Fixed Wing UAV System

Stratton Spirou - sspirou@seas.upenn.edu

Advisor: Dr. Camillo J. Taylor - cjtaylor@seas.upenn.edu

Abstract

This is a systems design project intent on developing a fixed wing unmanned aerial vehicle (UAV). My UAV will carry a payload, interface with q ground control, and have at least one and a half hours of flight time. A core goal in this design is to have a reproducible system with easy deployment. The payload will contain compute and sensing in order to provide aerial mapping to robots on the ground below. In essence, this UAV project is the design of a flyable laptop system.

I explore the composition of hardware components to enable software solutions to autonomous flight. I identify a software stack that will provide control over the aircraft's hardware systems while communicating with a ground control station, Q Ground Control. I report on the regulations that apply to small unmanned aircraft like this one. I describe the software background, challenges, and methodologies to achieving an on-board detect and avoid system to enable beyond visual line of sight operations of this aircraft. I detail solutions for a safe retrieval system. Finally, I provide a cost breakdown and societal relevance for UAV projects like this.

Table of Contents

Introduction

UAV systems are unmanned aerial vehicles that provide radar and mapping information to units on the ground. I explore an improvement to an existing quad rotor system via a change to a fixed wing design. The two alternatives come with different trade offs. The purpose of exploring a fixed wing system is to deliver a large improvement to flight time and possible ground coverage.

The main obstacle a quad rotor system is better at overcoming is safe retrieval. Due to the ability to hover in place, a quad rotor UAV can land anywhere. The aerodynamics of a fixed wing aerial vehicle require a runway or landing strip for standard takeoff and landing procedures. I will explore options for fixed wing UAV retrieval in the event that a runway is not available. Many desirable use cases for UAVs and drones do not permit a pilot to have a visual of the vehicle. I will use a computer vision approach to offer a system that is capable of beyond visual line of sight (BVLOS) operations to overcome this.

Motivation

The GRASP lab currently has a quad rotor UAV system that provides radar mapping and q ground control communications. The shortcoming of this existing UAV system is flight time, which negatively impacts lengths of deployment as well as possible coverage area. Quad rotor designs require more power consumption than fixed wing systems. This is problematic because there is a trade-off between weight and charge capacity with batteries. Dr. Taylor expressed a desire to double the flight time of the quad rotor UAV, which can be better accomplished by switching to a fixed wing system.

My personal motivation to pursue this project comes from merging some of my hobbies with my undergraduate work and future career interests. I have had a passion for exploring aviation for a long time. During my middle school years, I began building remote control helicopters. I started with four channel systems, and then I graduated to single rotor six channel systems. I would buy individual components and design my own builds ranging from nitro size to 700 class vehicles. All of the helicopters I built were electric, powered by lithium polymer batteries. In high school I trained for my private pilot license at a local airport. Since becoming certified, I have made a habit of pursuing further ratings on my license. Currently I fly small propeller airplanes.

I am hoping to pursue systems software in aviation post-graduation. My previous work experience has dealt with graphics, image processing, and game engines through work for a virtual reality company. Some of this background helps in the future of aviation software, and this particular thesis project will allow me to apply that when accounting for detect and avoid functionality for my UAV system.

Aircraft Structure

Overview

My UAV will be a propeller driven, fixed wing aircraft. In essence, the structure of this UAV will be similar to that of an airplane. A fixed wing aircraft includes 4 main control surfaces: the elevator, rudder, ailerons, and flaps. The elevator controls the vehicle's pitch, allowing it to point the nose up and down. The rudder controls the vehicle's yaw, allowing the nose to rotate left and right. The ailerons and flaps are both located on the vehicle's wings. The ailerons rotate opposite directions, controlling the vehicle's roll. Lastly, the flaps rotate in sync to provide extra lift and resistance to operate at slower speeds. Smaller aircraft, like this one, can omit their flaps, so this design will not include them.

The structural design of this UAV is a mechanical engineering task that is left for a later stage of this project, so exact placement and measurements of the body parts of this aircraft are not included in detail here. Instead, I will describe the body parts involved and identify where on the general aircraft structure each control surface is located. I will also identify the general areas where the hardware components will be housed so that they can effectively sense surroundings and manipulate the control surfaces.

Aircraft Body

The fuselage is the main body of the aircraft. It is generally a hollow tube structure with a pointed nose. In an airplane, the cockpit is located at the tip of the fuselage. As a single engine, propeller driven aircraft, our motor will be housed at the tip of the fuselage.

The wings are attached to the fuselage, running perpendicular to it. The wings are responsible for generating lift, to allow the aircraft to fly. The back of the wings, typically on the outer half, house the ailerons. The ailerons are responsible for adding equal and opposite air resistance to each other, such that the angle (up and down) that they are positioned at will rotate the aircraft about the fuselage's axis. This can be thought of as banking left and right. It is referred to as the roll of the aircraft. By rotating the airplane in this fashion, the lift from the wings allows the aircraft to turn left and right. Two servos, one per wing, will be located in the fuselage near where the wings attach. They will be responsible for manipulating the rotation of the ailerons.

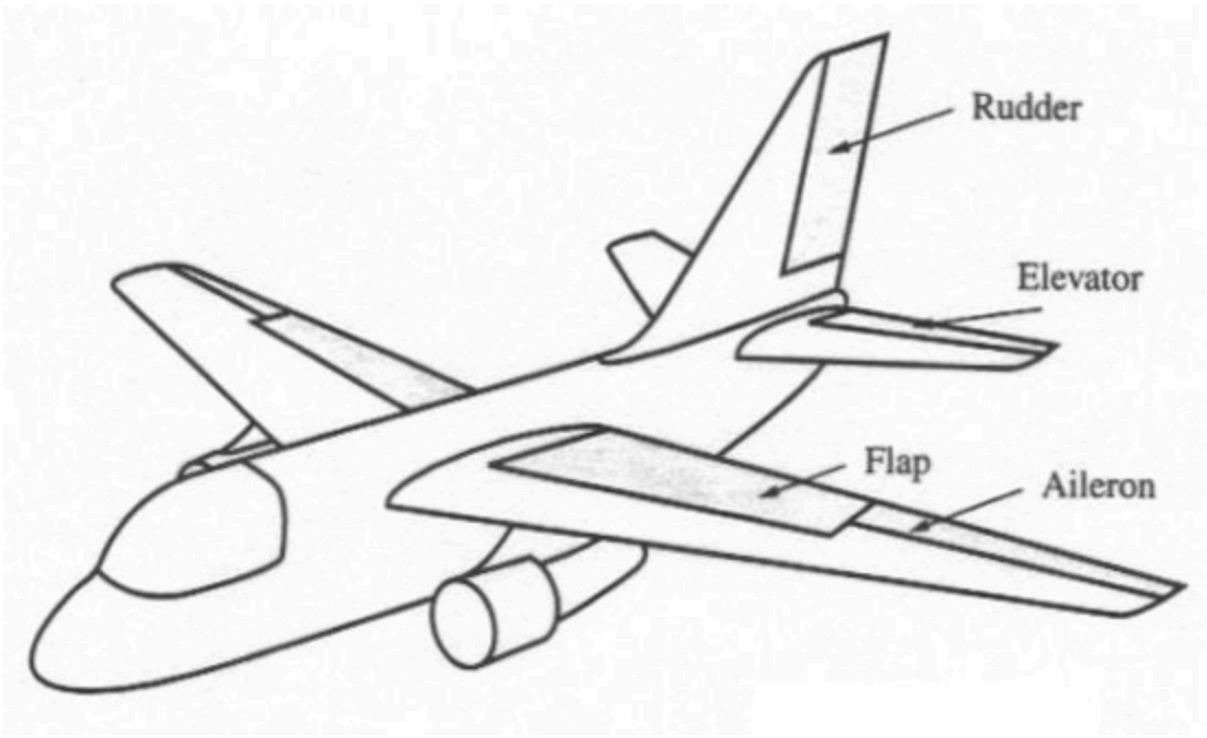
The tail of the aircraft is located at the rear end of the fuselage. The tail houses the 2 remaining control surfaces. As such, it will also house the 2 remaining ailerons that are responsible for controlling the elevator and rudder movements.

The elevator runs parallel with the wings and rotates up and down. When the elevator rotates up, it creates resistance that forces the aircraft to pitch its nose up, allowing it to climb. When the elevator rotates down, the nose pitches down, forcing the aircraft to descend. The axis of rotation controlled by the elevator is referred to as the pitch of the aircraft.

The rudder extends vertically perpendicular to the fuselage and rotates left and right. It is responsible for orienting the aircraft along its third axis of rotation, referred to as the yaw of the aircraft. When the rudder is used, the aircraft will rotate clockwise and counterclockwise if it were being viewed from directly above or below. The rudder allows the aircraft to keep forces balanced while maneuvering. It is also helpful in dealing with cross wind or in recovering from dangerous scenarios such as a tail spin.

Diagram

Below is a diagram of a fixed wing aircraft, identifying the 4 main control surfaces (Aerospaceweb.org, n.d.). The payload for this UAV will be placed mid-fuselage to keep the added weight on the center of gravity of the aircraft. The fuselage will contain voided areas on the bottom to allow housing of any additional downward facing sensors that are desired for particular use cases.



Operational Regulations

The Federal Aviation Authority (FAA) oversees regulations of airspace and areal vehicles. All FAA regulations are published under Federal Aviation Regulations (FARs). The specific regulations that apply to most Unmanned Aerial Systems (UAS) is FARs part 107. The UAV design detailed here weighs less than 55 pounds and is intended for research use. Per the FAA's user identification tool, this UAV falls into the general small unmanned aerial system classification (FAA, What kind of drone flyer are you? , n.d.) and is therefor regulated by FARs part 107.

FARs Part 107 Overview

The main rules that should be followed per FARs part 107 are:

- Fly in Class G (uncontrolled) airspace
  - This means that drones cannot be flown within 5 miles of most airports
- Fly at or below 400 feet above ground level (AGL)
- Maintain visual line of sight (VLOS) during operation
- Carry proof of test passage of The Recreational UAS Safety Test (TRUST)
- Have a current FAA registration and mark drone with registration number
- Give way to and do not interfere with other aircraft
- Do not fly over any individuals who are not involved in the operation

(FAA, *Certificated remote pilots including commercial operators* , n.d.)

In order to deviate from any of the above regulations, prior FAA authorization or waivers must be obtained. There are other specific regulations that apply to special operations such as flying over buildings and flying at nighttime. I will assume that we will not be operating this UAV under these special circumstances. I will note, though, that if it is desired to use this UAV under such circumstances, the design may be updated with appropriate indicator lighting to accommodate this without affecting much.

Airspace Overview

When operating any aerial vehicle, it is important to have a basic understanding of airspace. The FAA maps and classifies airspace based on latitude and longitude coordinates in conjunction with altitude. Important ground locations such as major cities, government buildings, and airports are often located at the center of controlled airspace.

The airspace will often be controlled within a small radius of these locations all the way to the ground. Airspace at larger radiuses from these centers will only be controlled down to a certain altitude (i.e. 500 feet and up at 2 miles from center; 2,000 feet and up at 5 miles from center). In this manner, airspace can be visually thought of as an upside down wedding cake. The purpose of this is to allow air planes to ascend

from and descend to runways in controlled airspace, while allowing recreational activity within the airspace below.

Any operators of this aircraft should keep this in mind when piloting the UAV or setting flight plans. Although prior authorization to operate within controlled airspace can be secured, it is necessary to know that a flight path will require such authorization beforehand. During flight at these low altitudes, there are many possible reasons why this UAV might have to deviate from a pre-planned flight path to avoid collision. By planning operations to allow for adequate deviation safety buffers, operators can better ensure that they will not break FAA regulations.

BVLOS Waiver

This UAV design does wish to obtain waivers for Beyond Visual Line of Sight (BVLOS) operations. The process for obtaining a BVLOS waiver from the FAA requires the applicant to detail the specific operations and show that they can be conducted without being a danger risk to individuals and property in the operational area. The FAA will review waiver submissions and reward them if the operation is deemed to properly mitigate risk.

This drone will offer 2 features that will help justify BVLOS operations to the FAA:

1. Remote Point of View (POV) control of the UAV
2. Automated Detect and Avoid system

The remote point of view control of this UAV is a standard feature among commercially available drones today. The way this works is that the drone operator has a controller, in this case it will be a tablet, smartphone, or computer, that provides the operator with a live video feed from the drone's point of view. The drone operator has forward sight as well as full control over the drone through the mission control software on the controller. This provides a human in the loop safety mechanism for pre-set flight paths, or full pilot in command control for standard flight.

The automated detect and avoid system is much more complicated. Detect and avoid systems are used by more advanced UAV systems and larger aircraft. They are not a standard feature yet with commercially available drones. This UAV will offer detect and avoid through a series of on-board cameras and a computer vision approach that will be discussed in its own section later on. For now, I will note that in the Hardware Components section I will include a Graphics Processing Unit (GPU) that will be necessary in order to permit our computer vision approach to a detect and avoid system.

Definitions and System Overview

Over the past decade, there has been a massive growth in research and availability of drones, remote control vehicles, and self-controlled robotics systems. Throughout the next few sections I will use language that has developed in these fields and respective communities, so I will define them first.

- Autopilot - This term refers to a system that performs automated control of a vehicle. An autopilot system is used in self-driving cars, commercial airplanes, boats, and more. This UAV will make use of a commercially available autopilot system.

- Companion Computer - This refers to additional processing units that a vehicle design may optionally employ to do more intensive data processing. The flight controller, defined next, is intended to be as light weight as possible while permitting required flight control. This UAV will optionally carry a graphics processing unit (GPU) and companion CPU in its payload to perform any data processing tasks that are not directly related to the flight controller or detect and avoid systems. The flight controller chosen for this UAV will be able to perform object detection and fast object tracking on the live video feed received from the UAV's front facing camera because it was chosen specifically with detect and avoid in mind.

- Flight Controller - The flight controller is the on-board computer and sensor setup that permits an autopilot system to work. It involves a limited computer processing unit that is capable of receiving signal and sending control output to the vehicle's movement systems. The flight controller is generally designed to be as light weight as possible so it limits its own power consumption. This UAV will use a PX4 compatible flight controller setup.

- Flight Controller Firmware - This firmware runs on the flight controller and is responsible for control of the vehicle. It offers control settings and functions that can be called externally to aid in control of the vehicle. This UAV will use PX4 autopilot as its flight controller firmware.

- Ground Control - A ground control system is used to communicate control instructions from a vehicle's operator to the flight control firmware onboard. Ground control programs are also responsible for reporting the readings from the onboard flight instruments back to the vehicle operator. This UAV will use QGroundControl. QGroundControl is an open source system that can be run on most standard operating systems. This particular design will run QGroundControl on iOS to allow the UAV operator to perform flight planning and piloting from their iPhone or iPad. This ground control software can also be run from Windows, Linux, or Android.

Hardware Components

This UAV will support autonomous flight through the PX4 autopilot system. There are PX4 equipped design packages available with prebuilt systems or preselected self-assembly kits. Most of these fixed wing kits have flight times of up to an hour. The target flight time for my UAV, 1.5 hours, is a little longer than the standard available. I have chosen a set of essential hardware components to use. My main focus is on the software approaches to automated flight, so I will leave more detailed analysis of individual hardware choices and options for a later stage of this project. For all information necessary to design a custom built UAV that supports PX4 autopilot firmware, the Pixhawk Autopilot v6X Standard documentation should be followed (Pixhawk, n.d.). The core hardware requirements for a fixed wing PX4 autopilot enabled UAV are:

- Processor - This is the 'Flight Controller' from the definitions page above. It can be a small CPU or a microcontroller. It needs to be capable of running simple control algorithms and interface between hardware components. This UAV will use the NVIDIA Jetson AGX Xavier. The NVIDIA Jetson is a complete System on Module device. It is a nano device as well, making it an attractive option for use cases like small UAVs. The Jetson provides a GPU, CPU, memory, and power management. It can be used for computer vision tasks, so depending on the computation requirements of the fast object tracking algorithm chosen, the Jetson may mitigate the need for a companion computer. The intended object tracking approach for this project is capable of running on the Jetson. One more positive to note about this choice is that the NVIDIA Jetson is compatible with OpenCV, a library that will be useful in any future updates to the computer vision portion of this project.

- Sensor System - Sensors are used to measure the UAV's position, orientation, and velocity. These measurements allow the flight controller firmware to make control decisions to keep the UAV on its flight plan. This hardware requirement is where design choices can become more complicated. Depending on which functionality is desired from the autopilot system, some components from this category can be omitted. Our fixed wing UAV will require an accelerometer, a gyroscope, a GPS receiver, and an altitude sensor. Addition of a magnetometer is not necessary, but it can enhance performance. I will include it for quality of VTOL operation.
  - The Adafruit 9-DOF LSM9DS1 is a 9-axis sensor that is fairly light-weight and provides up to 16g sensitivity with a measurement range of +/- 16g. The LSM9DS1 provides an accelerometer, a gyroscope, and a magnetometer in 1 compact device. I will use this to support those 3 sensor requirements (Industries, A., Adafruit 9-DOF LSM9DS1 breakout board - Stemma QT / Qwiic , n.d.).
  - The u-blox NEO-M8N is a popular option for GPS receivers on fixed wing drones. It is small and light weight, yet very high-performance. The NEO-M8N provides sensitivity up to -167 dBm and 72 channel tracking (u blox, 2022).
  - The Adafruit MPL3115A2 is a barometric pressure sensor, altimeter, and temperature sensor that works up to 50,000 feet. The MPL3115A2 is efficient and

provides high resolution. Its low power consumption makes it a great choice for this UAV design (Industries, A., MPL3115A2 - I2C barometric pressure/altitude/ temperature sensor , n.d.).

- Radio Communication - This hardware component is responsible for data transmission between the UAV and ground control. Important factors to consider are data transmission speeds and weight. This UAV design will use the FPVDrone 3DR 500MW Radio Telemetry Kit which can communicate up to several kilometers and boasts a data transmission rate of 57.6 kbps (FPVDrone, n.d.).

- Power Supply - The power supply is responsible for powering the entire system. Most of the hardware components that I have chosen were selected specifically for having low power consumption and light weights. The majority of the power consumption from the UAV will be done by the motor that powers the propeller. This UAV system will be powered with a Lithium Polymer (LiPo) battery. UAV's generally employ LiPo batteries because they have higher energy density and lower weights than other battery types. There are many options to choose from. The ZOP Power 4s 14.8 V battery comes in a wide range of charge capacities. The higher capacity, the heavier and more expensive the battery. Bigger batteries could be used to extend flight time, but the added weight does increase the energy consumption rate of the system (Zop Power, n.d.).

- Servos - Servos are electrical gear based components that are responsible for moving the UAV's control surfaces. This fixed wing aircraft will need 4 servos: 1 for each aileron, 1 for the rudder, and 1 for the elevator. We can use the same servo choice for all 4 of these purposes. The Hitec HS-5065MG is a high torque metal gear feather servo that will provide the force required for our UAV while keeping power consumption minimal (Hitec, n.d.).

These components will need to be stabile and secure on the airframe of the UAV. The processor, sensors, radio telemetry pieces, and power supply will need to be enclosed so that they are protected from any weather or debris. The servos are a little more durable than the other hardware components, however it would be best to keep them somewhat protected as well. The airframe design for this is a mechanical task left for a future stage of this project.

The individual choices of these components can be changed as necessary for projects that may require specific functionality that these components don't support. However, this provides a great starting point for a fully functional fixed wing UAV that enables PX4 autopilot. Without spending more time on hardware specifics, I will note that the capability of PX4 autopilot is limited, in the sense that it does not perform automated detect and avoid. For this reason, our payload will contain additional hardware that enables our computer vision approach for detect and avoid to be performed. Without the additional detect and avoid functionality, a drone with basic PX4 autopilot enabled will likely not be able to secure FAA waivers for beyond visual line of sight operations.

Interfaced Systems

As stated in previous sections, this UAV design will support the PX4 autopilot flight stack. It will interface with Q Ground Control to provide a ground control station to the drone operators. The payload will contain compute and sense hardware to provide images and mapping information to robots on the ground. The communication protocol that is used to exchange information between PX4 and Q Ground Control is called MAVLink.

PX4 Flight Stack

The PX4 flight stack will be used as the autopilot system and flight controller firmware. PX4 is a set of modular libraries that provide control functions and algorithms for a host of different vehicles and hardware setups. PX4 is actively developed by its community, and it is one of 2 standard options for flight controller firmware for drones. Many commercial drone manufacturers have adopted it too.

PX4 provides a real-time operating system, drivers for PX4 compatible hardware components, algorithms for automated control, navigation, and data processing, and a communication system that uses the MAVLink protocol. Previously, I chose hardware components that are compatible with the PX4 flight stack. This allows drone designs to focus on their core issues, like online detect and avoid for this UAV, rather than recreate the wheel of basic flight control.

MAVLink

MAVLink is a header based communication protocol designed specifically for remote systems like drones and robots. It is implemented in C, so it is very fast and powerful. MAVLink is well documented and has been adopted as the communication protocol standard by drone manufacturers and the do it yourself drone community. It is light weight, extensible, and protocol agnostic. This means that MAVLink requires limited storage space, can be extended for custom functionality, and can be used with a wide range of existing communication protocols (MAVLink, n.d.).

Data will flow through MAVLink in two directions. Key information about the current state of the UAV will be communicated from the vehicle to Q Ground Control. The key data reported will provide the UAV's attitude, altitude, battery level, and location to the user. The UAV operator will be able to see this information on the ground control interface. Navigation and flight plan instructions will be communicated from Q Ground Control to the UAV. This provides the operator with full control over the UAV's flight. Adjustments to the flight plan such as altitude, waypoints, and landing location will be sent in this manner.

QGroundControl

Q Ground Control is an open-source ground control station that is designed specifically for remote systems like drones and robots. It is commonly used with the

MAVLink protocol to enable operator to vehicle communication. This drone will be performing operations along with robots on the ground who also communicate to Q Ground Control with the MAVLink protocol. Q Ground Control provides a graphical user interface that can be run on a smartphone, tablet, or computer.

This ground control software provides support for: setup and configuration of the PX4 autopilot system, flight support, mission planning for autonomous flight, flight map displaying the flight track, waypoints, and vehicle telemetry information, video streaming from on-board cameras, and support for multiple vehicles per operation.

Aside from the basic telemetry status reporting and flight control, Q Ground Control allows custom plugins to expand functionality (QGroundControl, n.d.). This aspect of the Q Ground Control system makes it perfect for our use case, as it will provide the necessary mechanism for interfacing the payload with the units on the ground. The payload is described in the next section.

Payload

Overview

This UAV system will carry a swappable payload that will contain the desired compute and sense system. In essence, it will be a flyable and upgradable laptop. Payloads are designed and chosen for specific use cases, and can contain varying types of sensors and computation power. The purpose of the payload is to gather information about the UAV's surrounding environment, process the information, and communicate it back to ground control.

Our payload will connect to the flight controller through a serial bus. This will allow the detect and avoid system to provide control instructions to the vehicle. Connecting the payload to the flight controller will also allow the payload to transmit information to the ground control station using the MAVLink protocol and the PX4 system's radio communication hardware. Optionally, the payload could contain its own communication hardware, but that would add more weight and cost to the system. Without a specific need defined to justify this, it is preferable to rely on the existing communication system on board.

Detect and Avoid Components

For detect and avoid, our payload must always contain a front facing camera. This can be done using a stereo camera or a monocular camera. Stereo cameras use 2 angles to capture, allowing the resulting photo to be a 3D image. This is the camera of choice for our detect and avoid system because it allows more accurate object detection and classification. The other option, a monocular camera, is a cheaper approach that can also work. The drawback is that monocular cameras provide less accuracy in object detection and classification.

More advanced systems will also contain infrared cameras and thermal cameras. Infrared cameras allow detect and avoid to be successful in low light and at night time. Thermal cameras also permit object detection in bad lighting. The main benefit from thermal cameras is that they can be used in conjunction with the stereo or monocular camera to improve classification of the detected objects.

This drone will use the Zed 2 spatial AI camera to capture video feed for online detect and avoid. It is a high definition camera that is capable of measuring depth up to 20 meters. Stereo cameras are expensive, but relative to other stereo cameras that can be used for this purpose the Zed 2 is on the more affordable end of the spectrum. It also has the benefit of easy integration with our PX4 autopilot system. Further, stereo labs has designed all of their Zed stereo cameras to integrate seamlessly with NVIDIA Jetson processors.

The Zed 2 camera's depth range is on the lower end of usefulness in fixed wing aerial detect and avoid because our aircraft may be flying at up to 40mph. Fixed wing vehicles require a lot more space than their helicopter counterparts in order to perform maneuvers, so 20 meters does not provide much leeway. This is one of the more expensive components, and it may be worth cutting costs. I chose the Zed 2 for its

easy integration with the Jetson flight controllers but cheaper camera options are worth exploring.

Detect and avoid systems like this can take 2 different approaches for graphics processing. The first option is to send the payload's front facing video feed to the ground control station and process it there. This would require the UAV operator to be running Q Ground Control on a GPU enabled computer. The ground control station could identify, classify, and track objects, then send control decisions for collision avoidance back to the UAV. System configurations like this permit heavier graphics processing operations, but they can make operations planning more limited.

Alternatively, the payload itself can contain a GPU enabled processor. If possible, this is a better approach for autonomous flight because it removes the time difference incurred by sending data back and forth between the UAV and ground control station. Our PX4 processor choice, the NVIDIA Jetson AGX Xavier, contains GPU capability, and is suitable for our fast object tracking approach.

If the attached payload has graphics processing then we will opt to perform our fast object tracking on the payload's GPU to keep the flight controller's processing separate, though. However, we did choose the NVIDIA Jetson AGX Xavier because it is designed specifically for embedded vision systems like this one. If it is deemed more desirable to keep the graphics processing in the payload rather than on the flight controller, then the flight controller processor should be downgraded to save costs. The AGX Xavier comes in at about $2,500, but if graphics processing is not a requirement then we could get by with other flight controller processors in the $500 to $1,000 range. The AGX Xavier was chosen because it is the likely choice for vision payloads as well, so from a cost standpoint, it makes sense to utilize the same processor for as much of the flight requirements as possible.

General Payload Components

This system design is intended to support a swappable and upgradable payload, so I will not choose specific components in this section. Instead, I will provide some examples and overview of what UAV payloads can consist of for different use cases.

For most use cases of UAVs, the payload is what enables the aircraft to perform the desired tasks. Oftentimes this involves mapping and inspection of the environment below. A standard payload will contain a down facing camera, LIDAR, and RADAR. Down facing thermal and infrared cameras may also be used. Weather measurement equipment could be included if the drone is meant to make such observations.

LIDAR stands for light detection and ranging. It is often used by payloads in conjunction with visual and infrared cameras to perform high-resolution 3D mapping of the world below. LIDAR is a modern evolvement of RADAR, radio detection and ranging. In both technologies, signal is sent from the device, and distance is measured by the time differential that it takes for the signal to bounce off an object and return to the sensor. Front-facing LIDAR and RADAR can also be used for detect and avoid in an autonomous vehicle.

Detect and Avoid

Due to FAA regulations, this system contains detect and avoid functionality in order to permit beyond visual line of sight operations. Detect and avoid systems for aerial operations are uniquely tricky because there is no margin for tolerable error. A failure to detect and avoid collision while flying will lead to loss of the entire UAV in almost all cases, so we seek to implement the safest methodology possible.

In addition to minimizing error margin, it is important that we are able to perform the computations quickly, as the relative speeds of aerial traffic can be quite fast. If our UAV is approaching another moving object head-on, it is imperative that we recognize this and correct our flight path quickly. If our system is unable to do this with adequate speed, it is unlikely that we can secure BVLOS waivers from the FAA.

Fast Object Tracking

Computer vision has offered new ways to perform object detection, segmentation, and tracking, based on video feed. I will primarily focus on the objective of object tracking in real-time for this project. The goal of object tracking is to predict the location of an object in subsequent frames, given the location of this object in the initial frame. Since our goal is to make on the fly decisions based off of the information we stream from our on board video feed, we need to maintain object tracking in real-time. This is referred to as online.

Most approaches to object detection and tracking require pixel-level computation. First, objects are detected and classified in the initial frame. We isolate the objects with bounding boxes and then use a convolutional neural network to classify the objects. As we look at subsequent frames, we use the bounding boxes to predict the objects' locations. We perform classification on these regions of future frames. If we classify the same object type in the general predicted region of a subsequent frame then we infer movement of that object between the two frames. Traditional approaches like this can be very accurate, but they are not fast enough for in flight use.

The two main types of objects we need to be able to track will be stationary objects versus moving objects. Stationary objects will be poles, buildings, and telephone lines. Moving objects will be other aircraft and birds. In the air, we will be isolating these objects against the background of the sky, and they should be relatively infrequent. Our task of object tracking in the sky will deal with far fewer objects than when this task is applied in other domains like biological imaging or autonomous driving. Moving objects will be trickier to deal with because they may have unpredictable flight patterns. Thus, it will be important for us to not just identify obstacles in our path, but also we should classify them into subcategories starting with moving versus stationary.

The identification and classification of obstacles will require the use of artificial neural networks. I will provide an introduction to these graphing structures, and then I will expand upon the particular type of neural network that we will use, a convolutional neural network. After providing background and overview, I will present some recent techniques and research that have permitted fast online object tracking, and I will

discuss how we can make automated flight control decisions based on the results we receive from these algorithms.

Neural Networks Overview

A neural network is a computation system inspired by the biological study of the brain. It contains nodes that act as artificial neurons, which process input and pass output to other nodes. These artificial neural networks are structured as graphs. The connections between nodes, or edges of the graph of an artificial neural network, simulate synapses in a biological brain. The input of each node is a number, and the output is a non-linear function combination of all of the node's inputs. Nodes can be connected in various ways to achieve different results.

Activation functions are used on the output of each node in a neural network. An activation function is a non-linear function. Neural networks can be defined with different activation functions between layers, but commonly a single activation function is chosen. The three most common activation functions used are the sigmoid, rectified linear unit, and hyperbolic tangent functions. The purpose of an activation function is essentially to separate computed values at each node. When it comes to binary classification tasks, the sigmoid and hyperbolic tangent functions are particularly effective because they map inputs into the ranges 0 to 1 and -1 to 1 respectively.

Just like in many other graphing algorithms and approaches, edges in an artificial neural network graph can be weighted. The weighting of different edges is the mechanism that we use to allow a neural network to 'learn' when it is trained. Generally, nodes are organized into layers, and weights are applied in between layers. Weights are also generally adaptable to allow 'learning' to occur. This means that as data flows forward through the layers in an artificial neural network, some feedback can be propagated backwards to update the weight values based on the results of a computation. Just as the human brain learns to ignore the nose when processing input from the eyes, this approach allows an artificial neural network to adjust how much it values particular inputs when making a decision.

There are many different applications of neural networks. We can build different types of artificial neural networks, based on the number of nodes and layers used, the connectivity of the graph, the activation functions involved, and more.

For the purposes of automated flight, we are most interested in the fact that artificial neural networks can be used to perform classification and recognition tasks. One of the main use cases for them is image classification. The approach here is to build an input vector to feed into the neural network that identifies a set of features present in a given image. This is called the convolution operation. When provided with this feature vector, the neural network must process the data from layer to layer, and output a classification of what the image is. This classification step is referred to as labeling the image. Convolution can be applied at multiple layers in the neural network prior to the final fully connected layers of the network.

Training a Neural Network

A neural network does not initially have any conception of what a set of labels means. Consider the task of labeling a set of images into their proper classifications as road signs. At the beginning, a neural network cannot contain any knowledge of what constitutes a stop sign versus a yield sign. It must be taught how to distinguish these from each other. We use a ground truth to teach the neural network. The ground truth in this example is a set of images along with their correct labels. Each time we feed an image from the ground truth into our neural network, we get an output which consists of a probability projection, or guess. Then, we use the error calculation between this output and the correct answer to update weights in our neural network. As we process batches from our ground truth, our weights get updated, allowing our neural network to 'learn' the value of inputs throughout so that it better predicts as it sees more examples. This process is called training. Specifically for a task like the one proposed, we are doing what is called supervised learning.

Learning can be supervised or unsupervised. Supervised learning is the process of training a neural network with well labeled ground truth data like I described above. More explicitly, supervised learning is when both the input and output are provided to the model, so that weights within the neural network can be updated using an error each iteration. The most typical use cases for supervised learning are regression and classification problems. The other approach, unsupervised learning, is when only input data is given. This is used to identify patterns or trends within a dataset. A couple of examples of approaches for unsupervised learning are clustering and association.

The learning process is traditionally done offline, meaning that we train our neural network with input that is complete from the beginning. This task does not require fast computation, as it can be done over many days, or even weeks, prior to actually making use of our neural network in the air. What is most important here is a large, well labeled dataset for us to train our network with. Many such datasets already exist and are openly available. We wish to have a lot of examples of each label type from many angles, so that our network learns to identify the unique features much like our own brain can.

Convolutional Neural Networks

Classification and recognition problems commonly use convolutional neural networks. A convolutional neural network is capable of analyzing spatial relationships between pixels to learn features directly from images. A convolutional neural network is defined by its layers and connectivity. The early layers are capable of learning basic features such as object edges. They can identify lines and points. The higher level layers are responsible for learning more complex shapes defined by these edges.

The namesake of this type of neural network comes from the application of a convolution operation to the input data. The purpose of the convolution operation is to identify local patterns in the data and extract them as features to provide to the neural network. The methodology for doing this is to slide a small matrix over the image and compute the dot product of the pixel values within the matrix at each step. The matrix used for this is called a kernel, and its size is an adjustable hyper parameter of the

neural network. Another adjustable hyper parameter, stride, refers to how many pixels the kernel is translated by each time. Smaller kernel sizes, like 3x3 pixels, are used for identification of low level features likes edges. Larger kernel sizes are used to identify high level features like objects. An example of a larger kernel size would be 7x7 pixels. Strides can be chosen based on how many objects are expected in a frame. Since convolution operations are computed very quickly, we will use a low stride to avoid missing obstacles.

Regularization

When choosing a kernel size, it can be tempting to go with a bigger matrix. This is cautioned against because it could lead to a problem called overfitting. Overfitting refers to when a neural network performs well on the training data, but it performs poorly on the test data. This essentially means that the network learned the training set exactly. Larger kernel sizes require much larger amounts of training data to accurately fit a neural network.

Overfitting can also be mitigated through the use of regularization techniques such as weight decay and dropout. Weight decay is the idea of adding a penalty term to the original loss function that is used to update edge weights when training a neural network. The addition of a penalty term encourages low edge weights because it adds more loss for higher weighted edges. The reason this methodology works to combat overfitting is that the neural network becomes less likely to overvalue any individual feature. When used in conjunction with a larger kernel size, weight decay prevents the neural network from learning the training data exactly. The result is more generalized learning that will be more effective on test data.

Dropout is another regularization technique. Dropout is used only during training, and it is intended to prevent the neural network from becoming overly dependent on any individual or small group of neurons. The way this works is that different sections of the neural network are randomly weighted to 0 while training, such that the decision of that round or sets of rounds is not influenced at all by the neurons that were dropped out.

The dropout rate of a neuron is another hyper parameter of the neural network. Higher dropout rates mean that more neurons will be dropped out at any given time. Lower dropout rates cause the opposite to happen. Dropout prevents the co-adaptation of neurons during training. Just like weight decay, the use of dropout during training results in a more generalized learning that performs better on test data. When using drop out on a convolutional neural network, it is normally applied in between the convolutional layers and the fully connected layers.

Autonomous Flight

Now that I have established some background on training neural networks for various types of problems, I will highlight the approaches relevant to our task of autonomous flight. If we take a step back, we can again consider the problems that need to be solved to accomplish this.

First, it is vital that our UAV handles the basics of autopilot. This means two things: our UAV must be able to maintain altitude, and our UAV must be able to maintain compass heading. These two problems are fairly trivial, though, and only require explicit reaction to onboard instrument readings. Therefor, they do not require neural networks to make decisions. This part of autonomous flight is handled for us by the PX4 autopilot system.

Next, we have the issue I stated in the main topic of this section: detect and avoid. In a very broad sense, our aircraft must be able to sense obstacles in its flight path and navigate around them to avoid collision. As a fixed wing vehicle, our options for navigation are much more limited than those available to a quad rotor. I will discuss these at the bottom of this section. To recap, I will again describe the obstacles we are looking to detect.

While in the air, most of the space ahead of us is empty. We will need to avoid towers, buildings, birds, and other aircraft. This means we must be able to isolate them against the view of the rest of the sky and then predict their future positions. It will be useful to classify these objects into two main buckets: moving and stationary. Moving objects may have an unpredictable path, which poses a separate issue, while stationary objects can be fully predicted through our flight path once we recognize them.

Now, I will approach the question of ‘how do we recognize obstacles?’. Recall from earlier sections that our payload contains both compute and sense equipment. The latter of which, among other things, involves cameras. By having a forward facing camera, we have access to a live image feed of the path ahead of our vehicle. This will provide us with an ‘online’ data input stream. Using this data input stream, we need to recognize obstacles in a given frame, track them in subsequent frames, and predict their locations in future frames.

Our Jetson flight controller has the capability of performing the graphics intensive neural network operations. We will keep a pre-trained convolutional neural network on the Jetson system, and then we will input the image feed from the front facing Zed 2 Spatial AI camera. The images will be fed into our neural network, and we will perform object detection on each frame. For each object identified, we will keep track of its position across a series of frames and construct its movement path. These paths of movement should be assessed in a queue when making inferences on possible collision threats.

For each line of movement we have, we will extrapolate the path into the future based on its measured velocity and our own speed and flight path. As a fixed wing vehicle, our courses of corrective action are somewhat limited. To compensate for this, we will consider any obstacle that will come within 100 feet of us to be a possible collision. This is an appropriate safety buffer at scale in comparison to the 500 foot buffer used by small passenger airplanes.

If an object threatens to collide while approaching head on, we will make a left turn and divert our flight path in that direction until the obstacle is passed. Otherwise, we will turn right to avoid collisions on our left side, and we will turn left to avoid collisions on our right side. In all cases, if the approaching obstacle is above us and we have altitude to spare below, we will descend 100 feet to avoid collision. For all other cases, especially when the corrective action is uncertain, we will climb 100 feet and

maintain that new altitude. No matter the diversion tactic used, once all collision obstacles have been cleared we will return to the altitude and heading stated in the current operation's flight plan.

Current Fast Object Tracking Research

Online object tracking requires less resolution, but more speed than object recognition. Typically, objects would be masked in the first frame, and this would be used to train the neural network online. Tracking would then be done as I described above, by looking to identify the object in a predicted region of subsequent frames. When new objects come into view, they must be isolated like we did in first frame initialization. In order to track new objects that have entered the field of view, we would need to retrain our neural network on the new ground truth.

The approach above was quite effective at tracking, but somewhat slow. Approaches to speeding up this process were to remove the masking of objects for ground truth, and instead, attempt to isolate objects with a rectangular bounding box. Bounding box approaches proved to be much faster, but they lose accuracy. The problem became a trade off between speed and quality of object representation.

Correlation filters were introduced, and they have delivered significantly improved results. A correlation filter essentially computes rotations and translations of the ground truth representation of the object, allowing a convolution analysis of these against future frames (Bolme et al., 2010). The output of a correlation filter is an assignment of similarity score of the ground truth kernel to each location of the newly computed time frame. The highest score is the location we have tracked the object into in the new frame. The majority of newer online object tracking implementations have come to use this correlation filter approach.

The most interesting recent development on top of the correlation filter approach that I have seen is a very fast methodology of reintroducing binary masking (Wang et al., 2019) called SiamMask. This family of approaches uses siamese networks, a term that refers to the use of 2 twin neural networks at the same time. Internally, the 2 twin networks share the same edge weights, meaning they can be trained together quickly and learn to make similar decisions on similar inputs. Siamese networks are particularly useful in the task of image matching which is why they have been been adopted into this object tracking framework.

Siamese network approaches with correlation filters showed decent performance with offline training and online object tracking. They lost accuracy over the traditional object classification per frame methodology. SiamMask reintroduced per frame binary masking through its additional loss function, generating a significant improvement in accuracy while still operating at a fast enough frame rate for online object tracking. There are a few variants of SiamMask, trading speed for accuracy depending on how the network is set up.

At 30 frames per second, the human brain interprets images as continuous movement. One version of SiamMask achieves very good tracking accuracy while computing 55 frames per second. 55fps does not lose any reaction time in comparison to a human pilot. Therefor, this approach provides the output necessary for making change of flight path decisions plenty fast enough to be used in automated flight. I will

train and use an implementation of the SiamMask tracking network to detect obstacles and produce positions over time. These can then be used to calculate obstacle relative velocity and predict future obstacle positions. Ultimately this provides all of the necessary information I need to make collision avoidance decisions that will be passed to the flight controller.

Aircraft Retrieval

Overview

Safe landing and retrieval of this aircraft under various weather and location scenarios requires the design of a landing mechanism that does not depend on a runway. Currently, most commercial drones are quad rotor vehicles that do not require a runway to land. They are capable of receiving a landing coordinate from the flight plan or the drone operator, and landing precisely on the set location. This functionality is very desirable, and it has inspired some fixed wing vehicles, like the one described here, to adopt a vertical take off and landing (VTOL) system.

VTOL

This UAV will be able to support vertical take off and landing of the system through PX4, but it is not the most desirable solution. Under fair weather conditions in which battery consumption is not pushed towards its limits on a given operation, VTOL will be the desired retrieval mechanism. It is challenging to perform well with a single rotor on windy days, though.

Clear and calm weather conditions cannot be expected at all times. They are especially unlikely in some of the scenarios in which this alternative landing functionality is necessary. Further, one of the main goals of this project is to conserve power in order to lengthen max flight time. It is undesirable to add a second motor. Power consumption from vertical take off and landing with one rotor will be high. For these reasons, I have planned for an alternate landing system that will permit this UAV system to maximize its flight time when needed.

Parachute System

Zipline, a drone delivery company, has come up with their own unique landing system. Based on modern day aircraft carriers, their drone recovery system provides a tail hook to catch the drone mid-flight and stop its forward movement within seconds. The drone is hooked between a platform of four extended poles that then allow the drone to dangle above the ground to be retrieved. This system demonstrates the creativity that can be applied to fixed wing UAVs, but it does have a drawback that my UAV can’t afford. Zipline’s drones must land at a designated retrieval pad.

Designing a sturdy yet transportable retrieval pad to use for operations that vary in location is not a trivial task. At best, it increases the transportation costs for each mission. Zipline’s system is very effective for their use case of package delivery to and from their facility, but for other drone use cases it is very limiting on the range of operations plans.

Cirrus, an airplane manufacturer, designs their planes with an emergency landing mechanism for situations in which no other landing possibilities can be performed safely. In a Cirrus aircraft, the pilot can pull a lever in the cockpit that punctures the

canopy of the airplane. After doing this, the pilot can deploy a parachute that is big enough to deliver the entire plane, and the people in it, safely to the ground.

In a passenger airplane, deployment of this safety system compromises the structural integrity of the aircraft. Once a Cirrus plane's parachute has been deployed, it is unlikely that the airplane will ever receive an FAA air worthiness certificate again. The forces in play at a smaller scale do not require an aircraft parachute to be contained within a pressurized canopy, though. At the size and speed of this UAV, a parachute can be employed in a reusable and easily deployable fashion that does not compromise the structural integrity of the aircraft at all.

Inspired by the Cirrus safety parachute, my UAV design will contain a small parachute of its own. This parachute will only add (up to) 2 pounds to the total system weight, and it will be deployable through a default safety procedure. It is preferable that the VTOL system is used because the location of landing with VTOL can be very precise. However, this redundancy in aircraft retrieval provides my drone system with a much more reliable guarantee that the aircraft, especially the swappable payload, makes it back to the operator safely.

General Landing Procedure

Upon takeoff, a default GPS pin will be saved. During flight, this pin can be overwritten by remote control or the units on the ground. Any change can be communicated to the aircraft through the ground control station. If the emergency retrieval process is engaged, the UAV will navigate back towards that pin, descend into slow flight, and then enter one of the two landing routines.

Slow flight, for fixed wing aircraft, is the orientation and control surface settings that allow the vehicle to maintain altitude with minimal forward velocity. In essence, slow flight configuration places the aircraft on the verge of stalling. This means that with any less forward velocity, the wings are not generating enough air resistance upward to keep the aircraft in the sky. This is a nose up orientation, or elevator down control surface configuration. For larger airplanes, slow flight configuration involves the use of full flaps. When a fixed wing aircraft is in slow flight, altitude can be controlled with engine power; Speed can be controlled with pitch.

If the battery level of the system is above 15%, and the flight plan has not specified differently, the aircraft will use the PX4 vertical landing routine. At the point of establishing slow flight, the UAV can use its elevator to pitch the nose up, and then increase engine power to enter a vertical orientation. More specifics of this procedure are omitted because PX4 already provides a functional VTOL module for us to use.

If the battery level of the system is critically low, or if the flight plan has specified the use of a parachute landing, the UAV will deploy its parachute once slow flight has been established. At this point, the parachute will endure minimal force from slowing the aircraft down, allowing the system to use a light weight parachute.

Cost Breakdown

Overview

The cost of building a custom, fixed wing UAV varies with a lot of factors. The main things to keep in mind are size, components, payload capabilities, and design materials. Larger aircraft will tend to be more expensive as they require more material, a bigger power source, and a bigger engine. The design materials will primarily be a choice between carbon fiber and aluminum. These can be cut in a machine shop. Aircraft that offer more capability, such as autopilot and complex sensor systems, will be significantly more expensive than more basic designs. Hardware component choices will depend on desired capabilities and have a large range of options and price points.

The range of commercially available small UAV's is huge. Straightforward aircraft can be assembled for a few hundred dollars, while drones used for more complex tasks can cost up to $10,000. Building your own UAV will keep unit costs lower and offers custom functionality, but it is a very complex project.

This UAV is intended to support swappable payloads which can vary significantly in terms of price and functionality. The payload itself is a compute and sense system. This means that depending on sensors and required on board computation power, the payload alone can add several thousand dollars to the aircraft cost. My cost breakdown will not consider the cost of the payload because it is intended to be swappable based on desired needs.

Design Specifics

We will use the NVIDIA Jetson AGX Xavier as our flight controller, which costs about $2,500. The sensor system has 3 parts. First, the Adafruit 9-DOF LSM9DS1 costs about $40. Second, the u-blox NEO-M8N costs about $30. Finally, the Adafruit MPL3115A2 costs about $10. Together, this UAV's sensor system comes in at $80. My communication system, the FPVDrone 3DR 500MW Radio Telemetry Kit, adds another $100. The ZOP Power 4s 14.8 V battery can be selected in a few different sizes. I will budget $30 for the 5500mAh 75C option. The servo system requires 4 Hitec HS-5065MG servos. Each one costs $50, so altogether this UAV's servo system will cost $200.

I cannot estimate the full payload cost because parts of it will be entirely use-case dependent. The front facing camera is a requirement for our detect and avoid system, though, so I will include that here. The Zed 2 spatial AI camera costs $450. All together, our hardware and minimal payload requirements add up to cost $3,360.

The software libraries involved do not add to the cost of this project. PX4 autopilot, MAVLink, and QGroundControl are all open source projects that can be built and installed from source code hosted on GitHub.

Materials costs cannot be estimated until a later stage of this project when a full mechanical design is completed. It is likely to add up to a couple hundred dollars to the system cost. We should be able to design a parachute system complete with an

electronic trigger for under $50. Also, I did not include a component choice for engine yet because that will be dependent on size and weight of the mechanical design. A UAV like this will use a brushless electric motor. Powerful motors for this purpose can range from $50 to $200, so I will bring the price estimate in full to ~$4,000.

Summary

It is important to note that at $4,000, over half of the system cost comes from the high-end AGX Xavier processor. For a minimal approach, the NVIDIA Jetson nano could be good enough and shave $2,000 off of this cost. I opted for the most powerful processor, as my main interest was cutting edge computer vision techniques. Likewise, the Zed 2 Spatial AI camera adds $450 and could likely be downgraded without affecting usefulness of our detect and avoid system. A more minimal front facing camera could lower the project costs a few hundred dollars.

Relevance

Importance of Computer Vision

Computer vision has offered many advancements in artificial intelligence lately. In whole, computer vision is the approach to allowing computer systems to analyze and make decisions based on images and image streams of a real world environment. The main objectives in computer vision approaches are object detection, classification/ recognition, pattern recognition, and object tracking.

The range of applications of computer vision technology is huge. When applied to image and video analysis, computer vision algorithms provide a new wave of data extraction. Faces and objects can be identified. This is useful for policing, security, and other tasks. In medicine and biology, computer vision is allowing for more accurate diagnosis as well as deepened understanding in research. When applied to robotics, like I have done in this project, computer vision promotes autonomous movement and enhanced interaction between robots and their environments. When used with emerging virtual and augmented reality technologies, computer vision allows for improved control and interaction of the user in the virtual environment. Emerging technologies, greatly enhanced by computer vision, are tackling existing problems and defining massive new markets.

Importance of UAV's and Automated Flight

UAVs and drones have been growing in popularity and use over the past decade. Originally, they were developed for military operations. Now that the hobbyist community has grown and open source software libraries have provided wider access to the core algorithms that permit reliable flight, drones have become a common entity in our world today. They already provide enhanced business operations from surveying and agriculture to small photography businesses. UAVs have made use of technological developments in sensor technology and cameras to provide cool and useful features to their users.

As autonomous flight becomes safe and reliable, we will see even more utilitarian and commercial uses of drones become commonplace. Perhaps the biggest focus of autonomous drone flight development is centered around drone delivery systems. Amazon, for instance, is focused on research and development of its own drone delivery fleet. Other companies, like Zipline which was mentioned previously, are also emerging with their own approaches to drone delivery.

Once the regulatory and safety obstacles are overcome, drones will make delivery more efficient while pulling cars off the street. This has the chance to improve traffic and energy consumption problems, which will greatly benefit our world. Drones are already helping improve food production and they are performing civic duties like surveying and infrastructure design. The continuation of technological development of UAVs will continue to benefit us far into the future. This is why projects like this one are important investments to make today.

References

Aerospaceweb.org. (n.d.). *Ask US - origins of control surfaces*. Aerospaceweb.org | Ask Us - Origins of Control Surfaces. Retrieved December 18, 2022, from http://www.aerospaceweb.org/question/history/q0103.shtml

Bolme, D., Beveridge, J. R., Draper, B. A., & Lui, Y. M. (2010). Visual object tracking using adaptive correlation filters. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/cvpr.2010.5539960

FAA. (n.d.). *What kind of drone flyer are you?* What Kind of Drone Flyer Are You? | Federal Aviation Administration. Retrieved December 18, 2022, from https://www.faa.gov/uas/getting_started/user_identification_tool

FAA. (n.d.). *Certificated remote pilots including commercial operators*. Certificated Remote Pilots including Commercial Operators | Federal Aviation Administration. Retrieved December 18, 2022, from https://www.faa.gov/uas/commercial_operators

FPVDrone. (n.d.). *Amazon.com: FPVDrone 3DR 500MW Radio Telemetry Kit 915Mhz Air and ...* Amazon. Retrieved December 19, 2022, from https://www.amazon.com/FPVDrone-Telemetry-Transmit-Pixhawk-controller/dp/B074V6FKZR

Hitec. (n.d.). *Hitec RCD 35065s HS-5065MG high Torque Metal Gear Feather Servo*. amazon. Retrieved December 19, 2022, from https://www.amazon.com/Hitec-RCD-35065S-HS-5065MG-Feather/dp/B001GR7YXC

Industries, A. (n.d.). *Adafruit 9-DOF LSM9DS1 breakout board - Stemma QT / Qwiic*. adafruit industries blog RSS. Retrieved December 18, 2022, from https://www.adafruit.com/product/4634?gclid=CjwKCAiAkfucBhBBEiwAFjbkr6Bvqag1e3YX9jW4HECax8YFgy--1zKqOsXcnU6Gp_uFt7voUpKH0xoCKE8QAvD_BwE

Industries, A. (n.d.). *MPL3115A2 - I2C barometric pressure/altitude/temperature sensor*. adafruit. Retrieved December 18, 2022, from https://www.adafruit.com/product/1893

Iris Automation. (n.d.). *Casia detect-and-avoid*. Retrieved September 8, 2022, from https://www.irisonboard.com/casia/

MAVLink. (n.d.). MAVLink Developer Guide. Retrieved December 18, 2022, from https://mavlink.io/en/

NVIDIA. (n.d.). *Nvidia Jetson TX2: High performance ai at the edge*. Retrieved September 8, 2022, from https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-tx2/

Pixhawk. (n.d.). *Pixhawk/pixhawk-standards: Pixhawk Standards*. GitHub. Retrieved December 18, 2022, from https://github.com/pixhawk/Pixhawk-Standards

QGroundControl. (n.d.). QGroundControl User Guide. Retrieved December 18, 2022, from https://docs.qgroundcontrol.com/master/en/index.html

u blox. (2022, May 20). *NEO-M8 series*. u-blox. Retrieved December 18, 2022, from https://www.u-blox.com/en/product/neo-m8-series

Wang, Q., Zhang, L., Bertinetto, L., Hu, W., & Torr, P. H.S. (2019). Fast Online Object Tracking and Segmentation: A Unifying Approach. https://doi.org/10.48550/arXiv.1812.05050

Zipline. (n.d.). *Technology: Zipline - Instant Logistics*. Retrieved September 8, 2022, from https://www.flyzipline.com/technology

Zop Power. (n.d.). *Zop Power 4S 14.8V lipo battery*. aliexpress.com. Retrieved December 18, 2022, from https://www.aliexpress.us/item/3256803472390562.html?spm=a2g0o.productlist.main.1.73d0459eaDbP2Y&algo_pvid=db880045-d2d9-40f6-8dc7-c4911958d576&algo_exp_id=db880045-d2d9-40f6-8dc7-c4911958d576-0&pdp_ext_f=%7B%22sku_id%22%3A%2212000026686477661%22%7D&pdp_npi=2%40dis%21USD%2110.47%2110.47%21%21%21%21%21%40210213c81671311277412 7716d0733%2112000026686477661%21sea&curPageLogUid=QMDtZa8co7TD