Robust Robotic Operations in the Presence of Uncertainties

A Thesis

Presented to

the Faculty of the School of Engineering and Applied Science

University of Virginia

In Partial Fulfillment

of the requirements for the Degree

Masters of Science (Computer Engineering)

by

Tony Xiaotong Lin

May 2018

Approval Sheet

This thesis is submitted in partial fulfillment of the requirements for the degree of

Masters of Science (Computer Engineering)

Tony Xiaotong Lin

This thesis has been read and approved by the Examining Committee:

Nicola Bezzo, Adviser

Joanne Dugan, Adviser, Committee Chair

Cody Fleming

Accepted for the School of Engineering and Applied Science:

Craig H. Benson, Dean, School of Engineering and Applied Science

May 2018

Abstract

Autonomous robotic systems often encounter varying degrees of uncertainty. Characteristics of an environment or mission, such as the definition of a terrain, the presence of an attacker, or the necessary sub-goals may have dramatic impacts on mission performance. Guaranteeing reliable execution demands an analysis of these uncertainties in order to ensure some form of robust operation, such as continual performance in the presence of failures. In this thesis, we present contributions related to robust operations of multi-robot coordination and autonomous exploration and an in-depth analysis of LiDAR-related uncertainties.

"The good life is one inspired by love and guided by knowledge." - Bertrand Russell

Acknowledgements

First, I extend my heartfelt thanks to my adviser, Dr. Nicola Bezzo, for his continual guidance and patience during my time as his student. His expertise and openness to discussion, no matter the time of day, has allowed me to develop a strong foundation from which I will expand upon for the rest of my career.

I also extend my thanks to Dr. Joanne Dugan for her considerable support throughout my time as a student at UVa, both as an undergraduate and as a graduate student, and to Dr. Cody Fleming for his willingness to serve on my committee.

I thank my friends and colleagues Esen, Mahmoud, Paul, Rahul, Shijie, and Andrew for the wonderful times and the long talks, and I thank my sister Vivian, my mother Jian, and my father Zongli for their everlasting love and support, without which I could not be who I am today.

Finally, to Erin I express my deepest love and thanks. Her steadfast warmth always fills my heart with joy.

Contents

List of Figures

Chapter 1

Introduction

Since the introduction of advanced computing systems, scientists and engineers have dreamed of autonomous robots capable of performing human-level tasks with ease, minimizing the potential for error and maximizing the potential for improved quality of life. In particular, these systems seek to take over dangerous or menial tasks and to enable a wider variety of capabilities. For example, autonomous cars may potentially lower the number of automobile-related injuries while relieving drivers of long commutes, unmanned ground vehicles (UGVs) and unmanned aerial vehicles (UAVs) may be able to provide improved services through automated teamwork, and specialized robots may be able to perform hazardous tasks such as underground survey and inspection without human intervention.

Effectively taking over these tasks though demands a careful analysis of the uncertainties inherent to each problem. How should a self-driving car deal with spontaneous weather conditions or unknown traffic hazards such as potholes or wild animals? How should multiple robots cooperate in order to benefit rather than hinder each other's progress? Beyond the scope of this thesis, how should robots make moral decisions such as when robot actions may result in both human benefit and human loss?

Besides operating in complex environments and making difficult decisions, robots may also need to deal with adversarial agents. Malicious cyber-attackers threaten to destabilize life-bearing systems and hold human lives hostage for either monetary gain or notoriety and may do so at any given moment when a robot is connected to the internet. While forcing a robot to be continuously offline is a possibility, growing Internet of Things influences and the need for edge/cloud computing make the possibility of a completely disconnected system in the future unlikely.

Mission task requirements may also incur uncertainties due to general runtime specifications. Robot teams tasked with pickup-and-delivery missions may not know the locations of the operation sites beforehand and

unknown environment characteristics may drastically alter how a mission should be completed. In all of these cases, assumptions about these problems must be made in order to handle these uncertainties. A self-driving car, for example, may assume that all potholes may be geometrically modeled with circles or rectangles. Adequate design of robust systems under uncertainty is thereby highly reliant on the appropriate assumptions being made. Improper assumptions on these uncertainties may lead to inadequate mission performance or human injury and loss-of-life.

In this thesis, we will discuss our ongoing work on robust operation under uncertainty in three domains: 1) multi-robot coordination for task completion with energy constraints, 2) autonomous navigation and mapping in hazardous and unstructured environments, and 3) vulnerabilities of LiDAR sensors and how these vulnerabilities may be manipulated.

This thesis is organized as follows. First, we briefly discuss existing literature on multi-agent coordination, autonomous navigation and mapping, and sensor security in Section 1.1 before discussing the main contributions of this thesis in Section 1.2. We proceed by briefly discussing the robot dynamics and mappings used throughout this thesis in Chapter 2. Then in Chapter 3, we will discuss the dynamics models and controllers for UAVs and UGVs used to facilitate the works on multi-agent coordination and tunnel mapping along with our approach to heterogeneous multi-agent task allocation. We will next discuss our work on autonomous mapping in which by leveraging Graph-SLAM and a virtual leader-follower approach a UGV is able to autonomously build maps while maintaining safety constraints. The proposed technique is validated with experiments inside a real and hazardous tunnel located in Virginia in Chapter 4. In Chapter 5, we will discuss our work on identifying LiDAR vulnerabilities due to specialized surface features and infrared interference. Finally, we will conclude in Chapter 6 with discussions on our future research in improving reliable operation of autonomous robots under uncertainty.

1.1 Related Work

In prior literature, multi-robot task coordination, autonomous navigation, and sensor cyber-security have been well-studied and a variety of existing problems and solutions have been described. We provide here a literature review of the state-of-the art.

1.1.1 Multi-Robot Coordination

Many proposed approaches to coordinating multi-robot systems found in the literature utilize decentralized control laws to produce emergent behaviors. For example, in [86] a biologically-inspired flocking approach is deployed to enable formation control in windy and noisy conditions, while in [69] a physics-based potential

force controller enables a team to create polygonal formations. Similarly, in [12] a group of homogeneous robotic systems is deployed leveraging spring-mass potentials to maintain a user connection with a base station. However, decentralized and distributed approaches suffer in guaranteeing some forms of optimality and emergent behaviors are often difficult to appropriately design.

In more recent work, multi-agent motion control has also been conducted through off-line pre-planned methods. Most notably, model-checking techniques have been exploited to model primitive agent actions and produce motion plans with safety guarantees. Since its introduction for single agent systems [30], satisfiability modulo theory (SMT) has also been leveraged to produce motion plans for quadrotor teams in discrete spaces [74, 73], offering the same provably safe motion guarantees but with reduced computational time. However, these approaches, while guaranteeing safety in the sense of decision-making, do not guarantee that a controller is able to track a synthesized trajectory, especially in the presence of disturbance.

Multi-robot energy management has also been explored in the literature. Authors of [77] optimized controller inputs to preserve energy for non-holonomic vehicles with considerations for the actual dynamics of the vehicles, while authors of [59] observed the energy constraints of a ground-air team during a task completion mission with a generalized travelling salesman solver. Authors of [11] demonstrated a learning-based method to solve an energy-constrained surveillance task for a multi-UAV team, which autonomously schedules recharge events for all agents.

1.1.2 Autonomous Mapping and Navigation

The problem of simultaneously localizing an agent within a growing map, also known as SLAM, is commonly considered a fundamental problem of robotics. Most state-of-the art work formulates the SLAM problem as a two part process: 1) a back-end graph is built using state estimates and environment measurements that can be optimized to minimize the error between actual measurements and expected measurements and 2) a front-end observer takes raw sensor data and creates representations of fixed features in an environment known as landmarks that allow the system to correct for accumulated error [85]. In particular, the optimization process is greatly enhanced when the mapping agent re-encounters a landmark, also known as closing the loop.

Authors of [49, 51], for example, demonstrate using the Graph SLAM framework with RGB-D sensors and cameras to create dense maps and perform accurate localization underwater. Due to the large amount of information contained in images, cameras also allow for more generic observation techniques. In [17], authors are able to produce semantic-level landmarks using deep neural networks, allowing the identification of chairs, windows, and doors to act as landmark cues. Other work [41] demonstrates a real-time slam approach

leveraging 2D LiDAR scans for high accuracy localization, while the work in [40] demonstrates a real-time 3D LiDAR slam approach design to augment weak GPS signals in urban environments. The use of active sensors to collect information from the environment is not altogether necessary though. The authors of [46] demonstrate the use of wifi sources as landmarks for SLAM, providing evidence that their approach, while more error-prone than LiDAR-based methods, is still able to achieve reasonable accuracy.

Another research topic within the SLAM community, known as active SLAM, involves the process of autonomously building a map such that some metric is optimized. Past approaches, such as those described in [83, 33, 52], may identify open regions (known as frontiers) and explore according to depth-first or breadth-first searches in these areas. However, more recent approaches have focused on using information gain to predict the value associated with certain frontiers and attempt to build the map greedily according to these values. For example, authors of [80] explored actively to ensure loop closures occurred with higher frequencies while authors of [88] explored while minimizing the number of necessary actions for a multi-robot team.

1.1.3 Cyber-Physical Security

Of particular interest to cyber-physical systems (CPS) security is the analysis and study of sensors as attack surfaces. As CPS depend on tightly coordinated data-driven architectures, sensors provide a key adversarial point of failure due to their inability to reject maliciously injected or falsely collected data. We review previous research and documentation on 1) cyber-physical attacks on sensors, 2) autonomous operation failures in unstructured and adversarial environments, and 3) LiDAR-specific attacks.

Sensor Attacks

Previous works have demonstrated that many commonly used sensors, such as GPS, cameras, radars, and inertial measurement units (IMUs), are extremely vulnerable to malicious actions.

Various GPS-based attacks have been explored in the literature in order to alter the navigational performance of unmanned aerial vehicles (UAVs). Authors of [50] and [81] were both able to use GPS spoofing to control and capture a UAV and forcibly redirect the UAV's trajectory without alerting GPS spoofing detectors.

Camera-based techniques for classification and obstacle detection have also been exploited and successfully compromised. Previous literature has demonstrated camera blinding and damaging attacks using focused pulses of light to over-expose the sensor and using low-powered laser pulses to damage the internal CMOS chip [67, 97]. The heavy reliance on deep neural networks for machine vision has also been exploited through the use of generated image noise to force high-confidence mis-classifications [70, 62].

Concerns for radar security have also increased due to research on spoofing, replay, and jamming attacks. The authors of [99] discussed the ease of spoofing, jamming, and replaying radar signals in order to disrupt adaptive cruise control and obstacle avoidance performance on autonomous vehicles while the authors of [47] demonstrated a UAV swarm capable of performing decentralized radar jamming against adversaries.

The IMU is another critical sensor capable of providing key orientation and angular velocity information that has been explored as a vulnerability in the literature. For example, the work in [79] demonstrated attacks on IMU dependent UAVs using focused sound, destabilizing hovering UAVs and causing crashes.

Adversarial Environmental Failures

Beyond existing work on sensor attacks, autonomous systems also encounter a variety of problems directly related to interacting with the environment. In recent years, various research and reports have suggested major difficulties in providing a comprehensive solution to autonomous operations. Between 2014 and 2015, Google's self-driving prototypes experienced 272 failures, 13 of which likely would have led to a crash without human intervention [39]. In 2016, Tesla's semi-autonomous vehicle was involved in a fatal accident when the vehicle's cameras were unable to detect the difference between a white truck and the sky [1].

As of 2014, over 400 U.S. military autonomous drones have also crashed due to various issues related to insufficient sensory capabilities [93]. Many of these drones failed during testing phases and were unable to complete their sample missions. More recently, military drones have been plagued by various environmentally related failures, causing disconnected communication channels leading to crashes or lost drones [92].

When presented with environmental weather hazards, autonomous systems may struggle even more. The work in [32] showed that rain had dramatic downsampling effects on LiDAR road scans, yielding lower quality mapping, feature extraction, and object detection functions. Other weather conditions also created dangerous losses of performance. The presence of snow made LiDAR or camera based lane tracking and road mapping behaviors nearly impossible [60]. To combat this, researchers in [23] used ground-penetrating radar to produce subterranean maps to enable effective lane-tracking in snow while researchers in [2] used principal component analysis (PCA) to reconstruct missing lane edges from weather-inhibited LiDAR scans. However, the use of radar for mapping in snowy environments acts as an expensive solution for a singular problem and PCA is still unable to handle greatly corroded scans produced by heavier rain conditions.

LiDAR-specific Attacks

While LiDAR sensors are capable of improving the autonomous functions of a system, they are still vulnerable to attack and failure.

Previous literature has attempted to solve LiDAR's inability to scan windows and mirrors by using a Bayesian filtering approach [98]. Their results, however, assumed a differing surface, such as a wooden frame, enclosed windows and mirrors and also failed to address malicious use cases.

Other work has explored the use of foreign laser signals in order to introduce scan anomalies. The authors of [67] have shown that replayed scans emitted from the original LiDAR sensor could be used to falsely convince LiDAR based tracking algorithms of the presence of obstacles. Their work was able to demonstrate that maliciously modified IR scans could be leveraged for replay, blinding, relay, and jamming attacks. However, the authors' approach depends greatly on the high-precision timing of replayed scans in order to align the previously captured scans with the emission phases of the LiDAR sensor. As such, this attack is extremely difficult to successfully execute. In addition, their work did not provide a deeper analysis into the impacts of these attacks on application level uses of LiDAR such as obstacle avoidance, mapping, and localization.

1.2 Contributions

Having discussed previous literature, we now discuss the main contributions of this thesis to the fields of multi-agent systems, autonomous exploration, and CPS security.

Heterogeneous Multi-Robot Systems

Primarily focused on the coordination of robot teams composed with differing members, we propose a distributed framework that leverages the heterogeneity of a team of robots to improve the operational runtimes of each robot. Contrary to the existing literature, we prioritize the safety of the system over the optimal completion of the mission and produce computationally inexpensive results that adapt during runtime without need for pre-planning. Our contribution in this respect is threefold:

- we develop a distributed control framework leveraging artificial spring-mass and magnetic potentials to coordinate agents while maintaining each agent's energy to be always positive;
- we develop a self-healing approach using the predicted dynamics of each agent to guarantee safety and positive energy even in the presence of one or more agent failures;
- we validate the proposed approach with experiments on a real heterogeneous robotic system composed of two different types of ground vehicles and one aerial vehicle.

This work on heterogeneous multi-robot systems was submitted and accepted to the 2018 American Controls Conference.

Autonomous Tunnel Mapping

In particularly hazardous conditions, autonomous robots need to be able to perform long missions reliably in order to minimize a human operator's exposure. In this work, we describe an approach to enable autonomous exploration and mapping in hazardous and GPS-denied environments. In these environments, robotic agents must be able to adapt online to unknown environmental characteristics and perform adequate localization despite sensing limitations in order to effectively explore. While some assumptions about the environment may be made (e.g. the terrain is rocky or wet, the environment is cylindrical in shape), no prior information about the internal layout or extent of the assumptions are known.

We also validate our described approach on a customized UGV operating in a real hazardous tunnel located nearby in Virginia. Our approach and platform were able to utilize 3D LiDAR data to autonomously create full-scale maps of the tunnel interior and avoid obstacles. Our contribution is three-fold:

- we develop a UGV testbed specialized for autonomous tunnel mapping;
- we develop an approach capable of performing autonomous mapping and navigation in hazardous environments;
- we deploy our testbed and approach in a real tunnel under active restoration.

This work is in preparation for a conference paper and has received significant media coverage [34, 48].

LiDAR Sensor Security

In much of robotics research, authors implicitly assume high-quality sensor data with general assumptions on noise characteristics and operational performance. However, due to malicious action, sensors are capable of providing false data. In this thesis, we discuss and analyze the LiDAR sensor, a widely used sensor for various safety-critical systems (e.g. self-driving cars, autonomous robots). Without a careful study of how LiDAR sensors are vulnerable, autonomous systems may be unable to reject false or maliciously injected data. The contributions of this work to the field of LiDAR security and subsequently autonomous system security are therefore as follows:

- reflective, absorbent, and semi-transparent surfaces can be easily leveraged by an attacker to manipulate LiDAR range and intensity measurements beyond the manufacturer's specification;
- IR-based interference can be easily leveraged by an attacker to silently manipulate LiDAR sensors and introduce abnormal noise beyond the manufacturer's specification;

- specialized surface and IR-based attacks are demonstrated through extensive testing on commercially available LiDARs and proof-of-concept in-lab experimental tests on autonomous robots.

This work on LiDAR security was submitted to the 2018 USENIX Security Conference.

Chapter 2

Preliminaries

Before discussing the main contributions of this thesis, we briefly discuss the robot models used to develop the approaches found in Chapter 3 and in Chapter 4. In this thesis, we focus on quadrotor aerial vehicles and four-wheel ground vehicles.

2.1 Quadrotor Dynamics

A quadrotor is a special type of rotorcraft with four identical rotors, two rotating clockwise and two rotating counter-clockwise. The angular speed of each rotor is denoted by ω_i. The thrust and moment produced by each rotor is proportional to their angular speed [14]:

$$F_i = \kappa_f \omega_i^2, \quad M_i = \kappa_m \omega_i^2, \quad i = 1, 2, 3, 4$$

where F_i and M_i are the thrust and the moment produced by each rotor respectively with the proportionality constant for thrust κ_f and for moment κ_m. For a quadrotor with arm length d, the net thrust and moments generated on the quadrotor are given by:

$$\begin{bmatrix} F \\ M_x \\ M_y \\ M_z \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} \kappa_f & \kappa_f & \kappa_f & \kappa_f \\ 0 & d\kappa_f & 0 & -d\kappa_f \\ -d\kappa_f & 0 & d\kappa_f & 0 \\ \kappa_m & -\kappa_m & \kappa_m & -\kappa_m \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \tag{2.1}$$

The state vector of the quadrotor is:

$$\boldsymbol{q} = \begin{bmatrix} \boldsymbol{p}_q^\mathsf{T} & \phi & \theta & \psi & v_x & v_y & v_z & \omega_x & \omega_y & \omega_z \end{bmatrix}^\mathsf{T}$$

where $\boldsymbol{p}_q = [x\ y\ z]^\mathsf{T}$ is the world frame position, v_x, v_y and v_z are the world frame velocities, ϕ, θ and ψ are the roll, pitch and yaw Euler angles and ω_x, ω_y and ω_z are the body frame angular velocities. An example quadrotor is shown in Fig. 2.1. The dynamics of the quadrotor are then described as follows:

$$\begin{aligned}
\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} &= \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \\
\dot{\boldsymbol{p}}_q^\mathsf{T} &= \begin{bmatrix} v_x & v_y & v_z \end{bmatrix} \\
\begin{bmatrix} \dot{v}_x \\ \dot{v}_y \\ \dot{v}_z \end{bmatrix} &= \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \frac{1}{m} \begin{bmatrix} \cos\phi\cos\psi\sin\theta + \sin\phi\sin\psi \\ \cos\phi\sin\theta\sin\psi - \cos\psi\sin\phi \\ \cos\theta\cos\phi \end{bmatrix} u_1 \\
\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} &= \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \\
\begin{bmatrix} \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z \end{bmatrix} &= \begin{bmatrix} \frac{I_{yy}-I_{zz}}{I_{xx}}\omega_y\omega_z \\ \frac{I_{zz}-I_{xx}}{I_{yy}}\omega_x\omega_z \\ \frac{I_{xx}-I_{yy}}{I_{zz}}\omega_x\omega_y \end{bmatrix} + \begin{bmatrix} \frac{1}{I_{xx}} & 0 & 0 \\ 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix} \begin{bmatrix} u_2 \\ u_3 \\ u_4 \end{bmatrix}
\end{aligned} \tag{2.2}$$

In order to control the quadrotor to track desired accelerations, we use a cascade of PID controllers to control position and attitude. Using a PID feedback controller on the error $e_k = \boldsymbol{p}_{k,\xi} - \boldsymbol{p}_k$ we can control the position and velocity of the quadrotor to maintain a desired state, as demonstrated in [14, 12].

2.2 Ground Vehicle Dynamics

We now consider the following differential-drive kinematical model to align with the type of ground vehicles that will be used throughout this thesis. However, our proposed approaches could be extended to cover any

Figure 2.1: The Ascending Technologies Hummingbird Quadrotor

other type of dynamics including unicycle models.

$$\begin{aligned} \dot{x} &= \frac{r}{2}(\omega_r + \omega_l)\cos\theta \\ \dot{y} &= \frac{r}{2}(\omega_r + \omega_l)\sin\theta \\ \dot{\theta} &= \frac{r}{L}(\omega_r - \omega_l) \end{aligned} \tag{2.3}$$

where r is the wheel radius, L is the wheel base, θ is the robot angle with respect to a global frame, and ω_r and ω_l are the angular velocities for the right and left wheels respectively. To control the UGV, we use a PID feedback controller on $e_\theta = \theta^{des} - \theta$ and $e_{\boldsymbol{v}} = \boldsymbol{v}^{des} - \boldsymbol{v}$. An example ground vehicle is shown in Fig. 2.2.

Figure 2.2: The Clearpath Jackal Ground Vehicle

2.3 Double Integrator Mapping

For the works described in this thesis, double integrator dynamics in the form $\boldsymbol{u}(t) = \ddot{\boldsymbol{p}}$ are used for planning purposes and then mapped to the low level dynamics of quadrotors and differential drive ground vehicles described in the previous sections. In Fig. 2.3, we illustrate the transformation of linear accelerations generated by a double integrator planner to velocity and angle inputs used by differential drive ground vehicles and quadrotors.

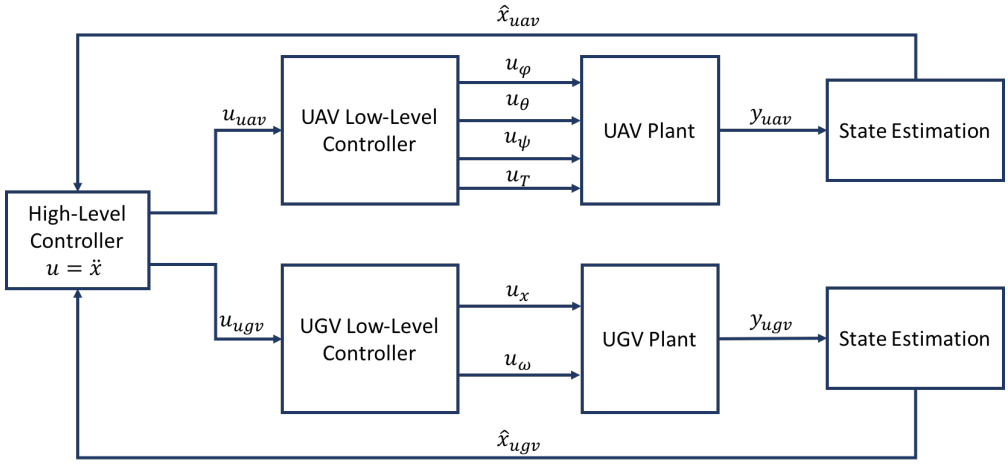


Figure 2.3: Input Mapping to UGV and UAV Dynamics

Chapter 3

Energy-aware Persistent Control of Heterogeneous Robotic Systems

Heterogeneous multi-robot systems, characterized by members with diverse dynamics, sensing, computation, and energy characteristics, offer a variety of benefits over teams composed of homogeneous members. Appropriate coordination of these systems can combine a team's differing capabilities and enhance a mission by extending individual agent limitations and optimizing performance. However, cooperative efforts may also lead to multi-agent failures. An aerial vehicle dependent on a ground vehicle for power recharging, for example, will fail if the ground vehicle fails. With these considerations in mind, we propose an online framework that prioritizes the safety of a heterogeneous team while accomplishing tasks of differing importance such as a medical-kit delivery task and an aerial photography task. These tasks are unknown before the mission and appear at random locations and times.

Our approach to this problem utilizes distributed spring-mass and magnet artificial physics and graph-theoretical reasoning to produce online controllers for the ground and aerial agents within a heterogeneous robotic system. In addition, by adapting the artificial forces according to agent energy, we are able to recharge agents as needed. We offer a proof of system stability using a Lyapunov analysis of our aerial to ground agent switching policy and demonstrate a persistent approach to dealing with multi-agent failures. During failures, dependent aerial agents seek out safe landing waypoints that delay energy depletion, allowing the team to safely reconnect stranded agents and maintain mission performance. Note that, here we consider heterogeneous systems composed of UAVs and UGVs because of the experimental testbed available in our Laboratory. However our proposed approach is general and works with any robotic system.

3.1 Problem Formulation

We begin by considering a heterogeneous team of $N = N_g + N_a$ agents composed of N_g unmanned ground vehicles (UGVs) and N_a unmanned aerial vehicles (UAVs). We assume the N_g ground agents may instantly recharge any of the N_a aerial agents by a battery replacement operation when two agents reside within a finite radius γ of each other. In addition, both ground and aerial agents may be similarly instantly recharged by residing within γ distance of a home/base. While ground agents may charge one aerial agent at a time, the home station may charge all agents simultaneously.

Consider the area of interest W, assumed to be a simple convex polygon defined as the maximum reachable area by any agent or combination of agents. Define $\mathcal{G}$ as the union of all desired tasks in region W, that is able to grow as new tasks appear. The objective is to eventually achieve tasks in the set $\mathcal{G}$ while avoiding inter-agent collision and maintaining energy on all agents. Each agent has limited energy decaying over time. For the sake of simplicity, we assume that energy of each agent E_i decays linearly as follows:

$$\dot{E}_i(t) = -k_d v_{i,M}^2 \tag{3.1}$$

where $v_{i,M}$ is a constant maximum speed for each robot and k_d is a damping decaying factor. Each agent's energy capacity E_i^0 (i.e. usable energy) varies. For example, a UAV will have lower energy capacity than a UGV and thus its operation will last shorter. Each agent i will have a task j assigned that corresponds to a goal location $g_j = [x_j, y_j]^T$ to visit. We define $\pi_{ji}(t)$ as the task completion status:

$$\pi_{ji}(t) = \begin{cases} 1, & \text{if } d_{ji} \leq \gamma \\ 0, & \text{otherwise} \end{cases} \tag{3.2}$$

where γ is a threshold distance an agent must be within in order to satisfy a task and d_{ji} is the distance between the agent and the task. Formally the problem which we are interested in solving can be formulated as follows.

Problem 1. ***Energy-aware Control of Heterogeneous Robotic Systems:*** *A heterogeneous team of N agents is deployed to visit a set of goals $g_j \in \mathcal{G}$ while ensuring that the energy of each agent remains positive during the entire operation even during the event of an agent failure. New goals may appear at unknown locations and times and are added to $\mathcal{G}$. Given $E_i(t)$ the energy of each robot i at time t defined as in* (3.1) *and $\pi_{ji}(t)$ the task j completion value for agent i defined in* (3.2)*, then we are interested in finding a control*

policy $\boldsymbol{u}_i$ for each agent in the heterogeneous system such that the following constraints are always guaranteed:

$$\begin{aligned} &\Box(E_i(t) > 0) \quad \forall i = \{1, 2, \dots N\} \\ &\Diamond(\pi_{ji} = true) \quad \forall j \in \mathcal{G}, i = \{1, 2, \dots N\} \end{aligned} \tag{3.3}$$

where we have used the notation $\Diamond$ and $\Box$ from traditional Temporal Logic syntax [30] to denote an event that eventually will occur at some unknown future time and an event that is always occurring, respectively.

3.2 Preliminaries - Proximity Graphs

In this section we present the use of proximity graphs to establish connections between agents, tasks, and other supporting agents.

3.2.1 Heterogeneous Proximity Graph and Task Allocation

Due to the heterogeneity of our robot team, we must define an appropriate proximity graph such that the edges correspond to allocated tasks and valid charging stations. For an agent i, the proximity graph $\mathcal{P}_i$ is the union of the sets $\mathcal{A}_i$ defining interactions with charging sources, g_i containing the allocated task (if any), and w_i containing the safety waypoint.

Agent Proximity Criterion

Given an agent i and a valid charging source k, i and k share an edge if k is the closest neighboring reachable source. Let $v_{i,M}$ be the maximum velocity of i and define the number of valid charging sources as N including the fixed home/base then a source k exists in agent i's reachable set $\mathcal{R}_i$ if the following condition holds:

$$k \in \mathcal{R}_i, \quad \text{if } d_{ik} < \min(t_i^E \cdot v_{i,M}, t_k^E \cdot v_{k,M}) \quad \forall k \in N \tag{3.4}$$

where $d_{ik} = \|\boldsymbol{p}_i - \boldsymbol{p}_k\|$ and t_i^E is the remaining time until the energy of agent i is fully depleted which can be estimated from (3.1). From $\mathcal{R}_i$, the closest source is selected to establish the agent-charging edge defined in set $\mathcal{A}_i$:

$$k \in \mathcal{A}_i, \quad \text{if } d_{ik} \leq \min(d_{il}) \quad \forall k \neq l, \text{ and } k, l \in \mathcal{R}_i \tag{3.5}$$

Since there exists only one valid charging station for mobile sources (UGVs in the context of this work), the set $\mathcal{A}_i$ for mobile sources at least contains an edge with the home/base besides edges with the dependent agents.

Task Proximity Criterion

We also establish edges with matching agents and tasks, for example aerial agents with aerial tasks. In this case, allocation is based on a reward function that considers the energy needed to reach a task and the importance of a task. Given an agent i and a type-satisfied task j, i and j share an edge if j is the closest neighboring task among the safely reachable tasks for agent i. Given a supporting agent k and nearest waypoint l, the set of safely reachable tasks for agent i, denoted by $\mathcal{R}_{ig}$, is found according to:

$$j \in \mathcal{R}_{ig}, \quad \text{if } (d_{ij} + \frac{d_{jk}}{v_{i,M}} \cdot v_{k,M} + d_{kl}) < t_i^E \cdot v_{i,M} \quad \forall j \in \mathcal{G} \tag{3.6}$$

where the worst case distance for an agent i to travel is d_{kl} and k is the position of a charging source that maximizes the distance to the closest waypoint. From this set of safely reachable tasks, we allocate the highest rewarding task for agent i. We use α_j to indicate the reward value for task j, assigned by a user. For example, higher rewards may be allocated to tasks related to medical emergencies. The reward of completing a task j is calculated as follows:

$$\mathcal{C}_{ij} = \frac{k_m}{d_{ij}} + k_\alpha \cdot \alpha_j \tag{3.7}$$

where j is a task in the set $\mathcal{R}_{ig}$, k_m and k_α are parameters for negotiating distance versus priority and α_j is the priority of the task. Given the set of reachable tasks $\mathcal{N}_i$ and the set of rewards for completing tasks in $\mathcal{N}_i$ according to (3.7) as $\mathcal{M}_i$, we now select the task that maximizes $\mathcal{C}$:

$$g_i = \{n \in \mathcal{N}_i | \mathcal{C}_{in} = \max(\mathcal{M}_i)\} \tag{3.8}$$

Waypoint Proximity Criterion

Finally, we also desire for our system to be tolerant to failures by continuously locating the closest waypoints. Define the set of all waypoints as $\mathcal{J}$, w as a waypoint in $\mathcal{J}$, and $\mathcal{K}$ as the set of all distances d_{wi}. Then the active waypoint w_i for agent i is given by:

$$w_i = \{w \in \mathcal{J} | d_{wi} = \min(\mathcal{K})\} \tag{3.9}$$

Having found the sets of neighboring agents, the allocated task, and the closest waypoint we now have the full proximity graph for agent i as the union of the sets $\mathcal{A}_i$, g_i, and w_i:

$$\mathcal{P}_i = \mathcal{A}_i \cup g_i \cup w_i \tag{3.10}$$

Fig. 3.1 shows a pictorial representation of a sample heterogeneous proximity graph. Tasks are indicated by colored $\triangle$ (red for UAVs and blue for UGVs). Dotted lines represent allocated task edges while dashed lines indicate energy supporting edges. The home/base station is depicted with a green $\Diamond$ while waypoints for the UAVs by pink $\otimes$.

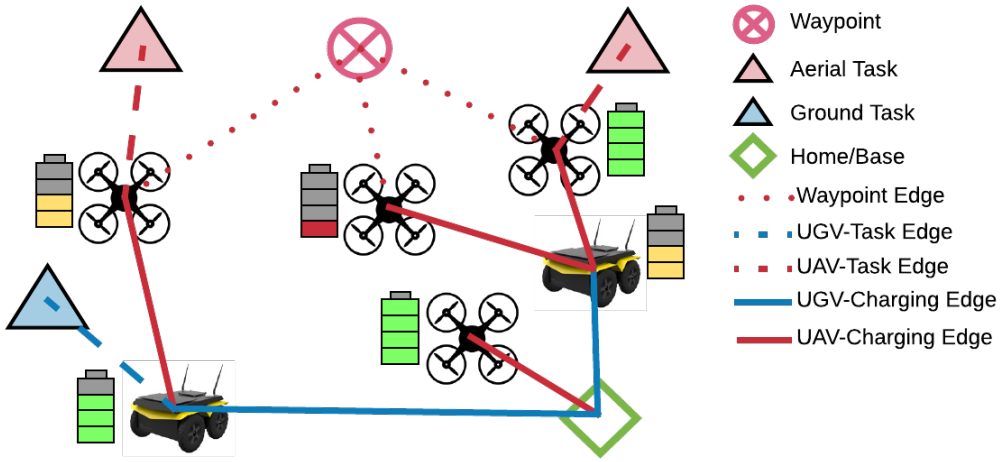


Figure 3.1: Sample proximity graph.

3.3 Distributed Control of Heterogeneous Robotic Systems

Having established the graph connections between agents, waypoints, and tasks, we now present our generalized approach to deploying heterogeneous robotic agents safely during a mission. We assume that mission tasks and robots occupy a planar workspace $W \subset \mathbb{R}^2$ and agents are bounded by a maximum velocity. For any agent i, the desired input is calculated as the sum of the various physics-based interactions according to the graph edges found in $\mathcal{P}_i$.

3.3.1 Spring-Mass and Magnetic Controller

Agent-Task Spring Interactions

Once allocated, agents minimize distances to tasks by using a spring-mass force with damping effects. Define $\boldsymbol{v} = [\ \dot{x} \quad \dot{y}\ ]^\mathsf{T}$ as a vector describing the velocity of an agent. The force on agent i pulling toward the allocated task j with the spring force constant k_{sp}^g and the damping constant k_d^g is described as:

$$\boldsymbol{u}_i^g(t) = \begin{cases} k_{sp}^g \frac{E_i(t)}{E_{max,i}} \cdot d_{ji}\vec{\boldsymbol{d}}_{ji} - k_d^g \boldsymbol{v}_i(t) & \text{if } \mathcal{R}_{ig} \neq \emptyset \\ 0 & \text{otherwise} \end{cases} \tag{3.11}$$

Agent-Agent/Base Spring Interactions

Inter-agent forces account for the system's ability to respond to decreasing energy and produce converging behavior over time. The force on agent i as a result of the inter-agent relationships in the set $\mathcal{A}_i$ with the spring force tuning $k_{sp}^{\mathcal{A}}$ and the damping constant $k_d^{\mathcal{A}}$ is:

$$\boldsymbol{u}_i^{\mathcal{A}}(t) = \sum_{k \in \mathcal{A}_i} \left[k_{sp}^{\mathcal{A}} \left(1 - \frac{E_k(t)}{E_{max,k}} \right) \cdot d_{ki} \vec{\boldsymbol{d}}_{ki} \right] - k_d^{\mathcal{A}} \boldsymbol{v}_i(t) \tag{3.12}$$

Inter-Agent Collision Avoidance

Inter-agent forces are also used to avoid collisions between homogeneous agents capable of presenting as obstacles to each other. Between agent i and all other homogeneous agents $\{1, 2, ..., N_h\}$, the repulsive force with the magnetic force constant k_{mag}^h and the damping constant k_d^h is:

$$\boldsymbol{u}_i^h(t) = \sum_{k}^{N_h} \Big[- \frac{k_{mag}^h \cdot E_i(t)}{d_{ki}^2 \cdot E_{max,i}} \vec{\boldsymbol{d}}_{ki} \Big] - k_d^h \boldsymbol{v}_i(t) \tag{3.13}$$

Agent-Waypoint Spring Interaction

During a ground agent failure, disconnected aerial agents minimize distances to waypoints by using a spring-mass force. The force on agent i due to the nearest waypoint w with the spring force constant $k_{sp}^{\mathcal{W}}$ and the damping constant $k_d^{\mathcal{W}}$ is given by:

$$\boldsymbol{u}_i^w(t) = \begin{cases} k_{sp}^{\mathcal{W}} \cdot d_{wi} \vec{\boldsymbol{d}}_{wi} - k_d^{\mathcal{W}} \boldsymbol{v}_i(t) & \text{if } \mathcal{A}_i = \emptyset \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

Agent-Waypoint Magnetic Interaction

After an aerial agent has successfully arrived at a waypoint, ground agents are exposed to a magnetic force from each stranded agent. The force applied to a ground agent i from the stranded UAV l with the magnetic constant k_{mag}^l and the damping constant k_d^l is described as:

$$\boldsymbol{u}_i^l(t) = \frac{k_{mag}^l}{d_{li}^2} \vec{\boldsymbol{d}}_{li} - k_d^l \boldsymbol{v}_i(t) \tag{3.15}$$

Assembled Controller

The summation of all components yields the final input for each agent:

$$\boldsymbol{u}_i(t) = \boldsymbol{u}_i^g(t) + \boldsymbol{u}_i^{\mathcal{A}}(t) + \boldsymbol{u}_i^h(t) + \boldsymbol{u}_i^w(t) + \boldsymbol{u}_i^l(t) \tag{3.16}$$

3.3.2 Stability Analysis

By constraining the system to produce a connectivity graph as described in Section 3.2, our system is able to construct and destruct spring relationships continuously. While advantageous, this presents a problem in proving the stability of the system. The behavior of this system in fact represents a switched system since there is a coupling between continuous dynamics and discrete events. Agents also establish spring relationships with fixed points in the environment that may appear/disappear in such a way that large amounts of virtual energy may be injected into the system. In this section, we use an appropriate Lyapunov function to prove system stability between switches, followed by an analysis of system stability during switches.

Theorem 1. (Energy Safe Stability of Heterogeneous Robotic Systems) *The virtual-potential-based controller in* (3.16) *with switching topology is safe if it reaches a rest state with constant potential and the system energy is always positive definite.*

In this case safety is equivalent to stability. We will prove the theorem by first showing stability between switches and then stability during switches.

Proof. **Inter-Switch Stability** Let the energy (Lyapunov function candidate) for agent i be given by:

$$\boldsymbol{U}_i = \sum_{k \in \mathcal{A}_i} \boldsymbol{P}_{ik} + \frac{1}{2}\boldsymbol{v}_i^{\mathsf{T}}\boldsymbol{v}_i \quad + \frac{1}{2}k_{sp}^g \frac{E_i}{E_{max}} {d_{ij}}^2 + \frac{1}{2}k_{sp}^{\mathcal{W}} {d_{iw}}^2 + \frac{k_{mag}^l}{2d_{il}} \tag{3.17}$$

where $\boldsymbol{P}_{ik} = \frac{1}{2}k_s \cdot d_{ik}^2$ is conservative. "Conservative" here is used in the sense of a conservative field, in that the integral of any two paths through a vector field with identical endpoints is equal and opposite. The total energy function of our system is then given by:

$$\boldsymbol{V} = \sum_{i=1}^{N} \boldsymbol{U}_i \tag{3.18}$$

Taking the first order derivative of (3.18) and inserting $\boldsymbol{u}_i = \ddot{\boldsymbol{p}}_i$ from (3.16), we obtain that each agent-agent force has an equal and opposite agent-agent force, cancelling all terms except for damping. Thus, with k_d

being the sum of all damping constants, the first order derivative becomes:

$$\dot{\boldsymbol{V}} = -\sum_{i=1}^{N} k_d \cdot \boldsymbol{v}_i^{\mathsf{T}} \boldsymbol{v}_i \tag{3.19}$$

which is as expected since in spring-mass systems energy is dissipated via damping. The second order derivative can be obtained similarly by deriving (3.19):

$$\ddot{\boldsymbol{V}} = -2\sum_{i=1}^{N} k_d \cdot \boldsymbol{v}_i^{\mathsf{T}} \dot{\boldsymbol{v}}_i \tag{3.20}$$

Using the Barbalat lemma [78] and [13], we obtain that our system is stable between switches if the following criteria are met:

- V is positive definite ($V > 0$)
- $\dot{V}$ is negative semi-definite ($\dot{V} \leq 0$)
- $\ddot{V}$ is bounded

which are all satisfied in our case, hence proving that the system is stable during switches and thus that each agent's energy decreases but never goes negative. □

Proof. **Switching Stability** In order to prove stability during topology switches, we use an argument similar to that presented by [78] and [13] in which authors introduce a positive definite energy reserve E to cancel out switching effects and prove stability. We define the energy change over a switch at time t as:

$$s_i(t) = \frac{1}{2}\left(\lim_{\tau \to t^+} \boldsymbol{U}_i(\tau) - \lim_{\tau \to t^-} \boldsymbol{U}_i(\tau)\right) \tag{3.21}$$

The total change in energy during a switch is defined as the summation of all switches that occur at time t

$$\sum_{i \in \mathcal{S}_i} s_i(t) = \lim_{\tau \to t^+} \boldsymbol{V}(\tau) - \lim_{\tau \to t^-} \boldsymbol{V}(\tau) \tag{3.22}$$

These switches occur when agents construct and destruct spring relationships with other agents, tasks, and the home/base. We use a new potential function to prove stability: $\boldsymbol{V}' = \boldsymbol{V} + E$, which is composed of the summation between the accumulated energy reserve over time and the new energy of the system after the switch. The reserve $E = \sum_{i=1}^{N} E_i$ where each local E_i, is initially set to the non-negative initial physical

energy of agent i and is the solution to the following differential function:

$$\dot{E} = -\sum_{i=1}^{N} k_d \boldsymbol{v}_i^{\mathsf{T}} \boldsymbol{v}_i \tag{3.23}$$

where $\dot{E}$ is directly found from the inter-switch stability proof and is the rate of virtual energy dissipated from the system. Contrary to the approaches of [78] and [13], our energy reserve decays according to the physical power expended by each agent. As such when an agent recharges, the agent's local energy reserve resets to maximum. By design, our agents always maintain enough physical power to return to waypoints in the event of failures during task allocations. Thus, our energy reserve always possesses enough energy to cancel out injected energy from introduced switches. The evolution of $\boldsymbol{V}$ and E during a switch at time t is given by:

$$\begin{aligned} \lim_{\tau \to t^+} \boldsymbol{V}'(\tau) &= \lim_{\tau \to t^-} \boldsymbol{V}(\tau) + \sum_{i \in N} s_i(t) + E(t) \\ E(t) &= \lim_{\tau \to t^-} E(\tau) - \sum_{i \in N} s_i(t) \end{aligned} \tag{3.24}$$

Now, substituting the expression of $E(t)$ into the first line of (3.24), we obtain:

$$\begin{aligned} \lim_{\tau \to t^+} \boldsymbol{V}'(\tau) &= \lim_{\tau \to t^-} \boldsymbol{V}(\tau) + \lim_{\tau \to t^-} E(\tau) \\ \lim_{\tau \to t^+} \boldsymbol{V}'(\tau) &= \lim_{\tau \to t^-} \boldsymbol{V}'(\tau) \end{aligned} \tag{3.25}$$

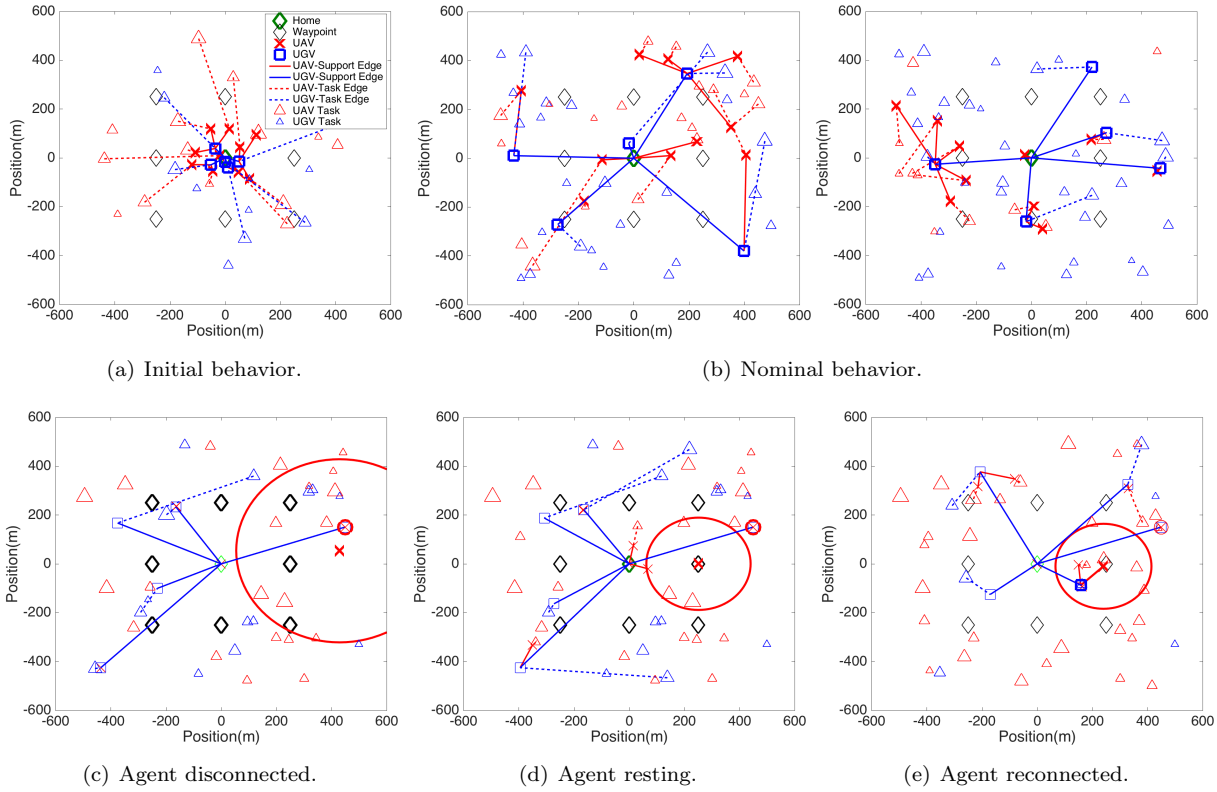
in which the energy injection due to switching has been cancelled and our Lyapunov candidate $\boldsymbol{V}'$ remains continuous. Since $\dot{\boldsymbol{V}}$ and $\dot{E}$ are both negative semi-definite and $\ddot{\boldsymbol{V}}$ and the dissipated energy of the system are also bounded, we find that $\dot{\boldsymbol{V}}'$ is negative semi-definite and $\ddot{\boldsymbol{V}}'$ is bounded. As such, our system is stable during and between switches. □

3.4 Simulation Results

In this section, we present simulation results to demonstrate the use of distributed control laws for task completion in heterogeneous aerial-ground robotic systems. In the simulations shown in Fig. 3.2, the workspace has a dimension of 1km×1km with resting waypoints marked with black ◊. Tasks (△ shapes) appear randomly with random priorities (different sizes of △) and are divided between aerial and ground tasks (different colors of the △ shapes). The base station is depicted with a green colored ◊ shape.

Our simulated team is composed of $N_g = 5$ UGVs with maximum velocities of 1 m/s and are able to recharge aerial vehicles, and $N_a = 10$ UAVs with maximum velocities of 3 m/s and have one twentieth the

operational lifespan of a UGV. We assume that robot and task states are continuously shared among the team and that take-off and landing times, recharge times, and communication delays are all negligible.



(a) Initial behavior.

(b) Nominal behavior.

(c) Agent disconnected.

(d) Agent resting.

(e) Agent reconnected.

Figure 3.2: Evolution of simulated mission completion using 5 ground agents and 10 aerial agents. (a) shows the initial condition. In (b) the heterogeneous system is deployed following our decentralized algorithm. In (c) an aerial agent is disconnected from the team due to a UGV failure and moves to the nearest waypoint and rests for an extended period (d). In (e), a ground agent is able reconnect the UAV to the graph to resume normal behavior. the shrinking circle shows the range of that specific UAV. We omitted all other ranges to make the figure less cluttered.

Fig. 3.2 shows snapshots of the simulated team's operation from a 30 minute mission. Dotted edges indicate task-agent forces while solid edges indicate agent-agent/base forces. Fig. 3.2(a) and Fig. 3.2(b) demonstrates agents adapting spring forces to complete tasks and recharge over time while Figs. 3.2(c,d,e) show an aerial agent being forcibly disconnected and the team's ability to adapt and maintain mission performance. The energy plot in Fig. 3.3 further reinforces the team's persistence by showing the energy of each agent during the UGV failure event.

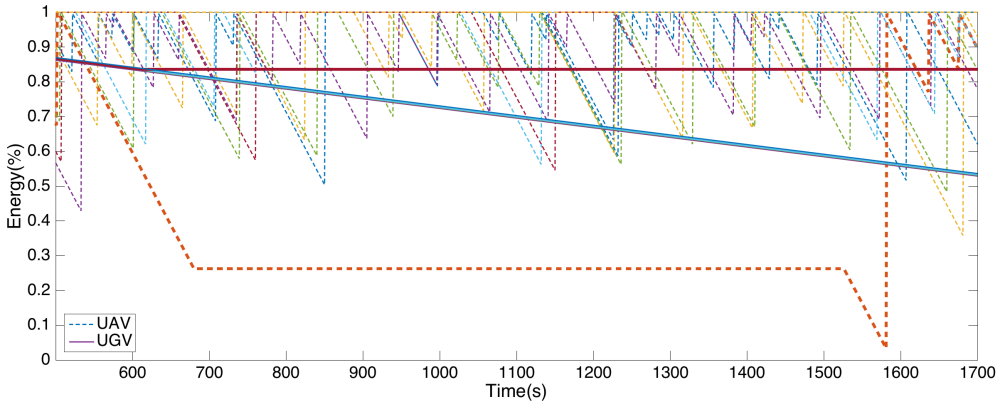


Figure 3.3: Energy percentage evolution of each agent during a UGV failure for a portion of the entire mission. Ground agents are indicated by solid lines and aerial agents are indicated by dotted lines. Bold lines indicate the UGV failure and the corresponding UAV response and eventual re-connection.

3.5 Experimental Results

In this section we present our experimental results using a team composed of one aerial AscTec Hummingbird quadrotor, one Clearpath Jackal UGV, and one Turtlebot UGV. We use the Robot Operating System (ROS) on a main Linux computer to interface with the 3 robots as depicted in Fig. 3.4.

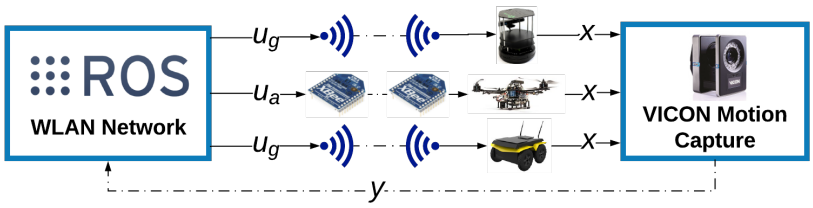
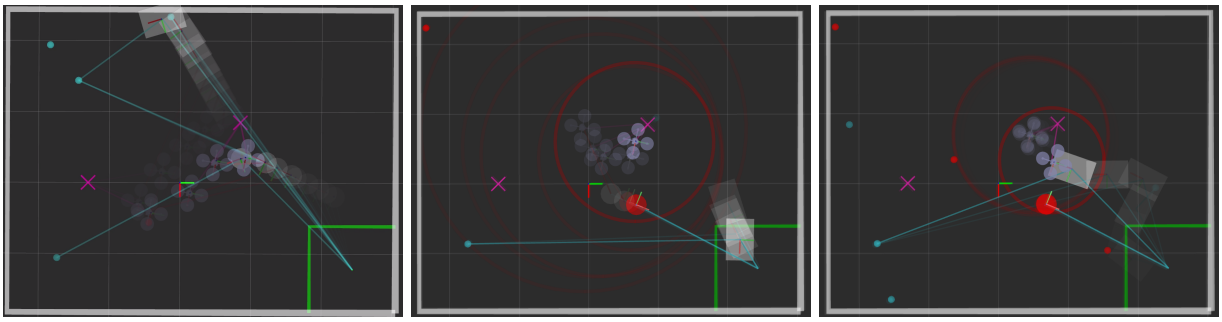


Figure 3.4: Architecture of the experiment.

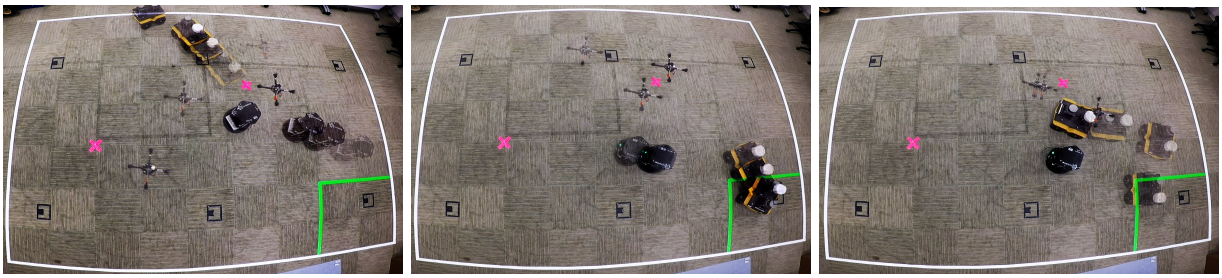
Because of the limited Lab space, each UGV was assumed to have 100s runtimes and 0.15m/s max velocities while the UAV was assumed to have a 25s runtime and 0.3m/s max velocity. States of all agents were provided by a VICON motion capture system and tasks were generated in a 5.5m×4.3m rectangular perimeter. Figs. 3.5(a,d) show overlapped sequences of snapshots for a heterogeneous mission running our proposed controller without failures. In Figs. 3.5(b,e), the turtlebot UGV fails (highlighted in red) forcing the Hummingbird UAV to re-route to the nearby waypoint denoted by the pink cross. Finally, in Figs. 3.5(c,f), the Jackal UGV is able to recover the stranded UAV and complete the mission with the control law in (3.16).

3.6 Conclusions and Future Work

In this chapter, we have presented a distributed controller for a heterogeneous robotic system that is capable of completing goals while balancing power constraints. While our approach is able to fulfill our desired



(a) Nominal Operation - ROS Rviz visualization (b) UGV Failure - ROS Rviz visualization (c) UAV Reconnection - ROS Rviz visualization



(d) Nominal Operation - Hardware visualization (e) UGV Failure - Hardware visualization (f) UAV Reconnection - Hardware visualization

Figure 3.5: Experimental results - nominal and recovery behaviors.

behavior, heterogeneous coordination is still a vastly complex problem.

Optimal solutions often scale poorly as the number of desired behaviors or agents increases, indicating a tradeoff between a heterogeneous team's ability to handle various situations and an approach's ability to render solutions for online performance. In future work, we will further explore this tradeoff and identify solutions that improve the generality of a system with multiple agents. We plan also to incorporate more complicated scenarios with fixed obstacles and cooperative task allocation.

Chapter 4

A Testbed for Autonomous Hazardous Tunnel Mapping

Having discussed an approach to coordinate multiple agents for task completion, we now expand more on the capabilities of a single agent. Classically, robots have been deployed to handle tasks in environments that are considered too dangerous for human operators, such as using drones to assess radiation levels following a nuclear meltdown or using tele-operated ground robots to defuse bombs. In underground environments, humans are exposed to potential dangers due to unknown and uneven terrain, low or no lighting, and falling rock. Normally, these hazardous elements would be enough to prevent frequent human operation in these environments. However, due to the possibility and severity of structural failure, underground inspection presents a problem that demands considerable and regular attention, forcing human operators to risk injury.

Figure 4.1: The Jackal UGV outfitted with the Velodyne VLP-16 LiDAR in operation.

Autonomous or tele-operated underground inspection therefore presents a desirable solution that is able to alleviate the risk for human harm while still providing adequate performance. Due to increasing urbanization and the need for infrastructure updates, the demand for automated underground inspection appears to be growing and is evidenced by the growing number of autonomous robot services for underground missions [29, 5, 71] and by the recent DARPA subterranean challenge designed to advance research for underground robots [25]. In this chapter, we detail our continuing

Figure 4.2: A watery section of the Blue Ridge Tunnel

work on designing an autonomous navigation and mapping approach

for hazardous environments, the construction of our specialized testbed, and our ongoing tests in a real underground tunnel environment. Fig. 4.1 demonstrates our specialized UGV autonomously navigating and mapping in the testing environment.The Claudius Crozet Blue Ridge Tunnel was initially constructed in 1849 and served as the longest US railroad tunnel until 1944 when a parallel route was built and the tunnel was decommissioned [24]. The tunnel is approximately 4264 feet long, 16 feet wide, and 20 feet tall and contains varying terrain features, ranging from smoothly paved to gravely and uneven. Fig. 4.2 demonstrates an example portion of the tunnel filled with water. In our experiments, we used LiDAR data to perform an extensive full mapping of the tunnel to aid the restoration process.

4.1 Specialized Underground Vehicle Testbed

In this work, we designed a platform to enable autonomous exploration and mapping in hazardous terrain. Specifically, our testbed vehicle is a Clearpath Jackal unmanned ground vehicle outfitted with a VLP-16 Velodyne "Puck" 3D LiDAR sensor, a vertically oriented Hokuyo UST-10LX 2D LiDAR sensor, a sensor payload with multiple IMU sensors and cameras, and a forward facing high-power lamp for vision and human operators. This platform was specifically designed with the Blue Ridge Tunnel in mind, and utilizes both LiDAR sensors to generate accurate maps of the environment. The Hokuyo LiDAR sensor augments the

VLP-16 LiDAR by mapping data missed in the Velodyne LiDAR's natural blind-spot. Fig. 4.3 shows our specialized testing platform.

Figure 4.3: Specialized UGV platform for underground tunnel experiments.

In the next sections, only the VLP-16 is used for localization and obstacle avoidance due to the preferred orientation of the VLP-16 and the greater data acquisition. However, the Hokuyo UST-10LX could be refitted to perform the same desired tasks.

4.2 LiDAR Mapping Through Graph-SLAM

With perfect state estimation, maps could be easily generated by integrating LiDAR scans according to the state of the robot at the time of each scan. However, due to sensor noise and drift, more robust techniques are needed to accurately map an environment as state estimation error accumulates over time, creating inconsistencies in a generated map. This problem is known in robotics literature as Simultaneous Localization and Mapping (SLAM) and tasks a robot with creating an accurate map of an environment while continuously identifying the robot's position in the map. In this chapter, we leverage the popular Graph-SLAM algorithm to estimate and eliminate accumulated drift through multiple observations of landmarks, also known as loop closures. Landmarks, in this case, refer to static objects in the environment that may be easily found multiple times. Over multiple loop closures, the algorithm is able to converge on the ground truth state of the robot and correct accumulated errors in the map.

4.2.1 SLAM as an Optimization Problem

Graph-SLAM focuses on constructing a directed graph of observations based on the states at which LiDAR scans are observed and the observations to landmarks in the environment. Nodes and edges therefore represent the beliefs on the accumulated observations and the relationships between these beliefs. Fig. 4.4 illustrates an example in which state estimates and landmark distance observations are represented graphically.

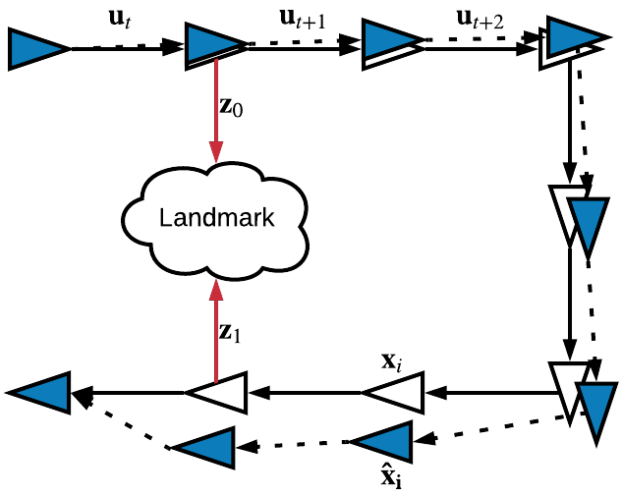


Figure 4.4: An example graph with a robot's true state denoted by white triangles, estimated states denoted by blue triangles, landmark observations denoted by red edges, and robot inputs denoted by black edges.

Having established a graphical representation of a robot's movements and associated LiDAR scans, loop closures allow the Graph-SLAM algorithm to eliminate errors that have accumulated over time [38]. By using multiple observations of a landmark to identify fixed points in the environment, Graph-SLAM is able to compare actual landmark measurements with expected landmark measurements. This allows the SLAM problem to be formulated as an optimization problem in which the error to be minimized is the difference between expected landmark distances and observed landmark distances. Formally, this error function between two nodes is described by:

$$\boldsymbol{e}_{ij}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{z}_{ij} - \hat{\boldsymbol{z}}_{ij}(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{4.1}$$

where $\boldsymbol{x}$ is a node denoting a robot state, $\boldsymbol{z}$ denotes a true landmark observation, and $\hat{\boldsymbol{z}}$ denotes an expected landmark observation. The minimization function is then the sum of all observation errors. For a set $\mathcal{C}$ that contains pairs of indices denoting a pair of landmark observations from two differing states and the information matrix Ω containing the sensor uncertainty, we seek to find the configuration of nodes $\boldsymbol{x}*$ that maximizes the likelihood of all observations:

$$\boldsymbol{F}(\boldsymbol{x}) = \sum_{\langle i,j \rangle \in \mathcal{C}} \boldsymbol{e}_{ij}^T \Omega_{ij} \boldsymbol{e}_{ij} \qquad \boldsymbol{x}* = \underset{\boldsymbol{x}}{argmin}\, \boldsymbol{F}(\boldsymbol{x}) \tag{4.2}$$

Under the ROS framework, we are able to utilize the Berkeley Localization and Mapping (BLAM) [61] package to integrate 3D LiDAR scans. BLAM utilizes the Georgia Tech Smoothing and Mapping (GTSAM) [26] optimization backend to perform efficient online Graph-SLAM as previously described.

4.3 LiDAR-Based Autonomous Navigation

Using Graph-SLAM to map the tunnel, we now introduce a controller to enable autonomous operation and obstacle avoidance. Leveraging knowledge of the interior of the tunnel, we designed a controller to track a reference signal that is a summation of an attractive force to the centroid of open space in the tunnel and a repulsive force from the closest point within a defined distance. Using the previously defined UGV model from (2.3), we design controllers on error signals for the two inputs, desired linear velocity and desired angular velocity. In both cases, we used LiDAR scans from the robot's local frame to perform obstacle avoidance in order to maintain a fast response.

In order to extract the centroid of open space, we first perform a spatial filter to remove the ground plane and to remove points behind the robot. We then use the average positions of all points to the left and right of the robot to extract the centroid of the open space. Formally, given a left set of filtered points $\mathcal{S}_l$ and a right set of filtered points $\mathcal{S}_r$, where each point is a 3-tuple describing position, extraction of the centroid C in the robot's local frame is found by:

$$C(t) = \frac{\bar{\mathcal{S}}_r(t) - \bar{\mathcal{S}}_l(t)}{2} \tag{4.3}$$

Having extracted the centroid for open space, we now calculate the desired linear velocity and heading of the UGV using a summation of the centroid C and the the closest obstacle point O, also a 3-tuple describing position, in the local frame:

$$\begin{aligned} \dot{x}_{des}(t) &= k_a C(t) + k_r O(t) \\ \theta_{des}(t) &= \arctan\Big(\frac{k_a C_y(t) - k_r O_y(t)}{k_a C_x(t) - k_r O_x(t)}\Big) \end{aligned} \tag{4.4}$$

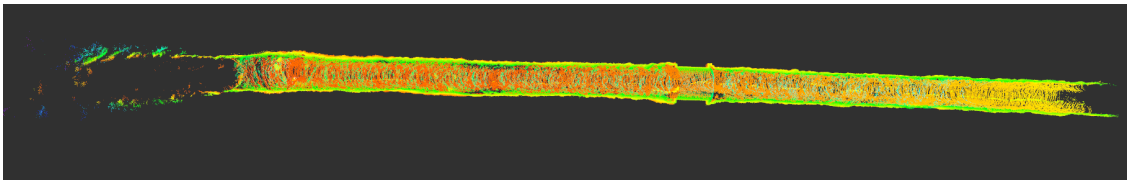
where k_a and k_r tune the sensitivity of the controller to obstacles.

4.4 Mapping Results and Future Work

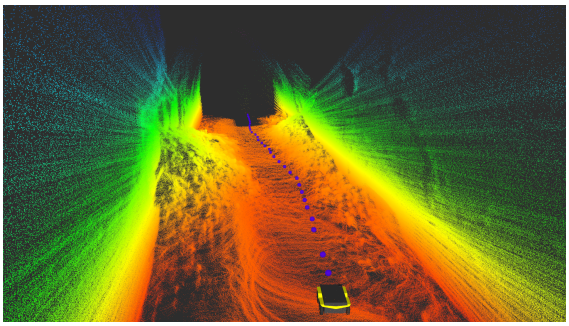
Over multiple experiments, we were able to extract a number of high quality structural maps using the Graph-SLAM algorithm and successfully perform autonomous avoidance in portions of the tunnel. In addition, we were able to generate maps using a vertically oriented 2D LiDAR as well based on the localization estimates provided by Graph-SLAM. In Fig. 4.5, the results of the autonomous mapping and obstacle avoidance experiments are shown.

Fig. 4.5(a) depicts the full mapping of the tunnel as extracted by Graph-SLAM while Fig. 4.5(b) depicts the autonomous avoidance and exploration controller adapting the robots trajectory to avoid a large pile of bricks visible in the accumulated LiDAR scans. In Fig. 4.5(c), a generated tunnel mapping using the

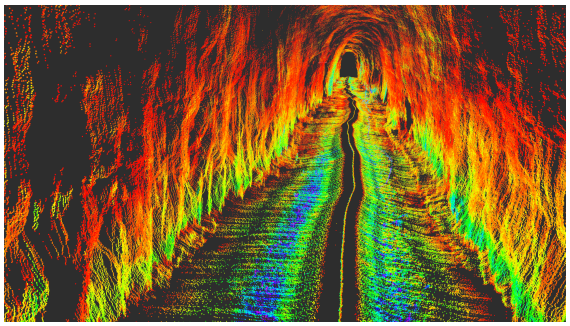
vertically aligned 2D LiDAR sensor is shown, demonstrating a differing map but with the same drift-free result.



(a) Full mapping of the Blue Ridge Tunnel.



(b) Demonstration of the autonomous avoidance and mapping.



(c) 2D Vertical Mapping of the Blue Ridge Tunnel.

Figure 4.5: Experimental Results of the Blue Ridge Tunnel Mapping

While we were able to extract excellent maps, we plan to continue this work by improving upon a number of limitations in our approach. By using BLAM, we were unable to directly control the state estimation method and were forced to use the package's internal state estimation based on the generalized Iterative Closest Point (gICP) algorithm, which identifies transformations between two overlapping scans [76]. However, in environments with similar features (such as a smooth underground tunnel), gICP may fail to accurately extract transformations. We therefore plan to integrate gICP with our existing sensor fusion techniques to provide improved state estimation with wheel encoder and IMU data. We also plan to improve our autonomous exploration controller since it is currently unable to handle extremely difficult terrain where identifying locations of smoother traversal may yield improved performance. Furthermore, in an underground tunnel, puddles and wet rock act as additional obstacles that may be difficult to avoid through our described obstacle avoidance approach. By using machine learning techniques, we may be able to infer from LiDAR scans or camera data possible locations of stable and dry ground.

Chapter 5

LiDAR Sensing Vulnerabilities

In the previous chapter, we relied on consistent and accurate measurements from LiDAR sensors to perform mapping and obstacle avoidance. However, in some cases, these measurements may no longer be trustworthy. In this chapter, we investigate LiDAR sensors and how these devices may be manipulated. Light Detection and Ranging (LiDAR) sensors are devices that measure distances to objects by emitting infrared (IR) laser pulses and measuring reflected light. Classically used for terrain mapping and meteorological studies [58, 37], LiDAR sensors generate depth images when used in organized vertical layers and spun at high speeds. At a low level, these depth clouds are capable of capturing important environment information such as road markings, trees, people, and cars. Additionally, due to their computational simplicity and high resolution, LiDAR sensors provide a variety of benefits over other depth-sensing methods such as stereo vision and radar. Inherently, this is extremely appealing for many autonomous applications including autonomous transportation, surgical robotics, and field robotics that depend on lightweight high granularity depth information to successfully perform their tasks. In particular, LiDAR sensors are most commonly used to enable automation and self-driving cars and are found on many research and commercial platforms [4, 7, 8, 20, 22, 89, 91, 95].

LiDAR sensors contribute to the autonomy of a system by enabling certain critical functions that other sensors are unable to offer. Due to the high accuracy of the generated depth images, also known as pointclouds, LiDAR sensors allow an autonomous system to build high quality 3D maps of an environment (see top illustration in Fig. 5.1). Intuitively, this ability has a variety of natural benefits for human observation as these maps would allow an operator to assess structural quality in dangerous areas or dynamic information in busy and difficult to reach areas. For autonomous systems though, these maps also offer high accuracy localization.

Once a map has been generated, autonomous systems may precisely pinpoint their current position relative

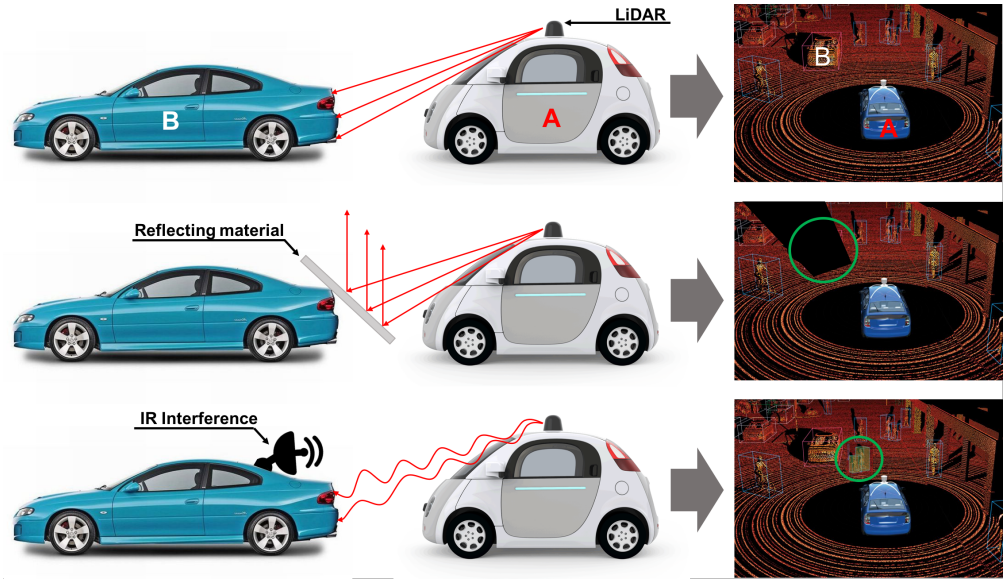


Figure 5.1: Pictorial illustration of how reflective materials and IR interference are able to compromise LiDAR scans. The top illustration depicts a normal LiDAR behavior. In the middle, a mirror hides objects by redirecting scans into the sky while in the bottom, IR interference is able to generate false measurements.

to the map by comparing new LiDAR scans with the previously accumulated scans encoded in the map. This process is formally known as localization and allows autonomous systems to locate themselves with high precision [55]. While never formally stated by commercial industries, the widespread use of map-based localization techniques for autonomous operation is seemingly validated in various blog posts, videos, and discussions presented by Waymo and Ford on self-driving cars [19, 90], the start-up company Civil Maps dedicated to creating localization maps for roadways [21], recent updates to Google's "street-view cars" designed to augment their roadside images with LiDAR scans [3, 63], and the wide variety of map-based localization literature in mobile robotics [9, 42, 66].

High-resolution pointclouds also allow autonomous systems to perform collision avoidance and path-planning behaviors in real-time. By observing static and dynamic obstacles in the vicinity, autonomous systems are able to identify safe open spaces. Radar, stereo cameras, and ultrasonic sensors are also capable of identifying obstacles but have various shortcomings as LiDAR depth information is still significantly more accurate and stereo cameras require additional computation.

On any autonomous system, high-precision localization and obstacle avoidance are highly critical for safe operation. However, despite the dependency of autonomous systems on LiDAR technology, cyber-physical security concerns and sensor vulnerabilities still continue to pose dangerous unanswered questions regarding the reliability of these systems. In particular, existing literature on LiDAR vulnerabilities has been extremely sparse despite its pivotal role in autonomous navigation and obstacle avoidance.

Figure 5.2: LiDAR sensors used on a variety of autonomous systems.

Like most other sensors, LiDAR sensors are unable to identify any abnormal physical characteristics of measured targets. Reflective, absorbent, and semi-transparent surfaces are able to create anomalous measurements by undermining the basic principles of LiDAR operation while foreign IR signals are able to produce anomalous measurements by introducing large amounts of interference. The pictorial representations in Fig. 5.1 depict how a mirror and an IR emitter opportunely placed in the environement are able to alter the true scans captured by a LiDAR sensor by redirecting pulses elsewhere (middle figure) and by creating false measurements (bottom figure).

With the growing prevalence of LiDAR technology on autonomous platforms, vulnerabilities may be able to cause dangerous failures. These systems which are shown in Fig. 5.2 and include quadrotor unmanned aerial vehicles, self-driving cars, and humanoid robots, may depend on the data acquired by LiDAR sensors to guarantee the safety of their behaviors. On self-driving cars, these sensors are even considered fundamental for achieving higher levels of performance.

As shown in Fig. 5.3, a report by Frost and Sullivan confirms that LiDAR technology is a designated necessity for achieving higher levels of self-driving automation [35]. Our analysis shows that by leveraging these vulnerabilities, an attacker may manipulate LiDAR measurements and thus compromise various autonomous system operations including localization, mapping, and obstacle avoidance. Relying on other sensors may not be sufficient to address this issue as existing literature has also already shown that radar, ultrasonic, and stereo camera sensors may be attacked with relative ease leading to a fully compromised system [97, 99]. We therefore explore the possibility of using these vulnerabilities to attack LiDAR sensors.

The LiDARs under investigation in this chapter include the Velodyne VLP-16 “Puck”, capable of generating 3D maps with a range of 100m and commonly found on self-driving cars, also depicted in Fig. 5.2, the Hokuyo UST-10LX, UST-20LX, and UTM-30LX, three 2D LiDARs capable of accurate 10m, 20m, and 30m range

measurements respectively and commonly used for autonomous robotics, and the Asus Xtion Pro Live, an RGB-D IR range sensor capable of accurate 3m range measurements commonly used for gaming and specific indoor robotic applications.

This chapter is organized as follows. Section 5.1 describes the mechanics and vulnerabilities of the LiDAR sensor and validates the vulnerabilities with experimental data. Section 5.2 discusses our LiDAR-based attacks on autonomous platforms and Section 5.3 provides some discussion on our findings with additional insight into other attack scenarios and their consequences. We conclude in Section 5.4 with our final remarks and plans for future work.

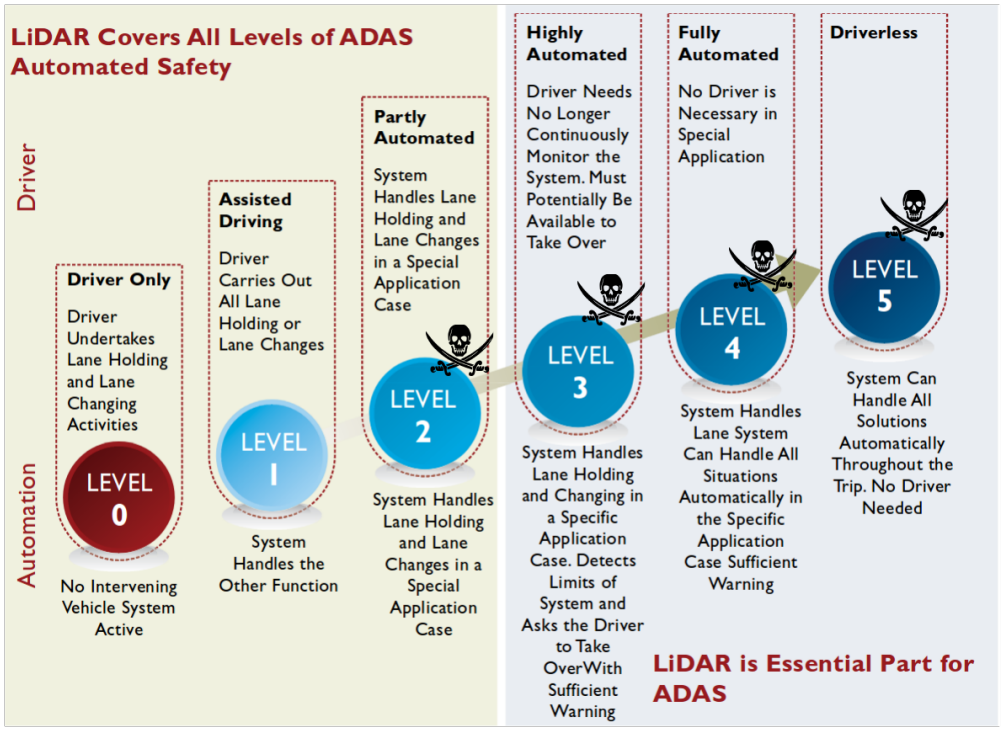


Figure 5.3: LiDAR necessity for higher-level ADAS according to Frost and Sullivan [35]. Compromising LiDAR sensors would impact levels 2-5.

5.1 LiDAR Vulnerability Analysis

In this section, we describe the underlying mechanics behind LiDAR technologies and the impacts of specialized surface features and foreign IR pulses as potential attack vectors. We validate our hypotheses with experimental data demonstrating the impacts of specialized surface features and IR interference.

5.1.1 Preliminaries: Principles of Operation

LiDAR sensors are composed of laser emitters and photodiodes that work together to make range measurements. The emitter creates a burst of light usually focused by a lens or lens assembly that collides with a surface in the environment. During this collision, the pulse of light is scattered in all directions including back along the original emission trajectory. This phase, in which the pulse is reflected back towards the original sensor, is known as *remission*. The photodiode detectors then measure the scattered photons that return along the original trajectory, capturing both the time of flight (t_f) and intensity of the returning pulse. Generally, the emitted pulses operate at either 905nm or 1550nm wavelengths due to restrictions in atmospheric transmission windows and high power pulsing sources [96].

Distance D to reflected surfaces is then calculated as a function of t_f by the following mathematical relation:

$$D = \frac{c \cdot t_f}{2n} \tag{5.1}$$

in which c is the speed of light and n is the refractive index of the travelling medium. In most cases, the travelling medium is air with $n = 1.000292$. Leveraging (5.1) to project pulses along differing trajectories, LiDAR sensors are capable of generating environmental representations as a collection of distance measurements. For example, many commercial LiDAR sensors capture data in a wide field of view by using a motor to rotate a set of vertically aligned emitters [57, 94]. Different variants offer differing capabilities in the form of measurement frequencies, measurement accuracy, and vertical/horizontal field of view.

Sensor	V. VLP-16	H. UST-10LX	H. UST-20LX	H. UTM-30LX	A. Xtion Pro Live
Range	100m	10m	20m	30m	3.5m
Wavelength	903nm	905nm	905nm	905nm	830nm
Nominal Freq.	10 Hz	40 Hz	40 Hz	40 Hz	30 Hz
Horizontal R.	360°	270°	270°	270°	58°
Vertical R.	15°	2D	2D	2D	70°
Error	$<$ 5cm	$<$ 3cm	$<$ 3cm	$<$ 3cm	$<$ 2cm

Table 5.1: Test sensors specifications

5.1.2 Experiments: Investigations into Specialized Surfaces and IR Interference

In this section, we describe our experimental procedures to validate the corrupting capabilities of reflective, dark, and semi-transparent surfaces and the noise-inducing capabilities of identical-wavelength IR interference. To validate the impacts of these vulnerabilities, we have tested commonly used commercially available LiDARs displayed in Fig. 5.4, specifically: 1) the Velodyne VLP-16 "Puck" 3D LiDAR sensor found on Uber and Ford's self-driving 2016 Ford Fusion prototypes, Bosch and Mercedes-Benz's partnered self-driving 2017

V-class prototype, and Boston Dynamic's 2017 Atlas humanoid robot (the Velodyne HDL-32E is also found on Boston Dynamic's Spot) [16, 18, 20, 28], 2) the Hokuyo UST-10LX, UST-20LX, and UTM-30LX 2D LiDAR sensors found on the 2016 AscTec Pelican, 2016 Clearpath unmanned ground vehicles, 2012 Black-I Landshark, and the 2011 Willow Garage PR2 robot [4, 15, 22, 95], and 3) the Asus Xtion Pro Live RGB-D sensor found on the 2016 Open Source Robotics Turtlebot 2 [65].

Specific details of all test sensors are listed in Table 5.1 and are shown in Fig. 5.4. While the Asus Xtion Pro Live does not qualify as a LiDAR sensor, the sensor also uses an IR-based mechanic to acquire range information. Commonly used for interactive gaming and robotic perception tasks, kinect-like RGB-D sensors project IR data in a dot matrix that may be retroactively mapped in order to detect depth [100].

Due to differing intensity measurement scales, Velodyne, Hokuyo, and Asus intensity measurements cannot be directly compared as Velodyne measurements are normalized, the upper limit of Hokuyo intensities is unknown, and Asus intensities are not measured. However, trends in intensity data can still be observed and compared with the expected error from their data sheets and are discussed in our findings. In each experiment, 50 samples are collected from each sensor in order to observe characteristics in the artificially induced noise, bias, and remission degradation. While each of our sensors is mounted to a robot, data collection and performance is unaffected and mounting is performed simply for ease of use. We also leverage a Vicon motion capture system to obtain ground truth measurements.

In our preliminary research, we also attempted to find how well the aforementioned vulnerabilities were recognized by our test sensor manufacturers, Velodyne and Hokuyo. Our searches found that the Velodyne VLP-16 was calibrated to measure dark, reflective, and semi-transparent surfaces by returning different packet signatures for higher and lower intensity values as found when measuring dark surfaces and retro-reflective surfaces but found no mention of IR-based interference or the limits of the VLP-16's ability to scan reflective, dark, and semi-transparent surfaces [87]. Our searches also yielded no information whatsoever on specialized surface scan errors or IR-based interference as potential sources of measurement degradation in the associated Hokuyo datasheets [43, 44].

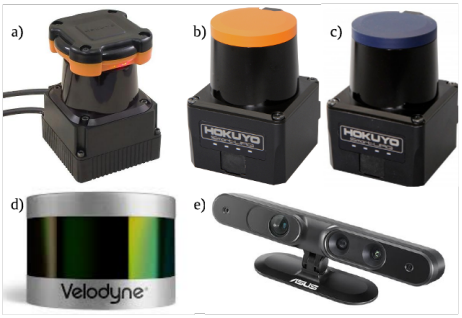


Figure 5.4: The Hokuyo UTM-30LX (a), UST-10LX (b), UST-20LX (c), Velodyne VLP-16 "Puck" (d), and Asus Xtion Pro Live (e) from top-left to bottom-right

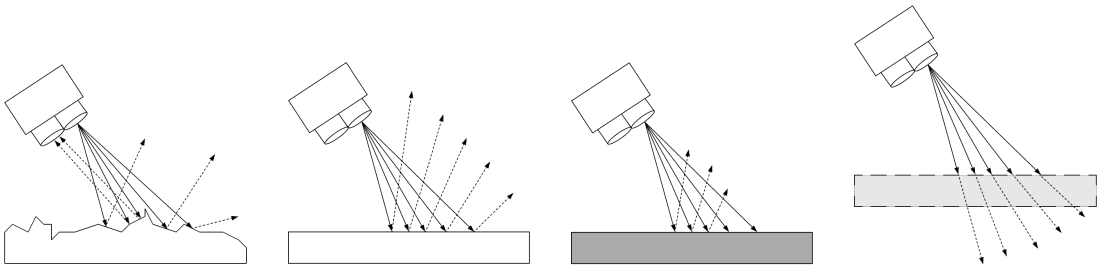
5.1.3 Leveraging Specialized Surface Characteristics to Compromise LiDAR

When analyzing LiDAR performance, we first consider the LiDAR pulse dispersion effects of certain specialized surfaces. Poor reflections off of surfaces are able to corrode a LiDAR sensor's ability to appropriately measure distances. The quality of LiDAR-generated maps and scans therefore is highly dependent on the textures of an environment. Rough surfaces scatter photons well and are clearly defined in the generated scans, as illustrated in Fig. 5.5(a). Smooth surfaces, however, redirect photons along new trajectories and create scan deformities (Fig. 5.5(b-d)).

In particular, we consider the unique effects that occur when a stream of photons impacts a smooth surface as a potential method of attack. First, smooth surfaces with high reflectivity (also known as specular reflectivity, like mirrors) are capable of redirecting emitted photons along new trajectories without dispersing the initial emission. The LiDAR sensor is incapable of detecting the trajectory disruption and instead incorrectly scans the environment (Fig. 5.5(b)). This phenomena would allow an attacker to poison LiDAR-generated maps, directly compromising LiDAR-based localization by injecting false objects at selected locations.

If the smooth surface has instead a low reflectivity (such as on a dark surface as depicted in Fig. 5.5(c)) then the stream of photons is partially absorbed before being reflected. All surfaces naturally absorb some amount of light but darker surfaces tend to absorb a larger amount, before reflecting to new surfaces and potentially causing a small or nonexistent return in data. The LiDAR sensor then interprets the data as missing and assumes that along that trajectory, no obstacle exists within its maximum range.

Finally, semi-transparent surfaces (commonly found on skyscraper exteriors and tinted vehicle windows), also known as one-way mirrors, are also able to impact the performance of LiDAR photon streams. Similar to dark surfaces, these surfaces allow the photon stream to pass through but partially absorb photons along the way (see Fig. 5.5(d)). As a result, semi-transparent surfaces do not appear in scans and objects masked



(a) Rough surface (b) Reflective surface (c) Dark surface (d) Semi-transparent surface

Figure 5.5: Comparison of reflection effects on various surface characteristics

by these surfaces yield lower remissions. By utilizing multiple layers or combining semi-transparent surfaces with dark surfaces an attacker may be able to mask targeted objects. In tandem with reflective surfaces, an attacker would be able to greatly alter a LiDAR sensor's perception of the environment as we demonstrate experimentally in the next section.

Reflective and Dark Surfaces

In these experiments, we consider the effects of reflective and dark smooth surfaces on LiDAR data. In particular, we investigate the degradation performance of these materials as dependent on the impact angle of the reflecting surface. We also provide our findings on the impacts of mirrors on RGB-D kinect-like sensors due to their usage of IR data. RGB-D sensors, however, do not rely on time of flight and may not be affected by some of the attacks. A cardboard panel coated with an off-the-shelf paint designed to absorb light [53] is used to investigate decreased remission returns and a 0.3m^2 mirror is used to investigate disrupted remission returns. In total, our tests investigate eight different angles ranging from 0°-70° that would theoretically yield optimal and sub-optimal scanning performance.

(a) Normal surface mapping performance

Sensor	Angle	Range(m)	Var(m^2)	Int.	Var.
VLP-16	0°	0.9227	0.00002	100.0	0.000
VLP-16	10°	0.9147	0.00002	99.08	4.153
VLP-16	20°	0.8979	0.00002	87.18	11.95
VLP-16	30°	0.9276	0.00002	75.70	9.010
VLP-16	40°	0.9323	0.00002	52.18	12.79
VLP-16	50°	0.8951	0.00003	40.16	5.414
VLP-16	60°	0.8607	0.00002	18.16	2.894
VLP-16	70°	0.8501	0.00004	7.580	1.124
UST-10LX	0°	0.9048	0.00002	1544.58	49.964
UST-10LX	10°	0.9250	0.00001	1464.48	92.930
UST-10LX	20°	0.9163	0.00003	1378.36	100.35
UST-10LX	30°	0.9212	0.00003	1273.26	125.47
UST-10LX	40°	0.9111	0.00003	1152.30	94.410
UST-10LX	50°	0.8909	0.00002	966.620	169.88
UST-10LX	60°	0.9052	0.00003	851.640	232.55
UST-10LX	70°	0.9337	0.00003	722.260	314.11
UST-20LX	0°	0.9164	0.00002	2497.48	84.849
UST-20LX	10°	0.9153	0.00002	2195.32	86.658
UST-20LX	20°	0.9184	0.00001	1967.96	120.24
UST-20LX	30°	0.9171	0.00003	1981.52	136.05
UST-20LX	40°	0.9111	0.00003	1762.58	194.76
UST-20LX	50°	0.9087	0.00002	1587.86	308.40
UST-20LX	60°	0.9099	0.00004	1367.70	417.77
UST-20LX	70°	0.9263	0.00002	1308.64	198.71
UTM-30LX	0°	0.9226	0.00001	4104.02	452.14
UTM-30LX	10°	0.9069	0.00002	3885.88	532.94
UTM-30LX	20°	0.9055	0.00002	3528.22	261.69
UTM-30LX	30°	0.9042	0.00002	3395.34	309.70
UTM-30LX	40°	0.8942	0.00001	3236.94	375.34
UTM-30LX	50°	0.8905	0.00002	3058.06	370.69
UTM-30LX	60°	0.8886	0.00002	3044.16	231.45
UTM-30LX	70°	0.8814	0.00001	2911.18	243.83

(b) Dark surface mapping performance

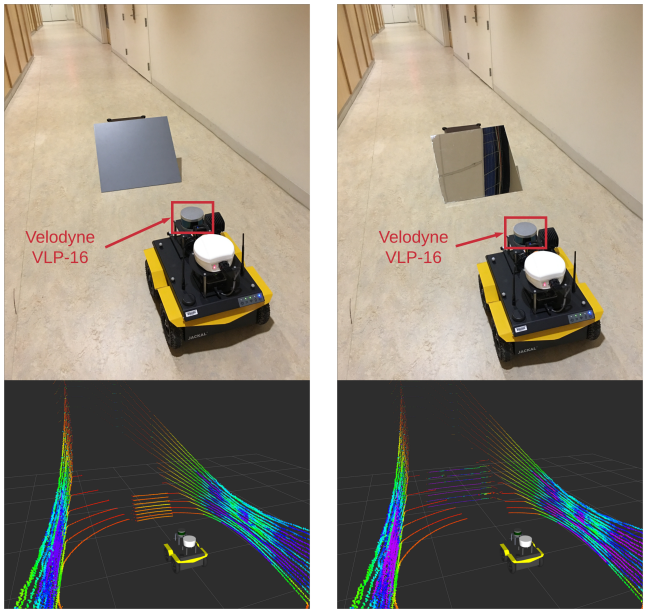
Sensor	Angle	Range(m)	Var(m^2)	Int.	Var.
VLP-16	0°	0.9218	0.00004	15.36	10.23
VLP-16	10°	0.9182	0.00011	6.600	4.400
VLP-16	20°	0.9149	0.00015	1.460	0.408
VLP-16	30°	0.9362	0.00158	1.040	0.038
VLP-16	40°	0.9642	0.00327	1.260	0.192
VLP-16	50°	0.9547	0.00643	1.940	0.056
VLP-16	60°	0.9748	0.00346	1.740	0.192
VLP-16	70°	1.3073	0.13076	1.660	0.236
UST-10LX	0°	0.9473	0.00004	581.960	156.918
UST-10LX	10°	0.9508	0.00005	502.800	58.240
UST-10LX	20°	0.9731	0.00002	485.860	30.061
UST-10LX	30°	0.9700	0.00004	443.660	13.662
UST-10LX	40°	Inf	n/a	0.0	n/a
UST-10LX	50°	Inf	n/a	0.0	n/a
UST-10LX	60°	Inf	n/a	0.0	n/a
UST-10LX	70°	Inf	n/a	0.0	n/a
UST-20LX	0°	0.9225	0.00004	1137.90	299.89
UST-20LX	10°	0.9223	0.00003	1042.20	41.320
UST-20LX	20°	0.9064	0.00003	868.060	78.416
UST-20LX	30°	0.9066	0.00004	699.660	45.224
UST-20LX	40°	0.9256	0.00007	565.280	62.602
UST-20LX	50°	0.9322	0.00009	510.720	77.922
UST-20LX	60°	0.9234	0.00020	463.440	98.326
UST-20LX	70°	0.9548	0.00023	456.640	157.63
UTM-30LX	0°	0.9062	0.00001	2720.42	226.83
UTM-30LX	10°	0.9244	0.00001	2535.64	238.91
UTM-30LX	20°	0.9265	0.00002	2017.38	298.15
UTM-30LX	30°	0.9360	0.00002	1368.22	272.61
UTM-30LX	40°	0.9343	0.00003	915.240	645.90
UTM-30LX	50°	0.9442	0.00002	985.760	492.50
UTM-30LX	60°	0.9484	0.00009	248.900	205.21
UTM-30LX	70°	0.9458	0.00009	195.000	196.04

Table 5.2: Surface impacts on measurement accuracy.

In Table 5.2, we report the average range and variances of our LiDAR sensors when tested against normal and dark surfaces. In our normal surface tests, we see that all LiDAR sensors maintain accurate range measurements with inconsequential increases in error and variance over all test angles. However, intensity measurements decrease significantly across all sensors achieving approximately 50% remission at 40° and 50° angles.

When tested on darker surfaces, these large remission drop-offs are able to create tremendous decreases in scanning capability. Of particular interest are the decrease in intensity of the Velodyne VLP-16, degrading to 15% of the original intensity when exposed to an optimal vertical 0° incident angle, and the decrease in intensity of the UST-10LX sensors, degrading to no remission at a 40° angle. The results of our experiments with the Velodyne Puck and the UST-20LX also demonstrate substantial measurement error and variance as incident angles increase. One potential explanation for these remission drop-offs could be due to non-uniform dispersion distributions when a LiDAR pulse impacts a surface. Larger incident angles will reflect fewer photons back towards the original emitter and more along the angle of reflection. An attacker's efforts could therefore be focused on leveraging LiDAR vulnerabilities to enhance remission degradation to a point that normally observable objects become completely cloaked.

Intuitively, increasing incident angles with reflective surfaces also creates measurement errors as scans map far away obstacles in the environment instead of the desired mirror surface. This effect is visually demonstrated by Fig. 5.6 in which the Velodyne VLP-16's 3D scans are compromised by a mirror that reflects the ceiling of the room, creating incorrect data. In Fig. 5.7 we graph the measurement data collected from the UST-20LX while exposed to a rotating mirror 1m away in an otherwise static environment. As the angles change, the range measurements change dramatically as scans sense the environment instead of the mirror. At 0°, however, we see that the measurement is accurate as the mirror is now reflecting pulses directly back towards the sensor.



(a) Normal pointcloud scanning (b) Mirror-distorted pointcloud scanning

Figure 5.6: A Velodyne 3D LiDAR compromised by a mirror

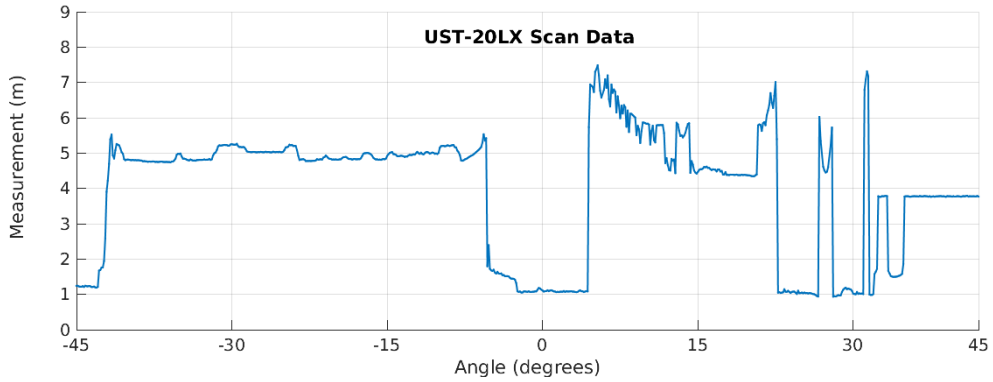
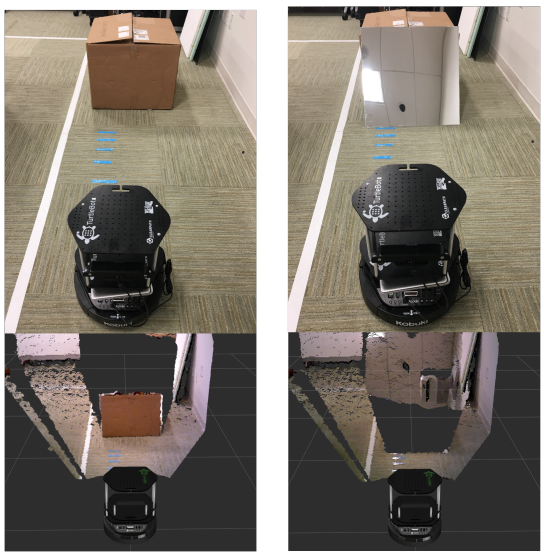


Figure 5.7: LiDAR distance measurements when corrupted by a rotating reflective surface in a static environment.

These mirror-based attack methods also have detrimental impacts on other IR based sensors. The Asus Xtion Pro Live RGB-D sensor [6] shown in Fig. 5.4(e) uses an IR based dot matrix to extract rough depth information from the environment. These RGB-D sensors are much more sensitive to extraneous light than LiDAR sensors but are able to extract continuous depth clouds instantly, making them suitable for gaming and some indoor robotic applications [6]. However, due to their dependence on IR-based registration, reflective surfaces are also capable of disrupting these sensors with similar results as for the LiDAR just presented.



(a) RGB-D scans of normal surfaces (b) RGB-D scans of mirrored surfaces

Figure 5.8: Asus Xtion scanning data corrupted by reflective surfaces.

Fig. 5.8 visually compares the normal and corrupted scans from the RGB-D sensor when exposed to a mirror. In our tests, the Asus Xtion Pro Live is mounted to a turtlebot 2 robot, a standard robotic research and education platform, which had no impact on the scanning capabilities of the sensor. We easily observe a missing hole in the data where the normally mapped box should be and the presence of the ceiling displayed further back. This effect is further demonstrated by a similar test in which a rotating mirror in an otherwise static environment is placed 1m from the sensor in Fig. 5.9. However, in this case, since the RGB-D sensor does not rely on time of flight, the sensor's range measurements are compromised over all incident angles.

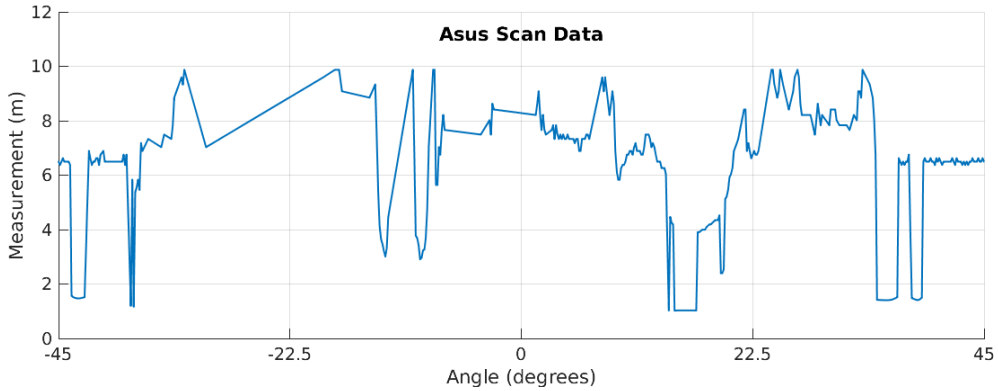


Figure 5.9: Asus Xtion range scans corrupted by reflective surfaces at different angles.

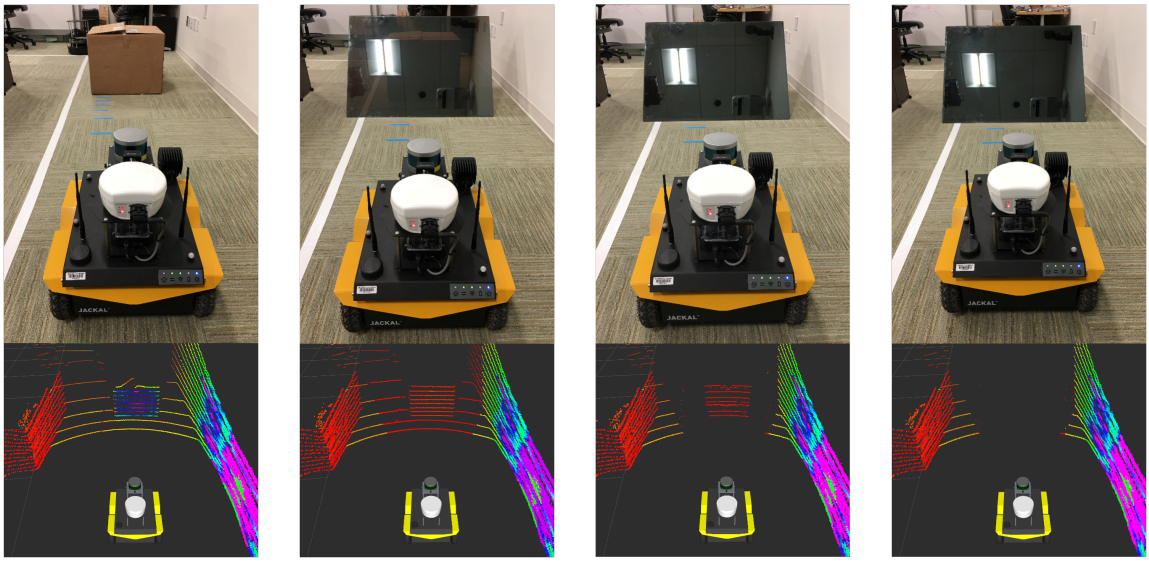
Semi-transparent Surfaces

In this experiment, we consider the remission-degrading impacts of semi-transparent surfaces. In our initial tests, we found that the impacts of these surfaces did not depend on the incident angle but instead on the depth of the semi-transparent surface. As such, our experiments are designed to investigate the degradation impacts of semi-transparent surfaces with respect to increasing depth. Each LiDAR sensor is placed approximately 1.5m from a cardboard surface. We use layers of one-way mirrors with a thickness of 5 mm in order to demonstrate the degradation properties of semi-transparent surfaces.

Sensor	Layers	Range(m)	Intensity
VLP-16	0	1.449	100.0
VLP-16	1	1.455	22.04
VLP-16	2	1.473	1.005
VLP-16	3	Inf	0.000
UST-10LX	0	1.468	1240.5
UST-10LX	1	1.453	488.24
UST-10LX	2	Inf	0.0000
UST-20LX	0	1.473	2374.9
UST-20LX	1	1.448	944.66
UST-20LX	2	1.446	464.38
UST-20LX	3	Inf	0.0000
UTM-30LX	0	1.574	4341.3
UTM-30LX	1	1.585	2089.1
UTM-30LX	2	1.593	195.44
UTM-30LX	3	Inf	0.0000

Table 5.3: Semi-transparent mapping performance.

Drawing from Table 5.3, we see that range measurements are minimally impacted with the largest change at approximately 3cm as presented by the UST-20LX. However, when considering remission degradation, we see much larger changes. Across all sensors, a single semi-transparent layer (Fig. 5.10(a)) is capable of causing a 50% remission degradation while three semi-transparent layers (Fig. 5.10(d)) are capable of causing 100%



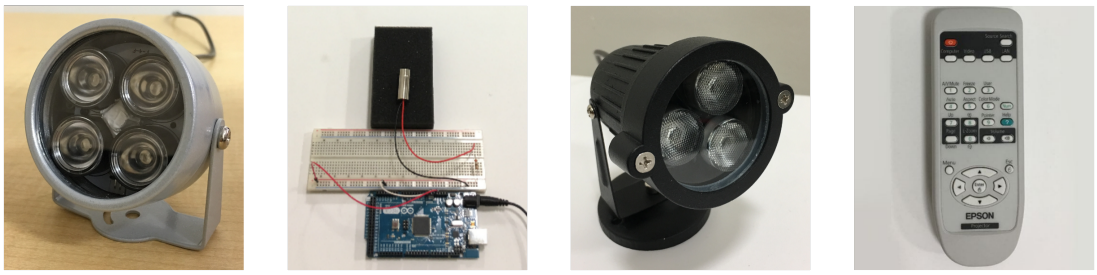
(a) 0 layer degradation (b) 1 layer degradation (c) 2 layer degradation (d) 3 layer degradation

Figure 5.10: Visual demonstration of layered semi-transparent window impacts.

remission degradation. In particular, the Velodyne VLP-16, the UST-10LX, and the UST-20LX experience large 80%, 40%, and 40% remission losses respectively when exposed to a single semi-transparent layer.

The effects of semi-transparent surfaces are also similar to reflective surfaces in that LiDAR scans are unable to detect these surfaces. However, semi-transparent surfaces differ from their reflective counterparts by absorbing portions of IR data as they pass through instead of reflecting them. As a result, these surfaces are capable of completely cloaking objects in the environment without leaving any trace of their presence. This effect is visually demonstrated in Fig. 5.10, in which the Velodyne VLP-16 scans are gradually corrupted more and more from normal scanning as shown in Fig. 5.10(a) to fully degraded scanning as shown in Fig. 5.10(d). Intensity data in these images is encoded using an inverted color scheme with red being lower intensities and violet being higher intensities. We can see that initially, the box is easily scanned but as we insert multiple panes of glass, scan returns become weaker and sparser before disappearing altogether.

An attacker may also leverage our previous findings to augment semi-transparent surfaces with dark surfaces and more efficiently degrade LiDAR remissions. The combination of these surfaces may be able to completely mask objects without alerting attack detectors. In particular, these attacks could have significant performance liabilities on localization and mapping algorithms that depend on LiDAR.



(a) The 850nm IR illuminator. (b) The 904nm laser. (c) The 940nm IR illuminator. (d) The Epson IR remote.

Figure 5.11: Test IR sources for measuring IR-induced noise.

5.1.4 Leveraging Foreign IR to Compromise LiDAR

Interference in LiDAR scans can also naturally occur when foreign IR signals with similar wavelengths are captured instead of the original pulse. This interference is capable of masking out real objects and inducing undesired noise into the scan data.

In addition, IR sources are relatively easy to procure and are purchasable as IR illuminators for late-night security cameras, hobby lasers, and even some IR based communication devices as shown in Fig. 5.11. Sample IR interference is shown in Fig. 5.12 in which an 850nm IR illuminator is used to introduce artificial noise. As foreign IR pulses activate the photodiode in the LiDAR sensor, the sensor registers erroneous range measurements.

Inherently unstable systems, such as quadrotors, utilizing LiDAR sensors for obstacle avoidance and positioning are particularly vulnerable. An attacker may easily purchase lasers with similar wavelengths to create highly erratic behavior that could lead to destabilization of these vehicles. In our experiments, we use two IR illuminators, a standard Epson remote controller used for TV and projector interfacing, and a hobbyist IR laser component used to enable larger IR-based projects.

IR Induced Interference

In these experiments, we investigate the noise-inducing impacts of foreign IR signals on LiDAR scan data. In particular, we construct our experiments in order to identify the impacts of foreign IR sources when directly pointed at a LiDAR sensor, and accordingly, with how a potential attacker might use a hand-held laser to disrupt nominal LiDAR-enabled operation.

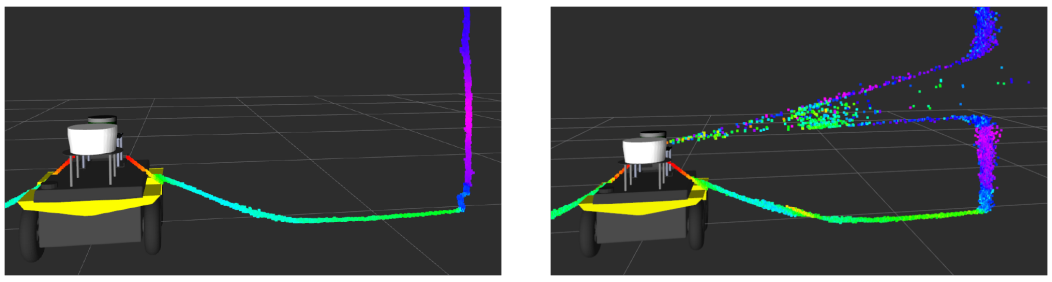
Each LiDAR sensor is also equipped with an internal wavelength-filtering circuit designed to reject light with wavelengths outside of the LiDAR emitter's own wavelength. As such, our selected IR sources emit pulses at wavelengths near the 904nm wavelength used by all of our LiDAR sensors. Each of our IR sources

are shown in Fig. 5.11 and include wavelengths of 850nm in Fig. 5.11(a), 904nm in Fig. 5.11(b), and 950nm in Fig. 5.11(c). In addition, we demonstrate that using a commonly found IR projector remote, shown in Fig. 5.11(d) [27], with an unknown IR wavelength is still capable of achieving some levels of noise, highlighting the ease of performing an IR based attack.

In contrast to our previous experiments, these attacks are oriented towards inducing extensive range variance rather than degrading LiDAR remissions. Intuitively, these attacks become more successful as the attacking interference's wavelength becomes closer to the LiDAR emitter's wavelength. As shown in Table 5.4, the range and intensity variances generally grew in order of similarity to the target 905nm wavelength, with the 904nm laser generating the greatest noise and the 850nm IR illuminator generating the least. Interestingly though, the UST-20LX is significantly more susceptible to the 850nm laser than both the 904nm and 940nm lasers, yielding 0.081m^2 variance compared to 0.012m^2 for the 904nm, 0.002m^2 for the 950nm, and 0.00004m^2 without interference.

The IR-induced noise also creates large measurement errors, yielding distance measurements ranging from millimeters to multiple meters. In terms of performing a truly successful attack, our experiments reveal that the 904nm laser, depicted in 5.11(b), is capable of inducing the most noise but is also the most difficult to continuously induce noise due to the narrow focus of the IR pulses. As such, an ideal attack would be a combination of both a wide field of effect and a matching wavelength.

Continuing our discussion on applications of IR-based interference attacks, an attacker could target obstacle avoidance protocols that are enabled by LiDAR sensors. By generating high variance range measurements, a system could overreact to false stimuli, creating erratic and potentially dangerous movements. In particular, inherently unstable systems that require finely-tuned software controllers to maintain safe operation, such as quadrotor aerial vehicles, are of noteworthy concern as small disturbances are capable of substantial impacts as we will show in the next section.



(a) Normal scan (b) IR-interfered scan

Figure 5.12: Impacts of foreign IR pulses on a 0.5 second period of accumulated LiDAR scans. (a) is unaffected, (b) is affected by an IR source.

Sensor	Wavelength	Norm. Ran(m)	Var(m^2)	Att. Ran(m)	Var(m^2)	Int.	Var
VLP-16	850nm	1.678	0.00002	0.932	0.00041	61.540	1741.45
VLP-16	904nm	1.679	0.00002	1.608	0.02503	97.700	174.090
VLP-16	940nm	1.416	0.00003	1.390	0.00166	63.780	135.132
VLP-16	Epson	0.913	0.00006	0.984	0.00308	21.440	632.726
UST-10LX	850nm	1.403	0.00002	1.441	0.01994	598.70	47953.3
UST-10LX	904nm	1.341	0.00004	1.197	0.06721	1618.7	37111.8
UST-10LX	940nm	1.312	0.00006	1.253	0.06122	689.10	66850.4
UST-10LX	Epson	0.913	0.00001	0.916	0.00028	1506.9	5104.71
UST-20LX	850nm	0.875	0.00003	0.953	0.08055	1913.1	76370.3
UST-20LX	904nm	0.727	0.00002	0.733	0.01227	3993.1	131363
UST-20LX	940nm	1.166	0.00005	1.152	0.00184	2701.0	6661.80
UST-20LX	Epson	0.881	0.00004	0.741	0.00224	1406.7	41364.3
UTM-30LX	850nm	1.267	0.00002	1.299	0.00018	4026.0	727253
UTM-30LX	904nm	1.551	0.00006	1.646	0.00054	3774.9	10465.4
UTM-30LX	940nm	1.127	0.00001	1.166	0.00021	4589.5	53166.6
UTM-30LX	Epson	1.402	0.00002	1.259	0.00034	3636.3	36071.9

Table 5.4: IR-based interference at differing wavelengths.

5.2 LiDAR Attack Impacts on Autonomous Operations

In this section, we demonstrate our analysis of surface-based and IR-based LiDAR vulnerabilities by synthesizing attacks on two autonomous robots enabled by LiDAR. We describe and execute two proof-of-concept attacks performed on commercially purchased robot platforms running the open source Robot Operating System (ROS) [68, 64]. While our exact implementations may differ from those found on commercial platforms, we use the exact LiDAR sensors found on many commercial applications and the localization and obstacle avoidance algorithms used are well studied in the literature and act as robust solutions to their respective problem spaces. We use such platforms in addition because most of the commercial platforms utilizing proprietary localization and obstacle avoidance software, such as a self-driving car, are not currently available for purchase. ROS software, however, is used to enable autonomous operation in several companies, such as Fetch Robotics and Savioke [31, 75], which provide autonomous warehouse and company-internal delivery services respectively, and is well maintained with new ROS distributions released alongside the newest Ubuntu Linux distributions.

5.2.1 Preliminaries: LiDAR-enabled Operations

In this section, we briefly discuss state estimation, localization, mapping, and obstacle avoidance methods as enabled by LiDAR before discussing our investigations into using our previously explored results to compromise two autonomous systems performing localization and obstacle avoidance.

State estimation, localization, and *mapping* are techniques designed to work in tandem in order to produce the best estimate of where a system is and where other objects are in relation to the system. State estimation fuses passively obtained data such as orientation measurements collected by an Inertial Measurement Unit (IMU) and velocity collected by wheel encoder measurements while localization and mapping fuses actively collected data such as LiDAR scans and stereo vision to create a representation of the environment. State estimation alone is fast and environment agnostic but drifts and accumulates an unacceptable amount of error. Mapping-based localization requires knowledge of the environment and is slower than state estimation but is drift free. Localization may also be performed through GPS but will fail when indoors and will yield lower accuracy estimates with approximately 1m to 2m of error [54, 84]. Combined together, state estimation allows a system to perform locally dependent behaviors such as path planning and obstacle detection while localization allows a system to correct and maintain absolute estimates of its current location.

The *particle filter*, a LiDAR-based triangulation algorithm, is a commonly used localization algorithm that is able to match new LiDAR scans with a previously generated LiDAR map of the environment to provide high accuracy estimation. Previous work by the authors of [84] has shown that 10cm localization accuracy is possible and may be used to enable the maneuvering of a self-driving car in urban settings. The technique utilizes a Bayesian framework by evolving hypotheses on the true state of the system. Over several iterations, the algorithm prunes out poor-fitting hypotheses and converges on ground truth [54, 55].

Obstacle detection and avoidance is another high level function that is enabled by LiDAR sensing. Given adequate localization and state estimation, a vehicle is now able to respond to dynamic obstacles in the environment in order to safely avoid collisions while reaching some desired goal. Other sensors, such as radar and ultrasonic sensors, are also able to enable this behavior but suffer due to reduced accuracy. With LiDAR, a system is able to safely identify open spaces for navigation even in cluttered environments. Generally, obstacle and collision avoidance algorithms prevent collisions by detecting objects and generating repulsive forces or new collision-free trajectories to maintain a system away from obstacles and other vehicles.

5.2.2 LiDAR Attacks on Autonomous Systems

In this section, we outline our proof-of-concept attacks on LiDAR-enabled autonomous systems. We continue our experiments by utilizing our findings from Section 5.1 to attack a 3D LiDAR equipped robotic unmanned ground vehicle (UGV) attempting to perform localization with the particle filter algorithm and to attack a 2D LiDAR equipped robotic unmanned aerial vehicle (UAV) attempting to perform stable hovering and obstacle avoidance. Our UGV and UAV test platforms are shown in Fig. 5.13(a) and Fig. 5.13(b), respectively.

(a) Clearpath Jackal equipped with Velodyne VLP-16.

(b) AscTec Pelican equipped with Hokuyo UST-20LX.

Figure 5.13: LiDAR-enabled autonomous systems used during the experiments.

Attacks on UGV Localization and Mapping

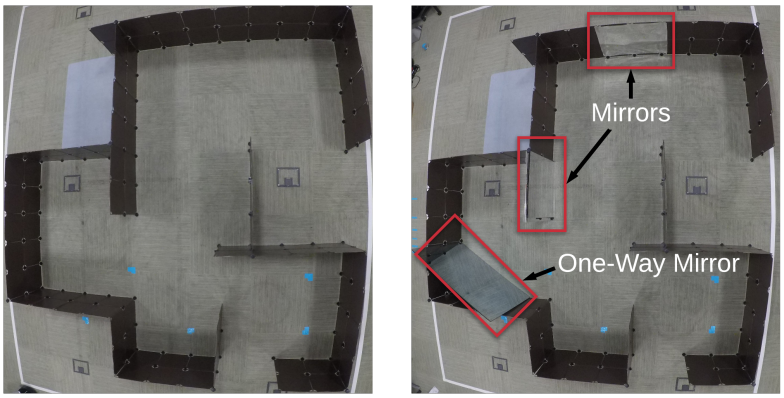
On autonomous systems, localization tasks rely on historical environment data to provide a ground truth reference. This data, known as a map, is generated beforehand and is used online to match newly acquired LiDAR scans. Building these maps is primarily done using LiDAR sensors and can be a non-trivial process if there are conditions that affect the accurate mapping of the environment. The accurate creation of these maps also does not guarantee good localization performance if the newly acquired scans are corrupted and cannot be properly matched to the map.

In this experiment, we propose two separate forms of passive attacks using mirrors, dark surfaces, and semi-transparent mirrors: 1) attack on a UGV's real-time scan data when localizing against a previously generated uncompromised map, and 2) attack on the map generation process for localization purpose in future operations.

Our test UGV is a Clearpath Jackal (Fig. 5.13(a)) that leverages the ROS framework with an open source implementation of the particle filter algorithm [36]. This system is equipped with IMUs, wheel encoders, GPS, and a Velodyne VLP-16 LiDAR sensor for localization. Computation for LiDAR localization, state estimation, and control is performed locally on the Jackal's onboard computer while a base station is used for visualization. Validation of localization is performed using a VICON motion capture system, capable of sub-millimeter position measurements, to evaluate the error in tracking performance. The test environment is constructed with distinct edges and corners in order to optimize localization performance. We also argue that due to similar dynamics and sensor payloads, our UGV robot serves as a valid model for a self-driving car operating in a tight environment as similar hardware and software are present on most commercial self-driving cars as presented in [56].

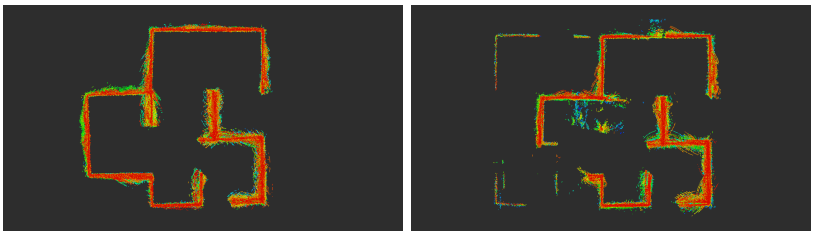
Attack Results

In these two attacks, we test the impacts of passively placing corrupting surface *traps* like mirrors to mismatch scan data and impair localization performance through falsely generated maps and through corrupted scans. The results of these attacks are measured through a tracking error on position. We first demonstrate normal localization performance when using the correctly generated map, as shown in Fig. 5.14(c), and without corrupting surface interference. As shown in Fig. 5.15(a), the filter is able to achieve 10cm pose accuracy with high confidence in approximately 30s. In contrast, localization with a corrupted map, shown in Fig. 5.14(d), over a 260s duration in our second experiment never converges.



(a) Clean environment.

(b) Corrupted environment.



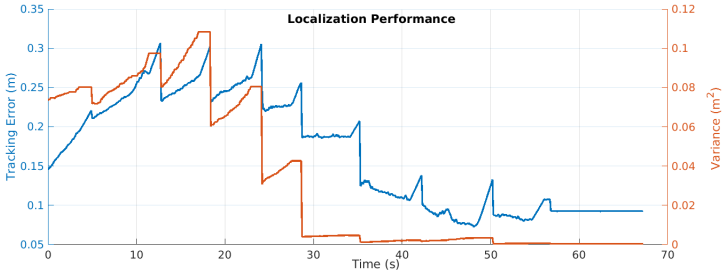
(c) Clean mapping.

(d) Corrupt mapping.

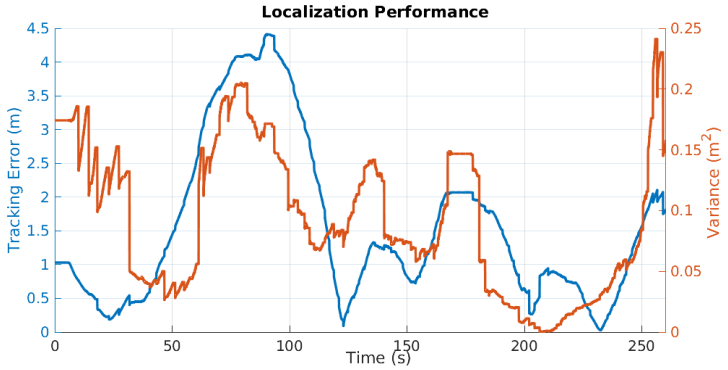
Figure 5.14: Comparison between clean and corrupted maps. The corruption on map generation is due to various detrimental surface features.

It is observed in Fig. 5.15(b) that the particle filter demonstrates a number of high confidence and high error estimates indicating the localization algorithm was confused by the aberrant map features found in the data. At other times, the algorithm yielded high tracking errors as much as 3.7m. When instead a normally generated map is given but the environment is filled with LiDAR-interfering surfaces, the algorithm is still unable to converge and again produces high confidence, high tracking error estimates. As seen in Fig. 5.15(c), the particle filter generates 7 high confidence estimates with below 0.05m^2 variance and over 1m error over a

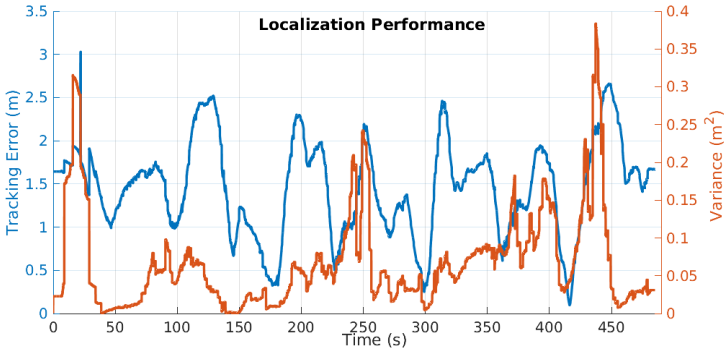
450s duration. On a self-driving car, these types of error would then be similar to that produced by a GPS and would be enough to place a self-driving car on a nearby sidewalk or the opposing lane.



(a) Normal Localization Performance.



(b) Corrupted Map Localization Performance.



(c) Corrupted Scan Localization Performance.

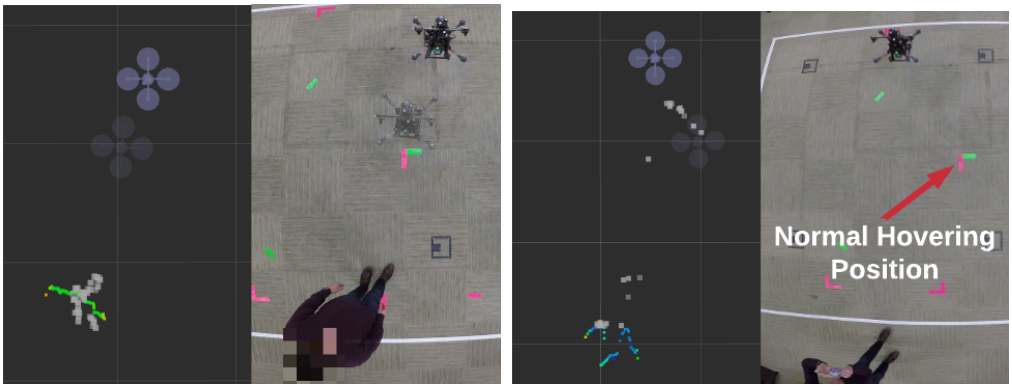
Figure 5.15: Tracking error and variance in hypotheses of particle filter-based localization over time.

Attack on UAV Obstacle Avoidance

In this experiment, we propose the use of foreign IR pulses, as presented in Section 5.1.4 to induce large error into an autonomous quadrotor UAV attempting to perform online obstacle avoidance. The quadrotor UAV is tasked with hovering and reacting to obstacles. This may easily be thought of as an aerial reconnaissance mission in which the quadrotor must autonomously survey a given area while avoiding collisions. In this case, the UAV is tasked with maintaining its position at 1.2m above the origin while avoiding obstacles. Our aerial test platform is an Ascending Technologies Pelican that also leverages the ROS framework to interface with a 2D Hokuyo UST-20LX and perform autonomous obstacle avoidance. We use the VICON motion capture system again to obtain high accuracy ground truth data of the UAV's position.

Attack Results

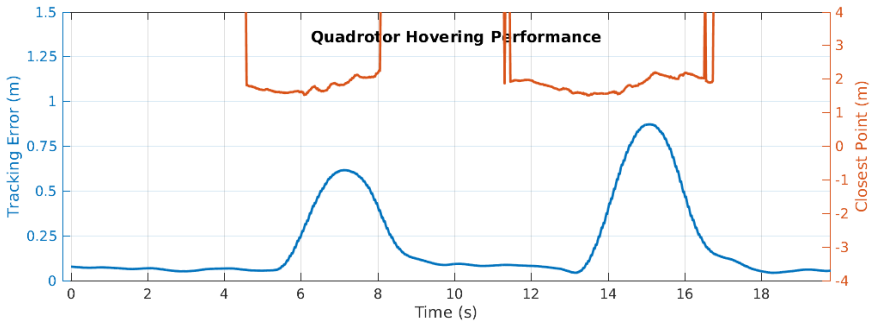
Our obstacle avoidance approach is first validated by assessing the normal collision avoidance and tracking ability of the quadrotor. We observe that our quadrotor is able to achieve approximately 5cm tracking error as seen in Fig. 5.17(a) and is able to maintain a 1m distance from an incoming obstacle, demonstrating that normal performance is able to simultaneously avoid obstacles and track a target position. This is visually demonstrated as well in Fig. 5.16(a), in which a person approaches and the quadrotor responds to maintain a safe distance.



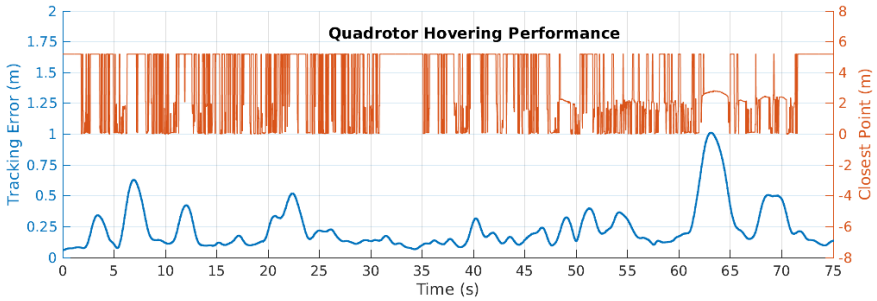
(a) Normal obstacle avoidance (b) Attacked obstacle avoidance

Figure 5.16: Normal and IR-compromised obstacle avoidance behavior.

When subject to IR interference though, our quadrotor demonstrates differing performance. As depicted in Fig. 5.17(b), the introduction of IR interference yields false range measurements as low as 0.01m. This noise is introduced by hand using the 850nm IR illuminator due to its high variance on the UST-20LX equipped on the quadrotor. This is also shown in Fig. 5.16(b), in which IR interference creates false obstacle beliefs indicated by the white dots. Over the full duration of the attack, our attacker is able to induce up to



(a) Normal obstacle avoidance plot.



(b) IR-interfered obstacle avoidance plot.

Figure 5.17: Tracking error and closest detected obstacle data during obstacle avoidance experiments.

1m of tracking error and create largely unstable behavior as the quadrotor struggles to maintain the desired hovering position. This effect is immediately eliminated once the interfering IR is removed.

5.3 Discussion

In this chapter, we presented our findings on passively attacking LiDAR sensors with semi-transparent, dark, and reflective surfaces, and actively attacking LiDAR sensors with IR pulses. In both cases, we were able to successfully induce error profiles higher than the claims given by the associated datasheets. The successful manipulation of LiDAR data elicits great concern for autonomous systems operating in adversarial and safety-critical environments.

Beyond malicious attackers, we must also consider the motivated average citizen who is simply interested in testing the limits of self-driving and autonomous technology as the ease of these attack materials enables any individual the ability to compromise these systems. Without careful insight into methodologies for securing sensory vulnerabilities, citizens seeking an advantage in a commute to work, a way to achieve more privacy, or even simple malicious enjoyment may create dangerous scenarios that may compromise autonomous systems that depend on LiDARs. Human drivers have already explored the limits of safety-constraints on some semi-autonomous cars. For example, a Tesla driver was able to convince his vehicle that his hands were

on the steering wheel just by using an orange [82]. This allowed the driver to use the vehicle's self-driving Autopilot features without paying active attention to the system's performance. Similarly, using the results found in this chapter, a pedestrian attempting to cross the road prematurely might use an IR source to forcibly open a clear path.

5.3.1 Attack Scenarios

In this section, we envision some other practical attacks that could be enabled by some specific high-risk scenarios and limitations in existing technology. Simply identifying the presence of a mirror or semi-transparent window, for example, may be difficult for autonomous systems using existing sensing approaches. Reflective surfaces may be difficult to detect with neural network based vision systems due to the difficulty in training to perceive a mirror, and radar systems may only be able to recognize the presence of an obstacle. Without being able to classify and observe the pose of these surfaces, LiDAR corruption would remain undetected. In difficult environments and situations, autonomous vehicles and robots could also be easily leveraged to facilitate an attack. Some sample attack scenarios are described in the following.

High Speed Autonomous Vehicle Attack

As autonomous aerial robots become more widespread and existing drone technologies become easier for consumers to utilize, a potential attack arises in the form of an autonomous malicious drone. A quadrotor equipped with an angled mirror or an IR illuminator would be able to convince an autonomous vehicle that another vehicle did not exist or that there was a large amount of random obstacles that suddenly appeared. Given that the vehicle could be travelling at high speeds, sometimes above 60 mph on an open highway, this could result in an autonomous vehicle moving into another car, potentially leading to injury, major damages, and widespread havoc among vehicles in the local vicinity.

Hazardous Terrain Attack

As autonomous robots begin to adapt to more rigorous environments, path-planning approaches have evolved to incorporate considerations into environmental hazards. For example, the authors of [45] presented a path-planning approach that considered a traversability index in order to pursue safer paths. Given surface detection is being conducted with the LiDAR sensor, the use of reflective or dark and semi-transparent surfaces in this situation could be used to convince an agent that the incoming surface had no supporting terrain, forcing an autonomous robot to take a desired route designated by a malicious attacker.

IR DoS Attacks

As autonomous robots begin to become widespread and easily accessible, IR-induced LiDAR attacks could be used to deny an autonomous system pertinent environmental information by preventing the accurate classification of road markings, traffic signs, and even pedestrians. Alternatively, these attacks could be leveraged to convince an autonomous system that particular paths are inaccessible, allowing an attacker to redirect a system at will.

5.4 Future Work

LiDAR sensors provide an extraordinary variety of functions to autonomous systems that include high precision localization to mapping and obstacle avoidance. The large density of high accuracy depth information with low computational penalty also provides attractive benefits over other similar methodologies such as stereo vision and radar. However, the over-reliance on LiDAR data can lead to dangerous security vulnerabilities that may destabilize a system. We found that using dark, reflective, and semi-transparent smooth surfaces could create decreased remissions in order to completely cloak objects in the environment while hand-held IR sources could be used to generate large range measurement variances with dangerous attack applications on obstacle avoidance protocols. These attacks were able to drive the error profiles of our test sensors well beyond the prescribed limits of operation as specified in the associated data-sheets.

To demonstrate the effects of these vulnerabilities, we implemented an attack with specialized surfaces on a LiDAR mounted UGV and an attack with IR-induced noise on a LiDAR mounted UAV. The results of our experiments showed that we were able to successfully induce large localization error when using corrupted scans or a poisoned map that directly resulted from our described passive attacks, and induce erratic over-correcting UAV behaviors with IR interference.

In future works, we will continue exploring solutions to these vulnerabilities by exploring the fusion of heterogeneous sources of data to identify specialized surfaces and aberrant sensor noise. While the singular use of LiDAR may be unable to identify these attacks, the differing observations by other sensors including cameras, radar, and sonar may be used in order to improve the safety and security of autonomous systems.

Chapter 6

Conclusions

In this thesis, we detailed our works on robust operation under uncertainty in multi-robot coordination, autonomous tunnel inspection, and sensor cyber-security. Specifically, our work involved safe motion planning for teams of UAVs and UGVs to complete tasks while observing energy constraints, developing an autonomous controller and mapping approach to allow a LiDAR-equipped UGV to assess structural quality, and an evaluation of LiDAR vulnerabilities and potential corresponding attacks. In each of these works, we also described how we intended to further each of these research lines. However, developing robust operation under uncertainty requires a great deal of evaluation and warrants a final discussion in this Thesis.

Drawing Appropriate Assumptions

In most applications, making assumptions about the problem space is a difficult necessity. Making stringent assumptions about a task, system, or environment could lead to improper modeling and poor performance while making no assumptions whatsoever may make the problem infeasible. In the classic obstacle avoidance problem for example, guaranteeing a robot will not collide with a group of pedestrians may either require the robot to take an extremely roundabout path due to the uncertainty of human behavior or require the robot to take no action at all. In essence, making no assumptions whatsoever yields an overly conservative result.

Making an assumption that humans have certain velocity boundaries though makes the problem more feasible by eliminating some of the possible solution space. However, making a drastic assumption that pedestrians walk in a straight line fails to capture the inherent stochasticity of human behavior and is insufficient to guarantee the robot will be able to safely travel near the pedestrians. Our future research will be directed towards developing algorithms that relax previously made assumptions in order to create more

general solutions and investigating previously made assumptions and whether those assumptions are sufficient for their given scenarios.

Machine Learning and Provable Guarantees

In terms of being able to relax assumptions, machine learning methods offer a great deal of potential. Since many assumptions are made for static situations, learning methods allow a system to respond to changes in the environment or the system and relax these assumptions. For the course of this discussion, when referencing machine learning we primarily refer to supervised learning methods.

At its core, machine learning offers highly accurate function approximation based on training and labeled data. Given a large enough sample size, a learning algorithm is able to predict an appropriate output when given new data. This methodology has already proven beneficial for creating optimal reinforcement learning controllers and highly accurate perception classifiers that have been used to fly quadrotors and enable self-driving cars.

Leveraging these learning-based approaches though still offers its own uncertainty. These learning methods only offer guarantees of learning convergence and in extremely nonlinear cases, these function approximations may make improper inferences. Contrary to traditional control theoretic methods in which deterministic controllers are able to offer formal guarantees of stability, machine learning methods offer potential optimality by better learning a system's true model. Combining control-theoretic methods with learning therefore allows a system to rigidly define its capabilities while fluidly responding to changes in the environment.

Some existing research on this topic, while still somewhat young, demonstrates promising results. Authors of [10] demonstrated a robust quadrotor controller that was able to offer continuous stability as a Gaussian Process based learning method optimized controller parameters online while the authors of [72] used a stable controller to supervise a reinforcement learning agent's actions as it learned to control a robotic arm.

In our future work, we will explore how to similarly integrate machine learning and control methods to offer the previously described optimality with guarantees and produce robust performance in multi-agent settings, in sensor cyber-security, and in adverse and dynamic environments.

Bibliography

[1] R. Abrams and A. Kurtz. *Joshua Brown, Who Died in Self-Driving Accident, Tested Limits of His Tesla*. https://www.nytimes.com/2016/07/02/business/joshua-brown-technology-enthusiast-tested-the-limits-of-his-tesla.html. 2016.

[2] Mohammad Aldibaja, Naoki Suganuma, and Keisuke Yoneda. "Robust Intensity-Based Localization Method for Autonomous Driving on Snow–Wet Road Surface". In: *IEEE Transactions on Industrial Informatics* 13.5 (2017), pp. 2369–2378.

[3] R. Amadeo. *Google's Street View cars are now giant, mobile 3D scanners*. https://arstechnica.com/gadgets/2017/09/googles-street-view-cars-are-now-giant-mobile-3d-scanners/. 2017.

[4] Ascending Technologies. *AscTec Pelican*. http://www.asctec.de/en/uav-uas-drones-rpas-roav/asctec-pelican/#pane-0-1. 2017.

[5] ASI. *ASI — Mining*. https://www.asirobots.com/mining/. 2018.

[6] Asus. *Asus Xtion Pro*. https://www.asus.com/3D-Sensor/Xtion_PRO/. 2017.

[7] R. Baldwin. *Audi takes its self-driving car where others dare not go*. https://www.engadget.com/2017/07/11/audi-takes-its-self-driving-car-where-others-dare-not-go/. 2017.

[8] R. Baldwin. *You can't buy a self-driving BMW until 2021 (and that's a good thing)*. https://www.engadget.com/2017/10/25/bmw-self-driving-7-series-2021/. 2017.

[9] Janusz Marian Bedkowski and Timo Röhling. "Online 3D LIDAR Monte Carlo localization with GPU acceleration". In: *Industrial Robot: An International Journal* 44.4 (2017), pp. 442–456.

[10] Felix Berkenkamp and Angela P Schoellig. "Safe and robust learning control with Gaussian processes". In: *Control Conference (ECC), 2015 European*. IEEE. 2015, pp. 2496–2501.

[11] Brett Bethke, Jonathan P How, and John Vian. "Group health management of UAV teams with applications to persistent surveillance". In: *American Control Conference, 2008*. IEEE. 2008, pp. 3145–3150.

[12] Nicola Bezzo et al. "A cooperative heterogeneous mobile wireless mechatronic system". In: *IEEE/ASME Transactions on Mechatronics* 19.1 (2014), pp. 20–31.

[13] Nicola Bezzo et al. "A decentralized connectivity strategy for mobile router swarms". In: *IFAC Proceedings Volumes* 44.1 (2011), pp. 4501–4506.

[14] Nicola Bezzo et al. "Online planning for energy-efficient and disturbance-aware UAV operations". In: *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE. 2016, pp. 5027–5033.

[15] Black-I. *Black-I Robotics*. http://www.blackirobotics.com/Home_Page.html. 2012.

[16] Boston Dynamics. *Boston Dynamics Robots*. https://www.bostondynamics.com/. 2018.

[17] Sean L Bowman et al. "Probabilistic data association for semantic slam". In: *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE. 2017, pp. 1722–1729.

[18] F. Brandon. *Lyft receives permission to test self-driving cars in California*. https://www.theverge.com/2017/11/22/16690148/lyft-self-driving-cars-california-dmv. 2017.

[19] C. Brewer. *Building Ford's Next Generation Autonomous Development Vehicle*. https://medium.com/self-driven/building-fords-next-generation-autonomous-development-vehicle-82a6160a7965. 2016.

[20] Business Wire. *Velodyne LiDAR Awarded Perception System Contract from Mercedes-Benz*. https://www.businesswire.com/news/home/20170912005760/en/Velodyne-LiDAR-Awarded-Perception-System-Contract-Mercedes-Benz. 2017.

[21] Civil Maps. *Sensor Fusion*. https://civilmaps.com/project/sensor-fusion/. 2017.

[22] Clearpath Robotics. *Clearpath Robotics Homepage*. https://www.clearpathrobotics.com/. 2016.

[23] Matthew Cornick et al. "Localizing Ground Penetrating Radar: a step toward robust autonomous ground vehicle localization". In: *Journal of Field Robotics* 33.1 (2016), pp. 82–102.

[24] Nelson County. *Claudius Crozet Blue Ridge Tunnel*. http://blueridgetunnel.org/. 2018.

[25] DARPA. *DARPA Subterranean Challenge Aims to Revolutionize Underground Capabilities*. https://www.darpa.mil/news-events/2017-12-21. 2017.

[26] Frank Dellaert. *Factor graphs and GTSAM: A hands-on introduction*. Tech. rep. Georgia Institute of Technology, 2012.

[27] Epson. *Epson 1547200 Replacement Remote Control*. https://epson.com/Accessories/Projector-Accessories/Replacement-Projector-Remote-Control-1547200/p/1547200. 2017.

[28] D. Etherington. *Ford outlines plan to build self-driving cars at scale to deploy with partners*. https://techcrunch.com/2017/08/22/ford-outlines-plan-to-build-self-driving-cars-at-scale-to-deploy-with-partners/. 2017.

[29] Exyn. *Exyn Technologies*. http://www.exyn.com/. 2018.

[30] Georgios E Fainekos, Hadas Kress-Gazit, and George J Pappas. "Temporal logic motion planning for mobile robots". In: *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. IEEE. 2005, pp. 2020–2025.

[31] Fetch Robotics. *Fetch Robots*. http://fetchrobotics.com/. 2017.

[32] A Filgueira et al. "Quantifying the influence of rain in LiDAR performance". In: *Measurement* 95 (2017), pp. 143–148.

[33] Friedrich Fraundorfer et al. "Vision-based autonomous mapping and exploration using a quadrotor MAV". In: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE. 2012, pp. 4557–4564.

[34] Emmy Freedman. *UVA Professor, Students Use Robot to Help Restore Claudius Crozet Blue Ridge Tunnel*. http://www.nbc29.com/story/37218808/uva-professor-students-use-robot-to-help-restore-claudius-crozet-blue-ridge-tunnel. 2018.

[35] Frost and Sullivan. *LiDAR: Driving the Future of Autonomous Navigation*. http://velodynelidar.com/docs/papers/FROST-ON-LiDAR.pdf. 2015.

[36] Johannes Garimort, Armin Hornung, and Maren Bennewitz. "Humanoid navigation with dynamic footstep plans". In: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE. 2011, pp. 3982–3987.

[37] GrindGIS. *LIDAR Data: 50 Applications and Uses*. http://grindgis.com/data/lidar-data-50-applications. 2015.

[38] Giorgio Grisetti et al. "A tutorial on graph-based SLAM". In: *IEEE Intelligent Transportation Systems Magazine* 2.4 (2010), pp. 31–43.

[39] M. Harris. *Google reports self-driving car mistakes: 272 failures and 13 near misses*. https://www.theguardian.com/technology/2016/jan/12/google-self-driving-cars-mistakes-data-reports. 2015.

[40] Sebastian Hening et al. "3D LiDAR SLAM Integration with GPS/INS for UAVs in Urban GPS-Degraded Environments". In: *AIAA Information Systems-AIAA Infotech@ Aerospace*. 2017, p. 0448.

[41] Wolfgang Hess et al. "Real-time loop closure in 2D LIDAR SLAM". In: *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE. 2016, pp. 1271–1278.

[42] Santosh A Hiremath et al. "Laser range finder model for autonomous navigation of a robot in a maize field using a particle filter". In: *Computers and Electronics in Agriculture* 100 (2014), pp. 41–50.

[43] Hokuyo. *Hokuyo UST-10/20LX*. https://www.hokuyo-aut.jp/search/single.php?serial=167. 2017.

[44] Hokuyo. *Hokuyo UTM-30LX*. https://www.hokuyo-aut.jp/search/single.php?serial=169. 2017.

[45] Ayanna Howard, Homayoun Seraji, and Barry Werger. "T*: a novel terrain-based path planning method for mobile robots". In: (2002).

[46] Joseph Huang et al. "Efficient, generalized indoor wifi graphslam". In: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE. 2011, pp. 1038–1043.

[47] Inmo Jang et al. "Cooperative Control for a Flight Array of UAVs and an Application in Radar Jamming". In: *IFAC-PapersOnLine* 50.1 (2017), pp. 8011–8018.

[48] Matt Kelly. *Into the Darkness: UVA Robot Maps Historic Tunnel*. https://news.virginia.edu/content/darkness-uva-robot-maps-historic-tunnel. 2018.

[49] Christian Kerl, Jurgen Sturm, and Daniel Cremers. "Dense visual SLAM for RGB-D cameras". In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE. 2013, pp. 2100–2106.

[50] Andrew J Kerns et al. "Unmanned aircraft capture and control via GPS spoofing". In: *Journal of Field Robotics* 31.4 (2014), pp. 617–636.

[51] Ayoung Kim and Ryan Eustice. "Pose-graph visual SLAM with geometric model selection for autonomous underwater ship hull inspection". In: *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE. 2009, pp. 1559–1565.

[52] Alexander Kleiner, Johann Prediger, and Bernhard Nebel. "RFID technology-based exploration and SLAM for search and rescue". In: *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*. IEEE. 2006, pp. 4054–4059.

[53] Krylon. *Camoflage Paint made with Fusion for Plastic Technology*. http://www.krylon.com/products/camouflage-paint-made-with-fusion-for-plastic-technology/. 2017.

[54] Jesse Levinson, Michael Montemerlo, and Sebastian Thrun. "Map-Based Precision Vehicle Localization in Urban Environments." In: *Robotics: Science and Systems*. Vol. 4. Citeseer. 2007, p. 1.

[55] Jesse Levinson and Sebastian Thrun. "Robust vehicle localization in urban environments using probabilistic maps". In: *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE. 2010, pp. 4372–4378.

[56] Jesse Levinson et al. "Towards fully autonomous driving: Systems and algorithms". In: *Intelligent Vehicles Symposium (IV), 2011 IEEE*. IEEE. 2011, pp. 163–168.

[57] LiDAR-UK. *How does LiDAR work?* http://www.lidar-uk.com/how-lidar-works/. 2017.

[58] Lidar-UK. *A brief history of Lidar*. http://www.lidar-uk.com/a-brief-history-of-lidar/. 2017.

[59] Neil Mathew, Stephen L Smith, and Steven L Waslander. "Planning paths for package delivery in heterogeneous multirobot teams". In: *IEEE Transactions on Automation Science and Engineering* 12.4 (2015), pp. 1298–1308.

[60] K. Naughton. *Self-driving cars succumb to snow blindness as driving lanes disappear*. http://www.autonews.com/article/20160210/OEM06/160219995/self-driving-cars-succumb-to-snow-blindness-as-driving-lanes-disappear. 2016.

[61] Erik Nelson. *Berkeley Localization and Mapping Source Code*. https://github.com/erik-nelson/blam. 2016.

[62] Anh Nguyen, Jason Yosinski, and Jeff Clune. "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 427–436.

[63] T. Ong. *Google has updated its Street View cameras for the first time in eight years*. https://www.theverge.com/2017/9/5/16254384/google-updated-street-view-cameras-ai-machine-learning. 2017.

[64] Open Source Robotics Foundation. *Robot Operating System*. http://www.ros.org. 2017.

[65] Open Source Robotics Foundation. *The Turtlebot2*. http://www.turtlebot.com/turtlebot2/. 2017.

[66] Francisco J Perez-Grau et al. "Multi-sensor three-dimensional Monte Carlo localization for long-term aerial robot navigation". In: *International Journal of Advanced Robotic Systems* 14.5 (2017), p. 1729881417732757.

[67] Jonathan Petit et al. "Remote attacks on automated vehicles sensors: Experiments on camera and lidar". In: *Black Hat Europe* 11 (2015), p. 2015.

[68] Morgan Quigley et al. "ROS: an open-source Robot Operating System". In: *ICRA workshop on open source software* 3.3.2 (2009), p. 5.

[69] Hamed Rezaee and Farzaneh Abdollahi. "A decentralized cooperative control scheme with obstacle avoidance for a team of mobile robots". In: *IEEE Transactions on Industrial Electronics* 61.1 (2014), pp. 347–354.

[70] Leigh Robinson and Benjamin Graham. "Confusing Deep Convolution Networks by Relabelling". In: *arXiv preprint arXiv:1510.06925* (2015).

[71] RedZone Robotics. *RedZone Robotics — Homepage*. http://www.redzone.com/. 2018.

[72] Michael T Rosenstein and Andrew G Barto. "Reinforcement learning with supervision by a stable controller". In: *American Control Conference, 2004. Proceedings of the 2004*. Vol. 5. IEEE. 2004, pp. 4517–4522.

[73] Indranil Saha et al. "Automated composition of motion primitives for multi-robot systems from safe LTL specifications". In: *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE. 2014, pp. 1525–1532.

[74] Indranil Saha et al. "Implan: scalable incremental motion planning for multi-robot systems". In: *Cyber-Physical Systems (ICCPS), 2016 ACM/IEEE 7th International Conference on*. IEEE. 2016, pp. 1–10.

[75] Savioke. *Savioke Robotics*. http://www.savioke.com/. 2017.

[76] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. "Generalized-icp." In: *Robotics: science and systems*. Vol. 2. 4. 2009, p. 435.

[77] Tina Setter and Magnus Egerstedt. "Energy-constrained coordination of multi-robot teams". In: *IEEE Transactions on Control Systems Technology* 25.4 (2017), pp. 1257–1263.

[78] Brian Shucker, Todd D Murphey, and John K Bennett. "Convergence-preserving switching for topology-dependent decentralized systems". In: *IEEE Transactions on Robotics* 24.6 (2008), pp. 1405–1415.

[79] Yunmok Son et al. “Rocking Drones with Intentional Sound Noise on Gyroscopic Sensors.” In: *USENIX Security Symposium*. 2015, pp. 881–896.

[80] Cyrill Stachniss, Dirk Hahnel, and Wolfram Burgard. “Exploration with active loop-closing for FastSLAM”. In: *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*. Vol. 2. IEEE. 2004, pp. 1505–1510.

[81] Jie Su et al. “A stealthy gps spoofing strategy for manipulating the trajectory of an unmanned aerial vehicle”. In: *IFAC-PapersOnLine* 49.22 (2016), pp. 291–296.

[82] J. Swinbanks. *You Can Apparently Fool Tesla’s Autopilot With An Orange*. https://www.gizmodo.com.au/2018/01/you-can-fool-teslas-autopilot-with-an-orange/. 2018.

[83] Tong Tao et al. “Motion planning for slam based on frontier exploration”. In: *Mechatronics and Automation, 2007. ICMA 2007. International Conference on*. IEEE. 2007, pp. 2120–2125.

[84] Sebastian Thrun. “Toward robotic cars”. In: *Communications of the ACM* 53.4 (2010), pp. 99–106.

[85] Sebastian Thrun and Michael Montemerlo. “The graph SLAM algorithm with applications to large-scale mapping of urban structures”. In: *The International Journal of Robotics Research* 25.5-6 (2006), pp. 403–429.

[86] Gábor Vásárhelyi et al. “Outdoor flocking and formation flight with autonomous aerial robots”. In: *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE. 2014, pp. 3866–3873.

[87] Velodyne. *VLP-16 Datasheet*. http://velodynelidar.com/docs/manuals/VLP-16%20User%20Manual%20and%20Programming%20Guide%2063-9243%20Rev%20A.pdf. 2017.

[88] Arnoud Visser and Bayu A Slamet. “Balancing the information gain against the movement cost for multi-robot frontier exploration”. In: *European Robotics Symposium 2008*. Springer. 2008, pp. 43–52.

[89] Volvo. *How our self-driving cars work*. http://www.volvocars.com/intl/about/our-innovation-brands/intellisafe/autonomous-driving/how-it-works. 2017.

[90] Waymo. *Building maps for a self-driving car*. https://medium.com/waymo/building-maps-for-a-self-driving-car-723b4d9cd3f4. 2016.

[91] Waymo. *Introducing Waymo’s suite of custom-built, self-driving hardware*. https://medium.com/waymo/introducing-waymos-suite-of-custom-built-self-driving-hardware-c47d1714563. 2017.

[92] C. Whitlock. “More Air Force drones are crashing than ever as mysterious new problems emerge”. In: (2016).

[93] C. Whitlock. *When drones fall from the sky*. http://www.washingtonpost.com/sf/investigative/2014/06/20/when-drones-fall-from-the-sky/?utm_term=.cdecba9186e8. 2014.

[94] wikipedia.org. *LiDAR*. https://en.wikipedia.org/wiki/Lidar. 2017.

[95] Willow Garage. *PR2 Overview*. http://www.willowgarage.com/pages/pr2/overview. 2017.

[96] J Wojtanowski et al. "Comparison of 905 nm and 1550 nm semiconductor laser rangefinders' performance deterioration due to adverse environmental conditions". In: *Opto-Electronics Review* 22.3 (2014), pp. 183–190.

[97] Chen Yan, Wenyuan Xu, and Jianhao Liu. "Can you trust autonomous vehicles: Contactless attacks against sensors of self-driving vehicle". In: *DEF CON* 24 (2016).

[98] Shao-Wen Yang and Chieh-Chih Wang. "On solving mirror reflection in lidar sensing". In: *IEEE/ASME Transactions on Mechatronics* 16.2 (2011), pp. 255–265.

[99] E Yeh et al. "Security in automotive radar and vehicular networks". In: *Microwave Journal* (2016).

[100] Z. Zhang. "Microsoft Kinect Sensor and Its Effects". In: *IEEE MultiMedia* 19.2 (2012), pp. 4–10.