A Low Cost Implementation of Autonomous Takeoff and Landing for a Fixed Wing UAV

Thomas Carnes
Virginia Commonwealth University

Follow this and additional works at: https://scholarscompass.vcu.edu/etd
Part of the Aerospace Engineering Commons

© The Author

Downloaded from https://scholarscompass.vcu.edu/etd/3540

This Thesis is brought to you for free and open access by the Graduate School at VCU Scholars Compass. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of VCU Scholars Compass. For more information, please contact libcompass@vcu.edu.
A Low Cost Implementation of Autonomous Takeoff and Landing for a Fixed Wing UAV

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at Virginia Commonwealth University

by

Thomas W. Carnes

Director: Dr. Robert H. Klenke
Professor of Electrical and Computer Engineering

Virginia Commonwealth University
Richmond, Virginia
August 2014
Acknowledgements

First and foremost I would like to thank my family for all the love and support they have given me throughout my life, without which I would have nothing. I would like to thank Dr. Klenke, for all his guidance, support, and patience with me, particularly during my first year at VCU. I would especially like to thank Tim Bakker all the many times he would take time out of his busy day and help me trouble shoot problems and give me much needed guidance. I would like to thank Garrett Ward for the numerous programming questions he answered and for all of his IT support around the lab. I would like to thank Siva Patibandla for the time we spent as lab mates and for all his encouragement. Lastly, I would like to thank John Picchi for his friendship and encouragement. I have very much enjoyed my time in the VCU UAV lab, where I’ve had excellent people to work with who have enriched my life, and who have helped me grow both personally and professionally. I thank the Lord for my time here and I feel very blessed to have the opportunity to have worked in such a unique environment.
# Table of Contents

Acknowledgements ........................................................................................................... ii  
List of Figures ................................................................................................................... v  
Abstract .............................................................................................................................. viii  
Chapter 1: Introduction ............................................................................................................... 1  
  1.1 UAV Overview .............................................................................................................. 1  
  1.2 Motivation .................................................................................................................... 2  
  1.3 Scope of Work ............................................................................................................. 3  
  1.4 Thesis Overview ......................................................................................................... 3  
Chapter 2: Background ........................................................................................................... 5  
  2.1 Traditional Takeoff and Landing: .................................................................................. 5  
    2.1.1 Traditional Aircraft Taxiing and Takeoff Phases: .................................................. 6  
    2.1.2 Traditional Aircraft Descent and Landing using ILS: ........................................... 7  
  2.2 Sensor Technologies ..................................................................................................... 8  
    2.2.1 Laser-Based Sensors: .............................................................................................. 8  
    2.2.2 Radar-Based Sensors: ............................................................................................ 10  
    2.2.3 DGPS with Backup Sensors: .................................................................................. 12  
  2.3 A Brief Survey of Commercial and Academic ATOL Systems: ...................................... 13  
    2.3.1 BAE’s Kingfisher Platform: ................................................................................... 14  
    2.3.2 Stellenbosch University’s ATOL System: ............................................................. 15  
    2.3.3 Seoul National University’s ATOL using only GPS: ........................................... 17  
    2.3.4 Brigham Young University MAV Autonomous Landing: ..................................... 19  
    2.3.5 Tohoku University’s ATOL: .................................................................................. 22  
    2.3.6 SAAB’s SHARC Demonstrator: .......................................................................... 23  
    2.3.7 Cloud Cap Technology’s Piccolo II Autopilot: ...................................................... 26  
  2.4 This Project’s Aircraft Platform, FCS, GCS, and HILS: ................................................... 27  
Chapter 3: ATOL Navigation and Waypoint Calculations ..................................................... 31  
  3.1 Navigation for Autonomous Takeoff and Landing: ....................................................... 31  
    3.1.1 Navigation for Autonomous Takeoff: ................................................................. 32
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1.2</td>
<td>Navigation for Autonomous Landing</td>
<td>32</td>
</tr>
<tr>
<td>3.2</td>
<td>Glidepath Altitude and Waypoint Calculations:</td>
<td>34</td>
</tr>
<tr>
<td>3.3</td>
<td>Example Runway with Calculated Variables:</td>
<td>40</td>
</tr>
<tr>
<td>Chapter 4: ATOL Algorithms</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>4.1</td>
<td>PID Control:</td>
<td>46</td>
</tr>
<tr>
<td>4.2</td>
<td>Autonomous Takeoff Sequence:</td>
<td>47</td>
</tr>
<tr>
<td>4.3</td>
<td>Autonomous Landing Sequence:</td>
<td>49</td>
</tr>
<tr>
<td>Chapter 5: HILS Testing</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>5.1</td>
<td>HILS Test – Autonomous Takeoff</td>
<td>56</td>
</tr>
<tr>
<td>5.2</td>
<td>HILS Test – Autonomous Landing with w/o Wind and Right Direction:</td>
<td>60</td>
</tr>
<tr>
<td>5.3</td>
<td>HILS Test – Autonomous Landing with w/o Wind and Left Direction:</td>
<td>65</td>
</tr>
<tr>
<td>5.4</td>
<td>HILS Test – Autonomous Landing with w Wind and Right Direction:</td>
<td>69</td>
</tr>
<tr>
<td>5.5</td>
<td>HILS Test – Autonomous Landing with w Wind and Left Direction:</td>
<td>74</td>
</tr>
<tr>
<td>5.6</td>
<td>HILS Simulation Conclusions:</td>
<td>78</td>
</tr>
<tr>
<td>Chapter 6: Conclusions and Future Work</td>
<td>79</td>
<td></td>
</tr>
<tr>
<td>6.1</td>
<td>Conclusions:</td>
<td>79</td>
</tr>
<tr>
<td>6.2</td>
<td>Future Work:</td>
<td>80</td>
</tr>
<tr>
<td>Bibliography</td>
<td>82</td>
<td></td>
</tr>
</tbody>
</table>
List of Figures

Figure 1: Aircraft Taxiing, Takeoff, Cruise, and Landing [9] ................................................................. 6
Figure 2: Three dimensional view of the ILS [13]. ...................................................................................... 8
Figure 3: Example of laser altimeter [16]. ..................................................................................................... 9
Figure 4: OPATS System (left) and UAV reflector x(right) [17]. ............................................................... 10
Figure 5: Roke Miniature Radar Altimeter (MRA) Type 2 [18]. ................................................................. 11
Figure 6: THALES MAGIC ATOL System [19]. ............................................................................................ 12
Figure 7: Example of Differential GPS (DGPS) Receiver: Trimble NavBeaconXL [20] ......................... 13
Figure 8: Stellenbosch University's Takeoff Sequence [4] ...................................................................... 16
Figure 9: Stellenbosch University's Landing Sequence [4] ..................................................................... 17
Figure 10: Cloud Cap Technology Piccolo II Autopilot [27]. .............................................................. 26
Figure 11: Carl Goldberg Mig Airframe .................................................................................................... 27
Figure 12: miniFCS developed at VCU. ........................................................................................................ 28
Figure 13: VCU’s Ground Control Station (GCS). ..................................................................................... 29
Figure 14: Hardware connection between the miniFCS, HILS, and FlightGear [28]. ........................... 30
Figure 15: Simple Takeoff Navigation in Relation to the Runway ......................................................... 32
Figure 16: Simple Landing Navigation with Traffic Pattern and Glidepath Waypoints ................ 34
Figure 17: Target Glidepath Altitude Calculation ...................................................................................... 39
Figure 18: Diagram of Runway, Showing Traffic Pattern and Glidepath Waypoints and Bearings .......... 40
Figure 19: Charles City Runway Running From South to North ........................................................... 41
Figure 20: Charles City Runway Bearings for South to North ............................................................. 42
Figure 21: Charles City Runway Running South to North with All Calculated Variables ................. 44
Figure 22: PID Controller [6] .................................................................................................................. 47
Figure 23: Complete Autonomous Takeoff Sequence ............................................................................. 49
Figure 24: Traffic Pattern Showing the Two Different Directions around the Runway .................. 52
Figure 25: Complete Autonomous Landing Sequence ............................................................................. 54
Figure 26: GCS with Charles City Showing Runway South to North (Left) ........................................ 56
Figure 27: GCS with Charles City Showing the Aircraft Shortly after Liftoff (Left) ..................... 57
Figure 28: Aircraft’s Altitude During Different Takeoff Mode Phases ........................................... 58
Figure 29: Aircraft’s Heading Error while on the Runway ................................................................. 59
Figure 30: Aircraft’s Cross-Track Error while on the Runway ......................................................... 60
Figure 31: GCS Showing Aircraft Following Right Traffic Pattern (Left) and FlightGear Image Showing Aircraft Turning from Traffic Pattern Waypoint 3 to 4 Confirming Right Hand Turning (Right)................................................................................................................................. 61
Figure 32: GCS Satellite Image Showing Aircraft during Glidepath and Flare Mode (Left) and FlightGear Image Showing Aircraft in Flare Mode Centered on the Runway (Right) ............. 62
Figure 33: Aircraft’s Altitude Error during Glidepath Descent........................................................ 63
Figure 34: Aircraft’s Cross-Track Error during Glidepath Descent ................................................. 64
Figure 35: Aircraft’s Climbrate Error during Flare Phase ............................................................... 65
Figure 36: GCS Showing Aircraft Following Left Traffic Pattern (Left) and FlightGear Image Showing Aircraft Turning from Traffic Pattern Waypoint 2 to 1 Confirming Right Left Turning (Right)................................................................................................................................. 66
Figure 37: GCS Satellite Image Showing Aircraft during Glidepath and Flare Mode (Left) and FlightGear Image Showing Aircraft in Flare Mode Centered on the Runway (Right) ............. 66
Figure 38: Aircraft’s Altitude Error during Glidepath Descent........................................................ 67
Figure 39: Aircraft’s Cross-Track Error during Glidepath Descent ................................................. 68
Figure 40: Aircraft’s Climbrate Error during Flare Phase ............................................................... 69
Figure 41: GCS Showing Aircraft Following Right Traffic Pattern (Left) and FlightGear Image Showing Aircraft Turning from Traffic Pattern Waypoint 3 to 4 Confirming Right Hand Turning (Right)................................................................................................................................. 70
Figure 42: GCS Satellite Image Showing Aircraft during Glidepath and Flare Mode (Left) and FlightGear Image Showing Aircraft in Flare Mode Centered on the Runway (Right) ............. 70
Figure 43: Aircraft’s Altitude Error during Glidepath Descent........................................................ 71
Figure 44: Aircraft’s Cross-Track Error during Glidepath Descent ................................................. 72
Figure 45: Aircraft’s Climbrate Error during Flare Phase ............................................................. 73
Figure 46: GCS Showing Aircraft Following Left Traffic Pattern (Left) and FlightGear Image Showing Aircraft Turning from Traffic Pattern Waypoint 4 to 3 Confirming Right Left Turning (Right) ............................................................................................................................... 74
Figure 47: GCS Satellite Image Showing Aircraft during Glidepath and Flare Mode (Left) and FlightGear Image Showing Aircraft in Flare Mode Centered on the Runway (Right) .................. 75
Figure 48: Aircraft’s Altitude Error during Glidepath Descent ..................................................... 76
Figure 49: Aircraft’s Cross-Track Error during Glidepath Descent .............................................. 77
Figure 50: Aircraft’s Climbrate Error during Flare Phase ........................................................... 78
Abstract

A LOW COST IMPLEMENTATION OF AUTONOMOUS TAKEOFF AND LANDING FOR A FIXED WING UAV

By Thomas W. Carnes, M.S.

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at Virginia Commonwealth University

Virginia Commonwealth University, 2014

Major Director: Dr. Robert H. Klenke, Professor of Electrical and Computer Engineering

The take-off and landing of an Unmanned Aerial Vehicle (UAV) is often the most critical and accident prone portion of its mission. This potential hazard coupled with the time and resources necessary to train a remote UAV pilot makes it desirable to have autonomous take-off and landing capabilities for UAVs. However, a robust, reliable, and accurate autonomous takeoff and landing capability for fixed-wing aircraft is not an available feature in many low-cost UAV flight control systems. This thesis describes the design of an autonomous take-off and landing algorithm implemented on an existing low-cost flight control system for a small fixed wing UAV. This thesis also describes the autonomous takeoff and landing algorithm development and gives validation results from hardware in the loop simulation.
Chapter 1: Introduction

1.1 UAV Overview

Much effort is currently spent on the research and production of unmanned vehicles, particularly those related to Unmanned Aerial Vehicles (UAV). The obvious advantage UAVs have over their conventional, manned aircraft counterparts is that UAVs do not need a pilot or crew to be physically present in the vehicle during operation. This fact keeps the pilot and crew out of harm’s way during potentially dangerous missions while also allowing the aircraft to be made smaller and exempt from all the hardware necessary to sustain life support. UAVs can also host a variety of sensors and payloads that can be tailored for a given situation or need. Due to the advantages listed above, UAVs have become very popular in military applications and more recently in civilian areas.

One need only look to the US involvement in Libya, Pakistan (or any other US military involvement within the past several years) to see the military and intelligence agency’s significant reliance on deploying UAVs to areas that are either too dangerous or controversial to send conventional aircraft. Depending on a given UAV’s sensor suite, that UAV can provide lifesaving surveillance (i.e. watching for militants to plant IEDs), provide mission critical target acquisition (i.e. identify a targets location), and can deliver a weaponized payload to a specified target. Although more recent civilian uses for UAVs have also become very popular and are suited for a wide range of applications including scientific and academic research, fire detection and management, law enforcement, search and rescue operations, digital mapping, land management, and air traffic control support [1].
1.2 Motivation

Early UAVs had to be remotely piloted from the ground by an operator acting as the pilot, but today’s technology advancements have allowed UAVs to often have the option of several levels of autonomy. After an operator sets certain parameters for the UAV (altitude, airspeed, coordinates, etc.) an autonomously operated UAV can fly itself to its destination, allowing the operator to monitor the aircraft instead of constantly controlling it. Even though VCU’s UAV lab has several platforms that have high levels of autonomy (navigation, collaboration, etc.), there is currently no platform available that is fully autonomous from take-off to landing. Such a platform requires a set of customized autonomous take-off and landing (ATOL) controllers in combination with existing flight control systems (FCS). High levels of automation, specifically ATOL based systems, provide several benefits such as higher levels of flight safety, simplified operations, lower operating costs, and reduced operator workload [2].

One of the many advantages of ATOL based systems is the elimination of the human operator (and consequently any operator error) during the crucial take-off and landing process. Human error alone accounts for roughly 60% of UAV accidents during operations [2], and incidents during take-off and landing account for over 50% of accidents even though they account for only a fraction of the flight phase [3]. Eliminating the role of the operator from manually controlling the aircraft during take-off and land and replacing him with an ATOL based system can greatly increase levels of safety during operations.

Training operators on how to take-off and land UAVs represent a significant investment in both time and money, and eliminating the need for that training could be beneficial to any host program. The operator is also limited by the conditions in which he can land the aircraft, such as
in nighttime or in heavy fog; having an automated system that does not rely on sight for landing purposes, but instead an onboard suite of sensors, provides a much more robust UAV platform[4]. Finally, the operator is able to focus his attention on other tasks and redirect his responsibilities to monitoring the aircraft, instead of taking his time to manually control the take-off and landing.

In military operations the ATOL system could cut down on the launching and recovery crews that are needed for the larger UAV platforms. This would allow the military to redirect valuable operators to other operational areas [5]. In long-endurance (20+hours) operations the human operator may suffer from fatigue that affects the operators decision making capacity, performance, and ability to focus – all of which are critical for landing [2]. Implementing an ATOL system eliminates any operator risk factors and replaces the operator with a system that neither tires nor fatigues.

1.3 Scope of Work

This thesis presents work for the creation and implementation of a low cost, fully autonomous aircraft – from take-off to landing – by implementing a set of ATOL navigation and control algorithms in conjunction with existing autopilot systems developed at the VCU UAV lab. This includes all the various sensors and software controllers added to the Mini FCS (Flight Control System) developed in [6] in order to achieve fully autonomous flight.

1.4 Thesis Overview

In the following chapters, the development and testing of autonomous take-off and landing controllers (ATOL) necessary for fully autonomous flight is presented. The first half of the thesis
concentrates on challenges associated with autonomous takeoff and landing along with algorithms developed to overcome those challenges. The second half of this thesis presents the flight simulations using a custom made hardware/software simulator, and evaluation of the system.

Chapter 2 provides the necessary background for this thesis. This section begins with a brief history of the origins of autonomous landing for a conventional aircraft and continues with an introduction to some sensor technologies that aid in autonomous takeoff and landing, a brief overview of current work being done in commercial, defense, and academic development of flight control systems featuring autonomous takeoff and landing, and finally an introduction to the aircraft platform, custom autopilot, custom ground control station (GCS), and simulation of the hardware and software used in this thesis.

Chapter 3 introduces how the aircraft’s runway is defined and how important variables such as target glidepath altitude, approach and approach leg waypoints, distances, and bearings are calculated in order to have a successful autonomous takeoff and landing.

Chapter 4 illustrates the autonomous takeoff and landing algorithms using the runway definition and variables calculated in Chapter 3. Chapter 5 discusses the simulation and validation results. Finally Chapter 6 outlines conclusions and future work.
Chapter 2: Background

The first recorded autonomous landing of a fixed wing aircraft occurred on August 23, 1937 at Wright Field in Dayton Ohio using the Army’s C-14B transport plane. This feat was made possible due to the efforts of Captain Carl Crane who developed the necessary instruments and radios on board the C-14B that interacted with a series of five ground radio beacons around the airfield. When the receivers onboard the aircraft were turned on, the airplane began its descent and was able to land unaided by its crew [7]. Since then there has been significant advancements in technology that has allowed for more complex, compact, and robust autopilot systems that can perform not only autonomous landings but also autonomous takeoffs. Even though there are several benefits when switching to an ATOL system, implementation can pose several challenges. The largest challenge in developing an autonomous takeoff and landing system is that of the localization problem. The localization problem is having the UAV know precisely where its position is (including elevation) relative to the runway, when taking-off or landing. Other challenges that need to be considered when designing an ATOL system include: runway centerline tracking, precision approach, ground effects, cross winds (crab and de crab angles), decent rate, and braking actions [8].

2.1 Traditional Takeoff and Landing:

In order to address the autonomous takeoff and landing for a UAV, it becomes necessary to understand the traditional takeoff and landing procedures for larger scale aircraft, typically commercial aircraft taking off and landing at a civilian airport, or military aircraft taking off and
landing at a military base. Typically the entire “flight” process begins with the aircraft taxis to the runway, followed by the takeoff procedure, cruising to its objective, and finished by the landing maneuver.

\[\text{Figure 1: Aircraft Taxiing, Takeoff, Cruise, and Landing [9].}\]

\[\text{2.1.1 Traditional Aircraft Taxiing and Takeoff Phases:}\]

Taxiing refers to the aircraft being propelled forward on the ground using the throttle to approach and line up with the runway. Steering is achieved by turning the nose wheel and rudder.

The takeoff phase of flight consists of the aircraft transitioning from taxiing to flying in the air. Usually the engines/motors are set to full throttle to achieve the takeoff speed, which varies with air density, aircraft weight, and the configuration of the aircraft frame. The speeds needed for takeoff are relative to the motion of the air, where a head wind will reduce the ground speed needed for takeoff, as there is greater flow of air over the wings generating more lift for the aircraft.
Following takeoff the aircraft has to climb to a certain altitude before it can enter its cruise altitude safely. The climb out is carried out by increasing the lift of the aircraft’s wings until the lifting force is greater than the weight of the aircraft [10].

2.1.2 Traditional Aircraft Descent and Landing using ILS:

The descent portion of flight is simply the aircraft lowering its altitude in preparation to land at an airport or runway of some sort.

Most airports today are equipped with an Instrument Landing System (ILS) that is comprised of several radio beacons placed on the runway allowing for vertical and lateral guidance to the aircraft during the aircraft’s glideslope and flaring maneuvers. Once the maneuver starts, the ILS guides the aircraft to a certain height, known as the decision height (DH) which is dependent on the airport’s ILS category and the ILS based guidance system on board the aircraft [11]. The decision height is the height at which the pilot must have adequate visual reference to the runway to decide whether to continue the descent to land or to abort the landing maneuver and try to approach the runway again [12]. The most advanced category of ILS is the ILS IIIc which allows for autonomy of the entire landing maneuver including guidance along the runway [11]. Figure 2 illustrates the ILS indicator to the pilot whether the aircraft’s altitude and position to the runway are acceptable enough to perform a successful landing.

Unfortunately ILS systems are very expensive and often only found at airports and military bases and are not often seen at smaller landing sites that would be more ideal for UAVs. With this in mind, this chapter will focus on some other alternatives for aiding the unmanned aircraft to accurately approach the runway and keep a correct glide path angle.
Figure 2: Three dimensional view of the ILS [13].

2.2 Sensor Technologies

Autonomous landing has the unique problem of having an accurate altitude measurement for very low altitudes (usually less than 20-30 feet) known as the height above ground (HAG) problem. In order to effectively address the HAG problem, several different sensor technologies must be examined for giving the UAV its position relative to the runway. The sensors that will be covered in this section include: laser-based, radar-based, and DGPS with backup sensors.

2.2.1 Laser-Based Sensors:

One of the many sensor options that could solve the HAG problem is a laser altimeter. Laser altimeters determine altitude by measuring the length of time needed for a generated pulse of light (laser) to travel from the generating instrument to the surface it is measuring and back [14]. For implementation on a UAV, the laser altimeter would be placed somewhere on the aircraft
(most likely the lower fuselage) that would have a clear view of the ground. The laser would fire, and the time it took for the laser to be reflected from the ground would be calculated to give a more accurate altitude for the UAV. The advantage of the laser altimeter is that it could be contained onboard the aircraft while giving reliable altitude information for lower altitudes, one of the disadvantages however is the price, usually laser altimeters can range from a couple of hundred dollars to well over a thousand dollars [15], making it not a viable solution for the low cost nature of this project.

Figure 3: Example of laser altimeter [16].

Laser based sensors do not necessarily need to be contained within the aircraft itself to give accurate altitude data for the aircraft. One laser based system developed specifically for autonomous takeoff and landing is the Object Position and Tracking Sensor (OPATS) developed by RUAG. The landing process is initiated by the UAV when it enters a predetermined window in the space. At that time the OAPTS takes continuous position measurements of the UAV by using an infrared laser (stationed on the ground near the runway) which echoes back from a passive retro reflector on the UAV. The measured position data is transmitted to the Ground Control Station (GCS), processed, and used to guide the UAV on its glide path to the runway [7]. The
equipment required for the OPATS system are the tri-pod mounted laser sensor, the sensor’s power supply, and two optical retro reflectors integrated on the front side (near the nose or under the wing) of the UAV.

![Figure 4: OPATS System (left) and UAV reflector (right) [17].](image)

The advantages of using this OPATS ATOL system include: reliability in GPS denied environments, reliability during any time of day and in most weather conditions, and the need for only one ground OPATS for multiple UAVs, where an aircraft mounted laser altimeter would require one laser altimeter for each UAV. These laser-based sensors, which cannot be jammed and are operable in GPS denied environments, are very desirable for military and defense based UAV projects.

Unfortunately, systems such as the OPATS are very expensive and require ground based platforms that are not an option for the low cost scope of this project.

### 2.2.2 Radar-Based Sensors:

Radar altimeters are another option when trying to solve the HAG problem. Radar altimeters transmit radio waves towards the ground and measure the time that the reflected signal
returns to the transmission source, similar to the method laser altimeters determine altitude. Radar altimeters such as the Roke Miniature Radar Altimeter (MRA) Type 2, offer altitude ranges from 100 meters down to 20 cm with maximum accuracy of 1 cm in a lightweight (13oz) package.

![Roke Miniature Radar Altimeter (MRA) Type 2](image)

**Figure 5: Roke Miniature Radar Altimeter (MRA) Type 2 [18].**

One radar based system that was dedicated specifically to address HAG issue is the THALES’s MAGIC ATOL system. MAGIC consists of ground radar placed on the edge of the runway, along with a beacon onboard the UAV and a separate ground beacon installed on the runway edge near a predetermined touchdown point. The beacon aboard the UAV gives the radar the UAV’s position reference, while the beacon on the runway allows for measurement of angular position of the UAV, and movement of the touchdown point [2]. The radar system employed by MAGIC consists of a wide fixed beam that allows for instantaneous detection in the entire coverage area.

The UAV landing phase is initiated when the UAV penetrates the radar’s antenna lobe. The UAV is detected and identified through the ground beacon’s fixed frequency at a range of 5 km. The radar operates in a continuous mode (CW) allowing for simultaneous emission of transmitted signals and reception from signals received by the UAV; the waveform includes two sequences working in conjunction with one another. The first sequence is referred to as the active mode in which the radar emits Frequency Modulated Continuous Wave (FMCW) signals in the
UAV’s direction, detects the UAV, and estimates the UAV’s position and speed. The second sequence is referred to as the passive mode. When the passive mode has been initiated, the radar listens successively to the signals emitted from the ground and UAV placed beacons. The radar then calculates the UAV’s position in regard to the touch down point. The active and passive modes, that calculate the UAV’s position in relation to the touch down point, create a system redundancy that greatly increases the chances for a successful landing. The downside, however, is that radar altimeters such as these are often very expensive and not possible for this project.

![Figure 6: THALES MAGIC ATOL System [19].](image)

### 2.2.3 DGPS with Backup Sensors:

One of the most commonly used ATOL HAG for UAV research systems is differential GPS (DGPS) in conjunction with a backup system, usually in the form of a laser/radar altimeter or an ultrasonic sensor. Differential GPS comes in many different forms, but the underlying idea is to transmit more accurate positional data to the GPS receiver than would be available for regular GPS systems). The need for a backup precision sensor stems from the fact that DGPS, on its own, has a number of weak points such as loss of GPS lock, and can be easily blocked or jammed. GPS lock and the possibility for jamming are not the only reasons why DGPS should not be used as the
sole ATOL sensor used in a UAV; GPS is less accurate in the vertical channel than the horizontal channel [2]. Inaccuracies in the UAV’s altitude would prove catastrophic while trying to land; consequently a precision laser, radar altimeter, or ultrasonic sensor, must be present to prevent inaccurate altitude measurements. Most differential GPS options require either a subscription for its service ranging in the several thousand dollars per year, or the use of a special GPS receiver and antenna that run several hundred dollars.

![Image of GPS receiver](image)

Figure 7: Example of Differential GPS (DGPS) Receiver: Trimble NavBeaconXL [20].

### 2.3 A Brief Survey of Commercial and Academic ATOL Systems:

In order to effectively address the design of an ATOL based system, it becomes necessary to examine the previous work from three of the main customers of autonomous UAV technologies: the defense industry, commercial applications, and academic institutions. Specifically, this section will focus on research and development of autonomous take-off and landing algorithms that have
been simulated and had successful flight validation, with brief mention of any special sensors used and the air platforms used. Since most ATOL research has been done in the field of rotary aircraft, the following sections focus on ATOL systems that have been implemented for fixed wing aircraft.

2.3.1 BAE’s Kingfisher Platform:

BAE Systems in [21] use the Kingfisher platform as its test UAV, whose fuselage and power plant are from the Brumby Mk3 delta wing configuration combined with a conventional wing and tail. Navigation of the Kingfisher is accomplished by the use of a suite of sensors comprised of a three axis silicon solid-state IMU, dual DGPS receivers, laser altimeters, and dual clinometers, which give the UAV its position and velocity.

For automatic take-off, BAE defines the runway alignment as a straight line joining two waypoints, which were surveyed using the navigation system during the UAV’s taxi test. Once the UAV has initialized the navigation system, the engine starts and the UAV is manually taxied to a holding point, at which point the ground control station (GCS) operator initiates the take-off sequence. The guidance system then applies full throttle and issued yaw control to main alignment with the runway centerline. Under normal take-off conditions, the UAV is airborne before the full pitch angle is achieved. The UAV’s lift-off indication was determined using the weight on its wheels, airspeed, and climb rate. Once the UAV is airborne, the guidance system transitions to a climb-out state, where the pitch angle and horizontal tracking are maintained by pitch and roll rate commands, respectively. The climb-out state, is completed when the UAV attains a certain altitude, at which point the guidance system transitions to normal control maintaining the altitude, horizontal track, and airspeed.
Autonomous landing is initiated after the UAV passes through the last waypoint indicating that the mission is complete, and that landing should commence. The landing approach trajectory is defined by a straight line linking two waypoints, and an end waypoint is located at the touchdown point. As the UAV is landing, the laser altimeter on board automatically corrects the altitude errors in the navigation solution. A flare maneuver is initiated when the UAV’s altitude has passed an altitude threshold that is set dynamically using estimated navigation errors and descent rate data. Once the flare maneuver is initiated, the UAV’s descent rate is reduced, crosswind induced crab angles are automatically limited by the rudder control, and the auto throttle is disengaged and brought to idle.

2.3.2 Stellenbosch University’s ATOL System:

In [4] South Africa’s Stellenbosch University designed a lightweight, low-cost system using COTS components with all ATOL sensors on board the UAV. To offset the use of the COTS components that may not be very precise or reliable, special models and controllers were implemented for their platform. The sensors used included a low cost GPS receiver (4 Hz update rate) for location and navigation, absolute and differential pressure sensors for altitude and airspeed, low cost MEMS angular rate gyroscopes and accelerometers for attitude information, and ultrasonic radar sensor for higher altitude accuracy at lower altitudes.

In this system, autonomous take-off is separated into 5 different phases, and begins when the operator places the UAV at the end of the runway with its nose aligned to the proper runway heading. The placement of the UAV on the runway serves as both the starting point for take-off and is used as the aiming point for landing. The first phase commenced with the UAV receiving
the command to move with a given ground speed, while maintaining a constant yaw angle. Phase 2 regulates the cross track position relative to the runway until the centerline is within an acceptable distance from the rhumline. Phase 3 initiates the ground run phase and the UAV accelerates up to take-off velocity while regulating cross track position. Phase 4 consists of switching to airborne lateral controllers and activating the climbrate controller, and switching to phase 5 when a predetermined altitude is reached. Phase 5 transfers longitudinal control over to the airspeed and climb rate controllers. The take-off sequence is finished when the UAV is at a determined altitude above the runway.

The autonomous landing sequence is comprised of 7 phases which begins when the operator initiates the landing sequence via their GCS. Phase 1 guides the UAV back to the runway by means of flight controllers developed in [4] and calculates the wind speed and runway heading information. The UAV calculates the surrounding wind speed and direction to ensure the largest head wind component during landing to minimize ground speed upon touchdown. Phase 2 begins by executing a slow descent over the runway until a specified altitude, where the ultrasonic altitude sensor overrides the barometric altitude readings. Phase 3 guides the UAV to the approach point, in relation to the runway based upon the wind speed. Phase 4 reduces the UAV’s airspeed and tracks the runway centerline before beginning the descent, when the glide path has been crossed.
the system enters Phase 5. Phase 5 regulates glide path altitude from the aircraft’s range to the glide path origin. Phase 5 also regulates the aircraft’s cross-track position to the runway’s centerline. Phase 6 commences once the aircraft goes below an altitude of 5 meters and the ultrasonic sensor takes over altitude measurements. Once the flare altitude has been reached, the aircraft closes its throttle and is given 0 meters/second climb rate command. Finally, once the aircraft has touched down on the runway, and comes to a standstill, the landing sequence is considered complete.

Figure 9: Stellenbosch University’s Landing Sequence [4].

2.3.3 Seoul National University’s ATOL using only GPS:

In [22] Seoul National University implemented an autonomous take-off, taxiing, and landing algorithm without any IR sensors or accelerometer/gyroscopes for attitude information.
They accomplished this using only a single-antennae GPS receiver, DGPS reference station and an air speed sensor by way of a Pitot tube.

To substitute for attitude information, that would traditionally be given by IR sensors or inertial sensors such as accelerometers/gyros, an estimator was developed that would estimate the “pseudo-roll” angle, and “pseudo-pitch” angle of the aircraft based upon the GPS velocity measurements using a Kalman filter.

The UAV used for flight tests was a fixed-wing twin tail-boom aircraft with a 48 cc gasoline propeller engine. The UAV’s wing-span was 2.5 meters and weighed approximately 13 kg with payload. The flight control computer consisted of a COTS PC/104 module, while a DGPS reference station was positioned at the ground station to improve position accuracy.

In this system, autonomous takeoff is implemented using runway track and climbout controllers. The single-antenna GPS estimator starts to estimate attitude information when the UAV velocity becomes more than a pre-specified value. The runway track controller maintains the UAV’s alignment with the runway centerline, while maintaining a neutral roll control to keep the UAV from rolling over. During takeoff, the climb rate controller is used for longitudinal control while maintaining full throttle, and heading control is used for lateral control. Lift-off is dependent upon the airspeed and the climb rate. The climbout controller controls the climb rate of the UAV using the elevator control surfaces. The climbout is considered completed once the UAV achieves a pre-specified altitude, at which time the longitudinal controller switches to altitude control. Once the take-off sequence is completed, the UAV switched to waypoint mode. For horizontal path control, straight-line and circular controllers are used for used for linear and circular waypoint navigation.
Upon exiting waypoint mode, the aircraft switches to landing mode. Autonomous landing consists of a glide slope and flare mode. The glide path mode consists of the aircraft following a curved glide slope approach path, the end of which the aircraft is in line with the runway. After reaching a pre-defined altitude, airspeed, pitch, and altitude is controlled such that the aircraft touches down on the runway. In simulation and flight tests of this system, during the flare phase, the elevator input oscillated because the single-antenna GPS estimator did not provide true pitch values. The oscillation was solved by applying a low-pass filter to the elevator input. Due to the inaccuracy of DGPS vertical position accuracy, ground levels of touch down points were not consistent during several flight tests, but could be corrected with an ultrasonic sensor or a carrier-phase DGPS signal. In the end, Seoul University maintains that a single-antenna GPS receiver can be used as the main sensor for a low-cost UAV.

2.3.4 Brigham Young University MAV Autonomous Landing:

One of the largest issues for autonomous landing is determining the height above ground (HAG) when the aircraft is in its glide path or flare phases. To solve this issue, in [23]Brigham Young University investigated and field tested a HAG estimator using an optical flow sensor.

The landing algorithm developed and tested consists of two user defined waypoints: the approach and landing waypoints, respectively. The miniature air vehicle (MAV) orbits around the approach point and descends to a specified altitude, where it breaks from a circular descent path and follows a glide slope to the landing point.

The approach point is defined by the following parameters: the airspeed during the spiral descent path prior to entering the glide slope, the radius of the spiral descent, the starting altitude
to which the plane climbs or descends en route to the approach point, the end altitude where the aircraft exits the spiral descent and begins its glide slope. The approach waypoint is determined based upon the relative east coordinate from around which the plane orbits while descending from its starting altitude to its ending altitude, the distance north around which the plane orbits while descending from its starting altitude to its ending altitude, and finally the descent rate in meters/sec at which the plane descends while orbiting the approach point.

The landing point is defined by the following parameters: the airspeed during the glide slope from the approach point to the landing point, the flare height or the height above ground at which the aircraft cuts the throttle and attempts to hold zero roll while maintaining its glide slope by commanding pitch to control the desired altitude. The landing point also depends on the relative east and north coordinate from home at which the airplane should touch down.

For the portion of the landing sequence where the plane was heading towards the approach point, if the measured altitude is within a preset variation of the target altitude, then pitch from altitude and throttle from airspeed controllers are used. If the actual altitude is greater than the present distance from the desired altitude and less than the desired altitude then the pitch from airspeed controller is used with full throttle. Finally, if the actual altitude is greater than the present distance from the desired altitude and greater than the desired altitude, then the pitch from airspeed controller is used with zero throttle. Throughout the entire landing sequence, until the flare phase, heading is controlled using the roll from heading PID controller.

The glide slope consists of a simple linear glide slope calculated by decreasing the desired altitude as a function of distance from where the glide slope begins. This allows the desired altitude to be driven negative for distance from the initiation of the glide slope that are farther than the
desired landing point, making this robust with respect to negative offsets between measured HAG and actual HAG.

Optic flow sensors can be used to estimate HAG by relating the flow of features across an image array to the speed the image array is moving past the surface it is imaging and its distance from the surface. The number of pixels that a given object moves in the imaging plane can be combined with speed data from a MAV’s IMU or GPS to determine the HAG. The sensor that Brigham Young used was the Agilent ADNS-2610 optic flow sensor that runs at 1500 frames per second. The ADNS-2610 measures the flow of features across an 18 by 18 pixel imager and outputs $dx$ and $dy$ representing the total optic flow across the sensor’s field of view in both the x and y directions respectively. The flow data in the x direction can be combined with data from an IMU and GPS to determine HAG (h) using the equation below,

$$h = \frac{\delta x}{2 \tan \left( \frac{\gamma f o v}{2p_n} - \frac{\theta T_s}{2} \right)} \times \cos \theta \cos \varphi$$

where $T_s$ is the sampling period, $\gamma f o v$ is the average number of pixels in the field of view, $p_n$ is the number of pixels in the imaging array in the direction of motion of the sensor, $\dot{\theta}$ is the average pitch rate of the sensor over the sampling period, and $\varphi$ is the average roll of the sensor over the sampling period. Testing the optical flow sensor vs. a laser range finder showed that they were able to obtain acceptable attitude estimates for altitudes lower than 40 meters.

BYU’s flight test hardware consisted of an airframe with a 1.5 meter wingspan constructed of EPP foam and covered with Kevlar. The airframe can carry a 12 ounce payload and has a flight time of 30 minutes. The electronics consist of the Kestrel autopilot, batteries, a 900 MHz radio modem, a 12 channel GPS receiver, three optical flow sensors, video transmitter, a laser range finder with a range of 400 meters, and a small analog video camera.
Using the optic flow sensors, BYU was able to have twenty-seven consecutive autonomous landings from a single approach point. The average distance between the desired and actual touchdown points for landing using the optic flow sensors was 4.3 meters, representing over a 50% improvement over the average distance from the desired touchdown point using a previously used barometric altimeter.

2.3.5 Tohoku University’s ATOL:

In [24]Tohoku University designed and implemented autonomous takeoff and landing algorithms using visual feedback via OpenCV and a camera mounted on the fuselage.

The platform chosen to host the ATOL algorithms was a commercially available Hyperion Sniper 3D with an Armadillo-500FX Linux board that processed the data from the sensors and send commands to the servo controller. The servo controller was a Pololu MC013 received the data from the Linux board and sent PWM signals to the servo motors to control the control surfaces. The Microstrain 3DM-GX2 was used for attitude detection with angle accuracy of +/- 2 degrees. An ultrasonic sensor was used to detect altitude values less than 6 meters from the ground. A Garmin GPS 18LVC gave position information and absolute altitude, while a Logicoool Qcam Pro USB camera was used to detect the landing strip during the landing phase of flight.

During the autonomous takeoff phase, the throttle was kept open to maximize the aerodynamic lift and upward thrust necessary to perform takeoff. A reference pitch angle of 10 degrees was regulated by the elevator during the takeoff phase.
During the landing phase, if the UAV was at a greater altitude than what was desired, the UAV descended gradually by having a negative pitch angle and decreasing throttle. If the altitude was lower than the necessary value, the UAV would pitch up and increase the throttle value.

The technique for autonomous landing using visual feedback was finding the runway centerline via a captured image from the camera mounted on the fuselage. The autonomous landing strategy using image processing was as follows: first, the USB camera captured the image of the runway; second, the landing program binarized the captured image to monochrome to reduce processing time; third, the feature points of the runway centerline were detected by edge detection; fourth, the runway centerline was determined by Hough transformation techniques; fifth, the direction of the centerline was calculated and the UAV changed its direction toward the centerline; and finally the UAV attempted landing if the altitude was lower than 0.75 meters.

2.3.6 SAAB’s SHARC Demonstrator:

In [25] SAAB developed a military grade SHARC platform that is able to fly manually as well as autonomous mode, which is capable of fully autonomous takeoff and landing. When flying autonomously, safe guards were implemented that allowed for return to base (RTB) mode if link was lost between the GCS or from RC link with the safety pilot.

The SHARC TD is a 60kg jet engine driven aircraft with a fixed robust tricycle landing gear, COTS components were used as much as possible – engine, servos, valves, etc. The payload consisted of a forward looking color video camera for use in beyond visual range (BVR) flight. GPS was obtained using DGPS. The avionics were designed and manufactured by SAAB, based upon the Flight Test Instrumentation COMET 15 used in the Gripen and Viggen fighters.
The autonomous takeoff procedure was designed as follows: first, the operator lines up the aircraft in proximity to the runway’s centerline; second, after obtaining takeoff clearance the operator selects “AUTO” mode; third, the breaks, acceleration, rotation, and climb occurs autonomously while maintaining minimal lateral distance from the runway centerline until the UAV reaches a pre-determined altitude of 50 meters; The operator is able to abort the autonomous takeoff process at any time prior to the rotation phase, by applying the brakes until the aircraft comes to a standstill, however once the aircraft is in the rotation phase and the takeoff needs to be aborted, the operator can simply switch back to manual mode and control the aircraft manually.

The autonomous landing procedure was designed as follows: first, a precision flight path following mode is engaged around 2 km before the appointed touchdown point (video goggles allowed for BVR flight), where at 150 meters the aircraft tracks a descent path of 4 degrees aligned with the runway’s centerline. At 30 meters in altitude, the flight path changes in order to keep a glide angle of 2 degrees, if a link loss below 30 meters in detected, the landing continues, else the aircraft goes into RTB mode. At 4 meters in altitude, the FCS switches to a vertical speed mode, holding a constant vertical speed of -1.2 meters/sec until touch down – no flare is attempted. Touchdown is detected by angular speed sensors mounted on the wheels of the main landing gear, and when the aircraft has determined it is on the ground, the engine is set to idle and the breaking phase is initiated.

Hardware in the loop simulations were performed by connecting the BS-001 aircraft to a SUN workstation. All of the aircraft sensors were disconnected and replaced by digital inputs generated by the simulation from the SUN workstation. The inputs to the workstation consisted of the positions of the control surfaces of the aircraft, measured by potentiometers, allowing for all avionics to be tested in a more realistic environment. The sensor models included noise
properties, along with the typical blocks composing a flight simulator – aerodynamics, engine performance, landing gear, atmospheric and turbulence data. The simulator possessed the capability to introduce simulated failures such as engine flame-out, sensor failure, GPS failures, etc. Also, a ground effect model was added based upon accurate aerodynamic analysis of the available takeoff and landing data collected during previous flights. The analysis showed that ground effects were very noticeable during the takeoff portion – influencing the rotation phase, while it was negligible during the landing phase.

Ground tests included manual taxiing; autonomous taxiing where the aircraft was lined up on the runway’s centerline and autonomous accelerations were tested, interrupted at increasing values of ground speed by autonomous decelerations; autonomous taxiing with a side offset where the same test points from the previous test but the aircraft started with a lateral offset from the runway’s centerline from 5 to 10 meters; and finally autonomous taxiing with heading offset where the same test points were used but started with a heading offset from 10 to 30 degrees relative to the runway’s centerline.

Flight tests took place at a testing range in southern Sweden in a restricted and controlled airspace. To provide flexibility to the software during testing, Flight Test Functions (FTTs) were employed, where a large number of parameters necessary to the control laws were listed in text files so they could be edited without having to recompile the software. The flight test program consisted of the following: high speed rolls, both manual and autonomous, to tune the roll and yaw controllers; manual landing patterns to collect data from a Miniature Radar Altimeter (MAR); autonomous landing “on the cloud” where complete autonomous landing was tested with a 30 meter altitude offset on the nominal flight path, once touch down point was reached at 30 meters the operator could take make manual control and perform a manual landing; and finally complete
autonomous flight including autonomous takeoff, pre-programmed navigation routs, and autonomous landing. Flight tests were successful and were carried out under several wind conditions, during which the ATOL controllers showed repeatable and robust behavior.

### 2.3.7 Cloud Cap Technology’s Piccolo II Autopilot:

The Piccolo II Unmanned Avionics System has the ability to perform two different types of autonomous landing and one type of autonomous takeoff. The Piccolo II offers an autonomous landing software option that interfaces with a laser altimeter which provides accurate altitude information allowing the vehicle to perform a soft flared landing, but the laser altimeter is sold separately. The second autonomous landing option comes in the form of DGPS autonomous landing, this extends the landing performance by using 2 cm accuracy DGPS, provided by NovAtel, and provides autonomous taxiing, rolling takeoff, and landing. The Piccolo II Autopilot is a military grade FCS which costs around $15,000 making it out of the price range for this project. However, in [26]Embry-Riddle Aeronautical University incorporated the Piccollo II on a flight platform for use in a student UAV competition.

\[\text{Figure 10: Cloud Cap Technology Piccolo II Autopilot [27].}\]
2.4 This Project’s Aircraft Platform, FCS, GCS, and HILS:

The target aircraft platform used in the implementation of this project’s ATOL system is a Mig style Styrofoam UAV using a detachable wing and a tricycle undercarriage. This airframe was chosen over some of the more popular balsa wood platforms in that the Styrofoam airframe is relatively cheap and easy to repair or replace. The tricycle undercarriage is also better suited for take-off and landing when compared to tail wheel type undercarriages that are more difficult to steer on the ground, and gliders that need to be hand launched and land on the underside of their fuselage. Figure 11 shows the Mig aircraft.

![Figure 11: Carl Goldberg Mig Airframe.](image)

The FCS for which the ATOL algorithms are developed is an in house designed, low-cost custom made FCS known as the “miniFCS”. The miniFCS is built around an AVR32 microcontroller incorporating the NMEA standard GPS for position information, absolute and differential pressure sensors for altitude and airspeed information, and three axis thermopiles for
roll and pitch estimation [6]. Communication with the ground control station is accomplished using 900 MHz XBEE modules. The miniFCS controls the direction of flight by using a system of cascaded PID loops that control airspeed, attitude, and heading, by controlling roll, pitch, and yaw. Figure 12 provides an image of a fully populated miniFCS flight control system.

![Image of miniFCS developed at VCU.](image)

The GCS is run on a laptop and provides the operator with a satellite image of the flight test area, along with allowing the operator to input and change waypoints, PID parameters and navigation and flight settings. Figure 13 illustrates the GCS with the satellite image of where field testing is done, with the runway centerline marked from South to North. The left hand side of the GCS illustrates the vehicle status such as plane link, GPS lock, mode of operation, position, airspeed, and attitude. The bottom of the GCS provides visual indicators for heading, airspeed, artificial horizon, altitude, and climb rate information respectively.

![Image of GCS interface with satellite image and vehicle status.](image)
Validation of the algorithms implemented on the miniFCS is accomplished by testing the miniFCS on a custom built hardware in the loop simulator (HILS) board. The HILS board provides an interface between the miniFCS and a software flight simulator running on a separate computer. This HILS board uses the aircraft state information from the flight simulator to simulate GPS coordinate inputs, absolute and differential pressure values, and the three axis thermopile values.
that are directed as inputs into the miniFCS. The miniFCS processes these input values and produces the appropriate servo command PWM values that are redirected back to the HILS board. The HILS board then converts these servo commands into control commands that are passed back to the flight simulator as inputs. The flight simulator then uses its flight dynamics model to update the aircraft state and the cycle begins again. The flight simulator used is the open source flight simulator FlightGear. The HILS board communicates via UDP packets with the FlightGear simulator. Testing algorithms on the HILS serves the dual purpose of allowing rapid development and testing of new flight control algorithms and serving as a means to conduct realistic tests while not having to perform actual field flight tests. Figure 14 illustrates the hardware schematic of the miniFCS, HILS, and FlightGear simulator integration.

![Hardware connection between the miniFCS, HILS, and FlightGear](image)

**Figure 14:** Hardware connection between the miniFCS, HILS, and FlightGear [28].
Chapter 3: ATOL Navigation and Waypoint Calculations

This chapter describes the navigation for takeoff and landing. This includes the calculations necessary to determine the distance between two waypoints given their GPS coordinates, the target glidepath altitude, the measured glidepath and landing bearings, and finally the approach leg and approach/glidepath waypoints with a runway showing all variables. The chapter ends with an example runway with all calculated waypoints, bearings, and altitude values.

3.1 Navigation for Autonomous Takeoff and Landing:

The general approach for takeoff and landing navigation is to create a series of waypoints in relation to the runway, based upon the direction the GCS operator sets for the aircraft’s takeoff and consequently the aircraft’s approach to the runway for landing. These waypoints are all based on the start and end of the runway position, the bearings from the start of the runway to the end of the runway and vice versa. The landing navigation sequence’s waypoints depends not only on the start and end of the runway points, but also on the calculated start of the glidepath altitude. The start of the glidepath altitude is determined by the target glidepath angle and the target maximum distance from the start of the runway that are both set by the GCS operator before takeoff and landing are initialized.
3.1.1 Navigation for Autonomous Takeoff:

Takeoff navigation starts by placing the aircraft at the starting position of the runway. Once the takeoff command is initiated the aircraft fully opens the throttle and the aircraft begins moving down the runway. PID loops are used to ensure the aircraft maintains proper yaw control so that the aircraft does not deviate too far off the runways centerline. Once the aircraft has reached a ground speed set by the GCS operator, the aircraft pitches up and begins gaining altitude. Once a variable climbout and cruising altitude has been reached the aircraft has completed its takeoff navigation. Figure 15 illustrates simple takeoff navigation with the start and end of the runway waypoints shown.

![Figure 15: Simple Takeoff Navigation in Relation to the Runway](image)

3.1.2 Navigation for Autonomous Landing

Prior to the start of landing navigation, the GCS operator must first choose a direction for the aircraft to approach the runway’ glidepath waypoint, either right or left in relation to the starting position of the runway. This direction of approach also dictates the direction of turns that the
aircraft is constrained to. For instance, if a right hand approach is selected by the GCS operator, the aircraft must approach the runway’s glidepath from its right hand side while only making right hand turns.

Landing navigation begins by having the aircraft exit its previous navigation mode (loitering, waypoint, cross-track, etc.) and traveling to one of the traffic pattern waypoints shown in Figure 15 that make up a traffic approach circuit around the runway. Once the aircraft is in the traffic pattern, the aircraft will travel in the direction and at the altitude that the GCS operator chose prior to initiating landing mode. For a right hand approach, the aircraft will continue to fly the traffic pattern until it has passed at least two traffic pattern waypoints, if the next traffic waypoint is waypoint #1 the aircraft will descend to the target glidepath altitude, the next waypoint for a right hand approach is waypoint #2, after it has reached waypoint #2 it will navigate to the glidepath waypoint, at which time it will line up with the runway and start its glidepath descent.

For a left hand approach, the aircraft will continue to fly the traffic pattern until it has passed at least two traffic pattern waypoints, if the next traffic waypoint is waypoint #4 the aircraft will descend to the target glidepath altitude, the next waypoint for a left hand approach is waypoint #3, after it has reached waypoint #3 it will navigate to the glidepath waypoint, at which time it will line up with the runway and start its glidepath descent. A simple runway with traffic pattern landing navigation and glidepath waypoint is shown in Figure 16.
3.2 Glidepath Altitude and Waypoint Calculations:

Before calculating any variables, it becomes necessary to define the units that are to be used for each variable. The GCS operator enters in all distances in feet, GPS coordinates in decimal degrees, angles in degrees, and speeds in knots. Once the values are set by the GCS operator, those values are converted by the miniFCS automatically to distance in meters, GPS coordinates in radians, angles in radians, and speed in meters per second. The equations below will be described as if the quantities were entered in by the GCS operator. The following variables listed provide the variable name and units that are used in the calculations for this chapter.
• $x$ – latitude, decimal degrees
• $y$ – longitude, decimal degrees
• $\beta$ – glidepath bearing, degrees
• $\Omega$ – “anti glidepath bearing”, degrees
• $R$ – Earth’s radius, feet
• $d/d_{max}$ – distance and maximum distance from the runway, feet
• $a/a_{start}$ – altitude and start of glidepath altitude, feet
• $\alpha$ – target glidepath angle, degrees

The approach/glidepath waypoint is the first waypoint that is calculated and is the basis for computing the approach leg waypoints. Before the approach/glidepath waypoint can be analyzed, the bearing from the end of the runway to the start of runway must be determined in order to have the necessary glidepath bearing. This is defined in Eq. 1, where the glidepath bearing [29] is $\beta$, the “from longitude” (end of the runway longitude) is $x_1$, the “towards longitude” (start of the runway longitude) is $x_2$, the “from latitude” (end of the runway latitude) is $y_1$, and finally the “towards latitude” (start of the runway latitude) is $y_2$.

$$\beta = \tan^{-1}\left(\frac{(\sin(x_2 - x_1)) \cdot \cos y_1}{(\cos y_1 \cdot \sin y_2) - (\sin y_1 \cdot \cos y_2 \cdot \cos(x_2 - x_1))}\right)$$

(1)

The approach/glidepath waypoint is calculated by using the start of the runway latitude $y_2$ and longitude $x_2$, $R$ is the earth’s radius, and $d_{max}$ is the maximum distance from the start of the runway set by the GCS operator. The calculation that provides the approach/glidepath waypoint latitude $y_3$ and longitude $x_3$ is seen in Eq. 2.
\[ y_3 = \sin^{-1}\left(\sin y_2 \cos \frac{d_{\text{max}}}{R} + \cos y_2 \sin \frac{d_{\text{max}}}{R} \cos \beta\right) \]

\[ x_3 = x_2 + \tan^{-1}\left(\frac{\sin \beta \sin \frac{d_{\text{max}}}{R} \cos y_2}{\cos \frac{d_{\text{max}}}{R} - \sin y_2 \sin y_3}\right) \]

(2)

In order to calculate the first and fourth traffic pattern approach waypoints, a new longitude \( x_4 \) and a new latitude \( y_4 \), must first be calculated that use the bearing \( \Omega \) which is 180 degrees opposite \( \beta \) so that the waypoint is at the end of the runway, and that is the maximum distance \( d_{\text{max}} \) from the end of the runway waypoint \((x_1, y_1)\). Equation 3 describes the calculation for both \( x_4 \) and \( y_4 \).

\[ y_4 = \sin^{-1}\left(\sin y_1 \cos \frac{d_{\text{max}}}{R} + \cos y_1 \sin \frac{d_{\text{max}}}{R} \cos \Omega\right) \]

\[ x_4 = x_1 + \tan^{-1}\left(\frac{\sin \Omega \sin \frac{d_{\text{max}}}{R} \cos y_1}{\cos \frac{d_{\text{max}}}{R} - \sin y_1 \sin y_4}\right) \]

(3)

The first traffic pattern waypoint is calculated using \((x_4, y_4)\), the maximum distance set by the GCS operator is halved, and 90 degrees added to \( \Omega \), making it on the right hand side of the runway. Equation 4 describes the calculations necessary to determine the latitude and longitude for the first traffic pattern waypoint, where \( y_5 \) is its latitude and \( x_5 \) is its longitude.
\[ y_5 = \sin^{-1}\left( \sin x_4 \cos \frac{d_{\text{max}}}{2R} + \cos y_4 \sin \frac{d_{\text{max}}}{2R} \cos(\Omega + 90^\circ) \right) \]

\[ x_5 = x_4 + \tan^{-1}\left( \frac{\sin(\Omega + 90^\circ) \sin \frac{d_{\text{max}}}{R} \cos y_4}{\cos \frac{d_{\text{max}}}{2R} - \sin y_4 \sin y_5} \right) \]  

(4)

The fourth traffic pattern waypoint is calculated using \((x_4, y_4)\), the maximum distance set by the GCS operator is halved, and 90 degrees subtracted from \(\Omega\), making it on the left hand side of the runway. Equation 5 describes the calculations necessary to determine the latitude and longitude for the fourth traffic pattern waypoint, where \(y_6\) is its latitude and \(x_6\) is its longitude.

\[ y_6 = \sin^{-1}\left( \sin x_4 \cos \frac{d_{\text{max}}}{2R} + \cos y_4 \sin \frac{d_{\text{max}}}{2R} \cos(\Omega - 90^\circ) \right) \]

\[ x_6 = x_4 + \tan^{-1}\left( \frac{\sin(\Omega - 90^\circ) \sin \frac{d_{\text{max}}}{R} \cos y_4}{\cos \frac{d_{\text{max}}}{2R} - \sin y_4 \sin y_6} \right) \]  

(5)

The second traffic pattern waypoint is calculated using the glides path waypoints latitude and longitude \((x_3, y_3)\), the maximum distance set by the GCS operator is halved, and 90 degrees added to \(\Omega\), making it on the right hand side of the runway. Equation 6 describes the calculations necessary to determine the latitude and longitude for the second traffic pattern waypoint, where \(y_7\) is its latitude and \(x_7\) is its longitude.

\[ y_7 = \sin^{-1}\left( \sin x_3 \cos \frac{d_{\text{max}}}{2R} + \cos y_3 \sin \frac{d_{\text{max}}}{2R} \cos(\Omega + 90^\circ) \right) \]

\[ x_7 = x_3 + \tan^{-1}\left( \frac{\sin(\Omega + 90^\circ) \sin \frac{d_{\text{max}}}{R} \cos y_3}{\cos \frac{d_{\text{max}}}{2R} - \sin y_3 \sin y_7} \right) \]  

(6)
The third traffic pattern waypoint is calculated using the glidepath waypoints latitude and longitude \((x_3, y_3)\), the maximum distance set by the GCS operator is halved, and 90 degrees subtracted from \(\Omega\), making it on the left hand side of the runway. Equation 7 describes the calculations necessary to determine the latitude and longitude for the third traffic pattern waypoint, where \(y_8\) is its latitude and \(x_8\) is its longitude.

\[
y_8 = \sin^{-1}\left(\sin x_3 \cos \frac{d_{\text{max}}}{2R} + \cos y_8 \sin \frac{d_{\text{max}}}{2R} \cos(\Omega - 90^\circ)\right)
\]

\[
x_8 = x_3 + \tan^{-1}\left(\frac{\sin(\Omega - 90^\circ) \sin \frac{d_{\text{max}}}{R} + \cos y_3}{\cos \frac{d_{\text{max}}}{2R} - \sin y_3 \sin y_8}\right)
\]

(7)

The general equation for calculating the distance \(d\) between two waypoints can be seen in Eq. 8 below, where \(y_f\) is the “from latitude”, \(y_t\) is the “to latitude”, and \(x_f\) is the “from longitude”[29].

\[
d = 2R \sin^{-1}\left(\sqrt{\left(\frac{y_f - y_t}{2}\right)^2 + \cos y_f \cos y_t \left(\frac{x_f - x_t}{2}\right)^2}\right)
\]

(8)

The starting glidepath altitude \(a_{\text{start}}\) is determined by the target glidepath angle \(\alpha\) and maximum distance \(d_{\text{max}}\) that the GCS operator sets before initiating landing mode. Depending on how many traffic pattern waypoints, and the next traffic waypoint (#1 for right hand approach, and #4 for left hand approach) the aircraft sets its target altitude to \(a_{\text{start}}\) until it reaches the glidepath waypoint. Once the glidepath waypoint has been reached, the target glidepath altitude \(a\) is calculated using the
The variable $d_{atol}$ is calculated using Equation 8 and can be seen in Equation 9, where $y_{plane}$ and $x_{plane}$ represent the latitude and longitude of the plane’s current position respectively.

$$d_{atol} = 2R \sin^{-1}\left(\sqrt{\left(\sin\frac{y_{plane} - y_2}{2}\right)^2 + \cos y_{plane} \cos y_2 \left(\sin\frac{x_{plane} - x_2}{2}\right)^2}\right)$$

(9)

Figure 17: Target Glidepath Altitude Calculation

Figure 18 illustrates a more detailed runway, with all calculated bearings, altitudes, and waypoints with the approach direction set to the right side of the runway.
3.3 Example Runway with Calculated Variables:

This section provides an example runway that uses real world latitudes, longitudes, altitudes, bearings, etc. This example runway is used for field testing purposes and is located in Charles City, VA, where the majority of the flight tests were conducted. The orientation of the runway used in this example is running from South to North, but the opposite orientation (North to South) could be just as easily calculated, or any orientation for that matter, as long as valid start and ending locations for the runway are defined. All of the pertinent landing variables are calculated automatically during their appropriate phase provided that the GCS operator has input a start and end runway position, target glidepath angle, approach direction, and maximum approach distance from the start of the runway.

The starting GPS position of the runway, with an orientation going from South to North is (-77.2368887, 37.3353168) where $x_2$ is -77.2368887 and $y_2$ is 37.3353168. The ending GPS
position of the runway is (-77.2364650, 37.33708862) where $x_1$ is -77.2364650 and $y_1$ is 37.33708862. The GCS operator has the option to either “capture” the GPS coordinates for the start and ending positions of the runway, by having the aircraft at each end and then press a “capture” button to capture the start and end runway locations, or enter in the GPS coordinates manually, or choose from a list of predefined runway start and end positions. All runway coordinates are sent from the GCS in decimal degrees and automatically converted to radians by the miniFCS. A start and end GPS location is all that is needed for autonomous takeoff navigation; Figure 19 illustrates the example runway with start and ending GPS coordinates, and the aircraft ready to start autonomous takeoff.

![Figure 19: Charles City Runway Running From South to North](image)

For autonomous landing navigation, the next step is to calculate the glidepath bearing $\beta$. Using the start and end runway latitudes and longitudes, the glidepath bearing $\beta$ is calculated to be 10.75 degrees, which is roughly due north. The runway is then split into two (right and left) based upon the glidepath bearing, and the landing bearing (180 degrees it’s opposite), which can be seen in Figure 20. From here the runway is split into two by a runway centerline, every bearing
from the start of the runway, to the aircraft’s current position can be classified as either on the right
or left hand side of the runway.

Figure 20: Charles City Runway Bearings for South to North

Since the start of the runway coordinates \((x_2, y_2)\), the glidepath bearing \(\beta\), earth’s radius \(R\)
have been defined, all the GCS operator needs to do is to set a target glidepath angle \(\alpha\), a maximum
distance from the start of the runway \(d_{max}\), and a direction the aircraft should approach the runway.

In this example, the GCS operator sets target glidepath angle of 10 degrees for \(\alpha\), \(d_{max}\) as
800 feet from the start of the runway, and the aircraft should approach the glidepath waypoint from
its right side, making only right hand turns. Now that the target glidepath angle, maximum distance
from the runway, and approach direction have been set, the glidepath latitude $y_3$ and longitude $x_3$ are calculated to be -77.23739624 and 37.3331604 respectively, giving an glidepath/approach waypoint of (-77.23739624, 37.3331604).

Before the traffic pattern waypoints are calculated, the latitude $y_4$ and longitude $x_4$ that are $d_{max}$ away from the end runway must be calculated. Using Equation 3, with a distance of 800 feet from the end of the runway and a bearing 180 degrees opposite 10.75 degrees provides a latitude and longitude of (37.339264, -77.236022) $x_4$ and $y_4$ respectively. Now that the glidepath/approach waypoint, and the end of the runway’s latitude and longitude used for traffic waypoints #1 and #4 have been calculated, the traffic pattern waypoints, and target glidepath altitude can be calculated.

The first traffic pattern waypoint can now be calculated using Equation 4 and ($x_4, y_4$) as its reference longitude and latitude. The first traffic pattern adds 90 degrees to $\Omega$ in order to for it to be placed on the right hand side of the runway, with a halved maximum distance of 400 feet. The first traffic pattern waypoint’s longitude $x_5$ and latitude $y_5$ are calculated to be (-77.2345963, 37.3390388).

Equation 5 calculates the fourth traffic pattern waypoint using ($x_4, y_4$) as its reference longitude and latitude. The fourth traffic pattern subtracts 90 degrees from $\Omega$ in order to for it to be placed on the left hand side of the runway, with a halved maximum distance of 400 feet. The fourth traffic pattern waypoint’s longitude $x_6$ and latitude $y_6$ are calculated to be (-77.2373047, 37.3394508).

Equation 6 calculates the second traffic pattern waypoint using ($x_3, y_3$) as its reference longitude and latitude. The second traffic pattern adds 90 degrees to $\Omega$ in order to for it to be placed on the right hand side of the runway, with a halved maximum distance of 400 feet. The
second traffic pattern waypoint’s longitude $x_7$ and latitude $y_7$ are calculated to be (-77.2360458, 37.3329544).

Equation 7 calculates the third traffic pattern waypoint using $(x_3, y_3)$ as its reference longitude and latitude. The third traffic pattern subtracts 90 degrees from $\Omega$ in order to for it to be placed on the left hand side of the runway, with a halved maximum distance of 400 feet. The third traffic pattern waypoint’s longitude $x_8$ and latitude $y_8$ are calculated to be (-77.2387543, 37.3333702).

The final calculation necessary for autonomous landing navigation is the target start of the glidepath altitude $a_{start}$. The target start of the glidepath altitude is calculated using the maximum approach distance from the start of the runway and target glidepath angle as set by the GCS operator. For a target angle of 10 degrees and a maximum approach distance of 800 feet from the start of the runway, the starting target glidepath altitude is calculated to be 141 ft. This is the target altitude that is set as soon as the aircraft is placed in landing mode. Figure 21 illustrates the Charles City runway with its orientation from south to north, with a 10 degree target glidepath, 800 feet maximum distance from the runway, and a right hand landing approach.

Figure 21: Charles City Runway Running South to North with All Calculated Variables
Chapter 4: ATOL Algorithms

Even though there are several benefits of switching to an ATOL based system, implementation can pose several challenges. The largest challenge in developing an autonomous take-off and landing system is known as the localization problem, which is a matter of having the UAV know precisely its position relative to the runway, in elevation and lateral motion, when taking-off and landing.

A low cost solution to the localization problem is to use a single GPS receiver that has been thoroughly tested in addition to a low-altitude sensor. The GPS receiver while in motion gives relatively accurate, repeatable positional and velocity information, but while stationary tends to report fluctuating position and velocity. While the aircraft is on the ground and in takeoff mode, the navigation mode should use heading error as a means of heading correction as opposed to target cross-track which returns the distance from the runway centerline. The addition of a low-cost magnetometer provides a more accurate heading reference to use when the aircraft is stationary or a low speeds, as in the initial takeoff run. Similarly, the addition of a low cost sonar sensor (for altitudes less than 20 feet) combined with the absolute pressure sensor (for larger altitudes) improves the altitude localization precision so that it is suitable for an ATOL system.

Before the aircraft can takeoff autonomously, certain parameters must be set by the operator in order for the aircraft to have enough information to start the takeoff sequence. Takeoff parameters include:
- target ground speed at which the aircraft should takeoff – knots
- target pitch angle to provide the necessary angle of attack for lift off – degrees
- target climbout altitude – feet
- target climbout airspeed – knots

Runway parameters must also be updated for the aircraft to have the correct locations for the start and end of the runway. The runway parameters on the GCS allow the operator to either “capture” the runway coordinates by placing the aircraft at the beginning and end of the runway, manually enter the GPS coordinates for the start and end of the runway, or select a predefined set of coordinates based on the direction of the runway and the starting location of the aircraft. Once the runway coordinates have been set, the aircraft is ready for the operator to initiate takeoff.

After the runway has been defined, the GCS operator sets the appropriate gains for the PID controllers that are the basis for the autonomous takeoff and landing controllers. This next section provides a brief overview of the structure of the PID controls that are used for the autonomous takeoff and landing algorithms.

### 4.1 PID Control:

The PID controllers calculate an error as the difference between a measured process and a target set-point. The control output is a weighted sum of a proportional term, an integral term, and a derivative term.

The proportional term is a response to the current error, and can be adjusted by multiplying the current error by its corresponding proportional gain “\( K_p \)”. A proportional gain that is too high will cause oscillations, while a proportional gain that is too small will have a slow response time [6].
The integral term is proportional to the error and duration of the error and is adjusted using the integral gain “$K_i$”. This accelerates the process output towards the target set-point and eliminates steady state error. Having a $K_i$ gain that is too high can cause the process to overshoot the target set-point [6].

The derivative term compensates for integral overshoot by responding to the rate of change in the error and is adjusted by the derivative gain “$K_d$”. Figure 22 illustrates a classical PID controller.

![Figure 22: PID Controller [6]](image)

PID controllers are used to control the roll, pitch, yaw, climb/descent rate, and glidepath of the aircraft and are key for the aircraft to make a successful autonomous takeoff and landing.

### 4.2 Autonomous Takeoff Sequence:

Takeoff starts with the aircraft positioned on the runway with the heading pointed in the direction of the runway it will traverse. The aircraft is placed in the idle state until the takeoff command is initiated by the GCS operator. During the idle state, the aircraft’s target heading is
calculated based on the start and end of the runway waypoints, and sets the ailerons, elevator, and rudder set to neutral, and the throttle to fully closed.

Once the takeoff command is initiated, the aircraft enters the first phase of takeoff – takeoff initialize. In this state, the ground level is set, and the target heading is updated to the end of the runway. Once these instructions have been completed, the aircraft enters phase 2 – pre liftoff.

The pre-liftoff phase updates the target heading and changes the controllers to Takeoff control until the target groundspeed set by the GCS operator has been met. Takeoff control sets the target roll and pitch to zero, while the rudder controls heading and the throttle is set to fully open. The purpose of keeping the target roll to zero is to prevent the aircraft’s wings from banking into the ground and disrupting the takeoff sequence, and the target pitch is set to zero to prevent a premature liftoff. Once the target takeoff ground speed parameter has been achieved, the aircraft enters phase 3 – liftoff.

The liftoff state consists of updating the target heading and changes to a set of Liftoff controllers until a settable target climbout altitude is reached. The liftoff controllers consist of reasserting the target roll to zero, neutral rudder, and the pitch degree is set to a settable target angle specified by the operator, allowing the elevators to provide the necessary angle of attack to get the aircraft airborne. Once the aircraft is airborne and has reached the target climbout altitude set by the GCS operator, the aircraft transitions to the fourth phase – climbout.

The climbout phase consists of reasserting the target heading and switching to a set of climbout controllers until the target cruising altitude set by the GCS operator is reached. The climbout controllers have the ailerons control heading, pitch controls airspeed, neutral rudder. Once the aircraft has reached the target altitude set by the GCS operator, the aircraft transitions
back to all regular cruise controls and enters waypoint mode. Figure 23 illustrates the complete autonomous takeoff sequence.

**Takeoff**

![Diagram of Takeoff Sequence](image)

Figure 23: Complete Autonomous Takeoff Sequence

### 4.3 Autonomous Landing Sequence:

Before the landing mode is initiated, certain parameters must be set by the GCS operator. Landing parameters include:

- Target glidepath angle – degrees
- Target traffic pattern airspeed – knots
• target traffic pattern altitude – feet
• target glidepath airspeed – knots
• maximum distance between the glidepath waypoint and the start of the runway – feet
• target flare altitude – feet
• target flare climbrate – feet/sec
• target ground speed at which the plane should idle – knots
• the direction the aircraft should approach the glidepath and the direction the aircraft should turn when in the traffic pattern – right or left.

Landing begins when the operator initiates the landing command from the GCS. The aircraft’s previous state prior to entering the landing mode could be any number of modes ranging from waypoint to loiter. Once the landing command has been initiated, the aircraft enters the first phase of landing – landing initialization, which consists of calculating the approach glidepath waypoint (Eq. 2), all the traffic pattern waypoints (Eq. 4-8) that are all necessary for navigation to the start of the glide path, and the target start of the glidepath altitude. Once all the waypoints, and glidepath altitude have been calculated, the miniFCS sends the waypoint coordinates to the GCS so they are displayed on the satellite image of the test area enabling the GCS operator to ensure that the aircraft is going to its appropriate location. After the miniFCS sends the waypoint coordinates to the GCS, the aircraft exits its landing initialization mode and enters the second phase of landing – enter traffic pattern.

The enter traffic pattern mode sets the target airspeed and altitude to the traffic pattern airspeed and altitude set by the GCS operator respectively. Once the target airspeed and altitude have been set, the aircraft measures the distance from its current location to all traffic pattern waypoints. The aircraft then compares the distance from its location to all traffic pattern waypoints, and depending on the shortest distance and the direction set by the GCS operator,
decides which waypoint to travel to. If the GCS operator selected a right hand approach, the aircraft will navigate to the next closest waypoint for the right hand traffic pattern. For example, if the aircraft was closest to traffic pattern waypoint #1, the aircraft would navigate to traffic pattern waypoint #2. If the GCS operator selected left as the approach direction, the aircraft would navigate to the next closest waypoint for a left hand traffic pattern. For example, if the aircraft was closest to traffic pattern waypoint #1, the aircraft would navigate to traffic pattern #4.

Once inside the traffic pattern, the aircraft reasserts its target airspeed and altitude to the values set by the GCS operator. The aircraft then navigates to each waypoint in the traffic pattern based upon the direction the GCS operator set. After reaching each traffic pattern waypoint, a traffic pattern waypoint counter is incremented. The purpose of the traffic pattern waypoint counter is to ensure that at least two traffic pattern waypoints have been reached before the aircraft enters its glidepath phase allowing enough time for the aircraft to reach the target glidepath altitude. If the traffic pattern waypoint counter is greater than two, after the aircraft passes the first traffic pattern waypoint for a right hand approach, or the fourth traffic pattern waypoint for a left hand approach, the target altitude is set to the start of the glidepath altitude. This gives the aircraft the distance between either traffic pattern waypoints 1 and 2, or 4 and 3 (twice $d_{max}$ and the entire distance of the runway) to reach its target start of the glidepath altitude. Figure 24 illustrates the traffic pattern around the runway.
For instance, if the aircraft enters the traffic pattern starting with traffic pattern waypoint #1, and a right hand direction is given, the aircraft will then travel to the second, third, fourth and back to the first traffic pattern waypoint and increment the traffic pattern waypoint counter after reaching each waypoint. After the aircraft reaches the first traffic pattern approach it will navigate to the second traffic pattern waypoint; since more than two traffic pattern waypoints have been passed, the aircraft will then set its target altitude to the start of the glidepath altitude. Once the second traffic pattern waypoint is reached, the aircraft enters its next landing phase – approach. The same idea follows for a left hand approach. If the aircraft enters the traffic pattern at the first traffic pattern waypoint, and a left hand direction is chosen by the GCS operator, the aircraft will navigate to traffic pattern waypoints 4, 3, and 2, 1 and back to the 4 respectively, and increment
the traffic pattern waypoint counter for each passed waypoint. Once at traffic pattern waypoint #4 for the second time, the aircraft will set its target altitude to the start of the glidepath altitude enter the approach phase.

The approach phase of the autonomous landing mode guides the aircraft to the approach/glidepath waypoint \((x_3, y_3, a)\) and when the aircraft is within a target distance, the aircraft enters the next phase – glidepath. The purpose of the target distance is for the aircraft to start turning before it reaches the glidepath/approach waypoint to avoid having to make a sharp turn.

Once in the glidepath mode, the aircraft switches from waypoint navigation and begins cross-track navigation, where the aircraft tries to regulate its position based on the rhumb line from the glidepath waypoint to the end of the runway. The aircraft maintains traditional cruising controls and measures the distance from the start of the runway (Eq. 5) and tries to maintain target altitude (Eq. 6) from the target glidepath angle and maximum distance the GCS operator sets.

Once the aircraft has reached a target flare altitude, the aircraft transitions to the next phase – flare, which regulates the target heading and cross-track distance from the rhumb line while switching to a flare controller. Flare control consists of turning the throttle fully closed, having a target roll rate of 0 rad/s, returning heading control to the rudder, and altitude control is determined by a climb rate controller. The climb rate control regulates the descent of the aircraft; since the throttle is fully closed and the aircraft is descending, the plane will pitch up slightly, allowing the back tricycle landing wheels to make contact with the runway before the nose wheel does.
Once the aircraft has touched down and reached a target ground speed set by the GCS operator, the aircraft has completed its autonomous landing and switches to idle – where the throttle is fully closed, and all other control surfaces are placed in neutral. Figure 25 illustrates the entire autonomous landing sequence.

Figure 25: Complete Autonomous Landing Sequence
Chapter 5: HILS Testing

The Mig aircraft, miniFCS flight control system, and all the electronics onboard the Mig that are necessary for flight (servos, sensors, BECs, motor, etc.) combined can be relatively expensive; to ensure the safety of the system and its onboard electronics, the miniFCS and its ATOL algorithms must be thoroughly tested. In order to carry out the most realistic testing without having to do field tests, hardware in the loop simulations (HILS) were performed numerous times, under many different scenarios.

In order to have an accurate representation of the length and width of the runway it became necessary to create a custom FlightGear airfield with corresponding scenery using X-Plane [30] and TerraGear [31] respectively. Having a custom airport/scenery also allows the GCS to center the aircraft on a satellite image of the actual test field according to the GPS information relayed by FlightGear. Having an accurate representation of the runway used in field testing and the ability to use the satellite image of the test area, make for the most realistic testing without actually performing field tests, which requires many hours to complete and has the possibility of crashing during the course of testing.

Figure 26 illustrates the GCS and HILS interacting with the custom FlightGear airfield to perform HILS testing. The GCS satellite image shows the start of the runway as waypoint 0, and the end of the runway as waypoint 1 (using the same coordinates from Chapter 3 Section 3), with a rhumbline connecting the two, and dividing the runway into two portions. The runway
orientation for this scenario is starting in the south and running north. This chapter outlines HILS test results for autonomous takeoff and landing along with all parameters input to the GCS and a discussion on target performance and actual performance.

![Figure 26: GCS with Charles City Showing Runway South to North (Left) and Custom FlightGear Airport for Charles City (Right) Prior to Takeoff.](image)

**5.1 HILS Test – Autonomous Takeoff:**

The first HILS test discussed is the autonomous takeoff sequence for the runway shown in Figure 26, running south to north with no wind being added to the HILS simulation. Upon starting FlightGear the aircraft is in idle and positioned at the beginning of the runway. The GCS parameters input for the takeoff options are:

- Target takeoff airspeed of 25 knots
- Takeoff pitch angle of 15 degrees
- Target climbout altitude of 75 feet
- Target climbout airspeed of 40 knots
- Target cruising altitude of 150 feet
- Target ground speed to idle of 13 knots

The runway’s start and ending points are then defined by the miniFCS, after which the runway is divided in two by a rhumbline. The start of the runway’s location is (-77.2368851, 37.3353157) and the end of the runway’s location is (-77.2364655, 37.3370895). At this point the aircraft is ready to be put into autonomous takeoff mode. Figure 27 shows the aircraft during autonomous takeoff mode, after the aircraft has lifted off the runway.

![Figure 27: GCS with Charles City Showing the Aircraft Shortly after Liftoff (Left) and FlightGear Showing Aircraft after Liftoff from the Runway (Right)](image)

Figure 28 shows the different phases of takeoff mode vs the aircraft’s altitude. Takeoff mode 0 is takeoff initialization, takeoff mode 1 is pre liftoff, takeoff mode 2 is liftoff, takeoff mode 3 is climbout, and takeoff mode 4 is takeoff completion, where the aircraft has finished its autonomous takeoff and enters into waypoint mode where the aircraft increases its altitude to the
target waypoint altitude which is 400 feet in this test. The entire autonomous takeoff sequence takes approximately 16 seconds to complete, with the aircraft only on the runway for approximately 3 seconds before it lifts off. The altitude shown during takeoff mode 0 is a result of FlightGear resetting and not an actual altitude greater than 0 feet.

![Altitude During the Phases of Takeoff Mode](image)

**Figure 28: Aircraft’s Altitude During Different Takeoff Mode Phases**

As can be seen in Figure 28, the aircraft is on the runway during the takeoff initialization and pre-liftoff phases between the 61st and 65th time stamp (takeoff mode 0 and 1 respectively). Figure 29 shows the aircraft maintaining proper heading while on the runway prior to liftoff for takeoff modes phases 0 and 1. The maximum heading error is 0.74 degrees. Figure 29 illustrates the runway rudder controller maintain proper yaw/heading for the aircraft while the aircraft is on the ground.
Figure 30 shows the aircraft maintaining proper cross-track control while on the runway prior to liftoff (takeoff modes 0 and 1). The maximum cross-track error was 2 feet away from the runway centerline. With a runway width of approximately 54 feet, the cross track error is more than adequate. Figures 29 and 30 illustrate the runway rudder controller maintains more than adequate heading and cross-track position for the aircraft traverse down the runway while the aircraft is on the ground prior to liftoff.
Figure 30: Aircraft’s Cross-Track Error while on the Runway

Testing using the same conditions and parameters, except with an 8 knot wind speed blowing due east (90 degrees) was attempted, but the aircraft was pushed to the side of the runway, even while in idle, making autonomous takeoff HILS testing not practice with a wind speed and direction. This is most likely due to a different ground model in FlightGear, but in air testing with wind is still possible.

5.2 HILS Test – Autonomous Landing with w/o Wind and Right Direction:

The second HILS test discussed is the autonomous landing sequence with the same runway orientation as the previous section. The landing parameters set in the GCS were as follows:

- Target glidepath angle of 10 degrees
- Target traffic pattern airspeed of 40 knots
- Target traffic pattern altitude of 300 feet
- Target glidepath airspeed of 10 knots
- Maximum approach distance from the start of the runway of 800 feet
- Target ground speed to idle of 13 knots
- Target climb rate for the flare phase of 3 feet/sec
- Right hand turns around the traffic pattern

Figure 31 shows the target area, with all traffic pattern waypoints around the runway with the aircraft following a right hand traffic pattern. Figure 32 shows the aircraft just after it has exited glidepath mode and has entered flare mode, aligned with the runway centerline.

Figure 31: GCS Showing Aircraft Following Right Traffic Pattern (Left) and FlightGear Image Showing Aircraft Turning from Traffic Pattern Waypoint 3 to 4 Confirming Right Hand Turning (Right)
The graph in Figure 32 shows the aircraft’s altitude error and the aircraft’s distance from the start of the runway during the glidepath descent. The maximum altitude error measured is 15.4 feet at a distance of 606 feet from the start of the runway. This is an acceptable altitude error because the maximum altitude deviation is less than the starting altitude for the flare phase set by the GCS operator (20 feet), this means that even if the aircraft maintains a maximum altitude error of 15.4 feet throughout its entire glidepath phase, the aircraft will not miss the flare altitude and crash into the ground. As the aircraft moves closer to the runway, the altitude error decreases significantly.
Figure 33: Aircraft’s Altitude Error during Glidepath Descent

Figure 34 illustrates the cross-track error in feet vs the distance from the start of the runway during the glidepath phase. This figure shows that at 573 feet away from the start of the runway, the aircraft has its largest cross-track error of 15.6 feet away from the rhumbline. This is more than adequate longitudinal control for a runway that is 58 feet wide, and has 29 feet on either side of the runway centerline. Even if the aircraft maintains a maximum cross-track error through the entire glidepath phase, the aircraft still has enough room on either side of the rhumbline to land on the runway. As the aircraft decreases its distance to the start of the runway, the cross-track error is reduced significantly.
The flare portion of the landing process is the last phase that needs to be considered to ensure the aircraft lands safely and successfully on the runway. Figure 35 illustrates the climbrate error in feet/second for the flare controller. Between 2180 and 2181 seconds, the set of controllers are switching from glidepath control to flare control, and accounts for the large spike in climbrate error during this one second period. From 2181 seconds on, the figure shows the aircraft trying to maintain its target climbrate of 3 feet/second, but since the aircraft’s throttle is fully off, the aircraft will continue to lose altitude, which is correct, until it finally is on the ground and not climbing or descending at all.
5.3 HILS Test – Autonomous Landing with w/o Wind and Left Direction:

The third HILS test discussed is the autonomous landing sequence with the same runway orientation and landing parameters as the previous section except the traffic pattern is a left hand direction as opposed to a right hand direction.

Figure 36 shows the target area, with all traffic pattern waypoints around the runway with the aircraft following a left hand traffic pattern. Figure 37 shows the aircraft just after it has exited glidepath mode and has entered flare mode, aligned with the runway centerline.
The graph in Figure 38 shows the aircraft’s altitude error and the aircraft’s distance from the start of the runway during the glidepath descent. The maximum altitude error measured is 15.7 feet at a distance of 600 feet from the start of the runway. This is an acceptable altitude error.
because the maximum altitude deviation is less than the starting altitude for the flare phase set by the GCS operator (20 feet), this means that even if the aircraft maintains a maximum altitude error of 15.7 feet throughout its entire glidepath phase, the aircraft will not miss the flare altitude and crash into the ground. As the aircraft moves closer to the runway, the altitude error decreases significantly.

![Aircraft's Altitude Error during Glidepath Descent](image)

**Figure 38: Aircraft's Altitude Error during Glidepath Descent**

Figure 39 illustrates the cross-track error in feet vs the distance from the start of the runway during the glidepath phase. This figure shows that at 569 feet away from the start of the runway, the aircraft has its largest cross-track error of 15 feet away from the rhumbline. This is more than adequate longitudinal control for a runway that is 58 feet wide, and has 29 feet on either side of the runway centerline. Even if the aircraft maintains a maximum cross-track error through the
entire glidepath phase, the aircraft still has enough room on either side of the rhumbline to land on the runway. As the aircraft decreases its distance to the start of the runway, the cross-track error is reduced significantly.

Figure 39: Aircraft's Cross-Track Error during Glidepath Descent

The flare portion of the landing process is the last phase that needs to be considered to ensure the aircraft lands safely and successfully on the runway. Figure 40 illustrates the climbrate error in feet/second for the flare controller. Between 221 and 222 seconds, the set of controllers are switching from glidepath control to flare control, and accounts for the large spike in climbrate error during this one second period. From 222 seconds on, the figure shows the aircraft trying to maintain its target climbrate of 3 feet/second, but since the aircraft’s throttle is fully off, the aircraft
will continue to lose altitude, which is correct, until it finally is on the ground and not climbing or descending at all.

![Aircraft's Climbrate Error during Flare Phase](image)

**Figure 40:** Aircraft’s Climbrate Error during Flare Phase

### 5.4 HILS Test – Autonomous Landing with w Wind and Right Direction:

The fourth HILS test discussed is the autonomous landing sequence with the same runway orientation and landing parameters as the previous right hand landing test except there is an 8 knot wind speed coming from a 45 degree direction.
Figure 41 shows the target area, with all traffic pattern waypoints around the runway with the aircraft following a right hand traffic pattern. Figure 42 shows the aircraft just after it has exited glidepath mode and has entered flare mode, aligned with the runway centerline.

Figure 41: GCS Showing Aircraft Following Right Traffic Pattern (Left) and FlightGear Image Showing Aircraft Turning from Traffic Pattern Waypoint 3 to 4 Confirming Right Hand Turning (Right)

Figure 42: GCS Satellite Image Showing Aircraft during Glidepath and Flare Mode (Left) and FlightGear Image Showing Aircraft in Flare Mode Centered on the Runway (Right)
The graph in Figure 43 shows the aircraft’s altitude error and the aircraft’s distance from the start of the runway during the glidepath descent. The maximum altitude error measured is 8.5 feet at a distance of 533 feet from the start of the runway. This is an acceptable altitude error because the maximum altitude deviation is less than the starting altitude for the flare phase set by the GCS operator (20 feet), this means that even if the aircraft maintains a maximum altitude error of 8.5 feet throughout its entire glidepath phase, the aircraft will not miss the flare altitude and crash into the ground. As the aircraft moves closer to the runway, the altitude error decreases significantly.

![Aircraft's Altitude Error during Glidepath Descent](image)

Figure 43: Aircraft's Altitude Error during Glidepath Descent

Figure 44 illustrates the cross-track error in feet vs the distance from the start of the runway during the glidepath phase. This figure shows that at 740 feet away from the start of the runway,
the aircraft has its largest cross-track error of 48 feet away from the rhumbline. If is maximum cross-track error were maintained for the entirety of the glidepath phase, the aircraft would not be able to land on the runway that is 58 feet wide, and has 29 feet on either side of the runway centerline. The extra cross-track error is due to the wind speed and direction. Thankfully, the closer the aircraft gets to the runway, the cross-track error decreases significantly, to a point that at 229 feet away from the runway, the aircraft has a cross-track error of 22 feet from the runway centerline, which is adequate for the given runway.

![Aircraft's Cross-Track Error during Glidepath Descent](image)

**Figure 44: Aircraft’s Cross-Track Error during Glidepath Descent**

The flare portion of the landing process is the last phase that needs to be considered to ensure the aircraft lands safely and successfully on the runway. Figure 45 illustrates the climbrate
error in feet/second for the flare controller. Between 230 and 231 seconds, the set of controllers are switching from glidepath control to flare control, and accounts for the large spike in climbrate error during this one second period. From 231 seconds on, the figure shows the aircraft trying to maintain its target climbrate of 3 feet/second, but since the aircraft’s throttle is fully off, the aircraft will continue to lose altitude, which is correct, until it finally is on the ground and not climbing or descending at all.

Figure 45: Aircraft’s Climbrate Error during Flare Phase
5.5 HILS Test – Autonomous Landing with w Wind and Left Direction:

The fifth and finally HILS test discussed is the autonomous landing sequence with the same runway orientation and landing parameters as the previous left hand landing test except there is an 8 knot wind speed coming from a 45 degree direction.

Figure 46 shows the target area, with all traffic pattern waypoints around the runway with the aircraft following a left hand traffic pattern. Figure 47 shows the aircraft just after it has exited glidepath mode and has entered flare mode, aligned with the runway centerline.

Figure 46: GCS Showing Aircraft Following Left Traffic Pattern (Left) and FlightGear Image Showing Aircraft Turning from Traffic Pattern Waypoint 4 to 3 Confirming Right Left Turning (Right)
The graph in Figure 48 shows the aircraft’s altitude error and the aircraft’s distance from the start of the runway during the glidepath descent. The maximum altitude error measured is 8.5 feet at a distance of 575 feet from the start of the runway. This is an acceptable altitude error because the maximum altitude deviation is less than the starting altitude for the flare phase set by the GCS operator (20 feet), this means that even if the aircraft maintains a maximum altitude error of 8.5 feet throughout its entire glidepath phase, the aircraft will not miss the flare altitude and crash into the ground. As the aircraft moves closer to the runway, the altitude error decreases significantly.
Figure 49 illustrates the cross-track error in feet vs the distance from the start of the runway during the glidepath phase. This figure shows that at 777 feet away from the start of the runway, the aircraft has its largest cross-track error of 38 feet away from the rhumbline. If is maximum cross-track error were maintained for the entirety of the glidepath phase, the aircraft would not be able to land on the runway that is 58 feet wide, and has 29 feet on either side of the runway centerline. The extra cross-track error is due to the wind speed and direction. Thankfully, the closer the aircraft gets to the runway, the cross-track error decreases significantly, to a point that at 148 feet away from the runway, the aircraft has a cross-track error of 21 feet from the runway centerline, which is adequate for the given runway.
The flare portion of the landing process is the last phase that needs to be considered to ensure the aircraft lands safely and successfully on the runway. Figure 50 illustrates the climbrate error in feet/second for the flare controller. Between 228 and 229 seconds, the set of controllers are switching from glidepath control to flare control, and accounts for the large spike in climbrate error during this one second period. From 229 seconds on, the figure shows the aircraft trying to maintain its target climbrate of 3 feet/second, but since the aircraft’s throttle is fully off, the aircraft will continue to lose altitude, which is correct, until it finally is on the ground and not climbing or descending at all.
5.6 HILS Simulation Conclusions:

As can be seen from the above HILS simulations, the aircraft performs successful takeoff and landing under several different directions of approach and under different weather conditions. The simulations with wind conditions have a higher cross-track error during the glidepath phase, but this could be brought down with different PID parameters than were used for the no wind simulations.
Chapter 6: Conclusions and Future Work

The first section of this chapter discusses the conclusions of the thesis project, ensuring that it is a feasible, low-cost system, and that HILS results are consistent with the constraints for longitudinal and altitude control for autonomous takeoff and landing.

The second section of this chapter discusses the future work, and any improvements that could be done to make this system more robust and reliable.

6.1 Conclusions:

Due to the inexpensive nature of the miniFCS and its components and the choice of a COTS ultrasonic sensor (around $30) the entire system retains its low-cost nature of less than $200. After examining the HILS simulations of Chapter 5, the aircraft maintains its target altitude, and adequate heading and cross-track error for a successful autonomous takeoff. Autonomous landing is also successfully accomplished by having adequate longitudinal and attitude control (Figures 33-36). Therefore, in simulation, this thesis has successfully accomplished a low-cost implementation of autonomous takeoff and landing algorithms for a fixed wing UAV, making the miniFCS a fully autonomous system from takeoff to landing. Also due to the setup of the traffic pattern waypoints, and the way the runway is defined by either capturing or entering in the runway coordinates, this system works for any runway orientation, making this a robust system.
6.2 Future Work:

There are several aspects of this project that could be improved further, first and foremost is successful flight testing. Even though HILS simulations performed more than adequately for successful autonomous takeoff and landings, actual flight have presented many problems.

One of the challenges that arises during actual flight testing is that the IR sensors that the miniFCS use for attitude estimation are very weather dependent, where an overcast day could interfere with attitude estimation, also IR attitude estimation is not as accurate as an IMU’s attitude estimation, where IMUs are traditionally much more expensive than IR sensors. To overcome this problem, the IR sensors could be replaced with a low cost attitude heading reference system (AHRS) using a Kalman filter based IMU that has been produced by the VCU UAV lab [32].

A newly developed flight control system developed at VCU provides improved peripheral support (I2C, SPI communication) for HAG sensors and has the AHRS developed in [32] already integrated into the board. The ARIES FCS [33] was under development in the VCU UAV lab during the course of this thesis work and was therefore not an option for developing these ATOL algorithms. The ultimate goal would be to port the ATOL algorithms designed on the miniFCS to the new ARIES FCS. The combination of a better peripheral support of the addition of HAG sensors and integration of the AHRS replacing the IR sensors would provide a much better option for flight tests.

The largest challenge is that even though PID parameters used in simulation perform more than adequately in HILS simulations, in actual flight tests the PID parameters used in simulation do not work in actual flight tests. The majority of time during flight testing of this project was spent tuning PID parameters for all the additional controllers added while the aircraft was in the air. This is due to the fact that the flight dynamic properties of the aircraft used in simulation are
not the same as the aircraft used in flight testing. It would be very beneficial to be able to model the Mig aircraft used for actual flight testing and be able to use that model in FlightGear so that the PID parameters used for simulation would also work for flight tests.

Finally, once the ATOL algorithms were successfully flight tested several times on the ARIES FCS, a very attractive safety feature could be added. In the event that the aircraft detects a GCS or RC loss (where the aircraft losses contact with the ground control station or with the remote pilot) due to modem failure or the aircraft has gone out of communication range, the aircraft is very likely to crash. With the implementation of the ATOL algorithms, if the FCS detects a loss between the GCS or RC pilot, the FCS could enter a return to base mode (RTB) where the aircraft could return to the GCS location and loiter for a short period of time trying to reestablish contact with the remote pilot or GCS; if contact is not reestablished in a reasonable amount of time the FCS could then autonomously start the autonomous landing sequence and safely land the aircraft. This is a very significant safety feature that could prevent the aircrafts certain destruction if link is lost between either the remote pilot or GCS.
Bibliography


84


"XPlane 10," [Online].

