

SANTA CLARA UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Date: June 12, 2021

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Connor Azzarello
Chris Gerbino
Ruchir Mehta

ENTITLED

Enhanced Sensing Methods for UAV-Based Disaster Recovery

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF SCIENCE IN COMPUTER SCIENCE & ENGINEERING



Thesis Advisor



[Nam Ling \(Jun 15, 2021 10:58 PDT\)](#)

Department Chair

Enhanced Sensing Methods for UAV-Based Disaster Recovery

by

Connor Azzarello
Chris Gerbino
Ruchir Mehta

Submitted in partial fulfillment of the requirements
for the degree of
Bachelor of Science in Computer Science & Engineering
School of Engineering
Santa Clara University

Santa Clara, California
June 12, 2021

Enhanced Sensing Methods for UAV-Based Disaster Recovery

Connor Azzarello
Chris Gerbino
Ruchir Mehta

Department of Computer Science & Engineering
Santa Clara University
June 12, 2021

ABSTRACT

Natural and human-caused disasters devastate and displace civilian populations. Over the past century, the rate at which these catastrophes occur has increased dramatically. Climate change and unsustainable human behaviors are large contributors to the occurrence of natural disasters, therefore it is likely this upward trend will continue. The region of the world where a disaster takes place often determines how severe the implications are for the affected civilians. The devastation that occurs from a disaster is much greater in low-resourced regions of the world.

Unmanned aerial vehicles (UAVs) are commonly used to assist first responders during disaster response. The existing UAV technologies focused on disaster recovery have proven to be quite effective, however, they are cost prohibitive, therefore limiting their use in developing regions of the world. Unlike the existing UAV-based disaster response technology, we propose a solution that enables indigenous first responders to reap the benefits of UAV-based disaster response technology.

Through extensive evaluation of cutting-edge single-board computers, UAV hardware, and computer vision models, we have created a UAV system that has comparable flight time and flight capabilities as the existing industry standard solutions. Additionally, our system capitalizes on the functional drawbacks of the current solutions. It is more modular, allowing for a single UAV to be used in a variety of disasters, and it boasts the capability of real-time computer vision. Most importantly, our system can be recreated for one-eighth the cost of a consumer alternative with similar functionality. As a result, first responders in low-resourced regions have access to affordable disaster response technology that can be used to save lives.

Acknowledgements

We would like to thank the School of Engineering at Santa Clara University for providing us with this exciting opportunity to research, design, and innovate. A special thank you must be given to our advisor, Dr. Behnam Dezfouli, who continually provided us with guidance and invaluable lessons throughout this year long project. Additionally, we would like to thank Cisco and Ron Snyder for providing imperative support. Finally, we would like to thank the Gregory Theyel, Allan Baez Morales, and the Frugal Innovation Hub for their indispensable guidance.

Table of Contents

1	Introduction	1
1.1	Motivation	1
1.2	Related Works	2
1.2.1	Research	2
1.2.2	Industry Standard Solutions	3
1.3	Problem Statement	3
1.4	Solution	4
2	Requirements	5
2.1	Functional Requirements	5
2.1.1	Critical	5
2.1.2	Recommended	6
2.1.3	Suggested	6
2.2	Non-Functional Requirements	6
3	Technologies Used	8
3.1	Description	8
3.2	Hardware	8
3.2.1	Single Board Computer	8
3.2.2	UAV Frame	9
3.2.3	Flight Controller	9
3.3	Software	9
3.3.1	Flight Software	9
3.4	Development Tools	10
3.4.1	JetPack Software Development Kit	10
3.4.2	NGC Catalog	10
4	System Evaluation	11
4.1	Computer Vision Performance	11
4.2	Price	13
4.3	Comparitive Peformance	13
4.4	Model Retraining - SSD MobileNet v2	14
4.5	Assembled System	14
5	Societal Impact	17
5.1	Ethical	17
5.2	Social	17
5.3	Political	17
5.4	Economic	18
5.5	Health and Safety	18
5.6	Manufacturability	18
5.7	Sustainability	18
5.8	Environment Impact	18

5.9	Lifelong Learning	19
5.10	Compassion	19
6	Limitations and Challenges	20
6.1	COVID-19	20
6.2	Testing	20
7	Conclusion	21

List of Figures

3.1	Overview of system architecture	8
4.1	Camera mounting and connection to the NVIDIA Jetson Nano	16
4.2	NVIDIA Jetson Nano mounting configuration	16

Chapter 1

Introduction

1.1 Motivation

Natural and human-caused disasters can cripple, displace, and diminish civilian populations. Disasters are not only physically damaging to the affected communities, they also create a major economic cost. In 2017 alone, there were 335 disasters and, on average, 68,273 deaths per year attributed to natural disasters. In 2017, there was a global economic loss of \$335 billion dollars due to natural disasters [1]. Over the past century, the rate at which these catastrophes occur has skyrocketed. Because global climate change and unsustainable human practices contribute to the occurrence of a disaster, it is likely this trend will continue.

Unmanned aerial vehicles, or UAVs, are used as technological solutions to increase the effectiveness of first responders during the search and rescue phase of disaster response. Existing UAV-based disaster response technology has proven to be quite effective when applied to various emergency situations. Currently, UAVs are being used for disaster relief situations such as hazardous chemical spills, the need for mapping high risk areas, assessing structural damage, delivering emergency infrastructure and supplies, and extinguishing wildfires.

Developing countries and non-governmental organizations (NGOs) need access to affordable disaster response technology. The current industry standard solutions are cost-prohibitive, which limits first responders in low-resourced regions to traditional search and rescue practices. These practices are less effective and provide much higher risk to the first responders.

At Santa Clara University, a Jesuit institution, we are expected to uphold a high ethical standard in both our professional and personal lives. This requires us to continually evaluate our moral duties, virtues, and obligations. Each member of this team recognizes that engineers have a moral duty to assist others who are less fortunate than us in areas of competence. Additionally, we understand that every contribution we make as engineers should benefit society and improve human well-being. This project provided an opportunity for us to fulfill our moral obligations as engineers and show the culmination of our Jesuit engineering education.

1.2 Related Works

We began by familiarize ourselves with the industries that are relevant to this project. Understanding the industries surrounding UAVs, computer vision, and disaster response was critical to ensuring our solution was novel. This section first discusses research projects that provided valuable information that was used during the design and construction of our solution. Second, we will address the industry standard UAV technology that was developed to be used during disaster response.

1.2.1 Research

Nanonets specializes in developing convolutional neural networks (CNNs) for aerial UAVs [2]. As it turns out, UAVs are used for a variety of industry applications such as surveillance over solar panel farms and monitoring progress in construction projects. This latter example proved to be very insightful for our project. This construction progress monitoring solution revolves around using computer vision models to both track the outlines of the building or other infrastructure being built within a construction zone, and using the density of human subjects within the zone to act as a proxy for the manpower currently working to make progress on construction tasks. Although there was minimal written documentation regarding this use-case, investigating *Nanonets* proved that sufficient research exists within industry applications to implement accurate and useful computer vision models on-device to solve real-world problems.

Microsoft is one multinational technology company that is investing a lot of resources into computer vision research using UAVs, especially those that utilize a very small chassis and low-power embedded computational hardware; these systems are referred to as micro aerial vehicles (MAVs). MAVs can be exceptionally powerful in practice because of their low cost, relative to those that use much larger frames, thereby allowing much easier deployment of vehicles "swarms" where many UAVs work in tandem to accomplish a task while communicating amongst each other throughout the process to remain efficient. Microsoft published research on one such project wherein MAVs swarms can be deployed using monocular camera systems and mapping entire environments in a distributed to fashion [3]. This paper was very useful in the ideas it explored regarding methods that could prevent the need for constant communication between UAVs. Since this was research originating from Microsoft, naturally, there was a fair amount of emphasis put on Microsoft AirSim as a testing platform for simulating results. Although we did not use this software suite in our approach, we believe it could be very useful for future projects within the field of UAV computer vision research.

A student research project from Indiana University [4] created a low-cost UAV-system that was capable of real-time object detection. The paper discusses how CNNs are the standard solution for solving image-related machine learning problems as this class of model is extremely powerful. However, the paper concludes that CNNs are too

computationally demanding to be used for real-time object detection onboard a UAV. This is because the CNNs require a high-performance GPU, which do not come in small form factors. As a result, the paper proposes that a bulk of the computation should be done on a computing cloud. This would allow for the UAV to still be capable of on-board low-level object detection, while, removing the need to have a powerful computation device on-board. This paper prompted us to examine cloud-based computing options for our UAV-system. However, this was not necessary as NVIDIA had released a single-board computer, after this paper was published, that meets the computing power requirements necessary to use CNNs to perform object detection on-board the UAV.

1.2.2 Industry Standard Solutions

Industry standard UAV solutions for disaster response, such as the DJI Mavic 2 Enterprise Advanced and the Parrot ANAFI USA, are cutting edge and available for purchase as a single off-the-shelf product. Our team extensively researched these systems to understand what attributes makes a UAV system effective in a disaster scenario. In other words, our research allowed us to define requirements that our frugally designed system must have to be considered a competitive market solution. These requirements are: autonomous flight, high maximum flight time, and real-time image capture.

Researching existing UAV systems for disaster response also allowed us to identify the shortcomings of the industry standard solutions and capitalize on them within our system. The shortcomings we identified are: low modularity, high cost, and the absences of on-board real-time image analysis. These drawbacks helped direct the project and define additional requirements for our UAV system. The additional requirements for our system are: high modularity, low cost, and the ability to perform on-board real-time image analysis.

1.3 Problem Statement

Natural and human-caused disasters are often more devastating to developing countries and low-income areas. These regions are more vulnerable to disasters because individuals residing in these areas often occupy housing that does not have the structural integrity to withstand earthquakes, damaging winds, or floods. In addition, low-income communities are more likely to be located on river banks, reclamation ground, and steep slopes. These locations increase the devastation, injury, and mortality caused by a disaster.

The technological gap between developed and developing countries is mirrored by the disproportionate devastation caused from a disaster. Technological innovation enhances human well-being, however, when technology is widely inaccessible only some can reap the benefits. Accessibility becomes increasingly important when the technology has the ability to relieve human suffering or save lives. Disaster response technology has these abilities, unfortunately, commercial UAV-based disaster response technology is cost prohibitive, and therefore it is not accessible by non-governmental organizations or first responders in developing countries.

1.4 Solution

The primary objective for our project is to make disaster response technologies accessible to NGOs and first responders in low-resource regions of the world. To do so, we designed and developed a low-cost UAV-based solution to assist in disaster response. Our solution greatly increases the accessibility of UAV-based disaster response technology because it is comparable to the current industry standard solutions, while costing an eighth of the price. Additionally, the solution is novel because of the high modularity, low-cost, and the ability to perform on-board real-time image analysis.

Throughout this project we will also be working in parallel with another group consisting of two members, Cameron Burdsall and Mark Rizko. Cameron and Mark are working to create a drone mesh Wi-Fi system for disaster scenarios. When both projects have been completed, we will combine the technologies to create a more complete solution for disaster recovery. Our combined solution will have the capabilities to assist in safely identifying victim locations, and resurrecting destroyed communication infrastructure.

Chapter 2

Requirements

2.1 Functional Requirements

2.1.1 Critical

1. Precise & Accurate Identification

A system based mainly on object-detection capabilities is only as useful as the accuracy of the resulting data. Most of the popular object-detection frameworks that currently exist display results based on a certain confidence threshold generated by the model itself. Therefore, it is imperative that this threshold be set to a very high value, in order to limit the number of false positive results. On the other hand, false negatives are similarly problematic and contingent on the training data used to train the model.

2. Fast Inference Speed

The term *inference* refers to the speed at which a computing device can process a set of data and output a result using a mathematical model. In this project, the “sets of data” that will be processed are digital images representing each frame of the video. The model that we will employ is likely to be one of the many industry standard deep learning models that are commonly used for computer vision. Inference speed in such a scenario is more accurately represented as the number of video frames that can be processed per second. Due to the state of low-power computing and wireless communication at this time, it is likely not feasible to display this video feed in real time to users, but every extra frame that can be captured and processed each second makes the resulting data more accurate and provides more data points to first responders.

3. Real-Time Notification of Detection

As mentioned above, the power usage and computing overhead necessary to transmit an annotated video stream to users is likely to high to be technically feasible in this application. However, the data that will be generated by the imaging array on each UAV is incredibly useful to disaster response teams. As a result, it was deemed a

critical requirement that the system must provide instantaneous notification to users when a human (or other object class, depending on circumstances) is detected by the system. This notification allows the disaster response teams to modify their flight path or, where necessary, take manual control of the flight in order to procure more data that may save additional lives.

4. Lowest Possible Power Footprint

The maximum flight time of the UAV is completely contingent on the battery capacity available to its DC motors while in flight. Therefore, it is imperative that any auxiliary functions require as little power as possible. Preliminary calculations showed us that the maximum power draw of most single board computers that exist today are significantly lower than that of drone flight, but every optimization is useful, and incredibly important, when such a system is deployed into a disaster scenario.

2.1.2 Recommended

1. Metrics

It is exceptionally difficult to build a UAV system from scratch that is capable of streaming video capture back to a base station in real-time, especially with the resources we have available, but it could be just as useful to provide users with real-time data that acts as a synthesis of the data captured and processed on the UAV. One such metric might be to transmit a count of the number of victims that have been identified along a flight path.

2. UAV Cooperation

Allow for multiple UAV's to be used simultaneously during a disaster response scenario. This would be useful as it would allow users to deploy a large fleet of UAVs in a disaster scenario and optimize their flight paths such that they do not overlap and can produce the largest amount of actionable data for each battery charge cycle.

2.1.3 Suggested

1. Multiple Sensing Technologies

Implement sensor fusion to utilize other classes of sensors, such as infrared (IR) cameras for heat detection or high sensitivity microphones to detect victims that may not be visible due to rubble or other physical obstructions.

2.2 Non-Functional Requirements

The system must be:

- Buildable and usable by non-technical users.

The UAV must be able to be assembled easily without expertise or the assistance of expensive machinery. Additionally, the documentation for each product should be clearly written so that others can recreate our system. The software required to make the system function as expected should not need to be altered in any way.

- Affordable

Hardware suggested by the team should be extensively researched and a cost-benefit analysis should be conducted to ensure that the components are affordable, accessible, and reliable. The cost of the complete system should be less than one-thousand dollars to ensure our system is not cost prohibitive.

- Reusable

All hardware components purchased for the UAV system should be reliable. Similarly, the hardware selected should be from a dependable manufacturer who offers replacements for individual parts.

- Modular

The system must have high modularity as it should be configurable for various types of disasters. This means that the selected components need to be universally compatible. Similarly, the single-board computer of choice should be able to handle a variety sensors.

Chapter 3

Technologies Used

3.1 Description

Figure 3.1 provides a visual representation of the main components of our system, their relation to one another, and the processes that each fulfills. This chapter will detail the important decisions made regarding hardware and software selections for our final implementation, as well as the rationale for why we believe a certain component was well-positioned to satisfy our objectives and requirements.

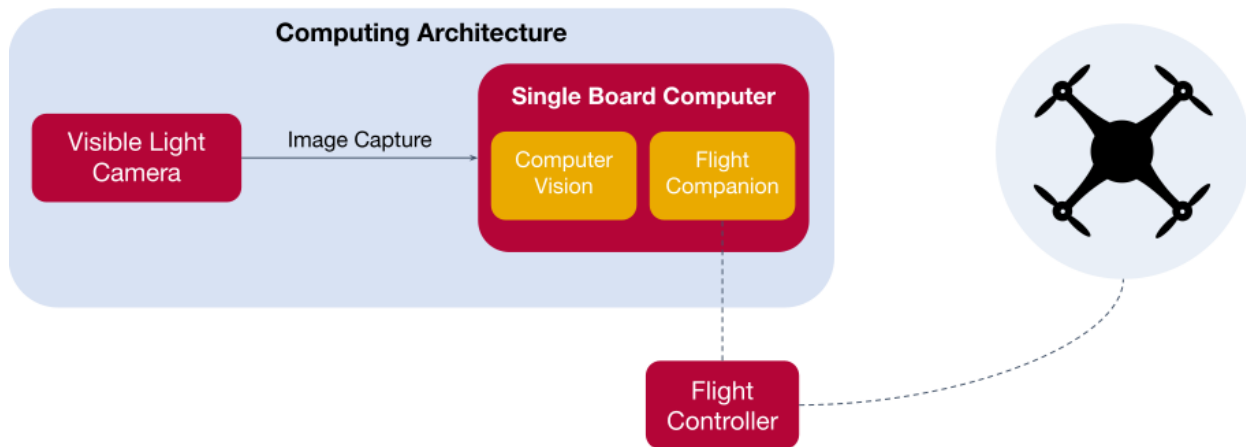


Figure 3.1: Overview of system architecture

3.2 Hardware

3.2.1 Single Board Computer

Nvidia Jetson Nano - 4 GB

A single board computer was imperative to achieve the non-functional design requirements outlined in Section 2, because of their low cost, energy efficiency, size and modularity. After researching performance metrics for single board computers (SBCs) across a variety of industry standard computer vision models, it was determined that the

NVIDIA Jetson Nano would be the most cost-effective and performant SBC choice, especially compared to competing platforms. On one hand, the Raspberry Pi series of SBCs would be satisfactory in regard to power usage, but significantly underpowered for computer vision applications. On the other hand, the Google Coral TPU Dev Board boasted incredible inference speeds where comparison was possible, but its major caveat consisted of only being capable of running Tensor Flow Lite (TFLite) models, which would have a significant detriment on the modularity and flexibility of the overall platform.

3.2.2 UAV Frame

HolyBro S500v2 Kit

In order to satisfy the requirements of high flight time, low cost, and suitable size to mount all components, the choice was quickly narrowed down to a *quadcopter*, meaning a UAV with four motors. A UAV with more than four motors, such as a hexacopter or octocopter, would provide a greater upward force, however, our drone will not need to carry objects. Additionally, more motors would cause our system to consume more power and therefore have a lower maximum flight time.

3.2.3 Flight Controller

Pixhawk 4

The flight controller acts as the *brain* of a UAV system. The PixHawk 4 was chosen because it provides the functionality listed below at a competitive price point.

- Autonomous flight
- Detailed logging
- Interface to companion computer

3.3 Software

3.3.1 Flight Software

QGroundControl

QGroundControl boasts a high degree of compatibility with many flight controllers and UAV systems, including the HolyBro S500 and PixHawk 4 flight controller. We selected QGroundControl for our project because of its capabilities that are listed below. Additionally, we tried other flight software such as MavProxy that have similar capabilities but we found that QGroundControl was the most intuitive and user-friendly.

- Various route-planning tools

- Universal compatibility
- Open-source software

3.4 Development Tools

3.4.1 JetPack Software Development Kit

- Jetson Linux Driver Package (L4T)
- Ubuntu 18.04 LTS Build
- CUDA accelerated libraries and APIs
- Samples, documentation and developer tools

3.4.2 NGC Catalog

NVIDIA Ecosystem Package Manager

Easy access to AI building blocks that allow for a more efficient development process. A subset of assets within the catalog are specifically built, optimized, and tested to work with the Jetson Nano family of single-board computers and allows for smooth deployment of these components:

- Machine Learning Models
- Development Containers
- Software Development Kits

Chapter 4

System Evaluation

4.1 Computer Vision Performance

The first step toward achieving the goals outlined for this project was to investigate capabilities of our chosen single-board computer, the NVIDIA Jetson Nano, in regard to deep learning inference performance. It was also very important for us to profile the power consumption of the computing hardware while under specific, realistic workloads. This was done by utilizing the benchmarking utilities provided by NVIDIA through their `jetson.benchmarks` utility, found on GitHub[5]. However, this utility only performs the benchmarking on the inference speed itself. To also log the power consumption of the hardware while running these benchmarks, we modified the codebase to also leverage NVIDIA's `jtop` system monitor utility and python package that allows a direct interface with power consumption and system load metrics throughout the running time of the benchmarking utility[6].

The results of these benchmarks can be seen in Table 4.1. One interesting observation we found after executing these benchmarks was that there was minimal, or even zero, performance penalty for using the 5 Watt power mode. This behavior can be attributed to the way in which the 5 Watt power profile functions in comparison to the Full Power (10W) profile on the Jetson Nano. The main difference between the two is related to how many CPU cores are enabled; in the 10W mode, all 4 cores are activated and usable, whereas the 5W profile disables two of those cores. Since computer vision and deep learning models are generally extremely GPU intensive, this result is rather logical, since it is likely that any performance bottlenecks are a result of components other than the CPU. This allowed our team to make the decision on proceeding with our implementation using the lower power consumption profile for its benefits to power consumption, and therefore UAV flight time, without concerns about large reductions in performance.

Our goal was to build our final implementation around the most versatile use case for disaster scenarios. For this reason, we chose to focus on *Object Detection*, because we believed that it would be the most useful in the widest variety of disaster recovery applications. Object Detection refers to computer vision models that take an image, or series of images in the case of videos, and produce classification of objects within the input image, while providing *bounding boxes* that can be used to overlay where the model believes that an object is within the image. This differs

from its nearest relative, *Classification*, because it has the ability to classify more than one object in a scene, rather than classify the contents of the image as a whole. In a disaster recovery scenario, it is entirely possible that there will only be one useful "object" to identify in a scene, but this is far from a given. If we chose to only pursue general classification models for our implementation, it would greatly hinder our system's ability to effectively provide information to first responders in situations with high victim density or with many objects in the scene, even non-human objects. The reason for this latter example stems from the nature of object detection models within computer vision. Generally, they tend to have the capability to detect many types of objects in a scene, such as dogs or cars, in comparison to simply attempting to identify humans.

The benchmarking application was designed to utilize the two most widely used and performant object detection models for embedded applications at the time of its design: TinyYOLO v3 and SSD MobileNet v1 [7][8]. YOLO is an acronym that stands for "You Only Look Once," pertaining to the underlying design of the model which requires only one pass of input image to produce a result, which is not necessarily the approach taken by many models designed for more powerful or accurate object detection algorithms. The "tiny" portion of the model's name describes the heavily optimized nature of the model's neural network, which grants significant increases in speed, at the cost of some accuracy. Although the Jetson Nano is by far one of the most powerful SBCs within its class, it will struggle to generate usable frame rates in practice due to the processing power of its GPU and its relatively small available RAM (4GB). Similarly, SSD stands for "Single-Shot multibox Detection" which is a different way of explaining the same approach utilized by YOLO. "Mobilenet" is also the signifier used to describe the optimized model of the original SSD model and carries the same advantages and disadvantages as the YOLO's "tiny" variant. In our testing, TinyYOLO performed slightly better in practice, while also using significantly less power on average compared to SSD Mobilenet. Although, our team decided to use the latter in our final implementation, because of its reduced model size and greater ability for end-user retraining in comparison to TinyYOLO, something we investigated and discuss in 4.4.

Model	Application	Power Profile	Inference Speed	Power Consumption	GPU Latency	GPU Load	CPU Load	
Inception v4	Classification	5W	10.58 FPS	5.029 W	94.477 ms	56.16%	85.03%	
		10W	10.60 FPS	5.570 W	94.340 ms	70.30%	40.31%	
5W		10.09 FPS	5.150 W	99.121 ms	58.92%	84.74%		
10W		10.04 FPS	5.004 W	99.596 ms	39.79%	49.45%		
ResNet 50		5W	36.67 FPS	5.029 W	27.268 ms	54.56%	86.90%	
		10W	36.88 FPS	5.564 W	27.116 ms	63.80%	41.56%	
SSD MobileNet v1		Object Detection	5W	42.56 FPS	4.360 W	23.498 ms	58.49%	94.82%
			10W	42.64 FPS	5.097 W	23.454 ms	77.88%	44.77%
TinyYOLO v3			5W	47.65 FPS	3.541 W	20.987 ms	14.25%	95.31%
			10W	47.60 FPS	4.325 W	21.008 ms	24.64%	50.62%
UNet	Semantic Segmentation		5W	16.59 FPS	3.281 W	60.266 ms	0.45%	92.68%
			10W	16.62 FPS	5.882 W	60.177 ms	89.76%	36.53%
Super Resolution	Image Processing	5W	15.26 FPS	6.396 W	65.524 ms	86.82%	78.00%	
		10W	15.26 FPS	5.943 W	65.547 ms	82.51%	36.94%	
OpenPose	Pose Estimation	5W	14.64 FPS	3.258 W	68.313 ms	0.26%	92.46%	
		10W	14.64 FPS	3.968 W	68.320 ms	0.00%	46.42%	

Table 4.1: NVIDIA Jetson Nano model performance

4.2 Price

One of our non-functional requirements, outlined in Section 2.2, was to produce a system that was affordable and accessible to those living in developing countries or working for a non-profit NGOs. It was critical that our system was accessible to these individuals as they might not have the funding to invest in enterprise solutions created by popular UAV manufacturers. It was hard to decide upon an upper limit to what our team could reasonably consider affordable, but our decision-making process when selecting components revolved around finding the most inexpensive option that did not sacrifice any of our other requirements in the process. After implementing our system, we strongly believe that we produced the most inexpensive system feasible given our constraints at the time of the project. All the components and their market prices are included in Table 4.2 which shows our system’s price was around \$835 before shipping and tax at the time of purchase. A comparison with the comparable UAVs in the enterprise disaster recovery segment can be found in Table 4.3.

Product	Price
NVIDIA Jetson Nano 4 GB	\$120
HolyBro S500 v2 Kit	\$355
Pixhawk 4	\$70
8000mAh Battery & Charger	\$270
IMX219-77 Camera	\$20
Total	\$835

Table 4.2: System pricing

4.3 Comparative Performance

The relative success of our implementation was contingent upon its performance when compared to similar UAV systems within the enterprise sector that are marketed toward disaster recovery use-cases. The two primary points of comparison were the DJI Mavic Enterprise Advanced and the Parrot ANAFI USA. Table 4.3 shows the comparison between our system and the average value for our primary requirements within these comparable systems. Although there are some features that were not achieved in our implemented system that exist within the enterprise systems, such as infrared imaging or high-quality video streaming, we accomplished our objective of meeting our critical requirements while also doing so at a fraction of the price of comparable systems. We were able to reach near parity with these systems on many dimensions and even surpassed them in some important categories, such as modularity, on-device computer vision, and the aforementioned pricing characteristics.

	Prototype	Industry Standard Averages
Flight Time	25.7 min	31.5 min
Computer Vision	Real-time up to ~480p 45 fps	No
Autonomous Flight	Yes	Yes
Modularity	High	Minimal
Price	\$835	\$6,750

Table 4.3: Comparison to industry standard enterprise disaster recovery UAVs

4.4 Model Retraining - SSD MobileNet v2

As was mentioned previously in Section 4.1, most object detection algorithms are equipped to detect a wide variety of different object classes within images. In a disaster scenario, humans are the top-priority and generally the only priority for rescue teams when rescuing victims. When models, such as SSD MobileNet, are attempting to distinguish between the 98 classes they support, there is a nontrivial amount of processing overhead to make this distinction that could be eliminated by focusing solely on humans. For this reason, we explored the process of retraining our model to only detect humans within a scene.

Although the benchmarking utility used to decide between TinyYOLO and SSD MobileNet utilized the original version of the latter model, we chose to utilize the more recent and efficient v2 of SSD MobileNet for retraining because we believed it would yield better results. We utilized portions of the documentation found in NVIDIA’s ”Two Days to a Demo” tutorial for Jetson systems to perform retraining on a pre-trained model designed for the Jetson Nano [9]. The model was retrained using PyTorch and leveraged 20,000 total images with 79,000 labels procured from the Google Open Images Dataset v6 [10]. Our model was retrained to only use the ”person” object class as this pertains to humans, the top priority in disaster scenarios. Our training set consisted of 17,058 images, our validation set contained 2,212 images, and our test set contained 730 images. The model was retrained over 30 epochs at a learning rate of 0.01 on the Jetson Nano itself, which took around 30 hours to complete. Although this took an exceptionally long amount of time, the training parameters used are viable for creating viable models, and it proved that it could be possible for end users to perform similar retraining without the need to purchase expensive and high power consumption graphics cards. Our results proved that this experimentation was successful in achieving this goal, providing an 8.16% improvement in inference speed compared to the baseline model, while also reducing the model size by 25.8%. With larger training datasets, more training time, and more powerful hardware, we expect that the improvements could be incredible for those who want to tailor their model to very specific use-cases, as long as they have these resources at their disposal.

4.5 Assembled System

After assembling the fully-functional system, shown below, we began performing an indoor test flight. This flight utilized the retrained SSD-MobileNet model previously discussed and allowed us to ensure that all components were

truly compatible and capable of flight in a simulated scenario. Although we were only able to complete this flight inside a large garage, the prototype implementation yielded nearly identical inference speeds to those found in our testing environment and produced power draw metrics in line with our estimated 25.7 minute flight time. The model was visibly able to lock onto the human subject with a high degree of accuracy both in a well-lit and darker environments. This modification was done by reducing the amount of ambient light through closing the garage door in order to only allow artificial lights to illuminate the environment. The prototype was flown at a designated height of 12 feet and was tested using both a static hover and a 4 m/s oscillating route perpendicular to the camera's field of view.

Figure 4.1 shows the method of mounting the IMX 219-77 camera to the UAV chassis. This was done using the angled mount that conveniently came with the camera sensor. This angle provides a fantastic mounting position to hold the camera for superior field of view at the altitude used for testing, although this may need to be adjusted if the UAV is planning to be deployed at a significantly higher altitude. The camera module is connected to the Jetson Nano via a Camera Serial Interface (CSI) ribbon cable. Since the Jetson Nano is specifically designed with embedded computer vision workloads in mind, it comes with two such CSI camera ports built into the module's carrier board. Although this camera connector is not often utilized in consumer-facing imaging products, like webcams, it is very common on camera modules designed for embedded systems applications because of the standard connection interface that allows the imaging sensors to directly interface with the module's CPU, without any hardware translation that may be necessary for more universal component standards such as Micro-USB or USB Type-A. This flexibility in connection methods further supports our objective of modularity as it provides three possible connection methods, CSI, Micro-USB, and USB Type-A, to the Jetson Nano and therefore allows a wide range of components to be utilized within a system using a Jetson Nano.

Figure 4.2 is indicative of the mounting position of the NVIDIA Jetson Nano on the top of the Holybro S500 UAV chassis. It was placed in this position out of necessity, in order to ensure proper weight distribution of the components and available space on the chassis. It was mounted using modified mounting holes on the bottom of the Jetson Nano's case and screws that were long enough to secure the case directly to the chassis, while also having room to use rubber, shock-absorbing spacers between the two mount points. This was done to limit the vibration that was transferred to the Jetson Nano, in addition to ensuring that the weight of the circuit board would not shift during flight. In our testing, a significant amount of oscillation can propagate throughout the system with components that were even slightly loose, since the flight computer will expect a center of gravity that is stationary throughout flight and over-compensate for this movement, creating a feedback loop that could very easily cause a severe malfunction if not addressed.

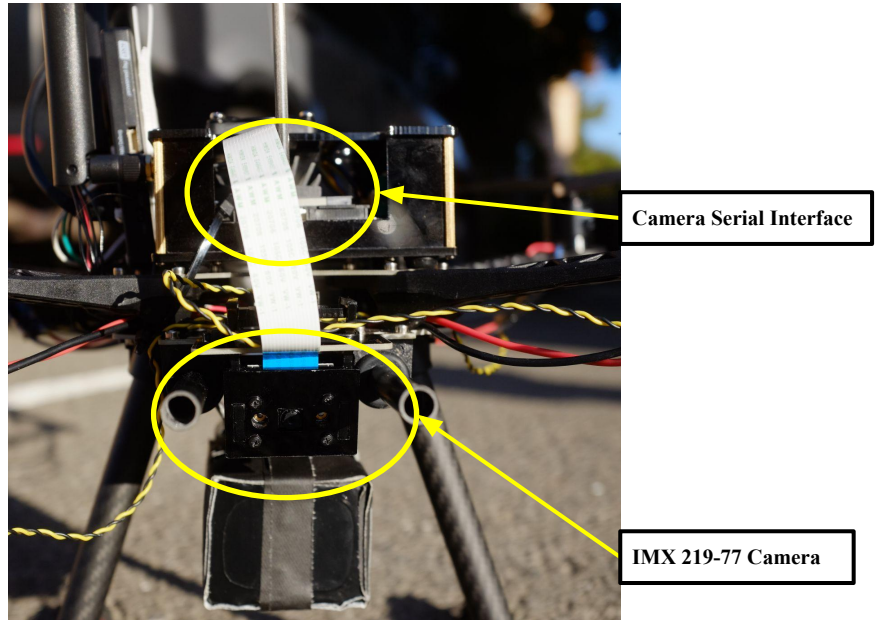


Figure 4.1: Camera mounting and connection to the NVIDIA Jetson Nano

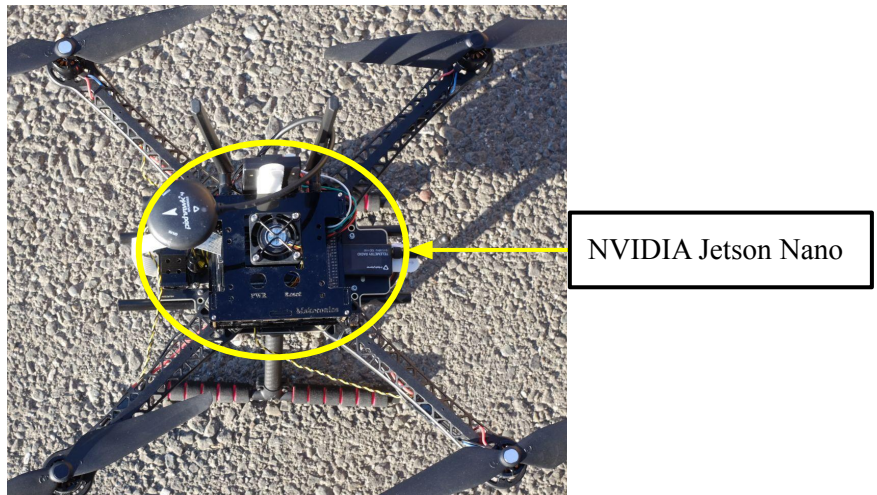


Figure 4.2: NVIDIA Jetson Nano mounting configuration

Chapter 5

Societal Impact

5.1 Ethical

The right to life is a fundamental human right. It is especially critical to uphold this right for individuals who are affected by natural disasters. Those who have been displaced or physically affected by natural or human-caused disasters are entitled to protection and assistance. The UAV-system presented in this paper enables first responders in developing regions to use UAV-based technology earlier in the disaster response phase. This will increase the amount of lives saved as it allows for survivors to be found in a shorter amount of time.

5.2 Social

Technological evolution has radically changed societies and improved the quality of life of individuals around the world. As technology continues to develop we, as engineers, have a social obligation to ensure that technology is widely available and not cost-prohibitive. This is critical to closing the technological gap that is currently seen in low-resourced regions. Our UAV-system fulfills this social obligation by enabling the use of life-saving technology in developing regions of the world.

5.3 Political

The Cisco tactical operations team responds to disasters world-wide. However, the team is primarily based in the United States of America which causes a delay before the team and their technology can assist local first responders. The delay exemplifies the need for government-sponsored disaster response teams in developing regions. These teams in combination with affordable disaster response technology would allow for more lives to be saved. The UAV-system our team designed is low-cost and easily replicated. Therefore, it is both affordable for, and, usable by government-sponsored disaster response teams in low-resourced regions.

5.4 Economic

Frugality was critical throughout the process of building the UAV-system as it is intended to be used by first responders in low-resourced regions of the world. To ensure our system was not cost-prohibitive, while maintaining high functionality, we extensively researched each component and the alternatives before selection.

5.5 Health and Safety

The UAV-system created during this project is intended to be used in disaster response scenarios and therefore it directly addresses the health, safety, and well-being of individuals. It is important to note that the UAV-system also poses a risk to those in the areas it is being flown. This is because, like all aerial vehicles, there is a possibility of failure during flight. Throughout our extensive testing the team has not experienced any errors during flight with the UAV-system. Our team combated the possibility of failure to the best of our ability by taking all necessary precautions and we recommend that any team working with UAV-based technology does the same.

5.6 Manufacturability

The UAV-system was built using off-the-shelf components that are available worldwide. Similarly, the components used in the construction of the system were all well-documented and user-friendly. The UAV-system is compatible with many computer vision models, many of which are open-source. The system created can be easily recreated by anyone with a average computer competency.

5.7 Sustainability

Industry standard solutions for disaster response are not modular as they have a limited number of compatible sensors. Ideally, UAV-systems for disaster response should have extensive sensor compatibility as different types of disasters will require the use of different sensors. Additionally, these industry standard systems must be sent back to the manufacturer if they are broken or non-functional. The solution we have built solves these issues as it is modular and built on open-source technology. Our system boasts vast sensor compatibility allowing for it to be effective in a variety of disasters. Similarly, non-functional or broken parts are easily replaceable.

5.8 Environment Impact

The UAV-system we have created can be used to minimize the devastating effects of environmental disasters. This paper primarily focuses on minimizing the effects to human populations, however, with slight modification it can be used to minimize the effects to the environment as well.

5.9 Lifelong Learning

This project required extensive knowledge of UAV-systems, hardware, and various other fields our team was not familiar with. Each member was challenged with the task of learning about these fields without the direction or assistance a university course would provide. This experience helped each member enhance their self-learning abilities which will be critical when we are in the workplace.

5.10 Compassion

Natural and human-caused disasters are devastating to any country or region of the world. However, the effects of a disaster are far more destructive to low-resourced regions of the world. Similarly, low-resourced regions are not as well-equipped to handle the aftermath of a disaster due to the technological gap and resources available. Our team recognized the suffering a disaster can cause, particularly in developing regions, and created a solution with the goal of relieving the distress felt in these communities.

Chapter 6

Limitations and Challenges

6.1 COVID-19

The COVID-19 pandemic has been a large challenge for our group. Each group member had to be isolated for the duration of this project. This meant that we could not hold in-person meetings or work on the project in a collaborative fashion. This was particularly challenging when multiple group members needed a specific component to make progress on their tasks. Similarly, only one member could be present when testing our complete system. This was a formidable task as the group member had to interact with the flight software, ensure the drone was functioning correctly, and act as a test subject.

6.2 Testing

Due to city and university regulations all testing had to take place in an enclosed space. Flying in enclosed spaces hard due to the close proximity of obstacles. This was a cumbersome process as we had to take extra precautions to ensure the drone was not damaged. Indoor testing also prevented us from creating our own test data, this meant we had to rely on third-party aerial imaging data sets to train our model.

Chapter 7

Conclusion

The UAV-system we have designed had three main objectives: high modularity, low-cost, and the ability to perform real-time computer vision. Throughout the process of designing our system we learned how important these aspects are when creating a technology that is accessible to individuals in low-resourced regions.

Modularity was one of our primary objectives. Throughout the project, we researched a variety of solutions or extensions to our computing architecture, including software packages and computer vision models, that proved to be essential to building the successful prototype detailed in this paper. The integration of these components would be exceedingly difficult or likely impossible without the highly modular system we chose in the form of the NVIDIA Jetson Nano and the accompanying software ecosystem provided by NVIDIA. Comparably capable single-board computers, such as the Google Coral Dev Board, have limitations in the software that can successfully be deployed to their hardware. Driver support was also a consistent issue our team has experienced throughout previous software engineering projects and was not an issue throughout this thesis because of the first-class support from the developers at NVIDIA, in addition to the passionate, helpful Jetson Nano community on the internet. Through this, we learned that modularity, whether it be through open-source software, easy system integration standards, or the many other relevant aspects to building a flexible system, is one of the most important aspects to designing and building a system that can serve the most people efficiently, easily, and at the lowest cost.

We selected this project because we saw the potential to make direct impacts on the lives of those in low-resourced regions. To make an impact on their lives we had to ensure that our system was accessible and not cost-prohibitive. Accessibility was achieved by emphasizing frugal design throughout the project, however, it was particularly important during purchasing decisions. When making our purchasing decisions, we had two main considerations. First, we had to analyze the price of the components in relation to their function within our system. For example, we selected the NVIDIA Jetson Nano instead of the cheaper Raspberry Pi. This was because we felt the extra money was worth the drastic increase in performance. The second consideration we made when purchasing components was their availability to a variety of international markets.

Real-time computer vision on low-cost hardware is one of the most exciting developments in computer science in the last several years. At the same time, the innovation within the consumer and low-cost UAV segment has also been essential to increasing the adoption of such systems for a variety of tasks. Throughout the preliminary research we did for this project, we observed that these two domains have failed to be bridged in a meaningful way, especially at a price point accessible to the low-resource users that could benefit the most from them. It is likely that this comes as a result of two major factors, both related to manufacturability. One, it is difficult to produce products that are on the cutting edge of technology while adhering to strict pricing targets without attempting to first manufacture solutions with the flexibility provided by a "flagship" product's development budget and resources. Second, even if a company can produce such frugal solutions to these market deficiency, it is very difficult to do so while generating a profit that makes their development worthwhile. For these reasons, it leaves such innovation to the hobbyist and DIY markets, but these engineers have even less resources to successfully produce a satisfying product with any amount of consistency. Although we believe that we accomplished the goals we outlined throughout this paper, it is very easy to see why that is not the case for every person or team that attempts to do so. With novel technology comes a severe lack of support and documentation, which requires experienced developers to overcome.

Bibliography

- [1] “Natural Disasters 2017,” tech. rep., Centre for Research on the Epidemiology, 30, Clos Chapelle aux Champs 1200 Brussels, Belgium, feb 2018.
- [2] G. Kaila, “How to easily do object detection on drone imagery using deep learning,” June 2018.
- [3] S. Vemprala and S. Saripalli, “Monocular vision based collaborative localization for micro aerial vehicle swarms,” in *IEEE International Conference on Unmanned Aerial Systems*, June 2018.
- [4] J. Lee, J. Wang, D. Crandall, S. Åabanovi, and G. Fox, “Real-time, cloud-based object detection for unmanned aerial vehicles,” in *2017 First IEEE International Conference on Robotic Computing (IRC)*, pp. 36–43, 2017.
- [5] ak nv, *Benchmarks Targeted for Jetson Xavier NX (Using GPU+2DLA)*, 2020.
- [6] rbonghi, *Jetson stats*, 2020.
- [7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in *Computer Vision – ECCV 2016* (B. Leibe, J. Matas, N. Sebe, and M. Welling, eds.), (Cham), pp. 21–37, Springer International Publishing, 2016.
- [9] dusty nv, “Jetson inference: Hello ai world,” 2021.
- [10] “Open images dataset v6 + extensions,” 2021.


Enhanced_Sensing_Methods_for_UAV-Based_Disaster_Recovery

Final Audit Report

2021-06-15

Created:	2021-06-15
By:	Bioengineering Department (bioengineering@scu.edu)
Status:	Signed
Transaction ID:	CBJCHBCAABAAE3Vwaf-qjMS8I6dahEN8FPMMMzgiQII

"Enhanced_Sensing_Methods_for_UAV-Based_Disaster_Recovery" History

 Document created by Bioengineering Department (bioengineering@scu.edu)

2021-06-15 - 5:40:35 PM GMT- IP address: 24.6.105.116

 Document emailed to Nam Ling (nling@scu.edu) for signature

2021-06-15 - 5:40:56 PM GMT

 Email viewed by Nam Ling (nling@scu.edu)

2021-06-15 - 5:56:32 PM GMT- IP address: 66.249.88.189

 Document e-signed by Nam Ling (nling@scu.edu)

Signature Date: 2021-06-15 - 5:58:08 PM GMT - Time Source: server- IP address: 75.4.202.62

 Agreement completed.

2021-06-15 - 5:58:08 PM GMT