents the probability of voxel n to be occupied given sensor measurement z_t.

Additionally, the octomap framework uses ray-casting from the sensor's field of view to occupied voxels to detect free or unoccupied space. This is a useful feature for path planning, since a configuration space with occupied and unoccupied partitions is required. Additionally, multiple resolutions from the same octomap can be queried. This is another useful feature, as there is no need to create multiple resolution scans of the same environment. Figure 1.2 show that the same octomap can be used at multiple resolutions.

1.4.2 Path Planning

Path planning seeks to find a collision-free trajectory from an initial location $p_{initial}$ to a target location p_{goal} within a reasonable amount of computational time, and with the shortest possible path. As previously

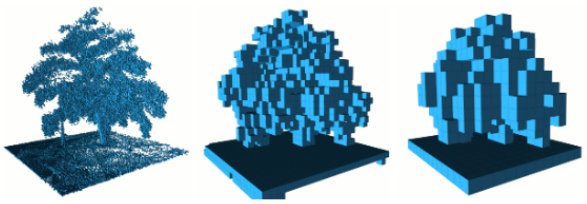


Figure 1.2: Octomap at different resolution levels [12]

stated, the first step for path planning is to discretize or obtain a map of a continuous environment. Using this map, a path planner can construct a graph considering cells on C_{free}, while simultaneously avoiding any cells on $C_{occupied}$. The constructed graph can be searched using path-finding algorithms such as depth-first, breadth-first search, dijkstra's or A* search algorithms to obtain a valid trajectory from $p_{initial}$ to p_{goal}.

Apart from graph searching, potential field methods for path planning are also found on the literature. In contrast to graph based methods, potential methods use gradient field vectors to find collision-free trajectories, much in the same way gravitational or electrical potential gradients are used to find body or electrical signals trajectories in space. However, these methods will not be covered in this thesis, so they are not part of the final study.

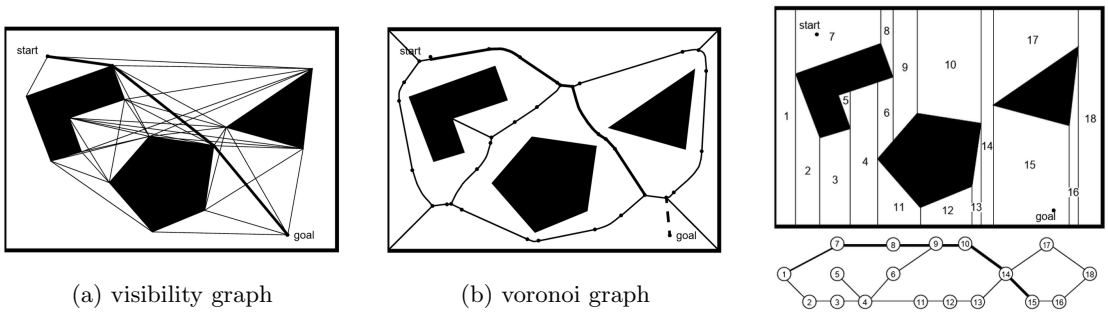
The next couple of sections introduce key ideas and strategies for graph construction and searching.

1.4.3 Graph Construction

Graph construction produces a graph $G = \{V, E\}$, where V is the set of unoccupied partitions or vertices on C_{free} and E is the set of edges that connects all elements in V to its neighbors [13]. Clearly, the first step to construct a graph G, is to find all the elements $V \in C_{free}$. For completeness, every element $V \in C_{free}$ should be considered, however, in practice this is often computationally expensive or impractical. For this reason, there are two main graph construction strategies: structured graph construction and randomized graph construction.

A structured graph construction leverages the topology of the discretized environment to search for $V \in C_{free}$. For instance, on a binary occupancy grid-map, elements in V can be found by searching all elements in C_{free}, which are represented by a boolean value of 0. Some other examples of structured graph searches are visibility, cell decomposition, voronoi diagrams, and lattice graphs. Figure 1.3 shows the different strategies methods for graph construction.

However, for higher dimensional spaces, finding all elements $V \in C_{free}$ can become impractical. Therefore, for this case, randomized graph searches on C_{free} are used. A randomized graph search randomly samples elements $V \in C_{free}$, so as to reduce the number of elements V on a graph. Rapidly Exploring



(a) visibility graph (b) voronoi graph

(c) cell decomposition graph

Figure 1.3: Different graph construction strategies for path planning [11]

Random Trees (RRT) is one example of a randomized graph search strategy.

1.4.4 Graph Search

Once a graph is constructed, graph search algorithms can be used to find a trajectory between $p_{initial}$ and p_{goal}. A graph search algorithm attempts to find an optimal path, which for most robotic applications, refers to finding the shortest path to reach p_{goal}. Some examples of graph search algorithms are breadth-first search, depth-first search, dijkstra's and A* [14].

Furthermore, incremental graph searching and planning can be found on robotic applications. These search algorithms use previous search results to refine subsequent search queries. Moreover, incremental path planning constantly updates both the map and graph, which ultimately, results in faster searches. Popular incremental search algorithms are D* and Life Prolonged A* (LPA*).

Although all search algorithms produce optimal results, in this thesis an A* search was selected. For the A* algorithm, see Appendix A.

1.5 Hierarchical State Machines

Reactive event-driven software systems are thoroughly used in robotic applications, as it concisely expresses the reaction nature of many of the robot's components. For instance, a robot must react to the different sensor stimuli through a set of pre-determined actions. Consequently, this behavior is often described through the use of Finite State Machine (FSM). However, expressing complex systems through the use of a FSM, causes a phenomenon known as state explosion [15], where many of the state machine's behaviors require an untractable amount of repeated states and transitions. Therefore, several extensions to traditional FSMs have been proposed by Harel [16], with the following requirements: