2022

PRECISE LANDING OF VTOL UAVS USING A TETHER

Jeremy W. Rathjen
jwr0014@mix.wvu.edu

PRECISE LANDING OF VTOL UAVS USING A TETHER

by

Jeremy W Rathjen

Thesis submitted to the
Benjamin M. Statler College of Engineering and Mineral Resources
at West Virginia University

in partial fulfillment of the requirements for the degree of

Master of Science
in
Mechanical Engineering

Guilherme A. S. Pereira, Ph.D., Committee Chairperson
Jason Gross, Ph.D.
Yu Gu, Ph.D.

Department of Mechanical and Aerospace Engineering

Morgantown, West Virginia
2022

Keywords: UAV, landing, tether, precise

ABSTRACT

Precise Landing of VTOL UAVs using a tether

by Jeremy W Rathjen

Unmanned Aerial Vehicles (UAVs), also known as drones, are often considered the solution to complex robotics problems. The significant freedom to explore an environment is a major reason why UAVs are a popular choice for automated solutions. UAVs, however, have a very limited flight time due to the low capacity and weight ratio of current batteries. One way to extend the vehicles' flight time is to use a tether to provide power from external batteries, generators on the ground, or another vehicle. Attaching a tether to a vehicle may constrain its navigation but it may also create some opportunities for improvement of some tasks, such as landing.

A tethered UAV can still explore an environment, but with some additional limitations: the tether can become wrapped around or bent by an obstacle, stopping the drone from traveling further and requiring backtracking to undo; the tether can fall loose and get caught while dragging on the ground; or the base of the tether could be mobile and the UAV needs to have a way to return to it. Most issues, like those listed above, could be solved with a vision system and various kinds of markers, but this approach could not work in situations of low light, where cameras are no longer effective.

In this project, a state machine was developed to land a tethered, vertical take-off and landing (VTOL) UAV using only angles taken from both ends of the tether, the tension in the tether, and the height of the UAV. The main scenarios focused on in this project were normal operation, obstacle interference, loose tether, and a moving base. Normal operation is essentially tether guidance using the tether as a direction back to the base. The obstacle case has to determine the best action for untangling the tether. The loose tether case has to handle the loss of information given by the angle sensors, as the tether direction is no longer available. This case is performed as a last-ditched effort to find the landing pad with only a moderate chance for success. Lastly, the moving base case uses the change in the angles over time to determine the speed needed to reach the base.

The software was not the only focus of this project. Two hardware components of this project were a landing platform and a matching landing gear to support the landing process. These two components were designed to aid in the precision of the landed location and to ensure that the UAV was secured in position once landed. The landing platform was designed as a passive funnel-type positioning mechanism with a depression in the center that the landing gear was designed to match. The tension of the tether is used to further lock the UAV into place when in motion.

While some of this project remained theoretical, particularly the moving base case, there was flight testing performed for validation of most states of the proposed state machine. The normal operation state was effective at guiding the UAV onto the landing pad. The loose tether case was also able to land within reasonable expectations. This case was not always successful at finding the landing pad. Particular methods of increasing the likelihood of success are discussed in Future Work. The Obstacle Case was also able to be detected, but the response action has yet to be tested in full. The prior testing of velocity following can be used as proof of concept due to its simplicity.

In conclusion, this project successfully developed a state machine for precisely landing a tethered UAV with no environmental knowledge or localization. Further development is necessary to improve the likelihood of landing in problematic scenarios and more testing is necessary for the system as a whole. More landing scenarios could also be researched and added as cases to the state machine to increase the robustness of the landing process. However, each current subsystem achieved some level of validation and is to be improved with future developments.

I dedicate this Master's thesis to my late Grandfather, Gerald Rathjen. You are loved and missed incredibly. Thank you so much for helping develop my interest in coding and robotics and for gifting me my first programmable microcontroller, a PBasic Parallax board. Your support ultimately led me to my greatest interest in the field of engineering and you were a personal inspiration. I also want to dedicate this to my wife's late Grandmother, Shirley VanFossen who I sadly never had the pleasure to meet. Kayla misses you dearly and I wish I had been able to meet you in person and get to know who it is that my wife admires so much.

Acknowledgments

Many thanks to the people who have helped me through the process of my degree. I first want to thank my parents, who encouraged me to go to college in the first place. My research advisor, Dr. Pereira, who got funding for me to do research, and get my Master's degree. He also taught me about robotics through his classes Robotic Manipulators and Robotic Motion Planning. I also want to thank the help of Bernardo Martinez, Rogerio Lima, and Kieren Samarakoon whom all helped me learn how to use Linux, Robotics Operating System, python/c++ coding languages, and LaTex without which I would have been dead in the water performing my research and writing this thesis. Above all, I want to thank my loving wife Kayla Rathjen who's been with me through the many trials and tribulations that I've faced since deciding to pursue post-graduate school. I love you and I wouldn't have been able to be successful without your healing hugs.

This thesis was partially sponsored by Alpha Foundation for the Improvement of Mine Safety and Health, Inc. (ALPHA FOUNDATION). The views, opinions and recommendations expressed herein are solely those of the authors and do not imply any endorsement by the ALPHA FOUNDATION, its Directors and staff.

Contents

List of Figures

Acronyms

2D	Two Dimensions or Two Dimensional
3D	Three Dimensions or Three Dimensional
EKF	Extended Kalman Filter
IMU	Inertial Measurement Unit
IR	Infrared Radiation
LED	Light Emitting Diode
LiDAR	Light Detection and Ranging
PID	Proportional Integral Derivative
PWM	Pulse-Width Modulation
ROS	Robot Operating System
UAV	Unmanned Aerial Vehicle
UGV	Unmanned Ground Vehicle
WVU	West Virginia University

Nomenclature

Symbol	Units	Description
norm		As a subscript is short for 'normalized', the velocity vector before being split into the coordinate axis'
r		Ray representing an obstacle in a homotopic word
t	seconds	Time
v	m/s	Velocity

Greek Symbols

θ	*radians*	Time dependent path curving function
γ	*radians*	Azimuth angle
ϕ	*radians*	Elevation angle

Chapter 1

Introduction

1.1 Motivation

This project is motivated by the gap in autonomous unmanned aerial vehicle (UAV) control caused by reliance on computer vision. Many UAV systems today use a camera or a 3D LiDAR to sense the environment and then formulate a way to navigate through it. However, when vision is inoperable or not included in the design process due to lighting, weather, complexity, or other limitations, there is very little that is left to rely on for navigation. Systems lacking a vision system haven't gotten much much development and so may seem highly unlikely to succeed. Such a system has to remember the path it took to its position without any significant knowledge of the environment. A physical link or a tether between two points, however, can make navigation between them much simpler. It may appear as though a tether provides even more complications to UAV flight control but it can be also bring many benefits. It could get wrapped around an obstacle or fall loose on the ground and get caught causing unnecessary perturbations to the UAV that would even make manual control difficult. However, if properly controlled, the tether can also provide an unexpected source of valuable information. Anytime two objects are tied together and separated by space and obstacles, whatever is tying them can be followed from the location of one object to the location of the other. This idea can be exploited in the case of landing a tethered UAV. The physical link created by a tether makes it such that following the tether, even in poor weather, will always lead to the base. Even though the tether may be swaying in the wind or tossed in the rain. As long as it is not loose on the ground, it doesn't matter what perturbations are occurring in the space between the UAV and the base. The UAV can just follow the direction provided by the tether and it will eventually reach the base.

One robotic application that this proposed methodology applies to is underground mine inspection. Robotic systems for mining applications are currently in development because of the risk to human lives inherent in these mines. Inspecting can be a dangerous operation because the inspector has to get close to potentially fatal and unknown mine conditions and structures. One of these systems is currently in development at WVU and serves as funding for this thesis. For the design of this system, an unmanned ground vehicle (UGV) and a UAV are paired together to drive into a mine, park at pillars or other structures, and fly up to inspect the structural integrity. The UAV then has to land back on the UGV for the robot to move on to the next location to be inspected. The prototype ensemble robot is seen in Figure 1.1. Mines are very dark underground locations and so it is a prime example of a UAV system that can't rely on a vision system for landing. This design also includes a tether to the UAV that provides power so the UAV can fly for longer periods and complete the inspection in less time. When landing the UAV, the tether can be followed back to the ground vehicle resulting in successful return.

The precise landing of tethered UAVs is the focus of this research. Ultimately this project hopes to begin to fill a gap created by the reliance on vision systems and to apply a simple yet elegant method to complex issues that have arisen within the topic of UAV navigation and landing. Tethered vehicles are far from a novel concept but navigating with them has always been a difficult engineering problem to approach. Some of those problems are approached in this thesis.

1.2 Problem Statement

Many autonomous UAV systems rely on vision or mapping to determine landing location. However, some systems operate in low visibility where computer vision is unreliable or inoperable. Battery limitations can also cause many UAVs to have a very short time of flight. One method that can serve as a solution for both of these issues is a power tether connected to external batteries. Such a tethering system attached to a UAV can extend flight time for the completion of the UAVs mission, and be used as a method for determining the landing location, solving both issues at the same time.

Determining the landing location can be done by measuring the angles formed by the tether at the UAV and the battery/landing pad, an illustration of which can be seen in Figure 1.2. When the tether is straight these values determine the transformation between the coordinate frame of the landing area and the UAV. This method of localization, while not perfect due to the catenary effect on the tether, directly points from the UAV to the landing area and inversely. This can fail due to

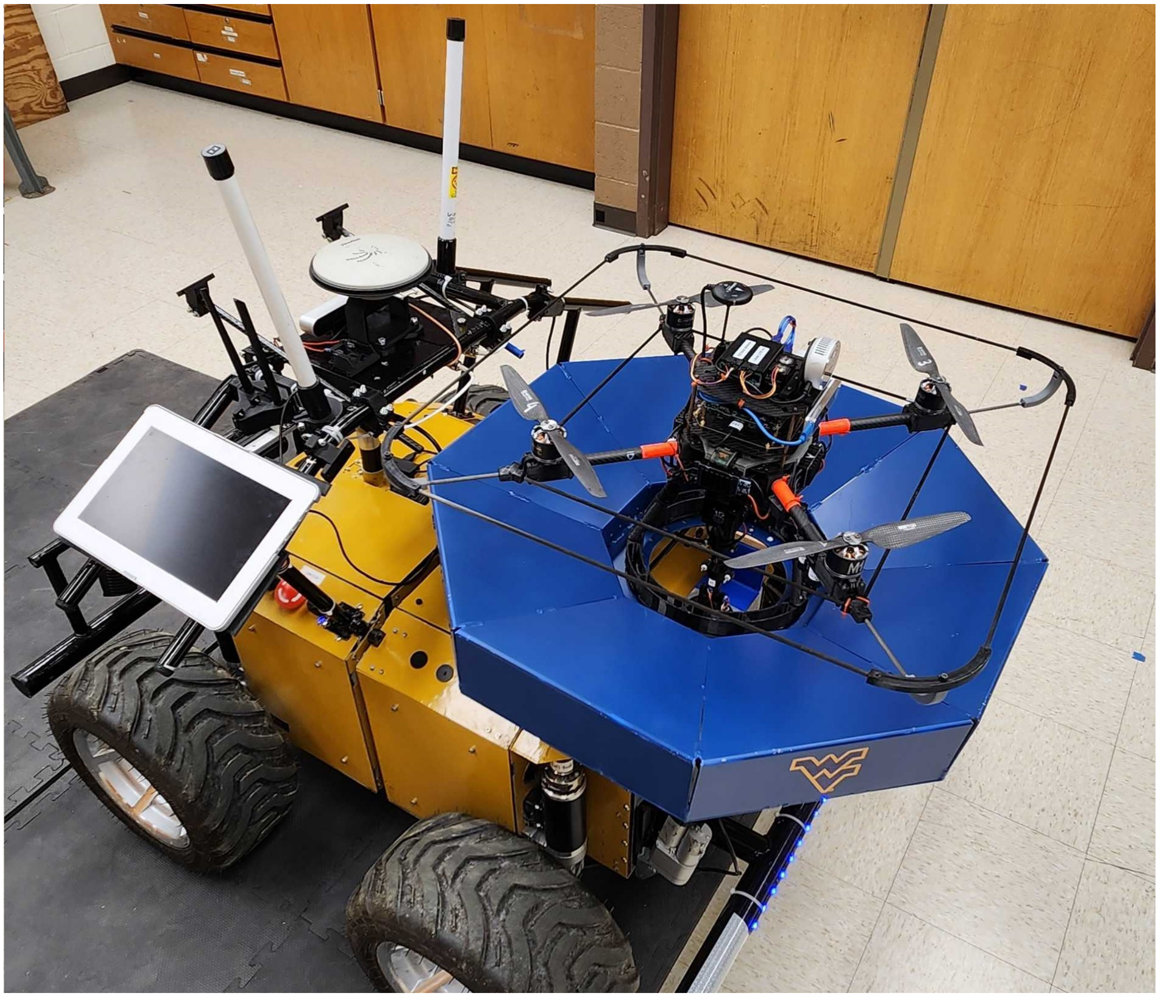


Figure 1.1: Ensemble of UGV and tethered UAV designed for stone mine mapping.

three primary reasons; the tether is no longer straight or it is bent, the winching or reeling system no longer keeps the tether taut, the catenary effect on the tether miscalculates the localization, or the battery end of the tether is on a mobile base. An algorithm for determining these states and the actions needed to successfully land has to account for these as potential problems.

Therefore, the problem approached in this thesis is landing a UAV precisely on a platform using the information gathered from a tether. Assumptions made to solve this problem include: 1) no computer vision system, 2) the presence of tether angle sensors as described before, 3) the presence, on the UAV, of a single beam LiDAR range finder pointing down, 4) the UAV has onboard a robust low-level controller that follows velocity set-points, and 5) the tether is managed by a system for winching/reeling system. The 'Loose Tether scenario operates under the assumption that the landing deck is located on flat or gently sloped terrain. Both the 'Moving Platform' and 'Obstacle' cases assume an accurate reading of the tether angles and that the calculations taken from these readings

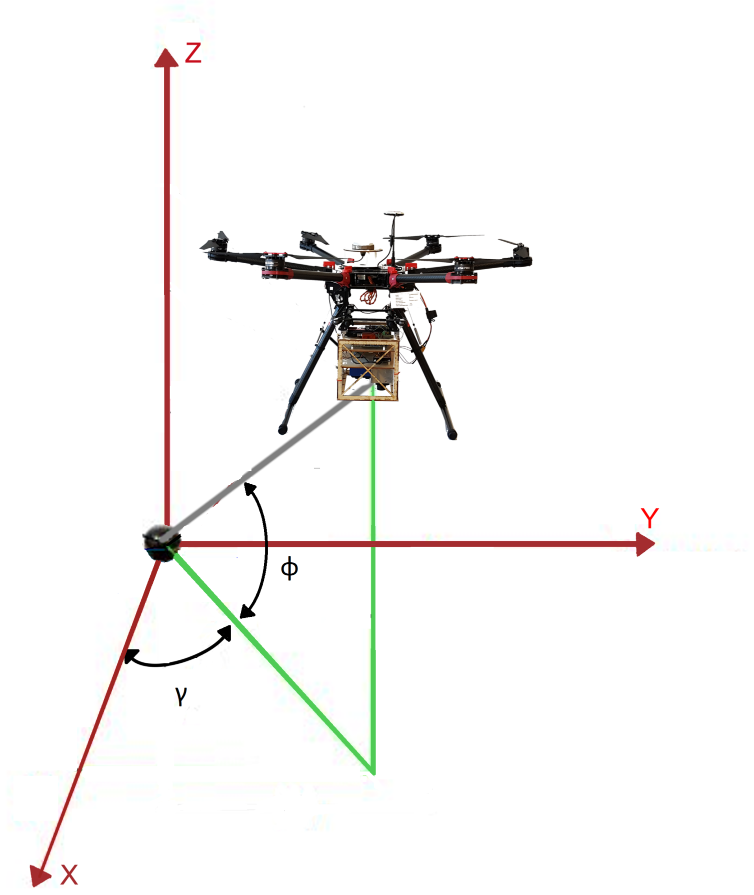


Figure 1.2: Image showing coordinate frame and tether interaction where $\varphi$is the elevation angle, and $\gamma$is the azimuth angle.

are also accurate.

In this thesis, a state machine was developed to determine these actions taken both in perfect operation and when a failure occurs. The state in which failure occurs needed to be determined according to the sensor data available. The winching system can calculate not only the tether's length but also the tension in the cable. On the winching end of the cable, a pair of potentiometers are used along with a cable guide to calculate the azimuth and elevation angles of the tether, and at the UAV a joystick (also using a pair of potentiometers) is used to calculate the azimuth and elevation angles of the cable. Changes in these data points (tether length and tension, and the azimuth and elevation angles at the landing area and UAV) will be used to determine bent or loose tethering or movement of the base of the tether, and to decide what actions are most likely to achieve landing at the designated area.

1.3 Contributions

In this thesis a methodology was developed for landing a tethered UAV without the use of a computer vision system. This method began with researching and designing hardware that would aid in the precision of the final landing location. A landing pad was designed as an octagonal funnel with a depression in the center for holding the UAV in place when in movement. This was integrated with a UGV that contained the winching mechanism for the tether. Mounting hardware and an auto-leveling system was also integrated for keeping the UAV stable when in motion and for landing when the UGV is on a slope.

A state machine was also developed to guide the landing process. This state machine also attempts to guide the UAV when problematic scenarios arise that make the tether an unreliable guide to the landing pad. Such scenarios include when the tether is loose, is bent by one or more obstacles, or when the UGV is moving with the landing platform. The loose tether, obstacle, and moving platform scenarios all make use of the information gained from the tether and a single beam LiDAR rangefinder mounted on the UAV pointed at the ground. Attempting to solve these issues increases the robustness of the tether guided approach and makes it a more viable solution to UAV navigation and landing without computer vision.

1.4 Literature Review

Autonomous UAV flight is a very difficult undertaking. Motion planning exists in many forms, some approaches take a 2D form such as [21] which is motivated by UAV motion planning but more general in its applications. This paper uses seventh-order Bezier curves to connect vertexes in a tree while effectively smoothing the path. Many approaches have been undertaken to extend these to 3D environments, such as [4] which can be applied to both 2D and 3D environments. Hugh Cover et al. [4] designed SPARTAN, an algorithm that performs fast motion planning with the ability to efficiently navigate a dense environment of obstacles such as dense vegetation. Another approach [31], seeks to improve Dubins Path in 3D environments to effectively decrease the number of helical curves and approach optimality. Fethi Belkhouche proposed a method of determining optimal trajectories in a dynamic environment in [3]. These trajectories, formulated using a virtual space representation, are reactive to the changing environment. Yet another motion planning solution, proposed in [7], uses Rapidly-exploring Random Tree(RRT) methods in conjunction with the Reduced Forward Model(RFM) of micro-UAVs to overcome the terrain and wind conditions in open-cut mining sites.

All of these methods require a map, either complete or in process of creation, to perform their planning. A tether provides different information and issues that result in several approaches to navigation for different types of vehicles.

The use of a tether can benefit many different types of vehicles, including cars, submarines, aircraft, and even space vehicles. Some of these are even used together and connected by a tether, such as [22], a UAV-UGV pair, and [11], a wheeled robot tethered to a support vehicle capable of position tracking. Another example of a tethered car exists in [19], which presents such a system to map extremely steep outdoor terrains. Research has begun to explore the idea of a 'hyper-tether', capable of self-launching/anchoring, power delivery, communication, as well as actively controlling tension and length. Edwardo F. Fukushima et al. proposed a prototype system in [8] for field works like cutting grass. One example of an underwater, or submarine, vehicle was used in [35], which developed a fiber optic cable to be used as the tether for communication between the underwater Hybrid Remotely Operated Vehicle(HROV) and a ship that operates it. In stark contrast, space vehicles have also been developed with tethered operations. Panfeng Huang et al. proposed a tethered space vehicle for debris removal in their paper [12]. Yet another example of a tethered robot comes from [23]. Louis Petit et al. [23] formulated a method for exploring 3D space while minimizing the distance traveled and unwound tether length. All of these systems were designed using various methods for mitigating the complications presented by the tether.

Homotopy is a common method for tracking the crossing of obstacles by mobile robots, made use of in [15] [16] [29] [30]. This is a means of classifying paths; two paths in an obstacle-populated environment are classified as homotopic or in the same homotopic class if they can be continuously deformed into each other without crossing any obstacles. In the case of tethered robots, the 'path' is equivalent both to the path of motion and to the tether itself. To determine the homotopic class, rays are drawn from each obstacle to a straight edge of the map and given a designation, typically $\mathrm{r_i}$. A paths homotopic class is the sequence of those designations in the order that the path crosses them, forming a 'word'. This is directional as well, if a ray is crossed left to right the notation remains $\mathrm{r_i}$, if the ray crossed right to left a superscript '-1' is added and the notation becomes $\mathrm{r_i}^{-1}$. A reduction can be done on the word if the same ray is crossed in the positive and negative direction sequentially, ie. $\mathrm{r_1 r_2 r_2}^{-1}\mathrm{r_3}$ becomes $\mathrm{r_1 r_3}$. However, if this ray is not crossed in sequence the word cannot be reduced, ie. $\mathrm{r_1 r_2 r_3 r_2}^{-1}$ cannot be reduced. This approach to tether navigation/approximation relies on a map either already known or in the process of creation. An example of this can be seen in Figure 1.3. The path τ_1 has homotopy class $\mathrm{r_1}$, path τ_2 has homotopy class $\mathrm{r_1 r_1}^{-1}\mathrm{r_1}$ which is simplified to $\mathrm{r_1}$, and path τ_3 has homotopy class $\mathrm{r_1 r_2 r_3}$. From this example, τ_1 and τ_2 are homotopic because they have the same simplified homotopy class.

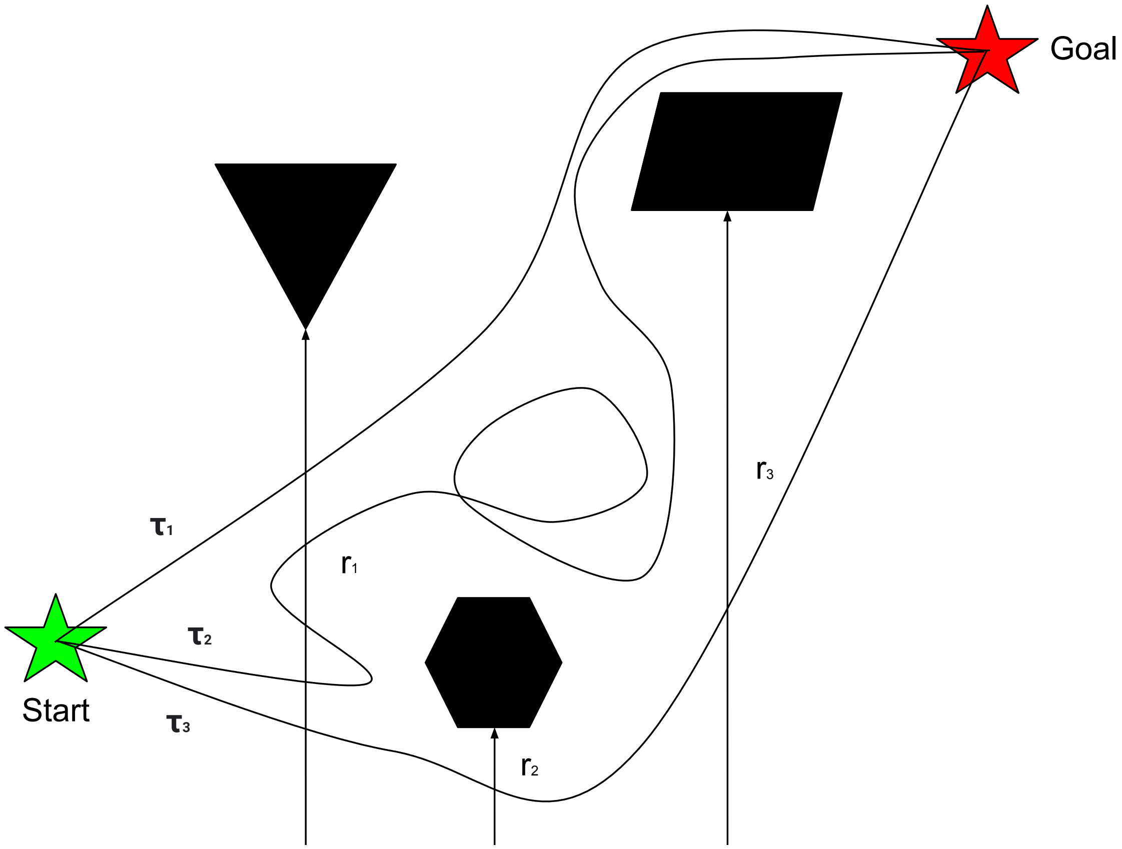


Figure 1.3: An example of homotopy used to classify paths τ_1, τ_2, and τ_3. Where the path τ_1 has homotopy class r_1, τ_2 has homotopy class $r_1 r_1^{-1} r_1$ (simplified to r_1) and , and τ_3 has homotopy class $r_1 r_2 r_3$. Therefore, paths τ_1 and τ_2 are considered homotopic.

It is also common for robotic systems to use inertial measurement units(IMUs) to calculate position. An IMU is a combination of sensors that output force, angular rates, or even a body's orientation. Specifically, the angular rates, forces, and accelerations are integrated over time to calculate velocity and position. This estimation is subjected to error accumulation and so is often paired with another position estimation system to decrease these error accumulations. UAV systems are even more complicated due to their operation being in 3-Dimensions. The IMU can be used to compute the tether's tension force, as it is in [24] and [25] on UAVs, which operate by converting the IMU data to output forces. These are used with the thrust input forces, calculated from the pulse-width modulation (PWM) signals to the motors, to give an approximation for the force of the tether on the vehicle. The elevation and azimuth angles that this force is acting in are then used with the tether length to get the position of the UAV with respect to the other end of the tether. The methodology in [25] (written by the same authors and as an improvement of [24]) is designed with the assumption that the force in the cable is unknown and making use of a 4-state Extended Kalman Filter (EKF) to make up for this, while [24] operates with the assumption that the cable force is known. This is not reliant on localization or mapping of any kind.

Tethered vehicles, and autonomous vehicles in general, are often operated in conjunction with mapping. This is typically to avoid tangling the tether around obstacles such as pillars. One unique way of using the environment map is presented in [33], which uses the map to do motion planning for a UAV specifically when the tether is straight or kinked. This involves taking points where the tether would contact an obstacle, or contact points, and storing them as way-points. This allows the UAV to travel the entire 3D space and remember where the tether is tangled when returning to the base. This return is performed by navigating back through the list of waypoints while avoiding the obstacles themselves until the UAV is in free space near the base. This return trip would typically end in landing which may come with its complications.

Vision systems are often included with methods for landing. There are many different forms of markers for camera recognition and image filtering to do so more efficiently. In [28], an ArUco marker detection system is used to perform localization for a drone. The image from a camera is first converted to greyscale, then to binary using the adaptive threshold method and then the image contours are recognized using the OpenCV library. From these contours, the marker is extracted and the position is calculated from its size and orientation in the image. This position is then used to direct the UAV to the landing location. References [6] and [32] do essentially the same thing except that the marker is an 'H'. For both of these, this is made more complicated, in [6] because the camera has a fisheye lens and is on a gimbal to make sure the image plane is parallel to the ground, and in [32] where the marker is on a platform moving to simulate the pitching and rolling of the deck of a ship. The system designed by Liyang Wang et al.,[32], also makes use of LEDs in the four corners of the 'H' symbol to make it detectable from a distance. Reference [2] is similar but it detects two circles, one red and one blue, in the image and is to land on the midpoint between them. The measurements needed are altitude, rotation, and horizontal distances. Altitude and rotation are both given directly by the camera and an ultrasound sensor while the horizontal distances must be calculated by transformations between the camera frame and the landing position frame. In paper [13], localization is performed with IR markers, however, the markers are on the UAV and the camera is on an unmanned ground robot (UGR) or a UGV. This is also done with a concentric set of IR markers to increase the redundancy of the system. In each system, the localization is then used to determine the action for landing the UAV.

Some research for UAV landing uses LiDAR for localization or landing area detection. This type of sensor, in its 3D incarnation, can be used along with a clustering algorithm to cluster the raw point cloud data from the sensor and track an object such as a UAV. In [14] the sensor is mounted on an unmanned ground vehicle (UGV) and a point-distance-based-segmentation method is used. This compares the Euclidean distance between points and classifies them based on a threshold. This localization is then used with a PID controller to get the UAV to the landing area. Inversely, in [34] the sensor is mounted on the UAV and used to determine a safe landing area not determined prior such as a UGV. The sensor is used to differentiate a flat, clear area without obstacles to land from other less safe landing options. In this paper principal component analysis and an improved region-growing algorithm are used to detect the flat, safe regions to land below it. These systems would be significantly impacted by the use of a tether. Depending on the implementation, this could be positive or negative

For vehicles that utilize a power-supplying tethering system, it is advantageous to get more usage from it than just receiving power, especially due to the navigational complications caused by using a tether. One such advantageous use of a tether that can aid in these navigational issues is localization. References [1] and [27] perform these calculations for helicopters, while [18] does the same for drones. Of these three papers, only [18] attempts to localize in the case of a loose or slack tether that forms a catenary. Both [1] and [27] assume a taut tether and utilize the fact that the angles taken from the tether below the helicopter will point directly to the winching end. This is used to direct the helicopter directly above the landing area, where the tether is vertical, then either the tether is winched to pull the helicopter down or the helicopter descends while the tether is kept taut. Rogerio Lima et al.'s paper, [18], designed a system for localization using a tether. There is no motion planning being performed, however, the winching system used to achieve localization is also being used in this thesis. This system for localization could be used in conjunction with a system like the one defined in [10], which details a method of drone landing using vector fields defined by the position of the drone.

Landing a UAV can be done anywhere that is flat or level. This, however, would leave the UAV in any, potentially unknown, location and may require retrieving by the operator. Autonomous systems, however, lack a present operator to perform this retrieval. Any system with a ground vehicle or charging station requires some design for a specialized landing station, landing pad, or retrieval system. Many of the common landing systems for UAVs or drones in use today are reviewed in [9]. These designs vary from passive or active positioning to charging stations. Of particular interest are the passive, funnel type because they require no actuation. This paper also delves into the pros and cons of each method of landing and summarizes them into a table that aids in defining the best method for any particular situation or drone design. Musa Galimov et al. wrote patents [17] [20] [26], which are referenced by [9], as passive funnels. Ran Krauss et al.'s patent, [17], is an octagonal funnel specifically designed with flat plates so that it can retract when landing is not the immediate function. This can be useful if the pad itself becomes a limiting factor to other parts of the design, such as the inclusion of a battery-swapping mechanism that requires more space around the drone to function. Another UAV positioning mechanism, [20], includes a cone in the center of the funnel for even more precise positioning in the drone's descent. Lastly of the passive funnel type landing pads, [26] is a funnel with a depression in the center to fit the foot of the drone not only for precise positioning but also for holding the drone in place. UAV landing, however, doesn't have to result from a descent. Sudam Chamikara De Silva et al.'s paper, [5], presents a design for a docking station that is inverted to take advantage of the ceiling effect that occurs when a drone flies upwards near a ceiling. This design makes use of a similar mechanism to one discussed in [9], the active positioning, W-shaped pusher mechanism. It also includes a battery swapping mechanism for when the drone's limited flight time is spent. This mechanism then stores the used battery for charging, keeping enough charging slots available for continuous operation.

The approach outlined in this thesis makes use of the winching mechanism from reference [18]. Instead of using the mechanism for localization, it will be used to achieve direction that will lead the UAV to the landing pad. This is used instead of mapping and motion planning. It also excludes the use of vision systems such as in [13], [6], [28], [2], and [32]. A state machine takes the tether information and height measured from a single beam LiDAR rangefinder, and outputs the actions that result in landing. Tether information refers to the elevation and azimuth angles at both the UAV and the landing pad, as well as the length and the tension in the tether. This information is used to determine the state that the UAV is in. These states are 'Normal Operation', 'Loose Tether', 'Obstacle', and 'Moving Platform', also referred to as cases. Each case has sub-states determined by variations in the tether and height data that output different actions to eventually result in a landing. These sub-states will be discussed further in the methodology section. The approach of this thesis is similar to [1] and [27], which use the tether angles for landing guidance, with the addition of problematic cases. This thesis also contributes the design of a funnel-type landing pad that passively positions the UAV as it lands, similar to [17] and [26]. This landing pad is used to increase the precision of the landed position. It was designed to mount onto the back of a UGV with the intent to be leveled automatically when the UGV is motion. In summary, this thesis presents a method for autonomously landing a UAV in a precise position without the use of computer vision.

1.5 Thesis Organization

The rest of this thesis further defines the hardware designs and state machine architecture, continues with results from testing, and ends with conclusions regarding the system and future work. Chapter 2 begins the 'Methodology' with sections 2.1,'Hardware', describing hardware designs, and 2.2,'State Machine'. Section 2.2, 'State Machine', has sub-sections defining the 'Normal Operation' case, and 'Landing in Problematic Scenarios' which itself has subsections defining the approach for the 'Loose Tether', 'Obstacle', and 'Moving Platform' cases. With this, the 'Methodology' is complete and the paper continues with Chapter 3, 'Results'. This chapter begins with section 3.1, 'Testing Conditions', which describes the conditions used to test this methodology. Next are sections describing results for the specific cases; section 3.2 is 'Normal Operation', 3.3 is 'Loose Tether', and 3.4 is 'Obstacle'. Chapter 4 is 'Conclusions'. This chapter has section 4.1, 'Discussion', which has subsections for each case; 4.1.1 'Normal Operation', 4.1.2 'Loose Tether', and 4.1.3 'Obstacle'. This chapter concludes with section 4.2, 'Future Work', which has subsections 4.2.1 'Loose Tether', 4.2.2 'Obstacle', 4.2.3 'Moving Platform', and 4.2.4 'Case Priority'. Lastly is the list of references, 'Bibliography'.

Chapter 2

Methodology

Much of this project relied not only on algorithm and state machine design but also on the design of the hardware on which that code operates. The landing pad is mounted to the back of a UGV for mobile applications, as seen in Figure 1.1. A tether, pulled by a winching system, runs through the landing pad with an angle sensor and is attached to the UAV via a similar angle sensor. Onboard the UAV is a single beam LiDAR rangefinder used to detect the height of the UAV. Using the information from these three sensors, the UAV will navigate to the landing pad from any location that it can reach with the tether, including those locations that bend the tether (or where obstacles are present). It will also be able to determine when the tether falls loose. Despite the tether being used as the primary source of direction, the UAV will also be able to find the landing pad due to the inclusion of the LiDAR rangefinder. The UAV will also be able to detect movement of the landing pad (because it's mounted on the back of a UGV) and increase flight speed to be able to reach the landing pad and continue the landing process.

2.1 Hardware

UAV landing begins with some knowledge of the landing location. For a tethered UAV, this is predetermined and can be utilized to ensure landing precision. Towards this end, a passive funnel-type positioning mechanism was designed as a landing pad with an octagonal shape for ease of manufacturing. The landing pad design and manufacturing is depicted in Figure 2.1. There is a depression in the very center where the UAV will be held in place when the landing pad moves, as it will be mounted on the back of a UGV. It was considered to add a central cone to the mechanism for even greater precision, however, it was concluded that this would restrict the sensors for the tether angles. Below the landing pad has a bracket for mounting this sensor and all around the base plate are holes for mounting additional sensors, lights, or buttons for future experimentation and addition.

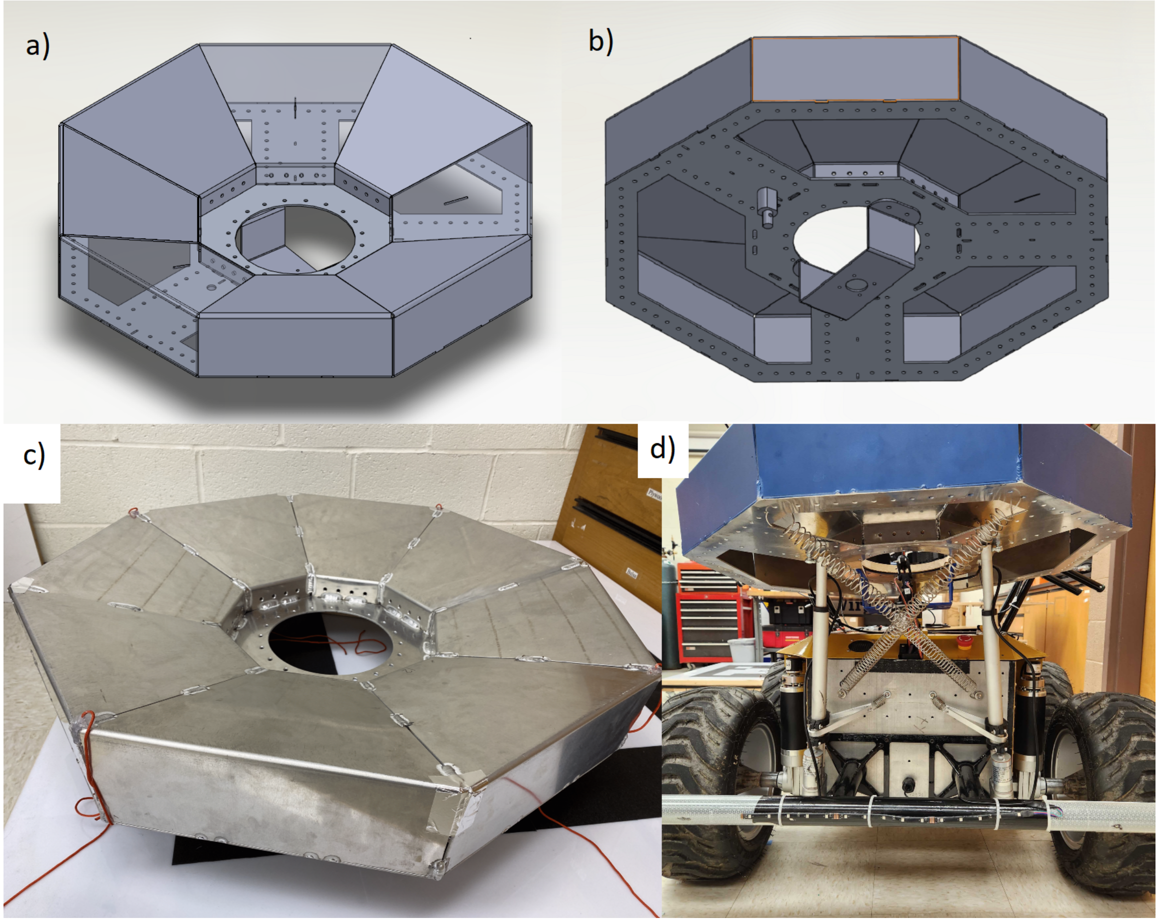


Figure 2.1: Landing pad in various states of design and manufacturing, a) is in Solidworks from above, b) is also in Solidworks from below depicting the tether angle sensor bracket, c) is the main body manufactured, and d) is the landing pad mounted on the UGV.

The primary interest of this project is the process of landing. This process will utilize the tether angles (measured on either end of the tether by a joystick) and the height of the UAV (measured by point LiDAR). This information can approximate the UAV position but is subject to error because of the catenary effect. A system that relies more directly on the angles and length/height could easily be used to lead the UAV in the direction of the landing pad and direct its behavior and flight pattern. The tether angles are output as azimuth and elevation angles both onboard the UAV and at the landing pad. For the implementation in this project, the elevation angles range between 60 and 90°(where 90°is positive or negative vertical) due to both the limitations of the measurement devices and the mounting location of those devices.

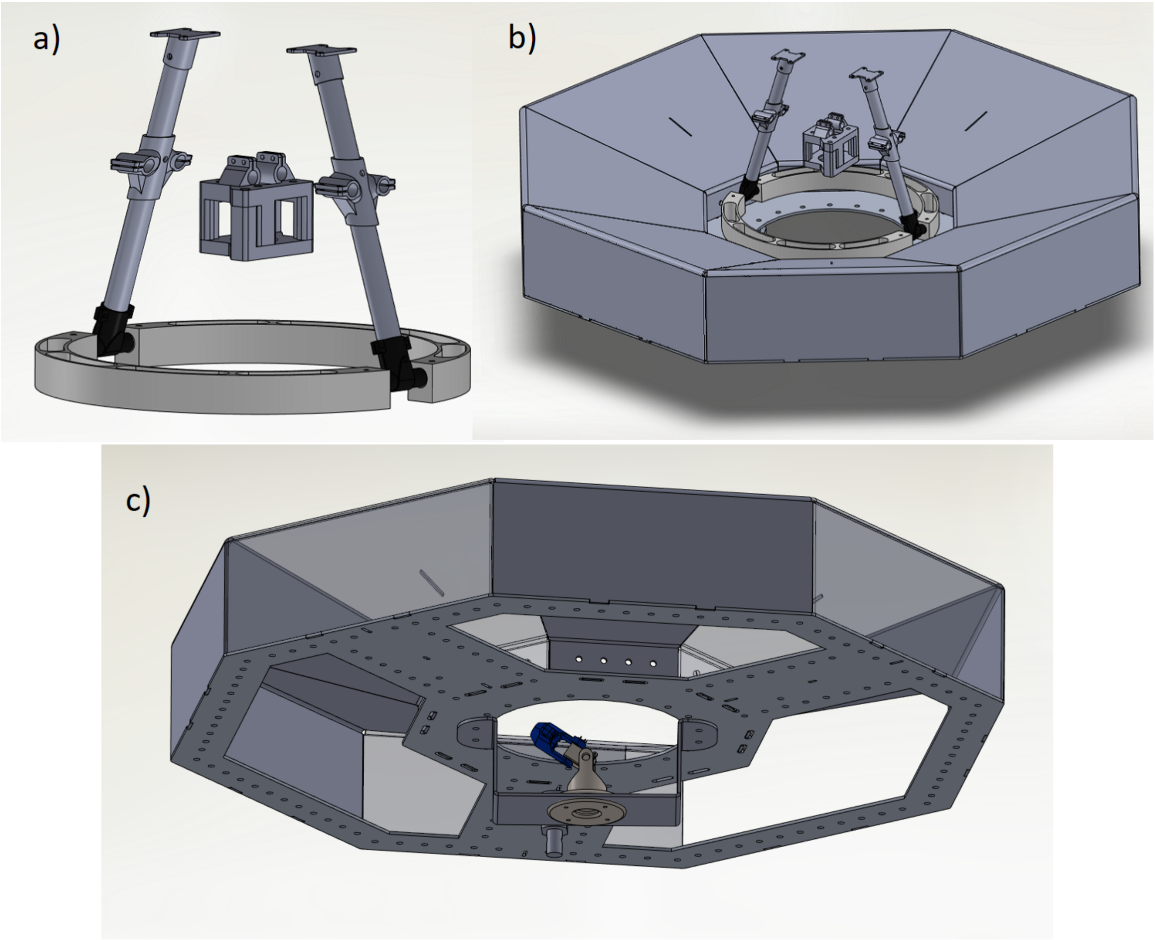


Figure 2.2: Landing gear design in Solidworks, a) is the landing gear alone with the joystick mount between the legs (the rods for holding the joystick mount are not visible), b) is the landing gear in on the landing pad to show how it fits and should be held in place, and c) is of the tether angle sensor mounted on it's bracket to show the angle limitations.

The angle sensor at the landing pad is mounted below the pad itself. The elevation angle is therefore limited to the hole that the tether has to pass through, shown in Figure 2.2(c). Similarly, the sensor on the UAV is mounted hanging below the main body of the UAV and above the landing gear. This landing gear was designed as a circular foot so it avoids orientation constraints when landing. A circular foot also fits the octagonal depression in the center of the landing pad which holds it in place when moving, as shown in Figure 2.2. This limits the elevation as the tether passes through the circular landing gear. Therefore the algorithm that controls the UAV to autonomously land must operate within these limitations.

The landing pad was designed with many mounting holes not only for future developments and additions, but also to be mounted on the back of a UGV which can be seen in Figure 2.1(d) and Figure 2.3. This design included linear actuators, heim joints, a large U-Joint, and springs for stabilization. The linear actuators are held to the landing pad with the heim joints and provide two mounting points as seen in Figure 2.3(c), while the third is the large u-joint as seen in Figure 2.3(b). This joint was custom threaded on one side to mount to the UGV and left smooth in the other for a custom designed pin. This pin is mounted to the platform and allows for quick removal of the landing pad when the UGV needs work.

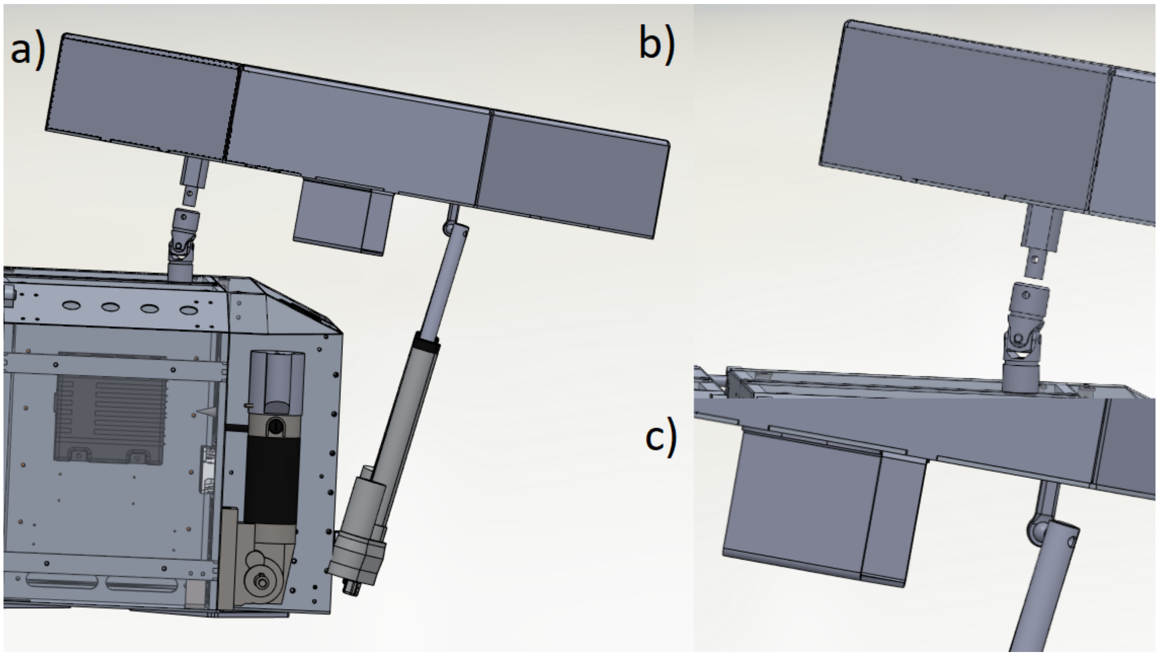


Figure 2.3: Solidworks design of the landing pad mounting to the UGV also seen in Figure 2.1(d), where a) shows the whole thing, b) shows the u-joint, and c) shows one of the heim joints.

2.2 State Machine

The state machine for autonomous landing operates in two different scenarios, normal operation, and problem-solving. Landing occurs in the 'Normal Operation' case, while attempts to solve potential problems occur in the 'Loose Tether', 'Moving Platform', and 'Obstacle' cases. Each scenario can be entered and exited with the clearing and detection of a new issue. This can be visualized in Figure 2.4.

2.2.1 Normal Operation

The state machine for landing primarily follows six states that depend upon the angles of the UAV-mounted joystick and the height measured by the LiDAR. These six states follow the flow of the landing process and cover any locations unhindered by obstacles around the landing pad. This

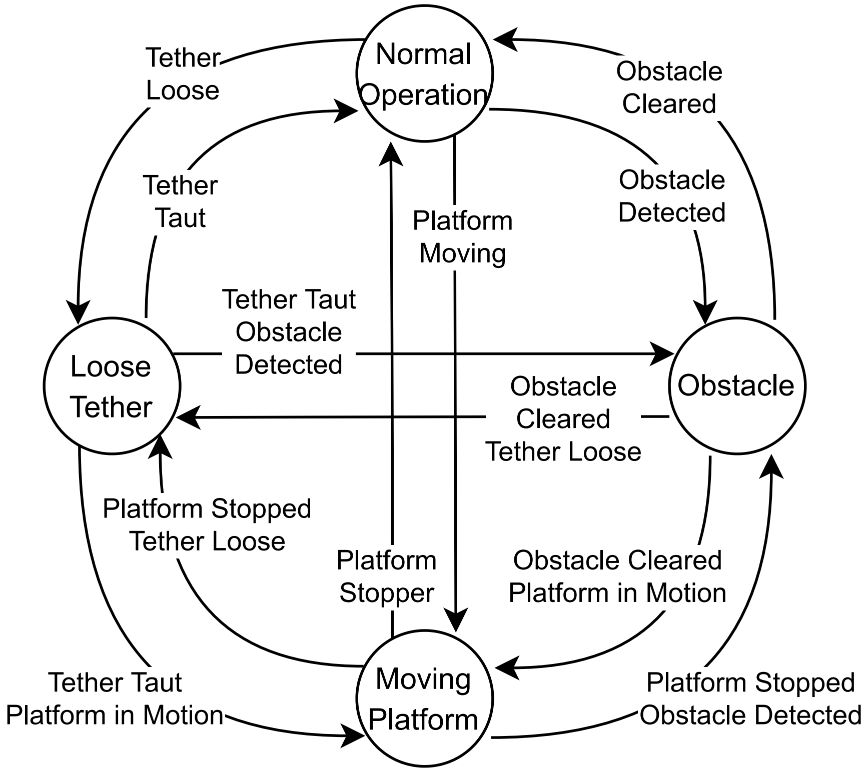

Normal Operation
Tether Loose
Obstacle Cleared
Tether Taut
Obstacle Detected
Platform Moving
Tether Taut Obstacle Detected
Loose Tether
Obstacle
Obstacle Cleared Tether Loose
Platform Stopped Tether Loose
Platform Stopper
Obstacle Cleared Platform in Motion
Tether Taut Platform in Motion
Moving Platform
Platform Stopped Obstacle Detected

Figure 2.4: Diagram for overall landing state machine, each case has an exit and entrance condition for all states. Notice that between 'Loose Tether', 'Obstacle', and 'Moving Platform' two conditions are required, one to exit the current case and another to enter the next.

process is shown in Figure 2.5.

The state '45°Ascent' is for positions below the height of the landing pad where the joystick will be measuring the minimum elevation angle (about 60 degrees in this project's implementation). The following states are all for positions above the height of the landing pad, where the elevation angle begins to increase. 'XY Approach' occurs when the UAV is far away and a horizontal approach is needed. '45°Descent' begins when the UAV is close enough to begin descending but far enough that it still needs to approach in the XY plane. 'Vertical Descent' begins when the UAV is detected to be above the landing pad, this state includes proportional control in the XY plane. 'Descent w/P Control', referring to proportional control of the z velocity, v_z, begins when the UAV is above the platform and is low enough that the UAV needs to slow its descent to lessen the impact of landing. Lastly, 'Landed' occurs when the UAV is above the platform and is low enough to stop the rotors from spinning and to land. The transition conditions are specific to the implementation and the limitations of the hardware but are based only on the elevation angle measured at the UAV and the height detected. Optimally, these six states would occur in succession in the order described, but some initial positions and paths will cause the algorithm to skip some states. For example, it's

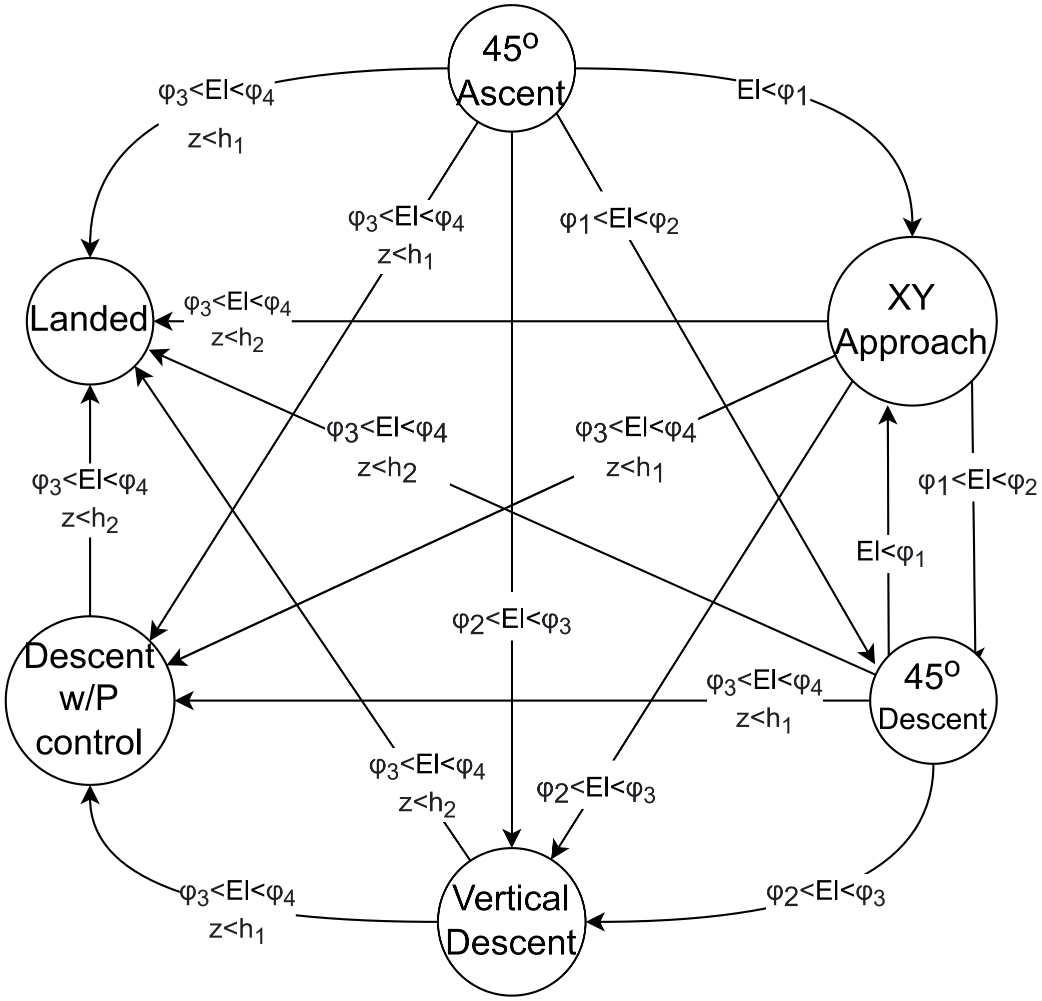

45^o Ascent
φ_3<El<φ_4 z<h_1
El<φ_1
φ_3<El<φ_4 z<h_1
φ_1<El<φ_2
Landed
φ_3<El<φ_4 z<h_2
XY Approach
φ_3<El<φ_4 z<h_2
φ_3<El<φ_4 z<h_1
φ_3<El<φ_4 z<h_2
φ_1<El<φ_2
El<φ_1
φ_2<El<φ_3
Descent w/P control
φ_3<El<φ_4 z<h_1
45^o Descent
φ_3<El<φ_4 z<h_2
φ_2<El<φ_3
Vertical Descent
φ_3<El<φ_4 z<h_1
φ_2<El<φ_3

Figure 2.5: Diagram for 'Normal Operation' state machine, where El is the measured elevation angle, z is the measured height, $h_{1,2}$ are height conditions, and $\varphi_{1,2,3,4}$ are elevation angle conditions

possible to begin not in '45°Ascent' but in 'XY Approach'. Some of these possible paths through the state machine are shown in Figure 2.6, along with the optimal state flow in Figure 2.6(a). These different state machine paths can also be caused by other external factors, such as a strong gust of wind, or latency's in communication between sensors and the UAV.

The flight path, if it follows the optimal state flow, is a 45° incline, followed by a horizontal line, which progresses to a 45° descent, ending in a vertical descent as shown in Figure 2.7. All horizontal movement is directed toward the landing pad, following the azimuth angle of the UAV. This flight path may shorten, lengthen, or skip states entirely, as mentioned above, depending on how close the UAV is to the landing pad, both vertically and horizontally, or the impact of external factors.

This path is described above as though taking place in a 2D plane, as seen in Figure 2.7, and while this is the goal of the behavior, the UAV is flying through real, 3D space. It must, therefore,

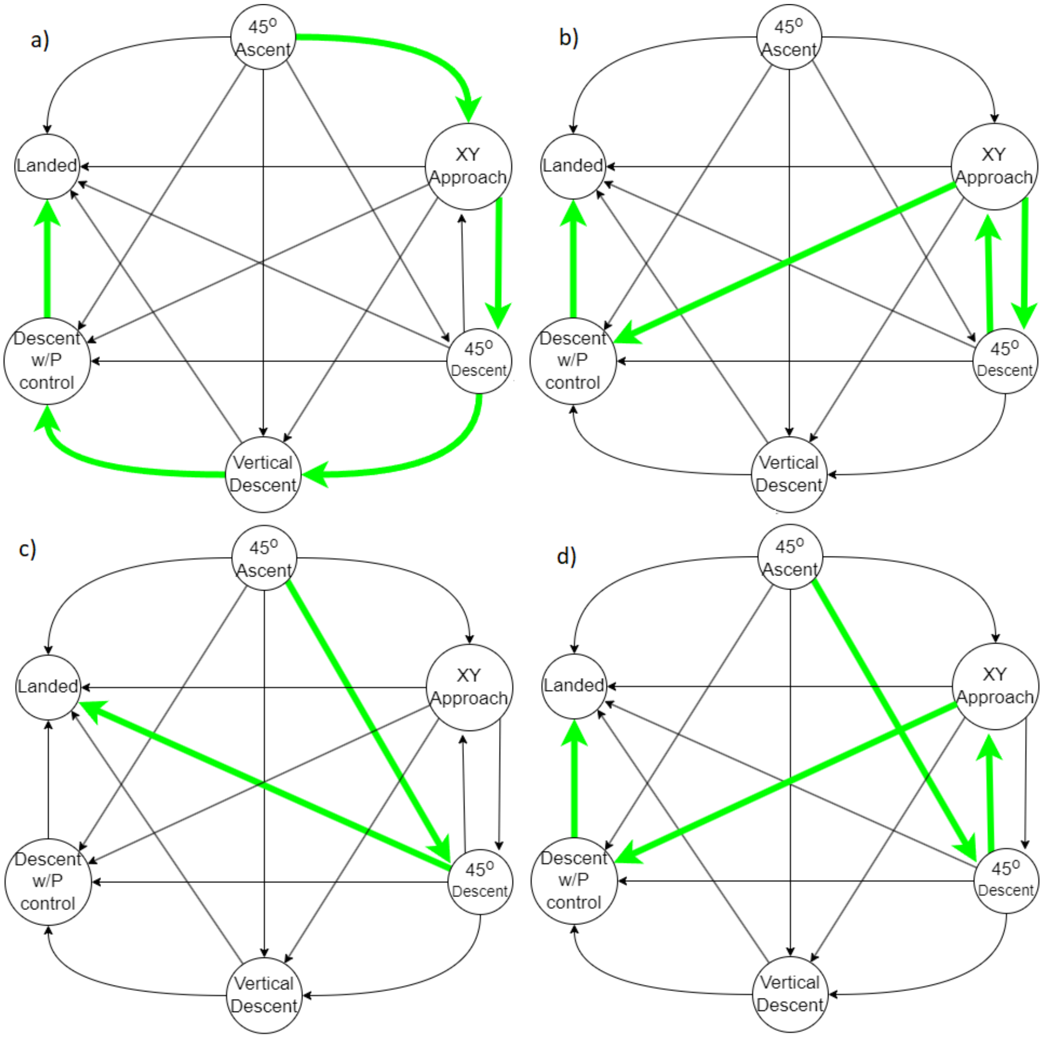


Figure 2.6: Paths taken in the state machine to achieve landing where a) is ideal, b), c), and d) are other possible paths.

be described by velocities along the three Cartesian axes, v_x, v_y, and v_z, with respect to the UAV's origin. These speeds are not only dependent on the elevation angle, φ, and azimuth angle γ, but also on time and height in some cases. To make the control of the UAV smoother, ramps are used based on timers within the code. These ramps incur that the velocities are also dependent on the time elapsed after switching the state, t.

Another way of visualizing the 'Normal Operation' landing process, similar to Figure 2.7, is in a vector field. Figure 2.8 is the first of these representations. This plot makes the landing process easier to understand because it is a direct visualization of the workspace. This graph is only limited to one plane above the horizontal plane, to get the full configuration space this graph would have to be rotated 180° about the z-axis. A 2D representation is much easier to view for this purpose.

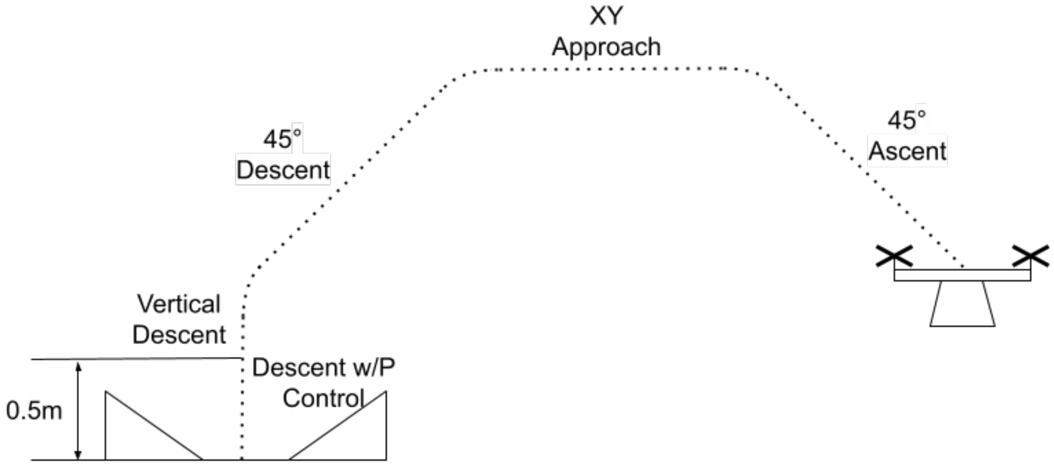


Figure 2.7: Illustration of the flight path in it's 2D form showing each sub-state of 'Normal Operation' case. Notice the curves where the state changes which are created using timers to increment direction changes.

In Figure 2.8, the elevation angle is calculated from the (X, Y) position and is used, along with the height, to determine the velocity vector from the 'Normal Operation' state machine, Figure 2.5.

The configuration space representation of the 'Normal Operation' case can be seen in Figure 2.9. This similar to Figure 2.8 except its not a direct representation of the workspace. Instead, the actual inputs to the state machine, the elevation angle, φ, and the height, h, are used to determine the velocity vector. As can be seen, the x-axis of the plot is the elevation angle in radians and the y-axis is the height in meters. This plot only goes from the angles 0 to π radians because when the elevation angle reaches π radians, or 90°, and the UAV continues in the same direction, the elevation angle decreases. This is because it is now measuring the angle from the horizontal plane opposite that which it was originally measuring from. Essentially, to get the full configuration space, this graph is rotated 360° to get the full configuration space for every azimuth angle, similar to Figure 2.8.

The piece-wise function that creates this path begins with Equation 2.1,

$$v_{\text{norm}} = \left\{ \begin{array}{l} 0.15t, t < 1 \\ 0.15, t \geq 1 \end{array} \right\} \tag{2.1}$$

which is the magnitude of the velocity of the drone. Notice that this is speed changes linearly with time, t, to avoid step inputs to the drone controller. The maximum velocity is 0.15 m/s and it takes 1 s for this speed to be reached, both of which are specific to this project's implementation. Whenever

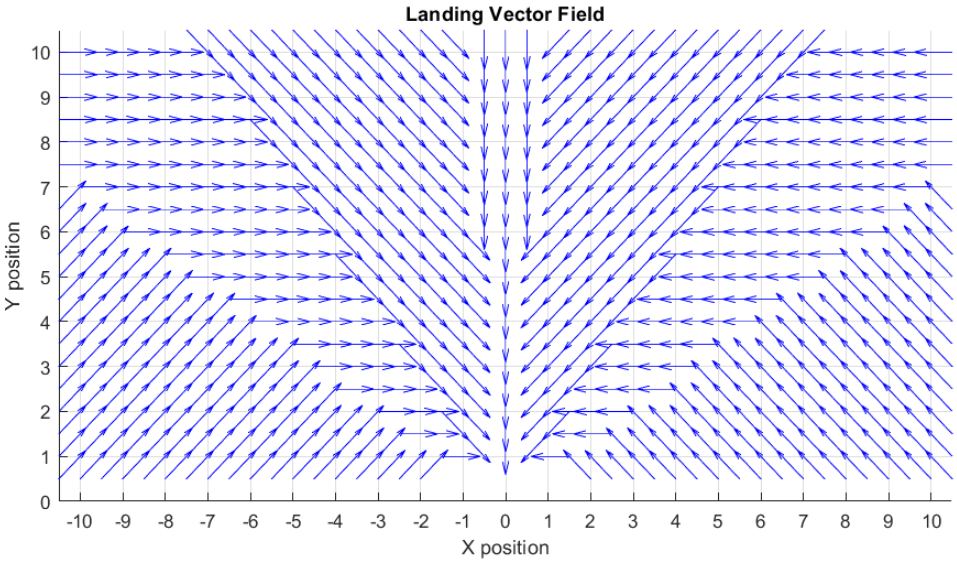


Figure 2.8: Vector field representation of the flight path. This is how the velocity is broken down in the 'Normal Operation' case, Figure 2.7 could be overlaid on top of this image to further illustrate.

the speed changes direction, particularly vertically (as shown in Figure 2.7), similar equations are employed within those states to ramp the speed up or down depending on the Cartesian direction that needs to change, most often vertically, z.

Equation 2.2 shows the 3D components of the velocity to create the path for '45°Ascent';

$$\begin{aligned} v_x &= \frac{\cos(\gamma) \cdot v_{\text{norm}}}{\sqrt{2}} \\ v_y &= \frac{\sin(\gamma) \cdot v_{\text{norm}}}{\sqrt{2}} \\ v_z &= \frac{v_{\text{norm}}}{\sqrt{2}} \end{aligned} \tag{2.2}$$

The velocity is at a 45° angle upwards and towards the landing pad, as detected by the azimuth angle, γ.

Velocities $\mathrm{v_x}$ and $\mathrm{v_y}$ are multiplied by the appropriate trigonometric function to point in the direction of the landing pad as guided by the tether. However, these speeds are divided by the square root of two to give the velocity a 45° angle above the horizontal plane of the UAV's coordinate frame and keep the magnitude constant.

Equation 2.3, for 'XY Approach' also has $\mathrm{v_x}$ and $\mathrm{v_y}$ multiplied by the trigonometric functions of

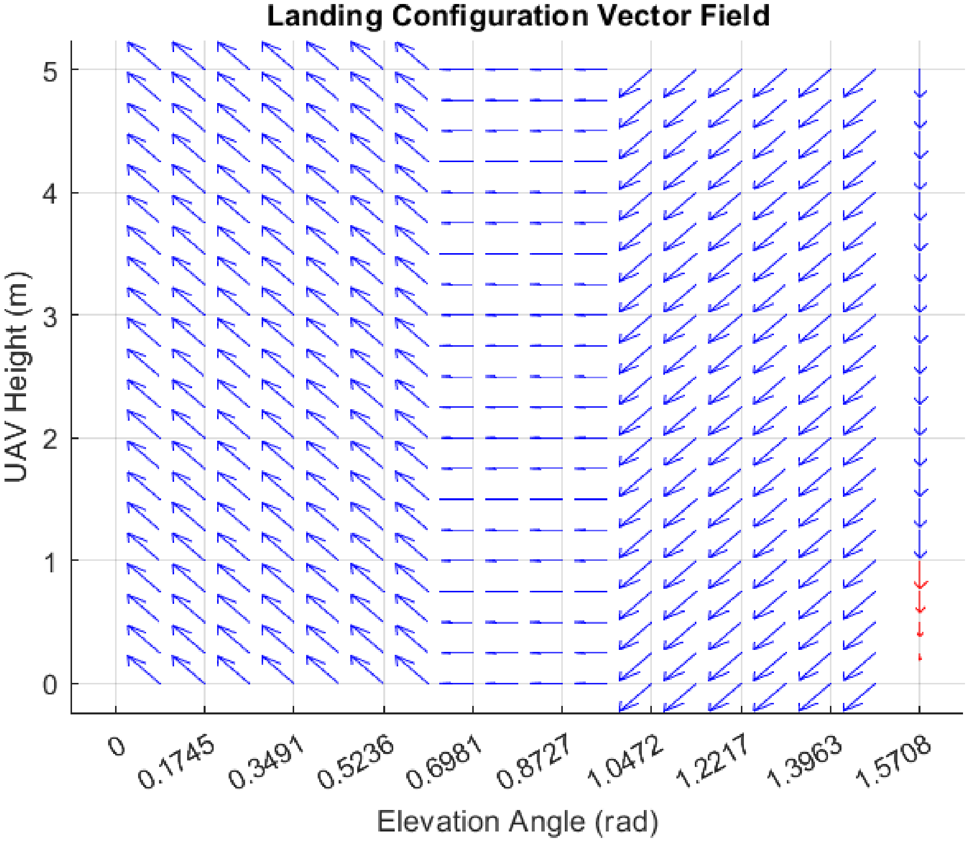


Figure 2.9: Vector field representation of the configuration space of the UAV. As the elevation angle increases, the UAV progresses through this vector field until it lands

γ.

$$
\begin{aligned}
v_x &= \cos(\gamma) \cdot v_{\text{norm}} \ \cdot \cos(\theta) \\
v_y &= \sin(\gamma) \cdot v_{\text{norm}} \ \cdot \cos(\theta) \\
v_z &= v_{\text{norm}} \ \cdot \sin(\theta) \\
\theta &= \left\{ \begin{array}{c} \pi/2 - t\pi/2, t < 1 \\ 0, t \geq 1 \end{array} \right\}
\end{aligned}
\tag{2.3}
$$

This is the first state that includes a time-dependent path-curving function. A new variable, ϑ, is used to ramp down the vertical velocity and transfer it into the horizontal plane beginning the horizontal line portion of the path and keeping the velocity magnitude constant. As time increases, up to 1 s after the 'XY Approach' state begins, the vertical velocity decreases, and the horizontal velocity increases until all 0.15 m/s are horizontal pointing towards the landing pad.

’45°Descent’ is very similar to the ’XY Approach’ and is described by Equation 2.4;

$$\begin{aligned} v_x &= \cos(\gamma) \cdot v_{\text{norm}} \cdot \cos(\theta) \\ v_y &= \sin(\gamma) \cdot v_{\text{norm}} \cdot \cos(\theta) \\ v_z &= -v_b \cdot \sin(\theta) \\ \theta &= \left\{ \begin{array}{c} t\pi/2, t < 0.5 \\ \pi/4, t \geq 0.5 \end{array} \right\} \end{aligned} \tag{2.4}$$

It includes the path curving function, noted by variable ϑ, except that in state ’45° Descent’ the vertical velocity further decreases to $-v_{\text{b}} sin(\pi/2)$ as time increases to 1 s after the ’45° Descent’ state began. This decrease in the vertical velocity creates a 45° angle below the horizontal plane of the UAV which is the start of the descent to the landing pad.

States ’Vertical Descent’ and ’Descent w/P Control’, described by Equation 2.5 and Equation 2.6, respectively, are both vertical descent.

$$\begin{aligned} v_x &= \cos(\gamma) \cdot \frac{\pi/2 - \varphi}{\pi/2} \\ v_y &= \sin(\gamma) \cdot \frac{\pi/2 - \varphi}{\pi/2} \\ v_z &= -v_{norm} \end{aligned} \tag{2.5}$$

$$\begin{aligned} v_x &= \cos(\gamma) \cdot \frac{\pi/2 - \varphi}{\pi/2} \\ v_y &= \sin(\gamma) \cdot \frac{\pi/2 - \varphi}{\pi/2} \\ v_z &= -2 \cdot v_{norm} \cdot h \end{aligned} \tag{2.6}$$

They both include horizontal velocity pointing towards the vertical axis proportional to the difference between 90° and the elevation angle. The only difference between these two is that in state ’Descent w/P Control’ the vertical velocity is proportional to the height, h, as detected by the LiDAR rangefinder. Lastly, state ’Landed’ is just to land, or set all velocities to zero and disarm the UAV, which stops the rotors spinning and ends the flight.

2.3 Landing in Problematic Situations

Landing a tether-guided UAV can easily go awry in several scenarios. The scenarios focused on within this research are a loose tether (a failed winching mechanism), an obstacle bending the tether, and a moving landing pad/target. Each of these represents a scenario where the sensor data available can no longer be relied on in the same way as before and so a new approach has to be followed, not only to attempt a landing but to detect these issues. This means the tether's azimuth angles, elevation angles, and height can no longer solely be used to determine the state and follow the normal operation state machine shown in Figure 2.5. Instead, one of the alternative states of the overall state machine shown in Figure 2.6 will be used.

2.3.1 Loose Tether

To detect a loose tether scenario, the tension calculation by the winching system will be used. The winching system requires a tether tension set-point, which is in place for other UAV operations that require much more precision and less disturbance by the tether. Landing is aided by the force of the tether, therefore this set point can be very high. Whenever the tension is detected to be significantly lower than this set point, it is considered a 'Loose Tether' case. This case can only be exited once the tether is detected to be taut again. Once in a 'Loose Tether' case, the most likely cause of the tether being loose is a temporary system fault and the tether will tighten up within a few seconds. Such a fault could occur because tether is computationally slower than the UAV and it needs to catch up. To fix this, all that has to be done is to wait, so the first action taken in the 'Loose Tether' case is to ramp the speed down to zero and pause until a timer runs out. When this timer finishes, some other action should be taken to continue the landing process.

If it is assumed that finding the landing pad is impossible, the UAV could just land on whatever ground is below it and let the system know that it failed. However, with the use of the height sensor onboard the UAV, it is conceivable that the landing pad can be found. If the landing pad is high off the ground (as it is in this scenario), a discontinuity with the previous values of the height could indicate that the landing pad is beneath the UAV, this can be seen in Figure 2.10. This is done by storing a vector of the previous height values, or the height history, and comparing the difference of each adjacent pair against a predetermined threshold. This threshold has to be similar to, but slightly less than, the height of the landing pad from the ground. However, the UAV first has to know which direction to fly in to achieve a height discontinuity.

To achieve some direction, several methods can be used. First, the UAV could use the last known direction of the tether when it was taut. This is imprecise due to any small perturbations that will rotate the UAV from the last heading while the tether was taut. To fix this, the UAV could fly upwards, or in the opposite direction that the tether last pointed, in attempts to tighten

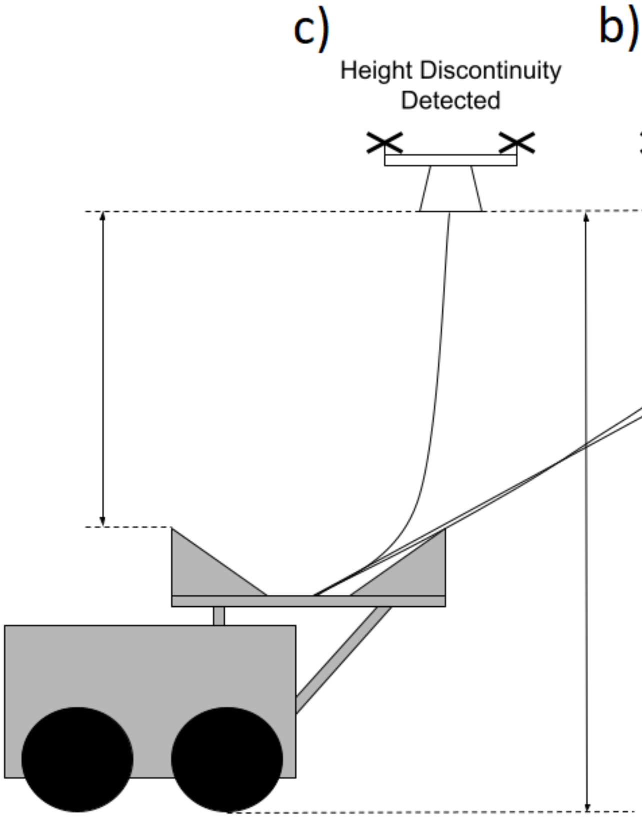


Figure 2.10: Illustration of the process for the loose tether case. Notice this process is the UAV moving from a) to b) and from b) to c).

the tether just to the point of giving a more accurate direction. Another useful method of improving the chances of finding the landing pad is to fly in a searching pattern pointed in the direction of the last known taut tether angle. All of these methods used in conjunction would greatly increase the chances of finding the landing pad.

Once a height discontinuity is detected, the UAV would fly a few more centimeters in the same direction, to ensure it doesn't land on the edge of the pad, and then descend. Beyond the possibility of landing on the edge of the pad, the design of the pad is used to ensure that the UAV lands precisely. If a height discontinuity is never detected after some period, the UAV should finally land in place and message the system of the failure and that retrieval is needed.

One significant issue with height detection is reliance on one source of the height measurement. If the UAV suddenly descends more than the discontinuity threshold, possibly due to drafts or minor control failures, a height discontinuity will be detected. In other words, the height discontinuity is reliant on the height of the drone from the ground or the altitude of the drone being constant. One way to circumvent this would be to add an altitude sensor or even a second point LiDAR pointing toward the ceiling. If both of these sensors detect a significant change in the height, then it's more likely that the drone has descended instead of the landing pad being detected. If only the downward-pointing LiDAR detects a significant change in the height, it's more likely that the landing pad is being detected.

2.3.2 Obstacle

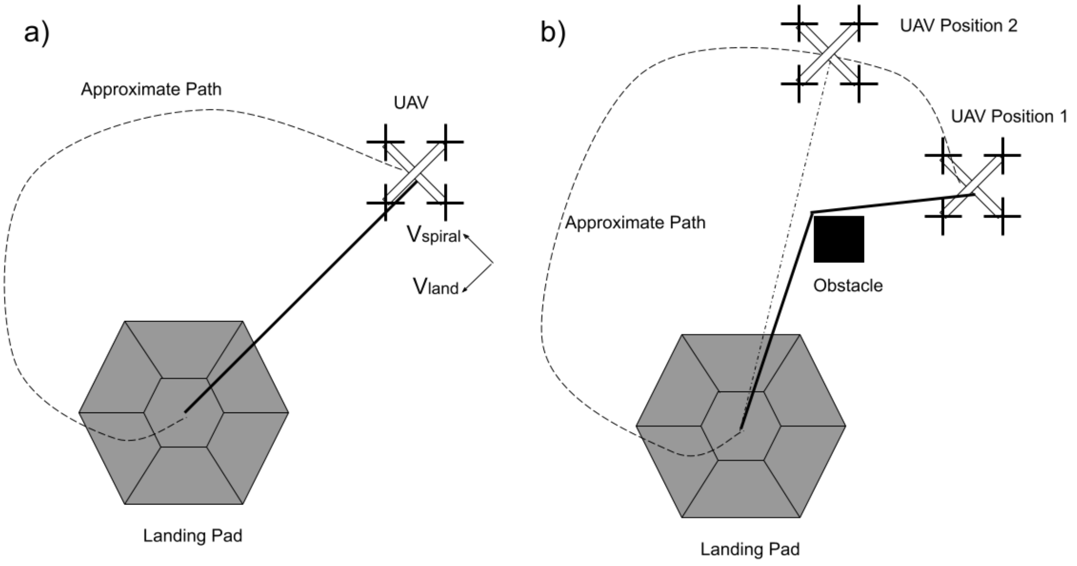


Figure 2.11: Illustration of the flight path for the obstacle case where; a) no obstacle bends the tether and the UAV spirals around the landing pad to vary the rate of change of the platform azimuth angle from zero, and b) when an obstacle is bending the tether and the rate of change of the platform azimuth angle is zero.

The obstacle case is when a pillar or some other obstruction causes the tether to bend. When the tether is bent, the rate of change of the azimuth angle on the rover side should be zero, or very near zero allowing for noise from the sensor. This can be used to detect that an obstacle is bending the tether, with the requirement that the UAV does not follow the tether in a straight path to the landing pad. Instead, this method introduces a slight spiral approach around the landing pad. This spiral gives the azimuth angles, particularly the azimuth angle at the rover, a significant rate of change. The UAV will fly with some velocity tangent to the tether which, when the tether is bent by an obstacle, causes a change to the azimuth angle of the UAV only. This is depicted in Figure 2.11. All measurements are subjected to some amount of noise, due to imperfections in sensors and therefore the angular velocity should be taken by applying some smoothing function to the previous values, or history, of the azimuth angle. Equation 2.7 shows that smoothing is achieved by taking the average of the first and second halves of a recent history of the azimuth angle and the associated time of each.

$$\begin{gathered}
\gamma_{h1} = \text{average}\left(\begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_{n/2} \end{bmatrix}\right), \gamma_{h2} = \text{average}\left(\begin{bmatrix} \gamma_{n/2+1} \\ \gamma_{n/2+2} \\ \vdots \\ \gamma_n \end{bmatrix}\right) \\
t_{h1} = \text{average}\left(\begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_{n/2} \end{bmatrix}\right), t_{h2} = \text{average}\left(\begin{bmatrix} t_{n/2+1} \\ t_{n/2+2} \\ \vdots \\ t_n \end{bmatrix}\right) \\
\dot{\gamma} = \frac{\gamma_{h2} - \gamma_{h1}}{t_{h2} - t_{h1}}
\end{gathered} \tag{2.7}$$

Once the UAV crosses the obstacle, it can once again continue the spiral pattern flight to land. This poses a potential issue as to direction. In several situations, the direction of velocity tangent to the tether can either further tangle around an obstacle or bend the tether in the first place. From figure 2.11b), if the spiral is reversed then the UAV would be flying from UAV Position 2 to Position 1 causing a bend in the tether. Further, once in Position 1, the UAV needs to know which tangent to follow to straighten the tether. This can be done, again, by using the rate of change of the azimuth angle at the platform. As seen in Figure 2.12, if the azimuth angle at the platform was increasing, or the rate of change was positive before the obstacle was crossed, the obstacle was crossed on the left side. A negative rate of change before the obstacle was crossed indicates that the obstacle was crossed on the right side. If the spiral direction of flight causes the UAV to cross an obstacle as is seen in Figure 2.12, then the spiral will be reversed to avoid the obstacle going forward.

This solution, however, relies on the fact that obstacles are crossed in the process of landing. Once the landing phase begins, there is no way of knowing what the rate of change of the azimuth angle at the platform was before the landing state machine takes over. The approach if this situation occurs is to choose a tangent direction arbitrarily and follow that tangent until one of the following occurs. Either the obstacle is cleared, or the UAV flies in that tangent direction far enough that it can be reasonably assumed that the obstacle should have already been cleared. In the latter case, the tangent direction is reversed and the UAV flies until the obstacle is finally cleared. The distance necessary to determine that the obstacle should have been cleared can be determined from the azimuth angle at the UAV. If this angle is approaching the same value that it was the moment

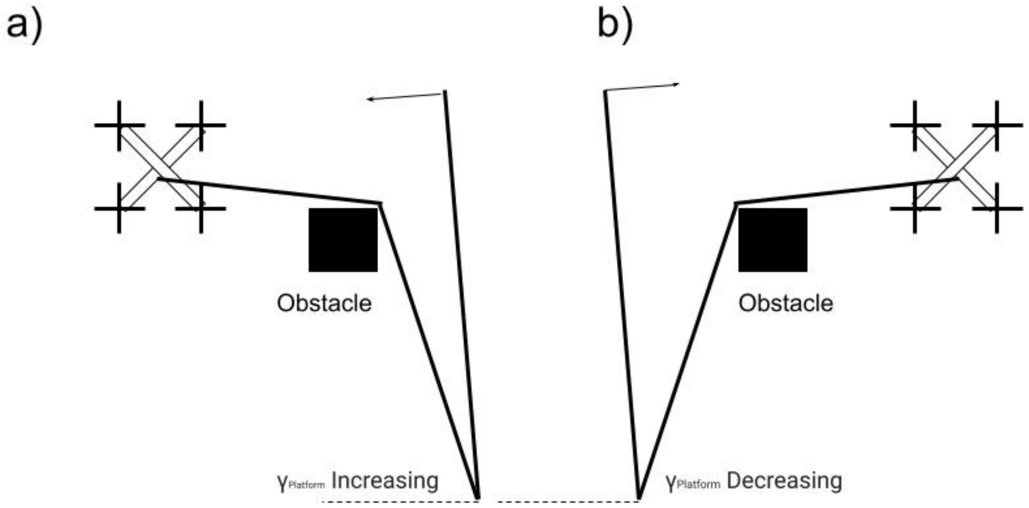

a)
b)
Obstacle
Obstacle
$\gamma_{Platform}$ Increasing
$\gamma_{Platform}$ Decreasing

Figure 2.12: Illustration of the issue regarding which direction the UAV crosses an obstacle.In a) the obstacle is crossed on the left, in b) the obstacle is crossed on the right.

that tangent flight began, then the UAV will have flown in a full circle around the obstacle without clearing it. This is subjected to perturbations in the yaw angle of the UAV. These can be assumed to be small, however, because the landing state machine has control authority and the yaw angle is never changed.

2.3.3 Moving Platform

A moving platform case is similar to the obstacle case in its detection but instead of using the rate of change of the azimuth angles, it uses the rate of change of the elevation angles. As can be seen in Figure 2.13, the movement of the platform causes the elevation angle to decrease or causes a negative rate of change in the elevation angle. When a negative elevation angle rate is detected, additional velocity is added to those that make up the horizontal plane to compensate. This additional velocity is proportional to the change of the elevation angle via Equation 2.8, where h is the UAV height, and $\varphi$is the elevation angle. This only occurs if the platform is moving against the UAV. If the platform is moving laterally to the drone, the 'Normal Operation' case will still direct the UAV to the landing pad.

$$v_{\text{tangent}} = h\left(\varphi_1 - \varphi_2\right) \tag{2.8}$$

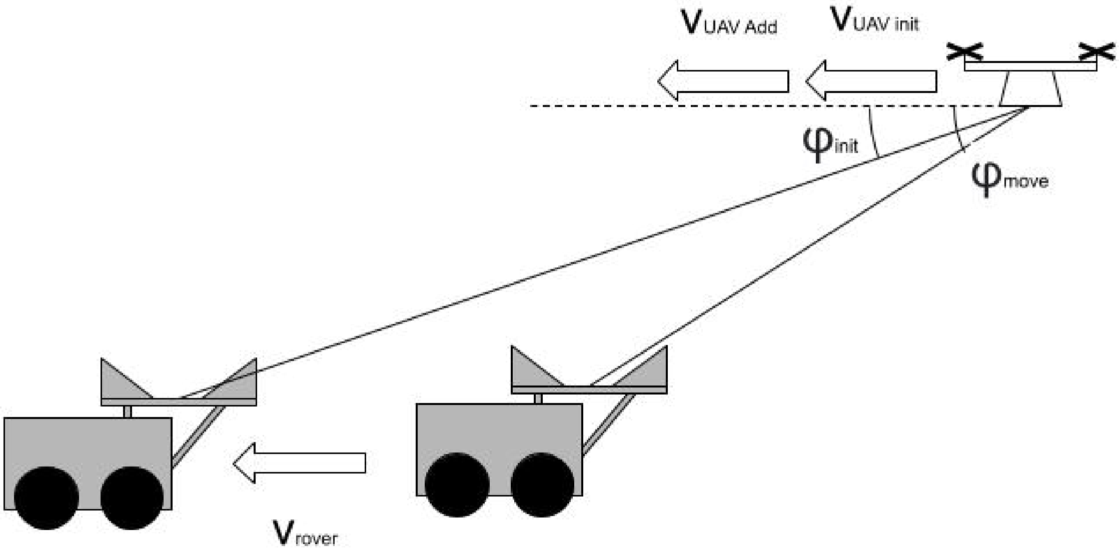
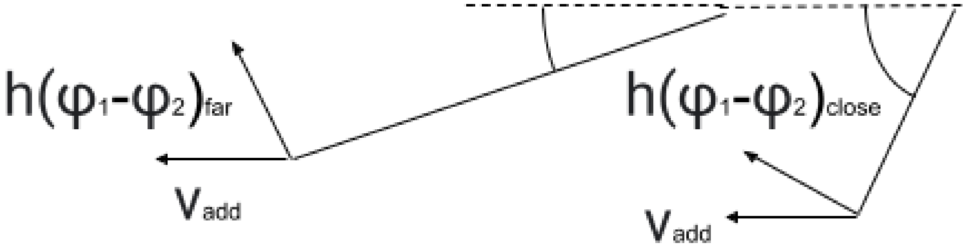


Figure 2.13: Illustration of the flight approach for the moving platform case. The UGV drives away from the UAV (from right to left) and the UAV has to increase speed to catch up and be able to land.

It's important to note that the actual velocity given by this equation is the velocity tangent to the tether. Therefore, it is only accurate, to compensate directly for the platform velocity, when the tether is vertical. However, the portion of this velocity that goes towards extending the tether is much lower at close angles and so can be neglected and allowed horizontally. The same cannot be said at far away angles, however, a higher speed at these angles can be used to more quickly overcome the movement of the platform. This phenomenon is visualized in Figure 2.14.



$h(\varphi_1-\varphi_2)_{far}$
$h(\varphi_1-\varphi_2)_{close}$
V_{add}
V_{add}

Figure 2.14: Illustration of the tangent velocity phenomena. Where h is the measured height of the UAV, $\varphi$is the elevation angle, and v_{add} is the additional speed calculated by Equation 2.8. Notice that Equation 2.8_{far} has more of a vertical component than Equation 2.8_{close}

A moving platform case can also be initiated by a zero elevation angle rate of change, meaning

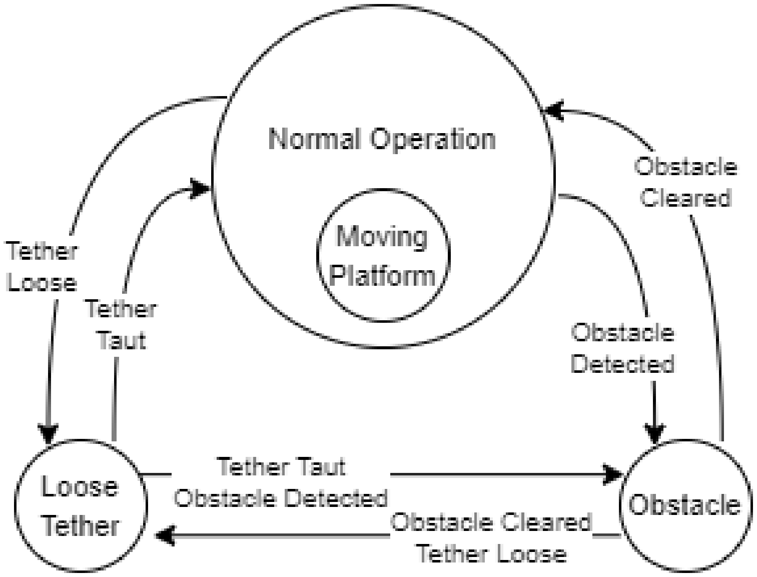


Figure 2.15: Amended overall state machine; Moving Platform case moved inside the 'Normal Operation' case to more accurately depict it's intended implementation.

the platform and the UAV are moving at the same speed. This is significant because this would mean zero additional velocity is added to the velocity command, according to Equation 2.8, and the UAV would be perpetually stuck in whichever landing phase it was last in. To counter this, the expected elevation angle rate of change should be compared with the calculated rate. Where the expected rate is calculated from the base velocity using the assumption that the rover isn't moving and the UAV is approaching the landing platform as normal. Any significant deviation initiates the Moving Platform case. This is also useful as the platform could be moving but slower than the UAV, causing the UAV's approach to take more time and decrease efficiency.

The moving platform case is special when compared with the other cases because it truly takes place alongside the Normal Operation case. It cannot operate at the same time as the Loose Tether case, because a loose tether renders elevation angles useless. In this case, if the platform were moving, it may aid as while the platform moves away, the tether would eventually tighten. In the other case, the 'Moving Platform' can operate at the same time as the Obstacle Case, however, the additional velocity would go tangent to the tether and speed up the process of crossing the obstacle. This may cause instability and unnecessary perturbations. It is therefore safer to allow the Obstacle case to operate on its own. In addition, if the tether is bent around an obstacle, the contact friction would make Moving Platform detection much more difficult.

In summation, the Moving Platform case operates in tandem with the Normal Operation. Therefore, it is important to note an amendment to the earlier proposed state machine diagram, shown in Figure 2.15.

Chapter 3

Results

3.1 Testing Conditions

To test the methodology proposed in this thesis, the ’Normal Operation’ case was first implemented using ROS in a Gazebo simulation environment to ensure control was successful before using a real UAV. The code was structured as a series of if-statements checking the UAV’s current state for the conditions shown in Figure 2.5. Once the current state has been categorized into a sub-state of ’Normal Operation’, the appropriate landing action is then sent to the UAV as a velocity command. For the simulation, a separate ROS node was used to calculate the tether information from the position of the UAV with respect to the origin of the simulation world (assuming that the origin is

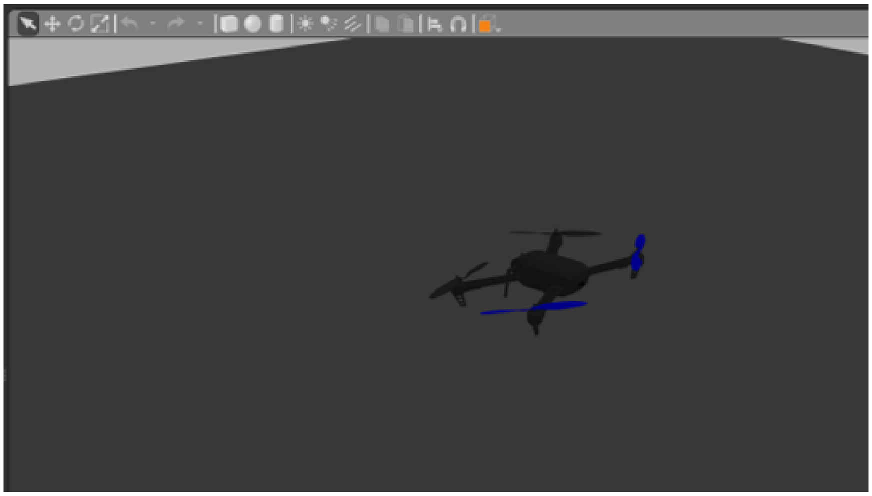


Figure 3.1: Simulation flight performed in Gazebo. To achieve landing a node sends velocity commands to the UAV. Another node simulates the tether information. In this simulation, only the ’Normal Operation’ case was tested.

the location of the landing pad. This setup was not used to perform tests for the 'Loose Tether', 'Obstacle', and 'Moving Platform' cases due to the complexity of simulating an actual tether. The simulation environment can be seen in Figure 3.1.

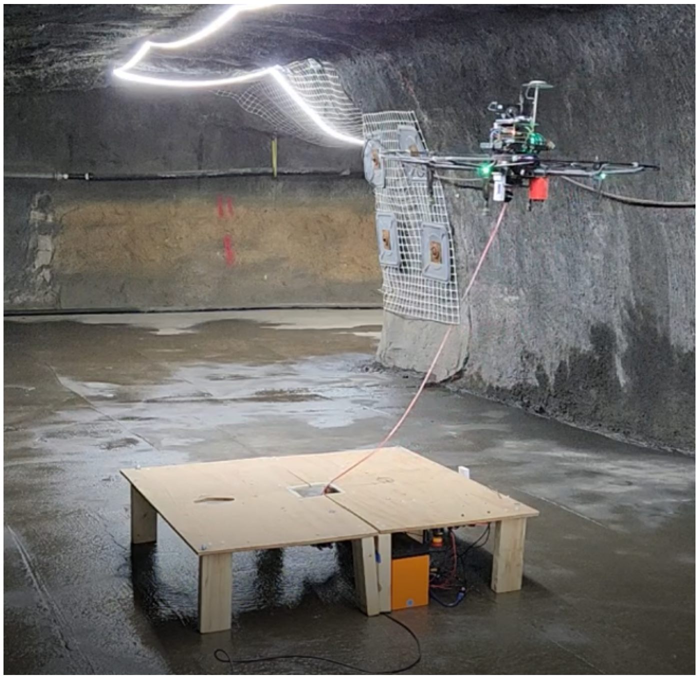


Figure 3.2: Image taken from testing of this project in NIOSH's experimental coal mining in Pittsburgh. This test was performed using a Matrice 100 UAV and a square wooden landing platform integrated with the tether angle sensor and winching mechanism.

Once simulation testing on the 'Normal Operation' case was successful, hardware testing commenced. The hardware used to test was a DJI Matrice 100 UAV and a square wooden platform assembled with the tether angle sensor. The Matrice 100 ran by battery operation so the power tether was simulated with a cord winched by a spring loaded mechanism. This test system did not make use of the landing platform and landing gear designed in this thesis. Initially, the UAV was set on a table with the rotors disabled and testing was performed by manually moving the joystick to check the expected velocity commands were being output. Once any adjustments were made to get the expected output, the UAV was set on the platform and testing was performed with flight. This testing set up can be seen in Figure 3.2, except most tests were performed in a laboratory, not in a mine. Following successful testing with this setup, the intended UAV with the designed landing gear and landing pad was then used for testing, as shown in Figure 3.3, which shows the progression of a successful landing.

To test the 'Loose Tether' case, the Matrice 100 and wooden platform setup was utilized with the addition of the intended winching mechanism because the tether tension measurement was needed.

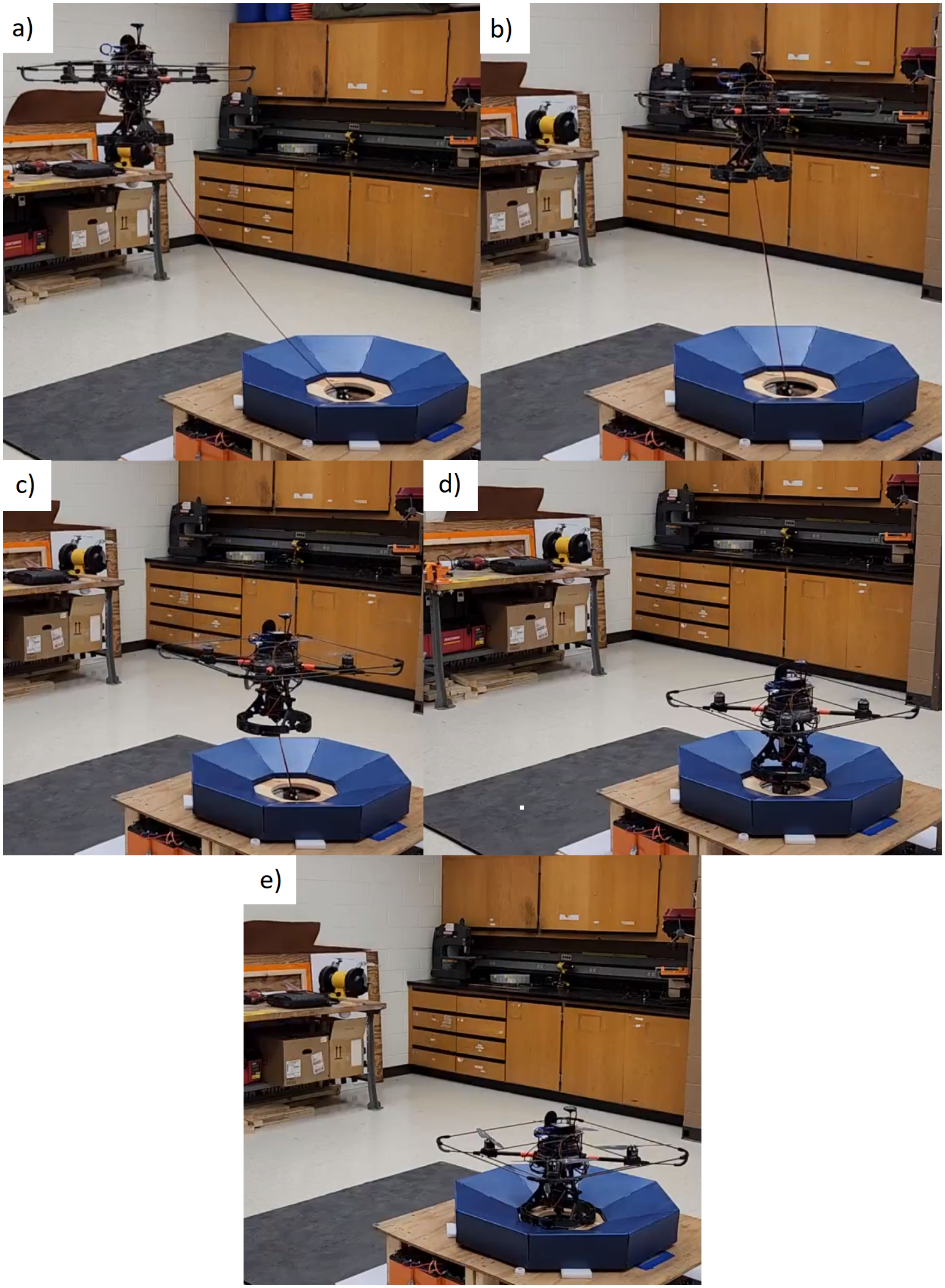


Figure 3.3: 'Normal Operation' case test flight performed with the intended landing gear and landing platform where the UAV is in the sub-state a) 'XY Approach', b) '45° Descent',c) 'Vertical Descent', d) 'Descent with Proportional Control', and e) the UAV has landed.

As before, this was first tested with the UAV on a table with the rotors disabled. For this test, the primary detection concern is the tether tension, so the tether was manually released to decrease the tension until the 'Loose Tether' case was detected. The UAV was then moved so the height sensor was detecting the floor (not the table) and the UAV was pushed back into place until the table height was again detected, resulting in a height discontinuity and completion of the 'Loose Tether' case. Once this was repeatedly successful, the testing was once again performed in flight. In order to keep the tension low enough for detection, the set-point for the tension of the winching mechanism was set to approximately 1.5 Newtons. This meant that the tether would always be loose and that the UAV would never enter 'Normal Operation' to detect a 'last known' direction. To achieve this, the UAV was flown manually against the tether to temporarily increase tension. As soon as the tension was high enough to get a 'last known' direction, the UAV was switched to autonomous mode, the 'Loose Tether case took over, and the test proceeded. The process for testing the 'Loose Tether' case can be seen in Figure 3.4. This case was never flown with the intended landing platform and landing gear.

The 'Obstacle' case was never flown. It was only ever simulated with a hardware setup. This was also performed with the Matrice 100 and wooden platform. The UAV was hung on strings connected to the platform with the cord and spring loaded winch mechanism. The platform had an angle sensor incorporated. On the ceiling where the UAV hung from was a greased track for the UAV to slide along in order for the azimuth angles to be changing (or not) and to detect an obstacle. On the platform was placed a vertical obstacle that would cause the tether to be stopped from the platforms side about halfway along the UAV's movement. This setup can be seen in Figure 3.5. To start the test, the UAV was swung back and forth in the same position on the track to cause change in the azimuth angles, then the code would be initiated so that the rotors were disabled. The UAV would then be pulled along the track until it hit the end and the UAV would again be swung back and forth in the end position on the track to cause the UAV's azimuth angle to change. As this test was never fully flown, it was also never performed with the intended landing platform and landing gear.

The last tests performed were to test the UAV landing on the platform on the back of the UGV for system integration. These are very similar to the 'Normal Operation' tests with the intended landing gear and landing pad with the only difference being that the platform is now mounted on/integrated with the UGV. These tests can be seen in Figure 3.6.

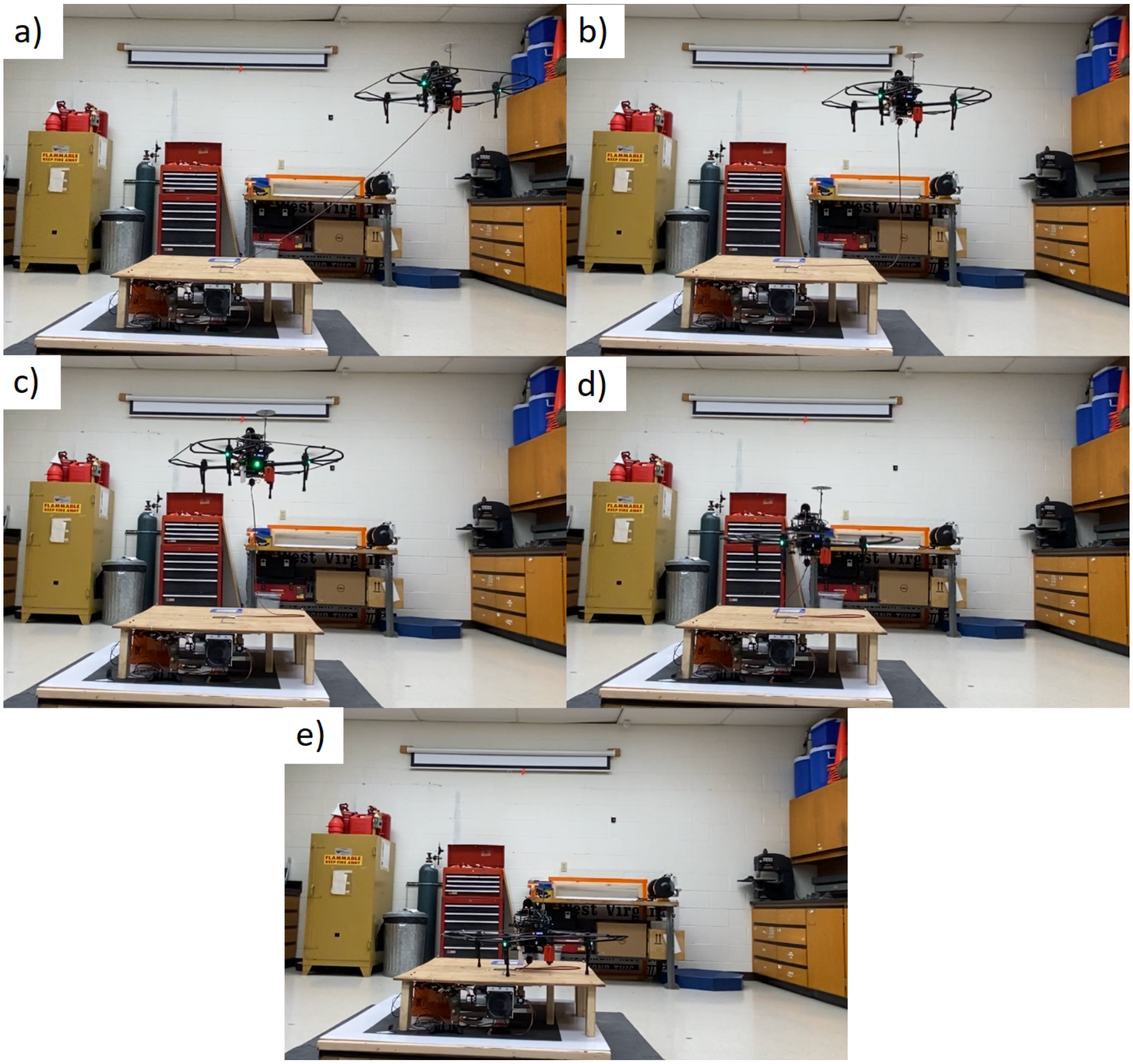


Figure 3.4: Testing the 'Loose Tether' case where a) manual flight is used to temporarily tighten the tether, b) autonomous control has taken over and the 'last known' direction is being used to find the landing platform, c) a height discontinuity has been detected indicating the landing pad has been found, d) the UAV has been commanded to auto-land and is descending, and e) the UAV successfully landed on the platform.

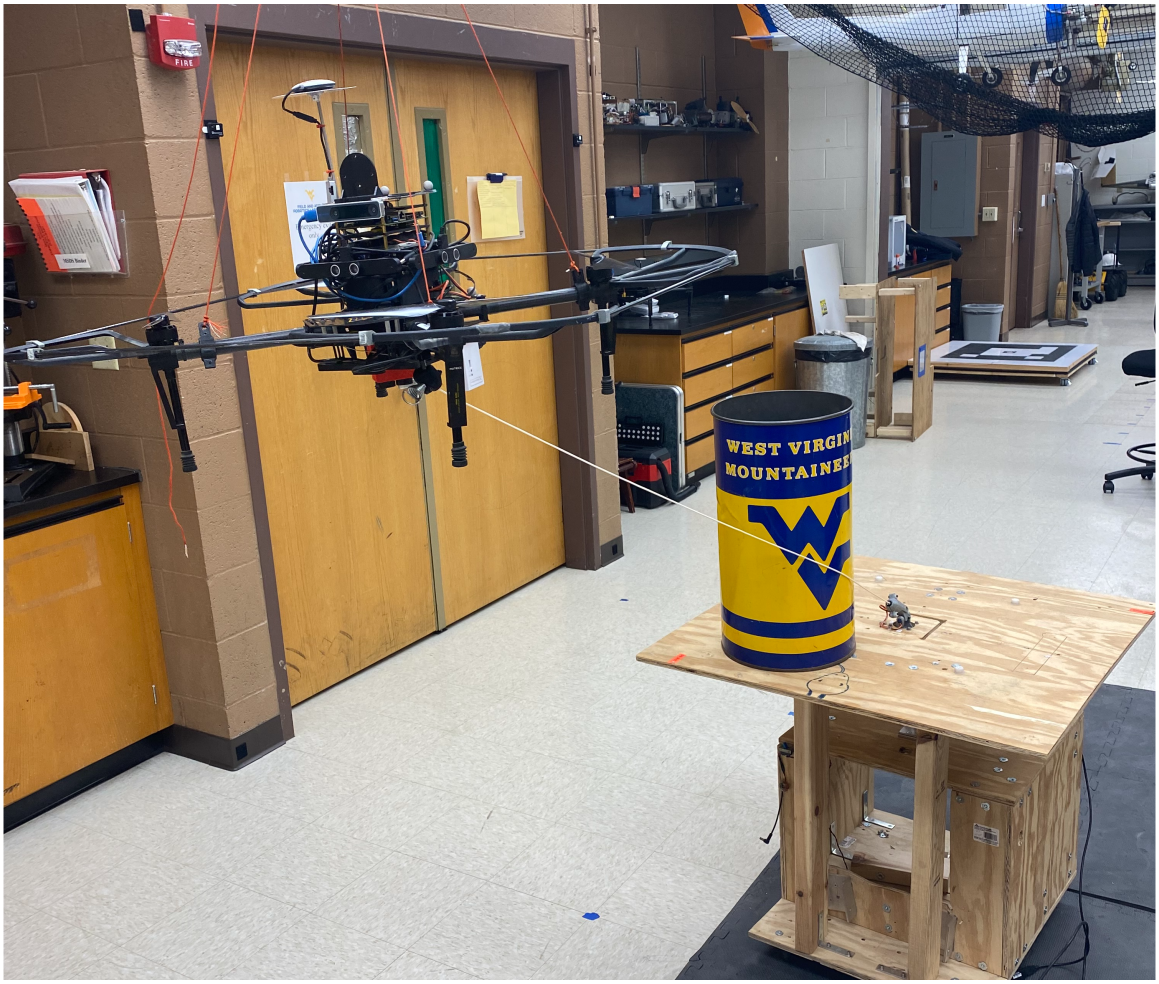


Figure 3.5: 'Obstacle' case testing setup with the Matrice 100 and wooden platform. Not seen is the greased track on the ceiling for pulling the UAV along causing change to the azimuth angles. On the wooden platform is the obstacle that stops the change in the platform azimuth angle and the 'Obstacle' case to be tripped.

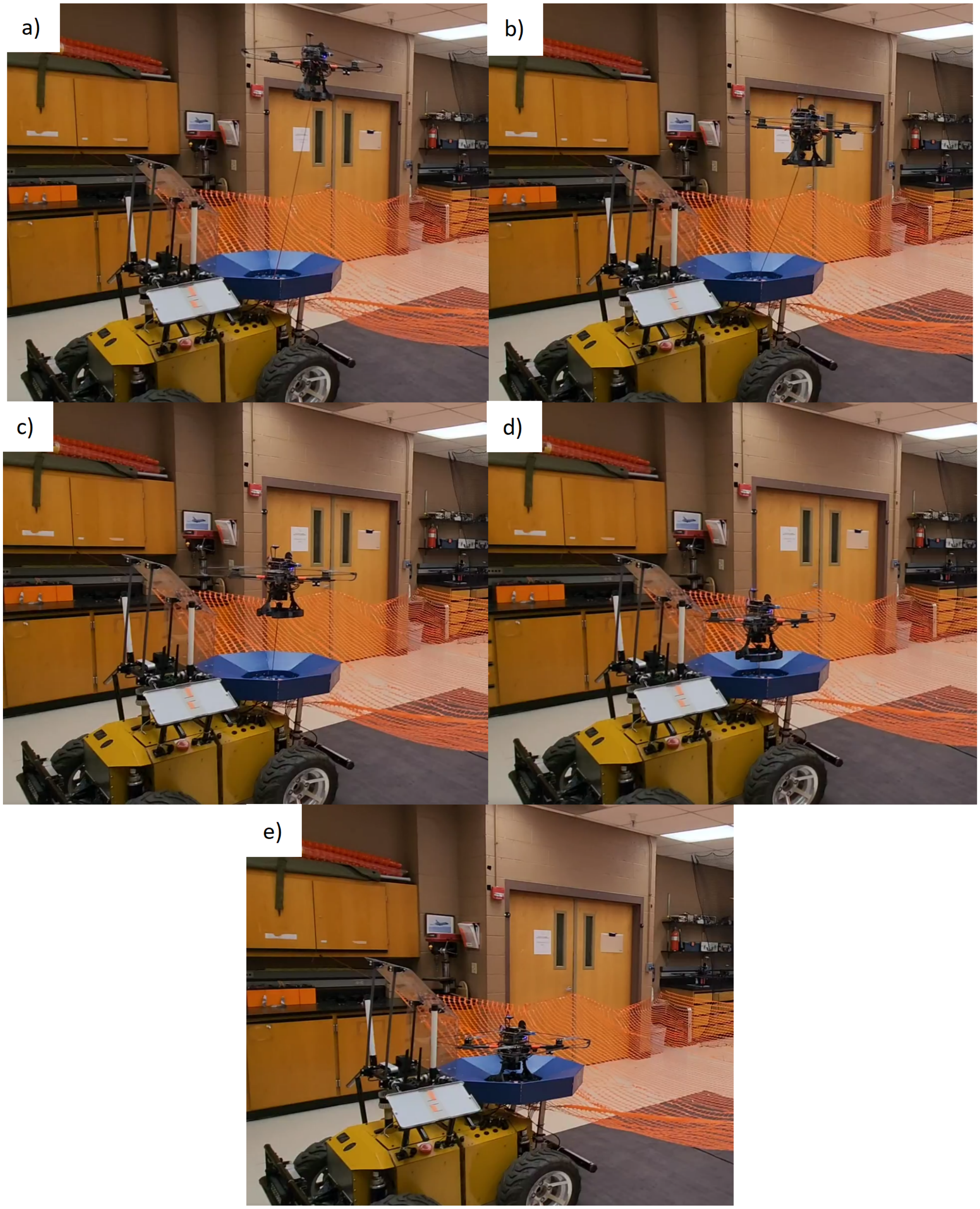


Figure 3.6: System integration testing with the UGV where the UAV is in the 'Normal Operation' sub-state a) 'XY Approach', b) '45° Descent',c) 'Vertical Descent', d) 'Descent with Proportional Control', and e) the UAV has landed.

3.2 Normal Operation

The process of landing, detailed in Figure 2.5, often followed a path of 'XY Approach', '45 °Descent', 'Vertical Descent', 'Descent w/P Control', and successfully reached 'Landed'. This is illustrated by the flight data in Figure 3.7a) and 3.8(a). A small portion of the plots may be a pre-landing phase as the states are based purely on the tether angles and don't show when landing processes properly took over. Figures 3.7b) and 3.8b) show the tether angles themselves, and Figures 3.7c) and 3.8c) show the velocity commands. These plots show the relationship between the angles, the state, and the commanded velocity through the flow of the landing process. As can be seen, the UAV proceeded through the landing states. Figure 3.7a) shows how states can be changed out of the aforementioned 'ideal' path, as it goes from '45 °Descent' to 'XY Approach' and back before entering 'Vertical Descent' and reaching the 'Landed' state.

The plots for the tether angles can be difficult to read because they can change very rapidly, however, Figure 3.8b) shows more clearly what is occurring. The blue scale is for the azimuth angles, and the red scale is for the elevation angles. In the legend, D and R refer to the drone and rover respectively, a solid line is for the drone, and a dotted line is for the rover. The rover angles are arbitrary because they do not influence the flight of the UAV in the 'Normal Operation' state, but it can be important to see how these angles interact for future cases. The azimuth angle at the UAV is also mildly arbitrary, its influence is only on the X and Y speeds, and these speeds direct the UAV toward the landing pad. It's important to notice how the UAV's elevation angle increases until it reaches above 85 degrees.

As with the tether angle plots, the commanded velocity plots can also be difficult to read due to rapid changes in the angles, however, Figure 3.8c) provides clear information. Again, X and Y are mildly arbitrary, other than to know that they point towards the landing pad. The most important thing to notice is the Z speed as it relates most closely to the elevation angle. The Z speed is constant for some time, this is '45 °Descent'. When the Z speed jumps up at the end, this is when 'Vertical Descent' and/or 'Descent w/P Control' take over, most likely solely 'Descent w/P Control' because the Z speed is decreasing with the height.

These tests were performed despite missing an important control factors, the ramp-up and down of the velocity in the z-direction. It may be difficult to see but both commanded velocity plots have instances where there is an immediate jump in the z velocity. The difference is clear in Figure 3.8c) at about 25 seconds, the x velocity has a more gradual slope than other instances where the z velocity is changing. This was improved for smoother control and the missing ramps were included

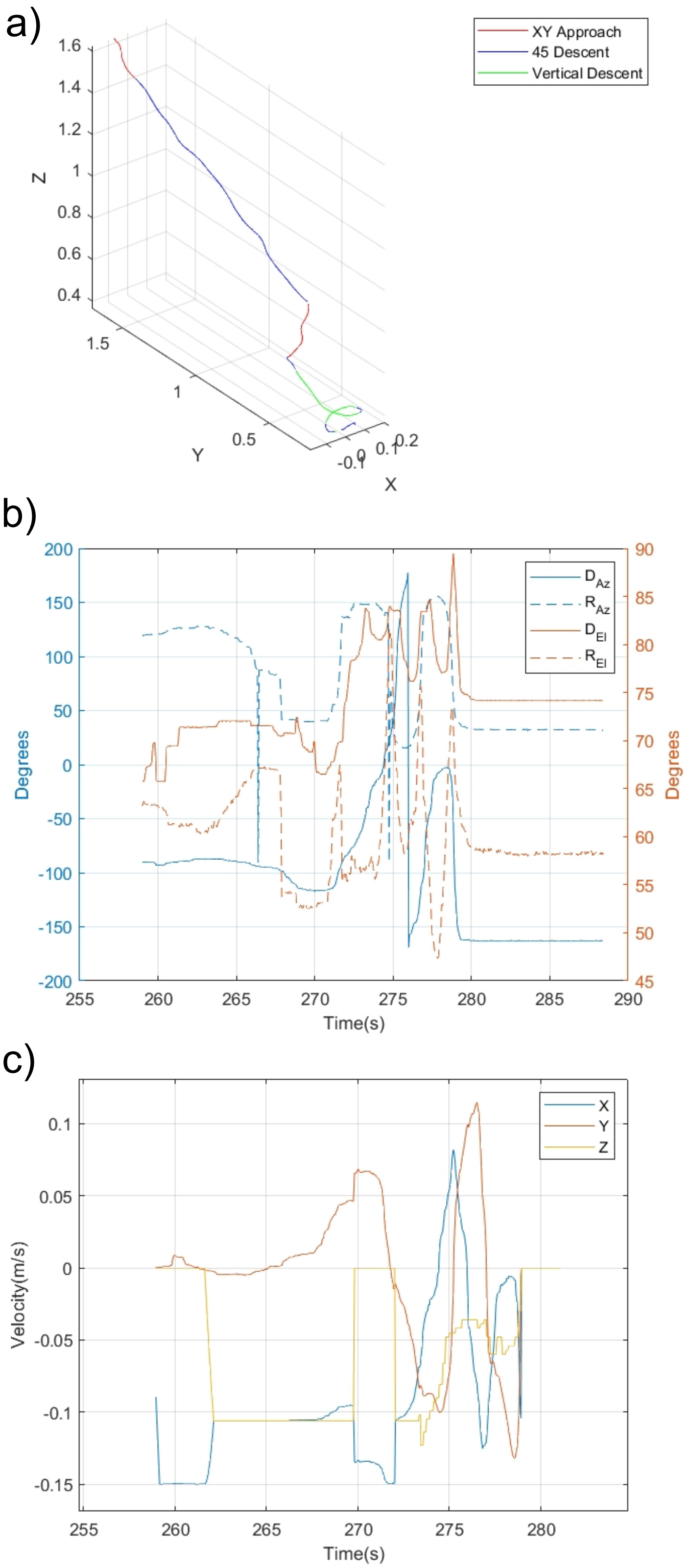


Figure 3.7: Data from a 'Normal Operation' case test, where a) is the position data, b) is the tether angle data, and c) is the commanded velocity resulting from the information in plots a) and b).

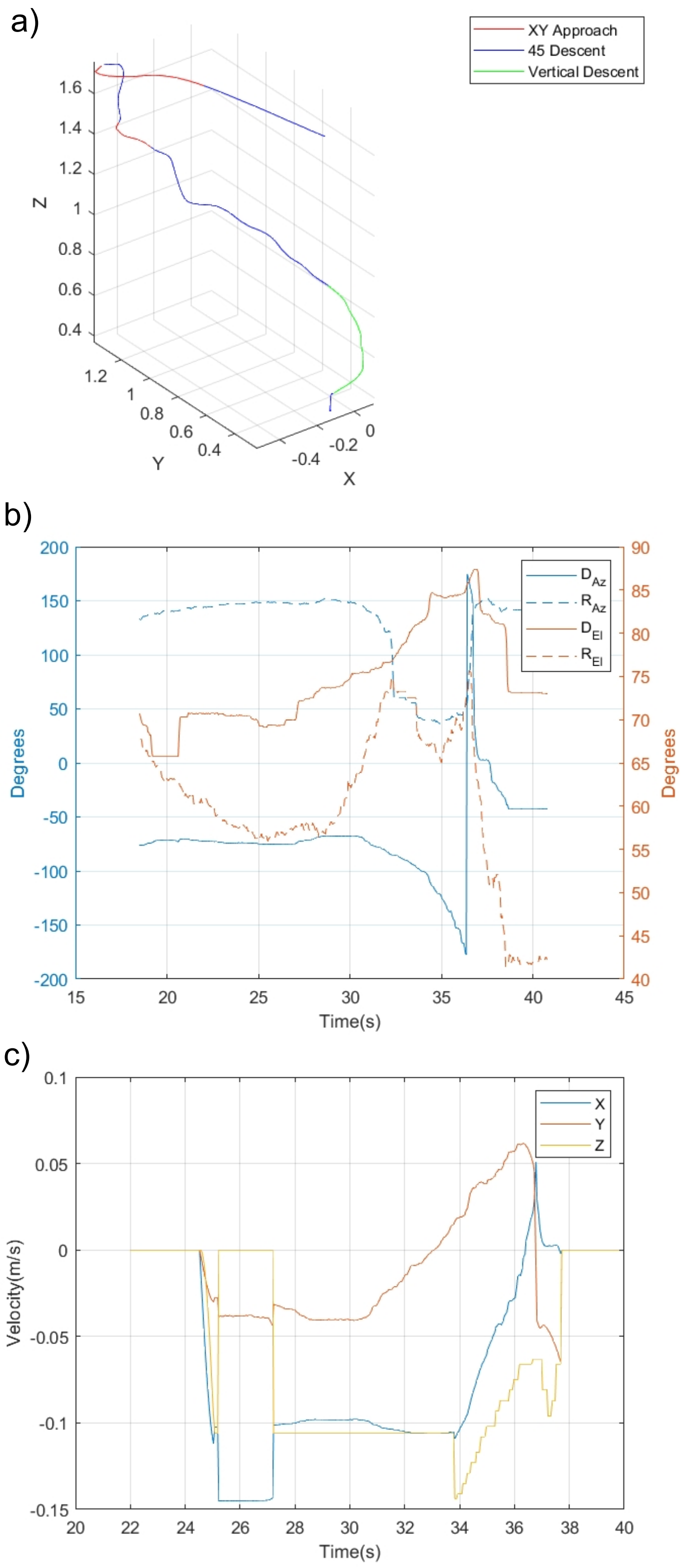


Figure 3.8: Data from another 'Normal Operation' case test, where a) is the position data, b) is the tether angle data, and c) is the commanded velocity resulting from the information in plots a) and b). The data was clearer in this test, less rapid changes in the tether angle data.

in later tests.

Examples of this improved control are seen in Figure 3.9. As can be seen particularly in plot b) at 568 seconds, the z velocity is decreasing with the same slope as the x and y velocities. Plot a) is more difficult to see because it ran for 40 seconds as opposed to the 12 seconds in plot b), but the slopes look very similar between the beginning and the z velocity increasing at 210 seconds.

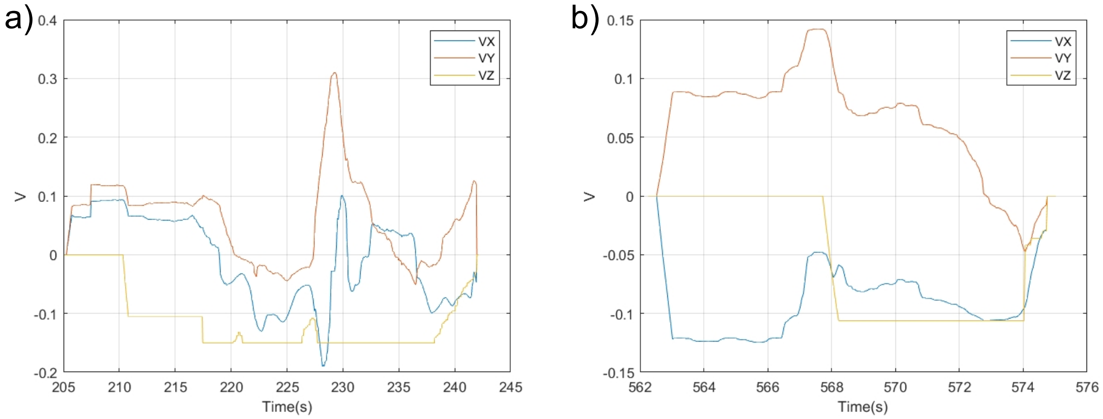


Figure 3.9: Plots of the velocity commands from two test flights that included improved velocity ramping.

3.3 Loose Tether

Figures 3.10, 3.11, 3.12, and 3.13 represent four successful tests of the loose tether algorithm. First, the position plots are shown in plot (a) of the figures, the velocity commands are shown in plot (b), and a sub-state tracking number is shown in plot (c). The position data begins where the tether is tightened to achieve direction towards the landing pad. The red asterisk on the plot shows where a height discontinuity is detected and the UAV was set to auto-land. The rest of the position data is just auto-landing and takes place after the loose tether algorithm has finished.

As can be seen from the position data, the tests in Figures 3.10, 3.11, and 3.12 all detected the landing pad and landed while the test in Figure 3.13 detected the landing pad and kept moving. This fault is because the height discontinuity was detected on the corner of the platform. Manual control had to be taken over to ensure the UAV did not try to land when it was not directly over the platform causing damage to the UAV. This issue will be discussed later in the Future Work section.

Another point in the data that may look incorrect is in the velocity, (b), plots where all the velocities drop to zero only to increase to the same or nearly the same velocity as before. This is due to the testing constraints. In the beginning, the tether is loose and the last velocity achieved from

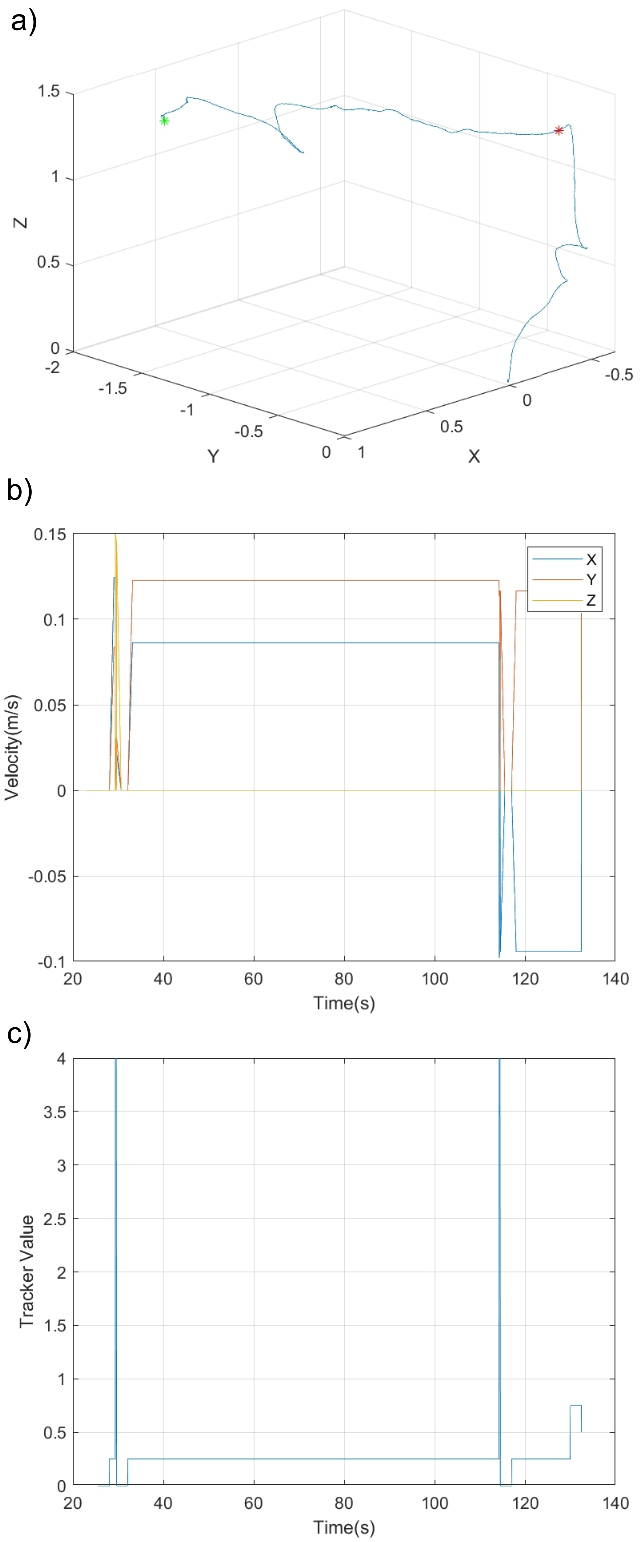


Figure 3.10: All data from the first of several tests of the 'Loose Tether' case, where a) is the position data, b) is the commanded velocity data, and c) is a sub-state tracker with 0 being the 'hiccup' pause, 0.25 is flying in the last known tether direction, 0.75 is when a height discontinuity has been detected, 0.5 is auto-landing, and 3 4 or 5 are the normal operation states.

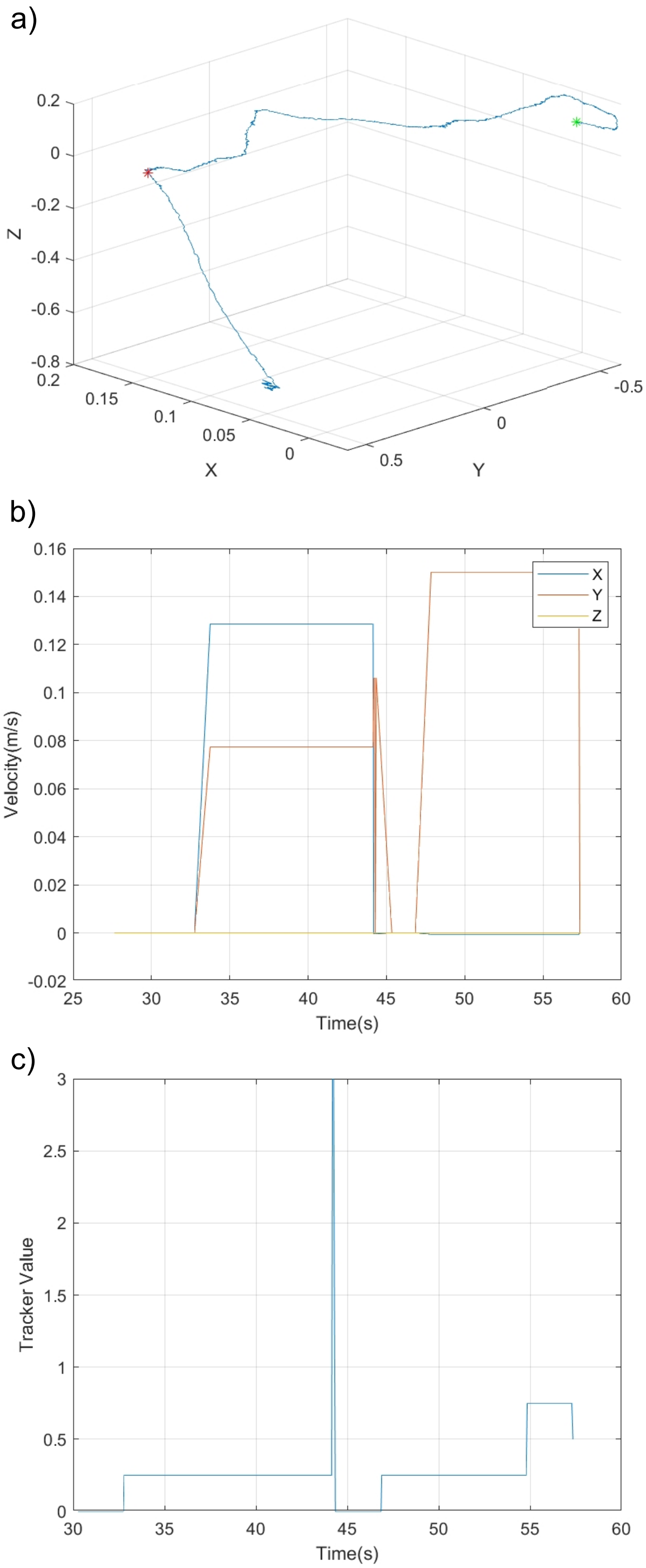


Figure 3.11: All data from a second of several tests of the 'Loose Tether' case, where the plots represent the same as the plots from Figure 3.10.

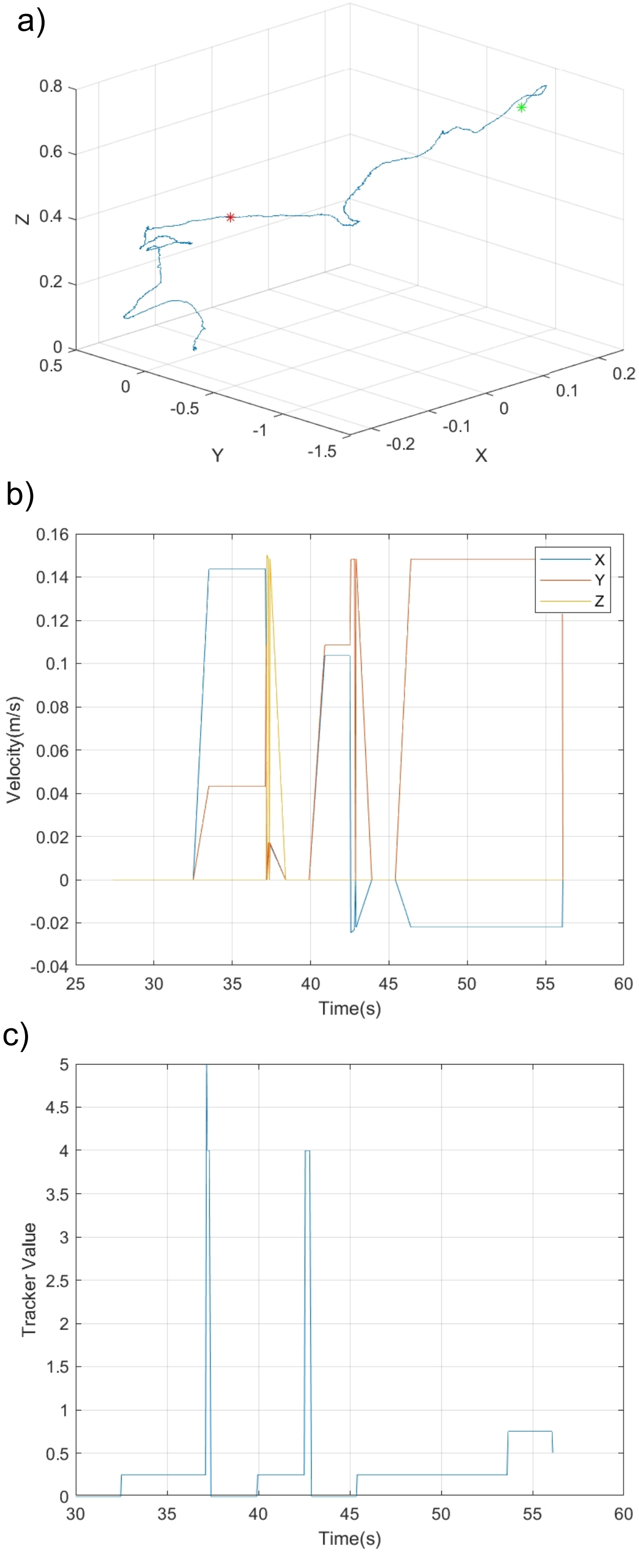


Figure 3.12: All data from a third of several tests of the 'Loose Tether' case, where the plots represent the same as the plots from Figure 3.10.

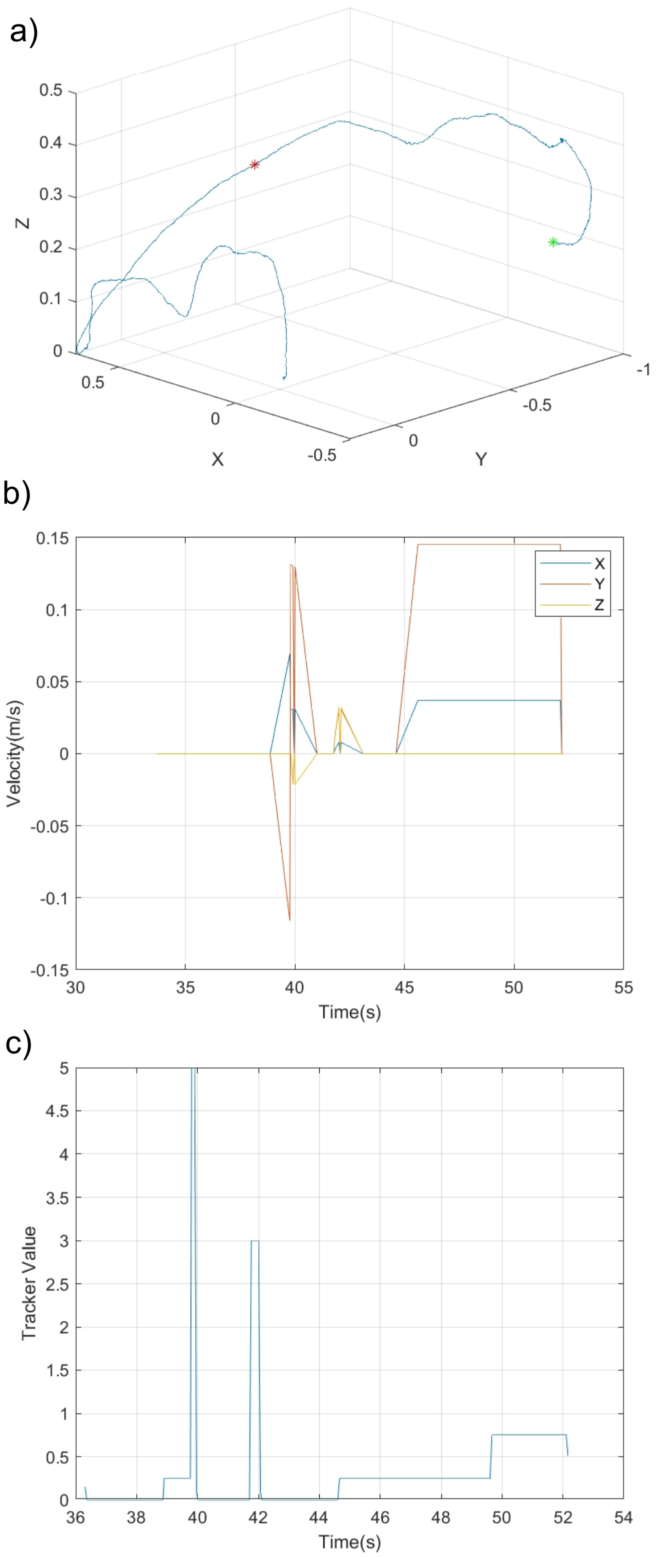


Figure 3.13: All data from a fourth of several tests of the 'Loose Tether' case, where the plots represent the same as the plots from Figure 3.10. This test was less successful because the height discontinuity detected was the corner of the landing platform.

the angle sensor is sent continually. This continues until the tether is tightened by manual flight and then immediately loosened again (to achieve a more accurate 'last known' direction), then the velocity drops to zero. This occurs because of the pause for temporary system faults. This pause was set to a short period to avoid unnecessary time in which conditions may change. After this pause, the speed ramps back up to the 'last known' direction and is continuously sent until a height discontinuity is detected, then zero speed is sent and the UAV is commanded to auto-land. This is depicted in each of the commanded velocity plots.

The sub-state tracking number is a floating point number that was added for these tests to track the sub-state that the UAV is in at any given moment. The value changes as the sequence of the 'Loose Tether' case progresses. A value of 0 is the pause for temporary system faults (very short for testing), 0.25 is sending velocity in the horizontal direction last known from the tether, 0.75 is for when a height discontinuity has been detecting and repeating that velocity to allow the UAV to move over the landing pad, and 0.5 is auto-landing. Any values higher than this (most often 3, 4, or 5 are normal operation states when the tether is tightened.

3.4 Obstacle Case

The obstacle case presented optimistically mixed results. This case was tested only in very simulated constraints, with the UAV hung from a track on the ceiling where it could move side to side with the platform nearby. The track that the UAV was pulled along was perpendicular to the platform so that it was not approaching the platform in its movement. Between the platform and the UAV

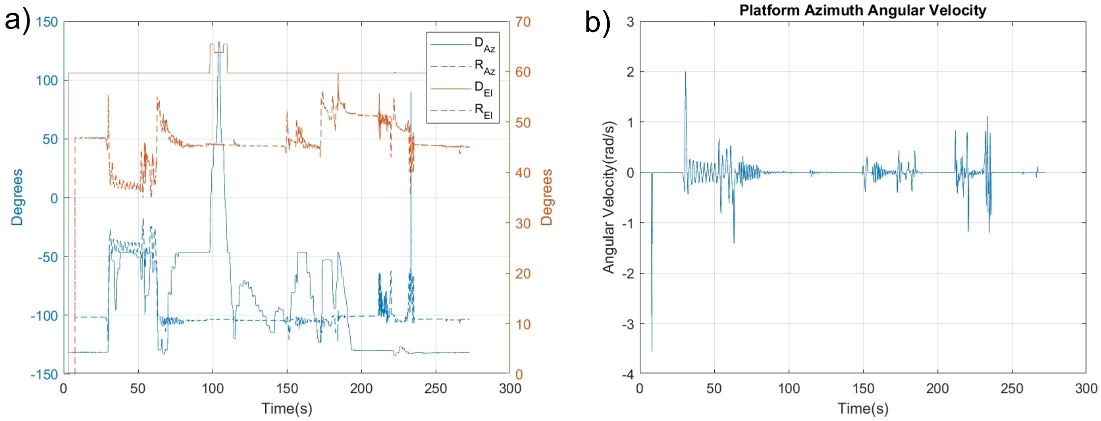


Figure 3.14: Plot a) of the angle information gathered from the tether, where a solid line is for the UAV, a dotted line is for the platform, a blue line is an azimuth angle, and an orange line is for an elevation angle and b) of the corresponding platform azimuth angular velocity.

was an obstacle that would bend the tether at about the halfway point of the track. To test the scenario, the UAV was pushed to wave back and forth, so the platform azimuth angle was constantly changing for a few seconds. It was then pulled along the track at about the speed expected from the spiraling motion required by this case. In the middle of this motion, the tether should be stopped by the obstacle and the 'Obstacle' case should be triggered. The tether information for one such test flight can be seen in Figure 3.14a). Unfortunately in this set up the UAV waving back and forth causes this to rapidly switch into the 'Obstacle' case and immediately back out because it's changing direction and therefore the velocity reaches zero. However, there are periods where the velocity reaches zero and remains there, these are where the obstacle has been crossed.

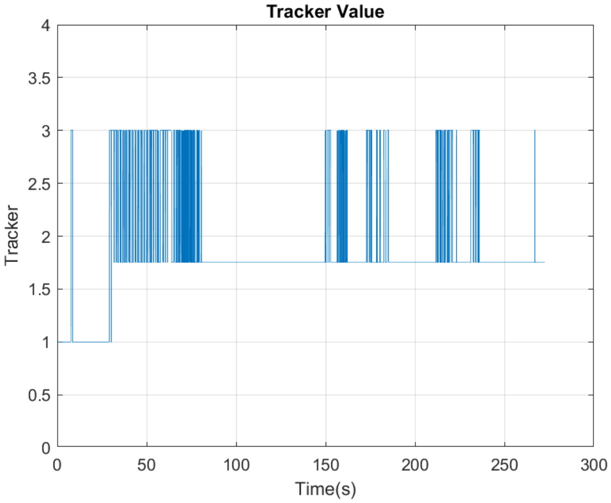


Figure 3.15: Plot of the state machine tracking value, where 3 represents the '45 °Ascent' sub-state from 'Normal Operation' and 1 represents 'Obstacle' case but no UAV azimuth change, 1.25 represents negative tangent flight, and 1.75 represents positive tangent flight.

Figure 3.14b) shows the angular velocity of the tether from the platform, it can be seen that from around 80 to 150 seconds, the angular velocity was zero, if not very near to it. The state machine tracker discussed above for other cases also ran in this case, seen in Figure 3.15, and the phenomena of incorrect obstacle case detection can be very clearly seen before 80 and after 150 seconds.

When 'Obstacle' case is detected and there is motion in the UAV, the tracker outputs a value of 1.25 or 1.75. These values represent the tangential direction that the UAV needs to be commanded to overcome the obstacle. As can be seen, the tracker always sends a value of 1.75, this is because the azimuth angle at the platform was decreasing before the obstacle was detected and positive tangent flight is being commanded to clear the obstacle.

Another similar test is shown in Figure 3.16. The corresponding tracker plot is shown in Figure

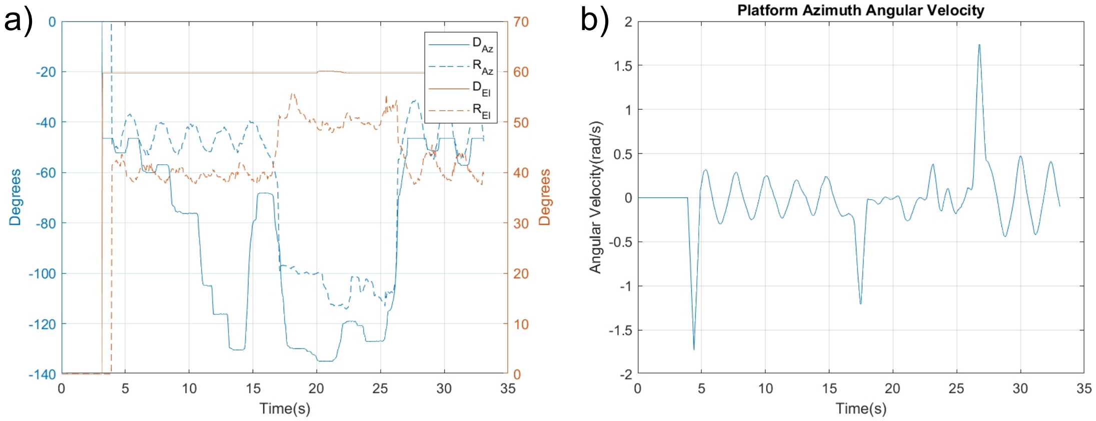


Figure 3.16: A second plot a) of the angle information gathered from the tether, where a solid line is for the UAV, a dotted line is for the platform, a blue line is an azimuth angle, and an orange line is for an elevation angle and b) of the corresponding platform azimuth angular velocity.

3.17. The tether crosses the obstacle where there is a sharp increase or decrease in the rover azimuth angle.

As can be seen, at around 18 seconds, the platform azimuth angle decreases significantly and at the same time, the velocity reaches the threshold for the obstacle case. However, this only lasts about 3 seconds, even though the azimuth angle should still be stopped by the obstacle. According to Figure 3.16, the rover azimuth angle is still in the area where it's being held by the obstacle until 26 seconds when it increases again. This can be attributed to sensor errors or the tether sliding on the obstacle causing movement to be detected.

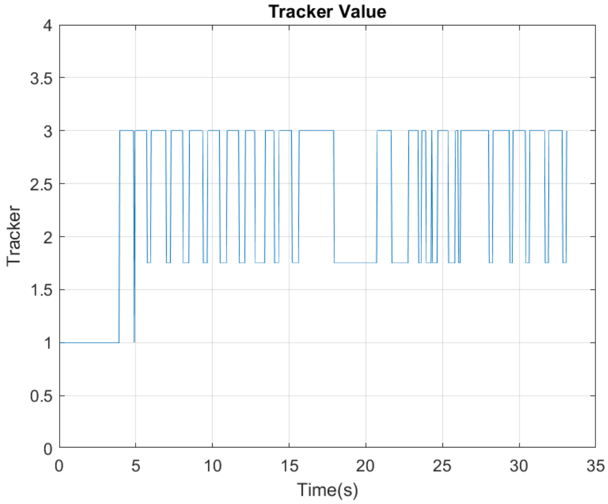


Figure 3.17: A second plot of the state machine tracking value, where the numbers represent the same states as Figure 3.15

Chapter 4

Conclusions

In conclusion, a method was successfully developed for landing a VTOL UAV using the tether as a guide. This method was not only able to land the UAV normally but also with a loose tether. With future developments and more time for implementation, it should also be able to overcome obstacles and find the landing pad when the tether is bent. To land with a mobile landing platform has not been significantly proven or implemented but future progress and implementation should prove fruitful. Throughout the course of this project, it has been considered that UAV flight has become reliant on vision systems and mapping the environment. While this can be very useful, not all UAV systems have that capability. Tethered UAVs, if lacking vision and mapping, can take advantage of the physical link between themselves and the landing location. Future developments on this will continue to prove this capability and increase the robustness in the case of other scenarios, such as obstacles bending the tether, and having to catch up with a moving platform.

The methodology developed is constrained by a few significant factors. The UAV's low-level controller must be robust and very responsive to command. The necessity for ramp inputs to the UAV was discussed as important because it reduces perturbations caused when using step inputs. This system also relies on low velocities. Currently, the speed is only 0.15 m/s and even this velocity has caused perturbations. The 'Loose Tether' case also relies on a flat or gently sloping ground for the height discontinuity detection to be an accurate indication of finding the landing pad. The current concept for the 'Moving Platform' case relies on an accurate reading of the elevation angle, as does the 'Obstacle' case for the azimuth angle. Path smoothing functions are able to solve this issue for noisy sensors. However, the current smoothing functions (Equation 2.7 for the 'Obstacle' case) may cause a small delay in the state machine response because the angle information is smoothed by taking the averages of the history vectors. Lastly, landing the UAV is only successful if it can follow its azimuth angle toward the landing pad. To ensure the azimuth angle is as consistent as possible, the UAV is never commanded to rotate about the z-axis. It's important that the azimuth angle of the UAV is subjected to small perturbations. This landing methodology operates within these constraints to attempt a precise landing, though future improvements can be made to mitigate them.

4.1 Discussion

4.1.1 Normal Operation

As shown in the results section, the normal operation state can successfully maneuver the UAV onto the landing pad from any point in the flight. This is especially important because if the landing pad is elevated, as it is for mounting the landing pad on a UGV, the UAV has to be able to consider this and not just fly and crash into the pad. The sub-states of normal operation are also designed to be adjusted so that the angles and height work for any system. The hardware used in the implementation of this thesis had a very narrow field of elevation angle detection at the drone and the 'XY Approach' state was specified for the narrowest area. This phenomenon can be seen in Figure 3.7a) and 3.8a) where 'XY Approach' is a very short segment of the flight. It can also be seen in Figure 2.7 as angles very similar to the angles in testing were also used to present the example vector field.

4.1.2 Loose Tether

The loose tether case was less successful but provided proof of concept. It was able to successfully land by detecting the discontinuity in the height measurements, but this had to be preceded by an accurate 'last known' taut tether direction. In testing, this was difficult to perform because of the winching mechanism. It was made such that a 'Loose Tether' state was difficult to initiate and maintain, even in simulated scenarios. To force a 'Loose Tether' state, the tension set-point was decreased. This made the tether loose from the beginning, and because of this, the UAV would never be able to get a direction. To overcome this, the tether was tightened momentarily by manually flying away from the platform. As soon as a direction was achieved, the UAV was set back to autonomous control. This manual flight caused more undo perturbation that might not be seen in a fully autonomous flight. Due to this, the number of successful 'Loose Tether' landings may be lower than if the UAV were fully autonomous and the tether fell loose by accident.

Another issue with the 'Loose Tether' case was proceeding with a landing when not directly above the landing pad. The UAV would sometimes reach the landing pad but the height discontinuity it detected was only a corner of the pad and it would land when the foot was not entirely over the landing pad. Part of this failure is again due to the testing constraints. As was described in the Testing Conditions section, the landing pad is a square wooden platform, very different from the landing pad designed to aid in the precision of landing shown in Figure 2.1 and 2.2. This issue can be improved by using the landing pad designed specifically for this purpose. The reason it wasn't used is that a different UAV had to be used in testing. The reason for this is that the intended UAV for this system was still under development. Of course, this issue is also caused by the imperfect 'last known' direction, which is the main reason for any unsuccessful landings. As mentioned earlier, possible improvements to improve the 'last known' direction and increase the likelihood of a successful landing will be discussed in Future Work.

As it is, the loose tether case is a last-ditched effort to land when the primary sources of information regarding the landing location are absent. Any success is more than would be achieved without this state. As it was successful for some of the tests with perturbations that only existed due to the constraints of testing, it can be reasonably assumed that without those constraints it would perform even better.

4.1.3 Obstacle

The 'Obstacle' case was tested up to the point of detection. This far, the case was successful and was also able to determine the correct tangent flight direction successfully based on the history of the angular velocity of the platform azimuth angle. The specific threshold for this angular velocity to determine an obstacle has stopped it may need some tuning, especially if, like in the case of testing, the obstacle allows the tether to slide up and down and some horizontally. As can be seen in Figures 3.16 and 3.17, this can cause the UAV to exit the 'Obstacle' case prematurely ultimately causing the UAV to collide with the obstacle it is trying to avoid. Outside of this, the case was successful and just needs full implementation.

The large downside to this case is a series of unsolved issues that require further development to fix. One such issue is that an obstacle crossed before the landing state initializes causes significantly more flight time to clear. This is a likely situation as UAVs perform complex tasks before landing that may result in obstacles being crossed. The logic required to solve an obstacle crossed midlanding is also problematic. If the UAV were to fly straight it wouldn't cross an obstacle to begin with. However, the rate of change in the platform's azimuth angle is beneficial when obstacles are crossed. This information is only available if the UAV flies in a spiral and it reduces additional flight time caused by an unknown tangent direction, as discussed in the Methodology. This could indicate that the landing state machine needs to run in the background of other UAV functions. The state machine would save the angle information before an obstacle is crossed and inform the tangent direction for clearing it. Another issue that this approach doesn't solve is the crossing of multiple obstacles. A solution could be devised using the rate of change of the UAV's azimuth angle. This information would be used to approximate the distance between the UAV and the obstacle. Large changes to the approximated distance could indicate another obstacle beyond the first. Such a solution will be further discussed in Future Work. Regardless of these issues, the fact that any solution exists without requiring mapping or vision systems, goes to show that reliance on such systems is unnecessary and that UAVs can function without them.

4.2 Future Work

4.2.1 Loose Tether

Several updates would improve the state machine's functionality. The focus of this section will be on improvements for the 'problematic' scenarios, loose tether, obstacle case, and moving platform. Starting with the 'Loose Tether' case, the most important improvement is to the accuracy or the 'last known' direction of the taut tether. It was mentioned in the methodology but this can be implemented by including an additional sub-state of 'Loose Tether' immediately after the pause for temporary system faults. This sub-state would be to fly in the opposite direction of the current 'last known' direction until the tether is taut and take direction from this. If following this state, the tether remains taut then 'Normal Operation' may resume, but if the tether loosens again then it proceeds in the updated direction until a height discontinuity is found. This same principle could be enacted by flying upwards until the tether is taut. However, both the opposite of the 'last known' direction and flying upwards have the potential to collide with obstacles. Another method of improving the 'Loose Tether' case, would be to improve the response to the 'last known' direction. Instead of trying to fix the direction, it would help to fly in a grid pattern over the area in which that direction points. Of course, this opens the UAV up to more perturbations because it extends the area over which the UAV is searching for the landing pad. In combination with improving the 'last known' direction, this should increase the likelihood of a successful landing for the 'Loose Tether' case significantly.

The other issue that needs addressing is to ensure that once a height discontinuity is found, the UAV lands near the center of the landing pad. This is addressing the issue of detecting corners referred to in the Discussion section. In the current implementation, the UAV only detects height discontinuities in which the height decreases. The way to address this issue is to have the UAV continue its flight until the inverse is detected, a height discontinuity in which the height increases again. This is the equivalent of detecting both sides of the landing pad. Once the inverse discontinuity is detected, the UAV would fly to about the midpoint of where the two discontinuities are detected. The midpoint would be approximated by saving the time it took to reach the inverse from the original discontinuity, halving it, and flying in the opposite direction that it took to reach the two discontinuities for that length of time. Once again this approach is imprecise but when paired with the improved 'last known' direction, the increased search area and search flight pattern, and the inclusion of the intended landing pad, it should result in a much-increased chance of successfully landing the UAV successfully and precisely.

4.2.2 Obstacle

The 'Obstacle' case still has to be fully implemented with the solution being tested in constraints much closer to a real situation. This testing should include obstacles crossing both on the left and right side of the tether and having to reverse the initial spiral direction. One addition to develop, as stated previously, is a response to and detection of multiple obstacles. The information necessary to approach this issue is presented in Figure 4.1. In this figure, v is calculated using Equation 4.1, where $\dot{\gamma}$ is the angular velocity of the UAV's azimuth angle, $\dot{v}$ is the velocity of the UAV tangent to the tether, and r is the approximated length of the tether from the UAV to the obstacle.

$$v = \dot{\theta} r \tag{4.1}$$

If a significant increase in r is calculated, it could indicate that another obstacle was crossed by the tether beyond the first. Likewise, if a significant decrease in r is calculated, it could indicate another obstacle was just crossed in addition to the one already blocking stopping the tether, this is another issue. It may be required in a situation as just presented and in more cluttered environments to decrease the rate of the spiral or remove the spiral entirely to clear obstacles. Overall this case needs to be made more intelligent with more sub-states to cover more situations in which the tether

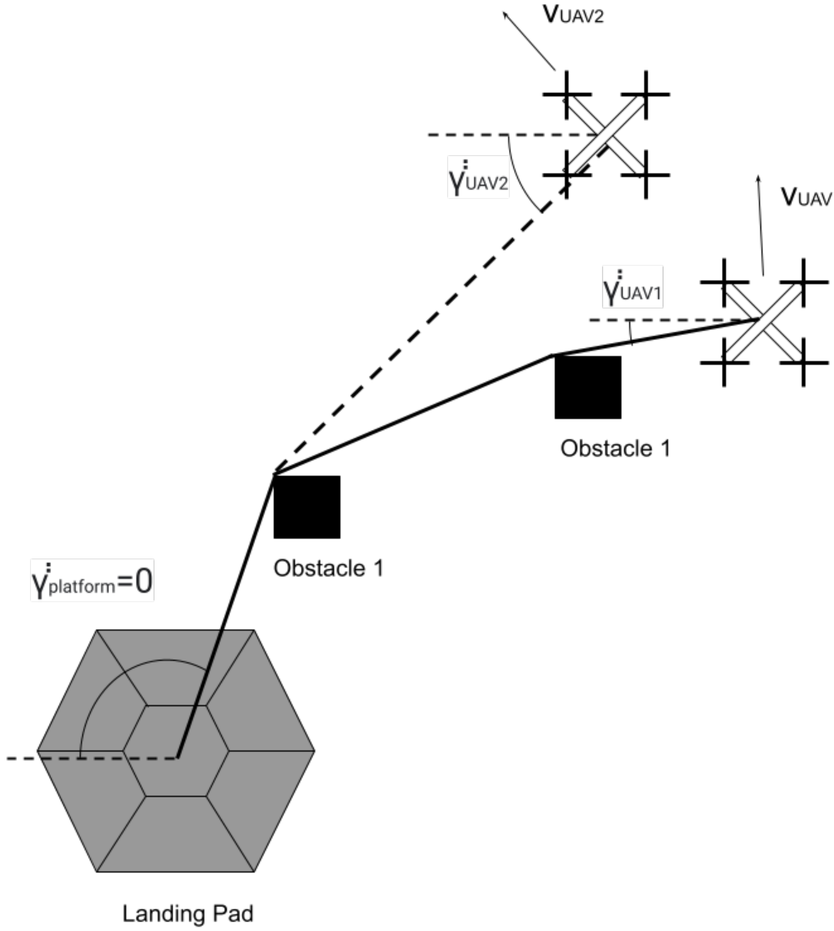


Figure 4.1: Illustration of multiple obstacles bending the tether, where $\dot{\gamma}$ is the UAV azimuth angular velocity, $\dot{v}$ is the UAV velocity tangent to the tether, and these are used with Equation 4.1 to calculate r, the approximated length of the tether from the UAV to the obstacle.

is bent by one or more obstacles in the environment.

4.2.3 Moving Platform

The moving platform needs implementation and testing. Due to the complexity of having a mobile platform, this case achieved the least development. It also requires much more space than is allowed in the confines of a laboratory to achieve velocity differences that can be detected. Especially with imperfect sensors that output noise requiring smoothing. In the case of all the noise, it may be advantageous to approach this case differently. Instead of calculating the elevation angular velocity, a set period could be assumed for each sub-state of 'Normal Operation', after which the UAV should have exited that case. If the UAV is lingering for too long in any sub-state, it means that the platform is moving away and the UAV needs to increase speed to catch up and land. After this detection, the same action would be taken as shown in Figure 2.13. Increase the speed of the UAV incrementally, if the UAV still lingers in the same state, increment again. After a long enough period if the UAV has remained in the same state it may be possible that there's another issue and failure is occurring. Another possibility is to receive communication from the mobile platform that it is moving and the UAV needs to adjust its flight speed. These should be implemented and tested along with the original method and compared for effectiveness.

4.2.4 Case Priority

The last issue for continued development is case priority. Some of these issues discussed in this thesis may be occurring at the same time, particularly the 'Loose Tether' and 'Obstacle' cases. It's also possible for the platform to be moving at the same time that either of these is occurring but as shown in Figure 2.15 the 'Moving Platform' case is a sub-state 'Normal Operation'. Testing so far has focused on the individual states and hasn't yet progressed to incorporating them all and the possibility of multi-state detection. The first determination is case priority, if two cases are occurring at the same time, which action needs to occur first, or is it possible to combine the actions in such a way that the landing is successful.

There's a lot of development that could be done to take this further and make this system more robust. This is a concept that needs development, despite how niche it may seem. Not all autonomous UAVs are capable of using vision systems for solving the issues presented so far in this thesis. This project presented that it's still possible to overcome these issues, if in a more mechanical or simplistic, yet elegant way.

Bibliography

[1] Francisco Alarcón et al. “A Precise and GNSS-Free Landing System on Moving Platforms for Rotary-Wing UAVs”. In: *Sensors* 19.4 (2019). ISSN: 1424-8220. DOI: 10.3390/s19040886. URL: https://www.mdpi.com/1424-8220/19/4/886.

[2] Roman Barták, Andrej Hraško, and David Obdržálek. “A controller for autonomous landing of AR.Drone”. In: *The 26th Chinese Control and Decision Conference (2014 CCDC)*. 2014, pp. 329–334. DOI: 10.1109/CCDC.2014.6852167.

[3] Fethi Belkhouche. “Reactive optimal UAV motion planning in a dynamic world”. In: *Robotics and Autonomous Systems* 96 (2017), pp. 114–123. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2017.07.006. URL: https://www.sciencedirect.com/science/article/pii/S0921889017302142.

[4] Hugh Cover et al. “Sparse Tangential Network (SPARTAN): Motion planning for micro aerial vehicles”. In: *2013 IEEE International Conference on Robotics and Automation*. 2013, pp. 2820–2825. DOI: 10.1109/ICRA.2013.6630967.

[5] Sudam Chamikara De Silva et al. “Inverted Docking Station: A Conceptual Design for a Battery-Swapping Platform for Quadrotor UAVs”. In: *Drones* 6.3 (2022). ISSN: 2504-446X. DOI: 10.3390/drones6030056. URL: https://www.mdpi.com/2504-446X/6/3/56.

[6] Malik Demirhan and Chinthaka Premachandra. “Development of an Automated Camera-Based Drone Landing System”. In: *IEEE Access* 8 (2020), pp. 202111–202121. DOI: 10.1109/ACCESS.2020.3034948.

[7] Ashray A. Doshi et al. “Development of micro-UAV with integrated motion planning for open-cut mining surveillance”. In: *Microprocessors and Microsystems* 39.8 (2015), pp. 829–835. ISSN: 0141-9331. DOI: https://doi.org/10.1016/j.micpro.2015.07.008. URL: https://www.sciencedirect.com/science/article/pii/S0141933115001076.

[8] Edwardo F. Fukushima, Noriyuki Kitamura, and Shigeo Hirose. "Development of tethered autonomous mobile robot systems for field works". In: *Advanced Robotics* 15.4 (2001), pp. 481–496. DOI: 10.1163/156855301750398374. eprint: https://doi.org/10.1163/156855301750398374. URL: https://doi.org/10.1163/156855301750398374.

[9] Musa Galimov, Roman Fedorenko, and Alexander Klimchik. "UAV Positioning Mechanisms in Landing Stations: Classification and Engineering Design Review". In: *Sensors* 20.13 (2020). ISSN: 1424-8220. DOI: 10.3390/s20133648. URL: https://www.mdpi.com/1424-8220/20/13/3648.

[10] Vinicius Mariano Gonçalves, Ryan McLaughlin, and Guilherme A. S. Pereira. "Precise Landing of Autonomous Aerial Vehicles Using Vector Fields". In: *IEEE Robotics and Automation Letters* 5.3 (2020), pp. 4337–4344. DOI: 10.1109/LRA.2020.2994485.

[11] Daehie Hong, Steven A. Velinsky, and Kazuo Yamazaki. "Tethered mobile robot for automating highway maintenance operations". In: *Robotics and Computer-Integrated Manufacturing* 13.4 (1997), pp. 297–307. ISSN: 0736-5845. DOI: https://doi.org/10.1016/S0736-5845(97)00002-1. URL: https://www.sciencedirect.com/science/article/pii/S0736584597000021.

[12] Panfeng Huang et al. "Dexterous Tethered Space Robot: Design, Measurement, Control, and Experiment". In: *IEEE Transactions on Aerospace and Electronic Systems* 53.3 (2017), pp. 1452–1468. DOI: 10.1109/TAES.2017.2671558.

[13] Ivan Kalinov et al. "High-Precision UAV Localization System for Landing on a Mobile Collaborative Robot Based on an IR Marker Pattern Recognition". In: *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*. 2019, pp. 1–6. DOI: 10.1109/VTCSpring.2019.8746668.

[14] Jonghwi Kim, Sangwook Woo, and Jinwhan Kim. "Lidar-guided autonomous landing of an aerial vehicle on a ground vehicle". In: *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*. 2017, pp. 228–231. DOI: 10.1109/URAI.2017.7992719.

[15] Soonkyum Kim, Subhrajit Bhattacharya, and Vijay Kumar. "Path planning for a tethered mobile robot". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. 2014, pp. 1132–1139. DOI: 10.1109/ICRA.2014.6906996.

[16] Soonkyum Kim and Maxim Likhachev. "Path planning for a tethered robot using Multi-Heuristic A* with topology-based heuristics". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015, pp. 4656–4663. DOI: 10.1109/IROS.2015.7354040.

[17] Ran Krauss and Meir Kliner. "Centering and Landing Platform for Hovering Flying Vehicles". U.S. pat. 2017130181.

[18] Rogerio R. Lima and Guilherme A. S. Pereira. "On the Development of a Tether-based Drone Localization System". In: *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*. 2021, pp. 195–201. DOI: 10.1109/ICUAS51884.2021.9476778.

[19] Patrick McGarey et al. "Developing and deploying a tethered robot to map extremely steep terrain". In: *Journal of Field Robotics* 35.8 (2018), pp. 1327–1341. DOI: https://doi.org/10.1002/rob.21813. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21813. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21813.

[20] Reese A. Mozer. "UAV docking system and method". U.S. pat. 10577126.

[21] Armando A. Neto, Douglas G. Macharet, and Mario F. M. Campos. "Feasible RRT-based path planning using seventh order Bézier curves". In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2010, pp. 1445–1450. DOI: 10.1109/IROS.2010.5649145.

[22] Christos Papachristos and Anthony Tzes. "The power-tethered UAV-UGV team: A collaborative strategy for navigation in partially-mapped environments". In: *22nd Mediterranean Conference on Control and Automation*. 2014, pp. 1153–1158. DOI: 10.1109/MED.2014.6961531.

[23] Louis Petit and Alexis Lussier Desbiens. "TAPE: Tether-Aware Path Planning for Autonomous Exploration of Unknown 3D Cavities Using a Tangle-Compatible Tethered Aerial Robot". In: *IEEE Robotics and Automation Letters* 7.4 (2022), pp. 10550–10557. DOI: 10.1109/LRA.2022.3194691.

[24] Amer Al-Radaideh and Liang Sun. "Self-localization of a tethered quadcopter using inertial sensors in a GPS-denied environment". In: *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. 2017, pp. 271–277. DOI: 10.1109/ICUAS.2017.7991436.

[25] Amer Al-Radaideh and Liang Sun. "Self-Localization of Tethered Drones without a Cable Force Sensor in GPS-Denied Environments". In: *Drones* 5.4 (2021). ISSN: 2504-446X. DOI: 10.3390/drones5040135. URL: https://www.mdpi.com/2504-446X/5/4/135.

[26] Sugumaran Ramanathan. "UAV docking system and method". U.S. pat. 9561871.

[27] L.A. Sandino et al. “Tether-guided landing of unmanned helicopters without GPS sensors”. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. 2014, pp. 3096–3101. DOI: 10.1109/ICRA.2014.6907304.

[28] Mohammad Fattahi Sani and Ghader Karimian. “Automatic navigation and landing of an indoor AR. drone quadrotor using ArUco marker and inertial sensors”. In: *2017 International Conference on Computer and Drone Applications (IConDA)*. 2017, pp. 102–107. DOI: 10.1109/ICONDA.2017.8270408.

[29] Danylo Shapovalov and Guilherme A. S. Pereira. “Exploration of unknown environments with a tethered mobile robot”. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 6826–6831. DOI: 10.1109/IROS45743.2020.9340993.

[30] Danylo Shapovalov and Guilherme A. S. Pereira. “Tangle-Free Exploration with a Tethered Mobile Robot”. In: *Remote Sensing* 12.23 (2020). ISSN: 2072-4292. DOI: 10.3390/rs12233858. URL: https://www.mdpi.com/2072-4292/12/23/3858.

[31] Petr Váňa et al. “Minimal 3D Dubins Path with Bounded Curvature and Pitch Angle”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 8497–8503. DOI: 10.1109/ICRA40945.2020.9197084.

[32] Liyang Wang and Xiaoli Bai. “Quadrotor Autonomous Approaching and Landing on a Vessel Deck”. In: *J. Intell. Robotics Syst.* 92.1 (Sept. 2018), pp. 125–143. ISSN: 0921-0296. DOI: 10.1007/s10846-017-0757-5. URL: https://doi.org/10.1007/s10846-017-0757-5.

[33] Xuesu Xiao et al. “Motion Planning for a UAV with a Straight or Kinked Tether”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018, pp. 8486–8492. DOI: 10.1109/IROS.2018.8594461.

[34] Lu Yan et al. “A Safe Landing Site Selection Method of UAVs Based on LiDAR Point Clouds”. In: *2020 39th Chinese Control Conference (CCC)*. 2020, pp. 6497–6502. DOI: 10.23919/CCC50068.2020.9189499.

[35] Chris Young et al. “Field Tests of the Hybrid Remotely Operated Vehicle (HROV) Light Fiber Optic Tether”. In: *OCEANS 2006*. 2006, pp. 1–6. DOI: 10.1109/OCEANS.2006.306819.