FlyCast

Autonomous Drone and Broadcast System

Alexander Pattison
School of Film and Animation
Rochester Institute of Technology
1 Lomb Memorial Drive, Rochester, NY, USA
arp7860@rit.edu

Benjamin Zenker
School of Film and Animation
Rochester Institute of Technology
1 Lomb Memorial Drive, Rochester, NY, USA
brz9605@rit.edu

***Abstract*—An out-of-the-box solution to dynamic broadcasting without the need of a multi-person crew (cameraman, boom operator, etc). The drone intelligently works alongside a single broadcast reporter and accurately frames, shoots, and broadcast a live signal to a variety of media platforms all without a drone pilot.**

Due to various obstacles we've encountered over our research, FlyCast is still very much a proof of concept and may require a few more years for the hardware, software, legal, and public acceptance of drones before the system could replace a professional broadcast crew for everyday reporting.

I. Introduction

With media platforms such as Twitter and Facebook, instant media coverage is an everyday occurrence. Today's news outbreaks aren't first on the six o'clock news, but rather thousands of on site photographs and 160 character excerpts. Technology today such as Ustream and Periscope make live broadcasting easier and more accessible than ever. Recent events such as the Baltimore Riots and Nepal Earthquake were first captured and streamed by amateurs surveying the field and conducting interviews from their iPhone's. Crowdsourcing broadcast journalism is an important and ever growing field; cost of entry is cheap and anyone with a modern phone can do it. Yet the professional touch of a reporter and trained broadcast crew is the best way to accurately present a story as it is unfolding. Unfortunately, in some instances it is not until hours later that professional crews arrived to broadcast on site. The best solution is to either get more professional and dynamic broadcast equipment to the masses or make the modern professional broadcast journalist more portable than ever. Combining this idea with the emerging field of "Aerojournalism", our senior project attempts to fill this gap. With the creation of a hardware and software solution, our drone autonomously frames and records broadcast quality video. This not only adds the ability to capture dynamic scenes never before possible with traditional broadcast and video equipment, but also cuts the size of a modern broadcast crew to one individual.

Fig. 1. Visual representation of FlyCast workflow

The basic workflow of our end product contains a broadcaster using his or her backpack or hardcase to easily transport the FlyCast System to the scene of the news. Upon arrival he or she can easily setup the system and get to work, reporting and broadcasting the news, while FlyCast does the rest. FlyCast itself is made up of three components. The 3DR Solo Quadcopter with gimbal, GoPro, and mounted screen for live monitoring. A microphone with attached audio transmitter and mounted Android device to send commands to the Solo. Last, a base station consisting of the the Solo controller acting as a video receiver, and an audio receiver. The two signals are synced and broadcast live via an ElGato streaming box with accompanying desktop application.

Our custom built Android app uses and builds upon many preexisting features made accessible by 3DR's open source software development kit (SDK). Via the smartphone app, the broadcaster is able to automatically launch the drone to hover at eye level, and a predetermined distance away from the reporter. They then have the ability to select various shooting modes such as single shot, double "interview" shot, group shot, landscape survey shot, and static movement mode. In single, double, and group interview mode the drone will be locked a fixed distance away from the reporter. The varying distance will affect the camera field of view and therefore the framing of the shot. Static mode locks the drone wherever it is positioned, much like a virtual tripod. Finally, survey shot will allow the drone to elevate 25, 50, and even 150 feet in the air to survey the scene below for a set amount of time. After surveying, it will safely return to the broadcaster and continue reporting. Additionally, micro adjustments will be available for the broadcaster to slightly adjust framing. While a "freeze"

button will always be visible on the app to lock the drone in place safely above people's heads to prevent a collision when imminent.

Fig. 2. The home screen of the FlyCast application

II. PREPARATION

A. Budget

The team's goal over the summer was to acquire funds of $2,500 in order to build a working prototype. $2,500 is a sufficient amount as it is the price of all current required parts, funds for spare equipment, and a 10% buffer if we go over our parts only budget of about $2,250. This is a state approved educational project that will be tax exempt and therefore NY state and local tax is not included. Our final budget and equipment list can be found in the Appendix. Despite a few workflow and equipment changes we came in under budget by about $300, with the entire project costing $2,146.

1) Applying to Grants

Our primary source of financing has been from the Chris A. Mondiek Student Support Fund, offered to Motion Picture Science students for use in their senior projects. We applied and received this award on November 23, 2015, for a value of $500.

2) Indiegogo Campaign

A secondary funding source was planned to come from a future IndieGoGo campaign. Our team began the process of creating a campaign in the Fall of 2015, but decided to cancel our plans based on the variability in our project direction. Some of the tasks we completed include creating application wireframe and project timeline graphics. As long as our product is still a proof-of-concept, an Indiegogo campaign is not viable.

Fig. 3. FlyCast logo created for the Indiegogo campaign

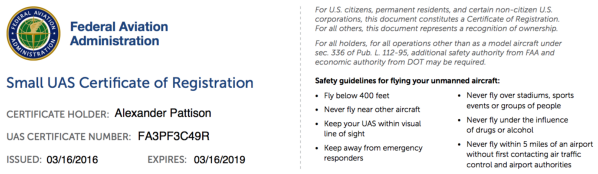
3) Educational Pricing

In order to accurately test the features provided by 3DR, we needed the native camera gimbal. This purchase was made using funds from our micro grant fund. Around the time of this purchase, 3DR announced an Educational Program that allows students researchers to purchase their products at reduced costs. We were the first participants in this program, and utilized the discount for our purchase of the gimbal.

B. Legal

Quadcopter technology is advancing at a pace that government and law cannot keep up with. Only a few years ago sub $1,000 drones would have been a fantasy. When we started our research in the Spring of 2015 it was the wild west in regard to the legal use of these devices. Our initial research led us to believe that the current New York State laws and Federal Aviation Administration (FAA) regulations in place, state that a controlled flying vehicle cannot operate outdoors within five miles from any airport and cannot fly above 400 feet. While these restrictions can be avoided by testing farther off campus (due to RIT being within five miles from an airport), a bit more research proved these regulations to not be legally binding. FAA document Advisory Circular 91-57A states that "Flying model aircraft solely for hobby or recreational reasons does not require FAA approval. However, hobbyists are advised to operate their aircraft in accordance with the agency's model aircraft guidelines." Within this document there is a provision that approves flight to happen within five miles of an airport, as long as Air-Traffic control has been notified beforehand. The Rochester International Airport Air-Traffic Control Center confirmed these provisions and was always notified a few hours before we performed tests within their airspace.

Halfway through our year-long research project the FAA initiated a mandatory "Small UAS (Unmanned Aerial System) Registration" in which all owner of UAS's, defined as a remote controlled aerial vehicle weighing between 0.55 and 55 pounds, must register their personal information as well as their vehicles information in a soon to be public database. Ethics aside, failure to register can result in in a $250,000 fine and up to three years in prison. Given these potential hefty charges, we registered our 3DR Solo with the FAA for a fee of $5.


Federal Aviation Administration

Small UAS Certificate of Registration

CERTIFICATE HOLDER: Alexander Pattison

UAS CERTIFICATE NUMBER: FA3PF3C49R

ISSUED: 03/16/2016 EXPIRES: 03/16/2019

For U.S. citizens, permanent residents, and certain non-citizen U.S. corporations, this document constitutes a Certificate of Registration. For all others, this document represents a recognition of ownership.

For all holders, for all operations other than as a model aircraft under sec. 336 of Pub. L. 112-95, additional safety authority from FAA and economic authority from DOT may be required.

Safety guidelines for flying your unmanned aircraft:

- Fly below 400 feet
- Never fly near other aircraft
- Keep your UAS within visual line of sight
- Keep away from emergency responders
- Never fly over stadiums, sports events or groups of people
- Never fly under the influence of drugs or alcohol
- Never fly within 5 miles of an airport without first contacting air traffic control and airport authorities

Fig. 4. FlyCast's UAS Certificate of Registration

III. HARDWARE RESEARCH AND DEVELOPMENT

A. First Flight Tests

After purchasing the 3DR Solo our first test was to get it airborne to asses it's performance in terms of stability, noise, and video quality. We also wanted to get familiar with Solo's system and learn about the drone's general usability. As

advertised by 3DR, the machine was easy to set up. A simple 'first flight' video tutorial explained the controls of the system and how to attach the propellers to the body of the drone. According to 3DR, assembling the drone only takes about a minute. The time required to connect the iPhone app to 3DR's WiFi and the drone to connect to a minimum number of satellites to operate, can take up to five minutes depending on a number of variables. Most notably is flight location and weather condition. Early on we realized that due to today's limitation in drone technology FlyCast would not be able to permanently replace a broadcast cameraman. In addition to needing a fairly large open space to operate, the drone can only acquire a GPS signal on clear days where it has a direct line of sight to the sky.

Flight number one using the 3DR solo, GoPro Hero 3+ Silver (without gimbal), and the native 3DR app being manually controlled went better than expected. After connecting to the app, the drone took just over two minutes to connect to eight satellites (a minimum of six is required for GPS enabled flight). Once airborne it was clear the audio was going to be a much larger issue than anticipated. At 15 feet away the sound of the hovering drone can only be described as an angry swarm of bees. The aggressive sound instills a feeling of uneasiness or danger, and was an issue that needed to be addressed when constructing FlyCast's audio components.

Fig. 5. Flight test one

From a video and stability perspective the drone performed worse than expected. The drone seemed to glide effortlessly in the air, but that did not directly translate into smooth video. The lack of a gimbal caused a vigorous shaking and 'jello' effect on screen. Interestingly enough the camera shake was always present, while the jello effect seemed to impact random portions of the video. Further research of GoPro's camera system and other reports [1] of similar issues by GoPro and 3DR users led us to believe that the issue was caused by how the GoPro Hero 3+ automatically calculates and compensates for correct exposure. The Hero has a fixed f/# and ISO, therefore the only other variable that can be internally altered for proper exposure is shutter speed. In direct sunlight the GoPro needs a very high shutter speed to cut down on light. This high shutter speed coupled with a vigorously shaking drone resulted in a very noticeable rolling shutter artifact.

Fig. 6. Flight test two

Test flight number two attempted to reduce the rolling shutter artifact and smooth out the footage as much as possible. As users of the GoPro Hero 3+ do not have control over internal exposure settings other methods to needed to be considered. First a lens hood was attached to the GoPro and to further cut down on light the flight took place during sunset. The video quality was drastically improved. Although the shaking was still present there was no jello effect whatsoever due to the lower shutter speed. Another costlier option to cut down on light to achieve the same result, was to use GoPro specific ND filters sold online. Finally, to cut down on camera shake the footage was run through Adobe Premiere's Warp Stabilizer which resulted in much smoother video. The one drawback with Warp Stabilizer is that the footage requires post processing and therefore would not be live broadcasted. Our later experiments show that post processing is not necessary with the addition of a gimbal which reduces almost all camera shake.

B. 3DR Weight Test

Our second experiment was a test of the 3DR's weight capacity in relation to its battery life. The Solo claims about 20 minutes of flight time with "average use" and a GoPro attached, but states that time is decreased dramatically as weight (typically in the form of a third party accessories or gimbal) is added on [2]. A harness was constructed out of dental floss and fishing weights to hold varying payloads evenly distributed across the drone. The total flight time was recorded at varying half pound weight values once the 3DR app indicated that there was 10% or less battery life. Additionally, in a tangential experiment, a four ounce weight was added to the front of the drone to test its stabilization abilities. This experiment aimed to give us insight as to how heavy our add-on components can be, as well as where they need to be placed, in relation to how much battery life we wish to achieve.

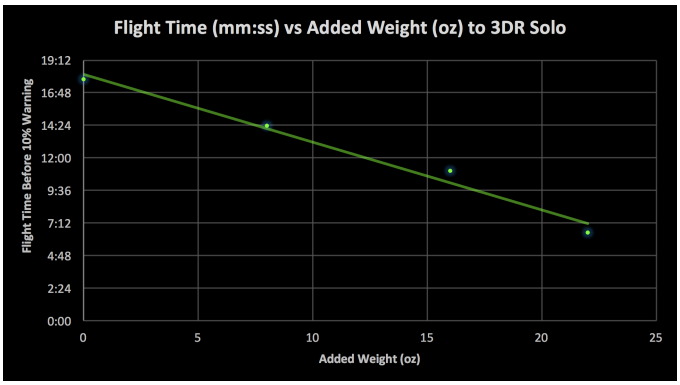


Fig. 7. Results of the 3DR weight test

In order to reduce the amount of variables affecting our data all flights were performed on the same clear day with no wind in a large grass field. The drone was set to take off and hover at the set distance of 10 feet off the ground and hold its position using a GPS lock. The timer started the moment the drone left the ground and stopped once the 3DR app indicated that the drone only had 10% battery left, at which point the app urges the pilot to land immediately.

Our first flight confirmed 3DR's claims as we achieved 17 minutes and 49 seconds of flight time while using 90% of the batteries power. Using the described method above we added two ounces of fishing weight to the front and rear of the drone, totaling one half pound of added weight. For each iteration one half pound was added and the total flight time recorded. Four flights took place until a clear trend emerged showing that the flight time to added weight relationship seemed to be linear. Additionally, the tangential experiment in which the four ounce weights were only placed on the nose of the drone showed that the internal gyroscope compensated for the added weight and continued to fly normally. Given this information and the usability of FlyCast, we made sure our final configuration of add-ons to the 3DR solo will not total more than one pound.

C. *Reference Monitor Mount*

The purpose of a small monitor mounted directly on the nose of the drone is so that a broadcaster can be aware of his or her surroundings as well as make any micro adjustments to framing, via our app, during a live broadcast. Initial plans were to use a small three inch LCD monitor typically used in car back up camera systems. The mini HDMI out port on the GoPro was originally planned to view a live output of what the camera is recording. Unfortunately, this monitor implementation had to be abandoned as the 3DR gimbal utilized this port to stream the signal to the controller and 3DR app. An attempt to use a splitter cable resulted in the video stream only going to one source.

An alternative method, later tested and implemented, was to use the 3DR Solo app solely for its live streaming capabilities. After multiple tests we realized that third party apps such as our own custom app, or an additional open source 3DR controller app called Tower, could control the drone while at the same time the video signal could be streamed to another device running 3DR's app. Therefore, an iPhone 5S running the 3DR Solo app was mounted to the nose of the drone and live streaming footage from the GoPro via WiFi. At the same time the broadcaster is sending control commands from another device running our custom app. Flight tests proved this to be effective live stream method with latency as low as 0.1 second. One drawback to this method is that the image on the live view monitor is not a mirror image, but flipped along the vertical y-axis. We looked for Android setting and software tweaks to fix this but were unsuccessful.

Fig. 8. Reference monitor and mounting system

D. *Gimbal Test*

After purchasing the gimbal, we performed a test flight to experience the added stability it would provide to our video stream. Even on a windy day with light precipitation, the drone managed to keep a level camera regardless of how the drone floated and moved through the air.

We performed a second gimbal test two weeks later which allowed us to begin exploiting the full usability of both the 3DR Solo app and the 3DR Tower app. We tested simple rotations and lateral movements, watching how well the drone was able to keep our mock-reporter in frame. These preliminary experiments provided inspiration for our GPS accuracy test, which will be described in a later section.

At the conclusion of our second gimbal test, our drone executed its auto-return-home function, which happens anytime the drone battery goes below 10%. This function caused our drone to crash full-speed into a large pine-tree, and caused a setback in our testing workflow. We were then tasked with getting new blades and testing each propeller motor, making sure that all hardware components still functioned properly.

E. *GPS Accuracy and Comparison Tests*

The aim of this test was to judge the accuracy of the GPS signal transmitted between an android device and the 3DR Solo controller. The mock-reporter would walk varying routes (figure 9) with the drone either in a "lead" or "follow" mode in which the drone would attempt to maintain a constant ten foot distance between itself and the 'reporter'.

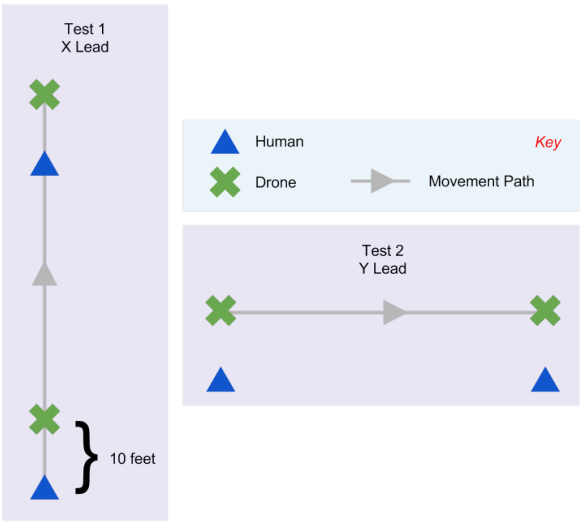


Fig. 9. GPS test paths

1) *GPS Comparison*

The two devices in question were a Google Nexus 7 Tablet (Version 2) and a Samsung Galaxy S5. Online reports have led us to believe that certain Android devices have lackluster GPS capabilities therefore resulting in potentially poor tracking [3]. By comparing two different devices, made by separate manufactures, we aimed to distinguish any difference between them. Each device ran through test one (figure 9) while in autonomous "follow me" mode to note its responsiveness and accuracy.

To our surprise, the Galaxy S5 performed much worse than anticipated. From a usability standpoint the device was barely able to run the Tower application used for testing. The app was sluggish and often unresponsive, sometimes taking up to five minutes to initiate any autonomous flight modes. Due to this factor alone, we decided to move forward with solely using the Nexus 7 for testing which experienced none of these issues.

2) *GPS Accuracy*

Once the Google Nexus 7 was chosen, the team moved forward with the remaining tests. Test one was performed in both a "lead" mode in which the reporter walked towards the drone, as the drone would continuously fly backwards at the same rate as the broadcaster. As well as "follow me" mode, where the 'reporter' walks backwards with the drone moving towards him.

3DR claims six feet of accuracy when using its autonomous flight features, yet we did not find this to be the case. Both flight modes would move the drone in bursts, and not in a smooth motion. If the reporter moved eight feet back, the drone would not maintain a constant ten foot distance but would suddenly move after a set amount of time. "Lead" mode seemed to be the preferred mode as it had a considerably lower latency as compared to "follow me". The former would reposition the drone every second or two, while the latter would only reposition the drone every four to five seconds. While this choppy motion to the drone is not ideal the onboard gimbal makes the apparent motion much less noticeable.

3DR also claims that while the accuracy of the GPS is only six feet [4], that when using autonomous flight features the subject will always be in frame. Later review of the footage proved this to be correct, as rotational capabilities of the gimbal aid in keeping to target in frame. While not a typical medium shot used in broadcast today, this unique angle could have many creative implications for broadcast in the future.

At the beginning of test two, our drone unexpectedly experienced its third major crash. While the tests performed have the drone ten feet from the subject and ten feet from the ground these settings need to be manually entered into our test application whose initial settings are 60 feet in the air and 40 feet from the subject. The drone initiated its autonomous mode before these changes could be applied resulting a 60 foot fall after hitting a nearby tree. The crash caused major damage to the exterior shell and the drones external compass housed in one of the legs. The 3DR Solo was completely disassembled and each component was individually tested before being put back together (figure 10).

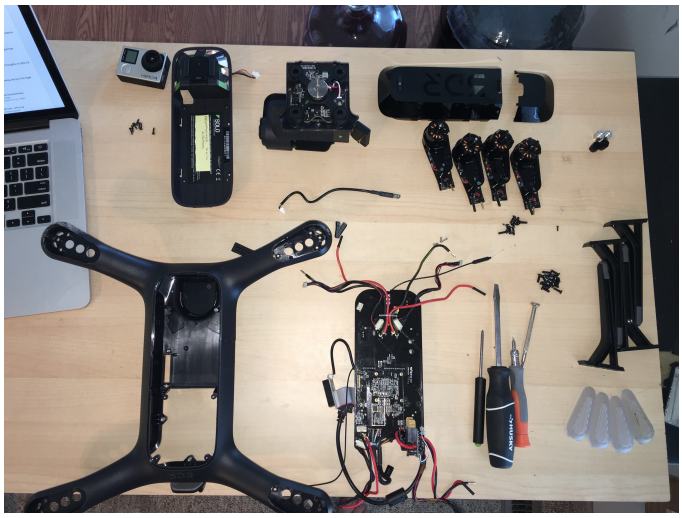


Fig. 10. Deconstructed 3DR Solo

F. *Audio Quality Test*

In order to cut down on noise caused by the quadcopter, a Takstar SGC-598 directional microphone was chosen to be used by the broadcaster. An audio quality test was required to determine a minimum distance the quadcopter should be from the reporter while not affecting audio quality and video framing.

Indoor and outdoor audio tests were conducted and multiple audio sources were measured. The onboard GoPro audio hum averaged at -3dB, while there were no plans to use this audio it was a good baseline to see how much we were able to reduce the unwanted hum from the drone's propellers. The directional microphone aimed at the subject's mouth, as opposed to an omnidirectional microphone used by conventional broadcasters, did a great job of reducing background noise. Without the subject speaking and the drone ten feet away, the audio levels coming from the microphone average at -22dB. Surprisingly the distance between the broadcaster and the drone did not have as much of an impact on hum loudness as expected. During our tests the distance between the drone and the subject ranged from 5 to 15 feet and the dB level only fluctuate +/- 3dB. In order to further reduce

the unwanted hum from the final live audio, the stream was sent through ElGatos “hum removal”, which resulted in our final hum level at -35dB.

G. 3D Printed Microphone Unit

FlyCast’s microphone unit contains three components that need to be held by the reporter during broadcast. A directional microphone, similar in size to microphones used by today's broadcast reporter. An audio transmitter, used to wirelessly transmit the audio signal to the base station where it will be synced up with the the recorded video. And lastly, an android device which is used to send commands to the drone while also acting as a GPS locator to keep the broadcaster in frame. All three objects need to be easily connected and easy to carry, therefore a custom 3D printed handle was constructed. All three objects were measured and drawn first on paper (figure 11) and later in Google SketchUp. Then a housing was digitally built to connect all three pieces and to appear to be a single unit. The goal was for the handle to appear as a large microphone with a horizontally positioned phone / tablet covering the broadcasters hand that would be holding the handle. The broadcaster is then able to hold, and speak into, the microphone with one hand while using the application with the other. Multiple iterations of the handle were constructed and printed using ABS plastic before a final design was ultimately chosen (Figure 12).

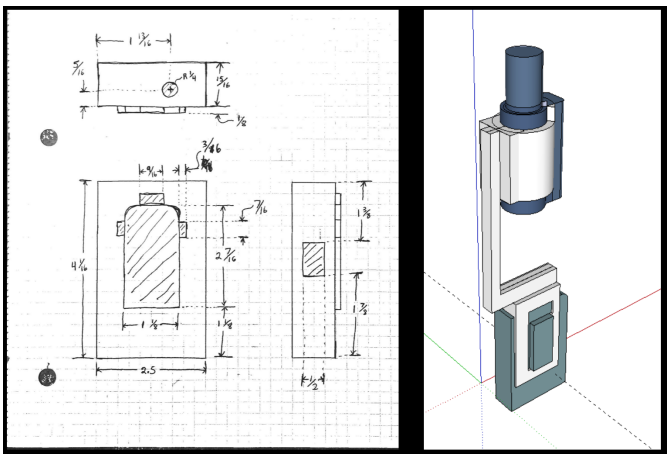


Fig. 11. Microphone unit version I

Fig. 12. Microphone unit version II

IV. Software Research and Development

A. HelloDrone Test

We began our software journey by exploring the resources already provided to us through 3DR’s open source packages. To begin building with their API, a suggested tutorial is provided, detailing the instructions to build a “HelloDrone” application [5]. HelloDrone allows for basic arm/disarm, take-off/land, and built-in flight modes to be executed with a stripped down user interface, as shown in Figure 13. We were not able to successfully implement the HelloDrone application through the provided tutorial. The tutorial was poorly written and had many outdated code snippets, which led to buggy software and hard-to-solve runtime errors. Instead, we download the open source code of the completed HelloDrone application from GitHub. Once downloaded, we were able to compile the app and launch it to our android device.

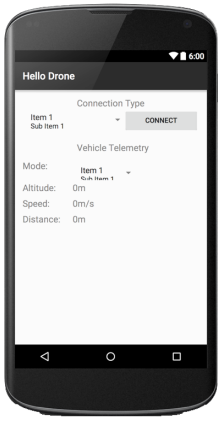


Fig. 13. HelloDrone application

The final step was to run the app and test the capabilities that it provided to the drone. We attempted this multiple times without any results, until one day it finally worked. Our theory is that if the drone’s satellite signal was too weak, the drone would not allow for any incoming connections, ie. the app. Once working, this milestone verified that we had direct ability to manage drone functionality within the Android development environment, and deploy it to an android device.

B. DroneKit-Android Research

The next step was to begin implementing our own functionality into the application. To do this, we started by investigating how the current ‘flight modes’ operated. This was a difficult task, as the implementation ran very deep, and each module was highly de-coupled. Through our own search, and through responses found in online forums [6], we realized that each flight mode was being executed from the Ardupilot API, a unique 3rd-party platform. This presented us with a dead-end, since we found no ways to implement our own flight modes using the DroneKit-Android API [7].

C. Tower Research

A new discovery was made by our team for a strategy that allows software developers to create new flight modes, as long as those systems are developed through 3DRs Tower app [8]. This was the holy grail that we had spent so long looking for. The way 3DRs software platform is constructed: Tower is

simply a GUI that allows actions to be executed through the 3DR Services app, which must always be running on the smartphone that is controlling the drone. 3DR Services will convert any commands into functions that the drone itself will understand, and also integrates the items utilized from Ardupilot. This discovery made us realize that originally, we had not found the highest level interface for working with the 3DR drone. Further testing confirmed that the Tower App, which is also open source, would give us the flexibility we needed with enough pre-built functionality to be as efficient as possible.

D. Development Environment

Android Studio was used as the development environment for building our final android application. As well built as Android Studio is, there were a few issues that caused interruptions in our software construction workflow. During the month of April, Android released a new version of Android Studio that organizes project resources differently and uses different compilers. This caused a major disruption to our software, and we had to restart our build from scratch because none of Tower's resources functioned properly with the new version of Android Studio. Another issue with Android Studio is that when you deploy your software to a physical Android device for testing, you can only have one test app at a time. This was an issue when we were comparing the differences between the native Tower-Dev, and our own FlyCast-Dev. Every time we wanted to deploy one of these to our device, the other would be deleted. Lastly, when we were developing a new graphical user interface, we were not able to use an emulator that matched our physical tablet. Every attempt to do so would lead us to a prompt to update our version of Android Studio, but we couldn't do that since it would destroy our software modules. In the end of the day, the learning curve of Android studio was lower than expected and it had enough power to do what we needed to create our custom application.

E. Final Application

The current release of our custom Android application is built atop 3DRs Tower application. In order to take an iterative software design approach while testing often, building and deploying to our physical drone would not suffice. 3DR offers a beautiful solution to this problem with a tool called DroneKit-SITL (Software-in-the-Loop) that creates a virtual instance of a drone in a local terminal. This virtual drone can then be controlled with the help of a tool called MAVProxy, which is a text-based Ground Control Station (GCS) that can send commands to both DroneKit-SITL and a real drone (when connected)(Figure 14,15).

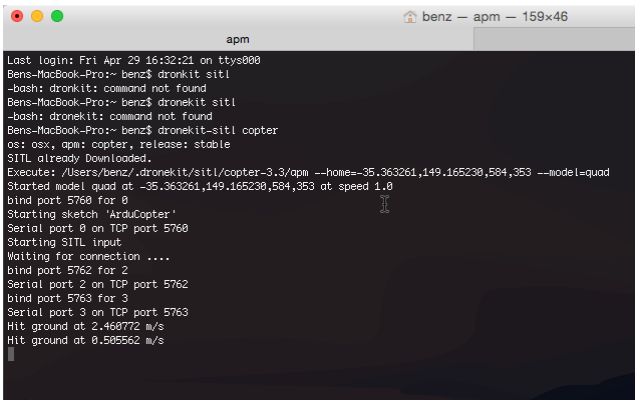


Fig. 14. DroneKit-SITL virtual drone

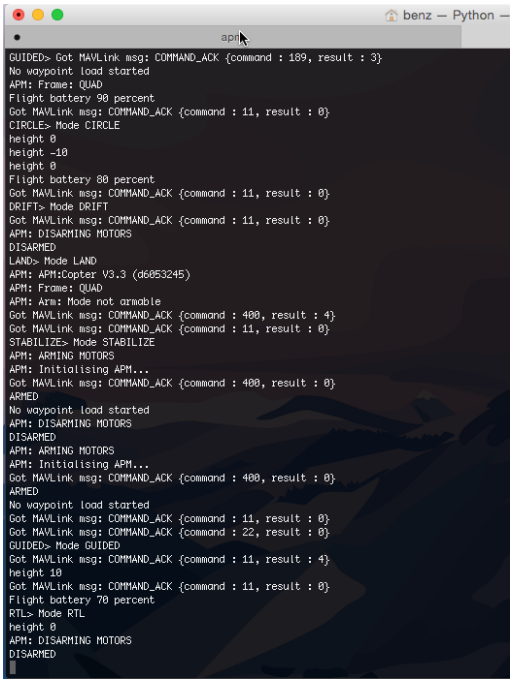


Fig. 15. MAVProxy ground control station

In order to confirm that the best way to frame our app was to build it atop Tower, we needed to make sure that we could take a development version of Tower (Tower-Dev) and control a drone with it. In the beginning, each test presented new problems, and we were unsure if a Tower-Dev would ever be able to control our drone or DroneKit-SITL. But enough persistence, research, and tinkering proved successful - we were able to control both our physical drone and DroneKit-SITL through Tower-Dev.

Our Android application takes the basic functions accessed by Tower and re-skins/re-purposes them for our custom use case. The base Tower app put a large amount of control in the hands of the user/pilot, but this is not ideal for our workflow. We adjusted the functionality of the app to connect with the drone and take-off in fewer steps and fewer 'taps', and provided the user (broadcaster) with very few additional controls aside from the broadcast modes. Our final broadcast modes include: Single, Double, Group, Survey, and 3D Tripod.

Single mode acts in place of Tower's "Follow Me" mode but puts the drone at a very close proximity to the user. Double and Group mimic this behavior but at further distances respectively, from the user. Survey mode takes the drone to a much further distance and height from the broadcaster, while moving in a circular path around the broadcaster. Lastly, 3D Tripod is equivalent to the Air-Brake. Being able to pause the drone at any moment gives the broadcaster the ability to place the camera anywhere in X, Y, and Z space. All of our final source code can be viewed at the following GitHub repository: [9].

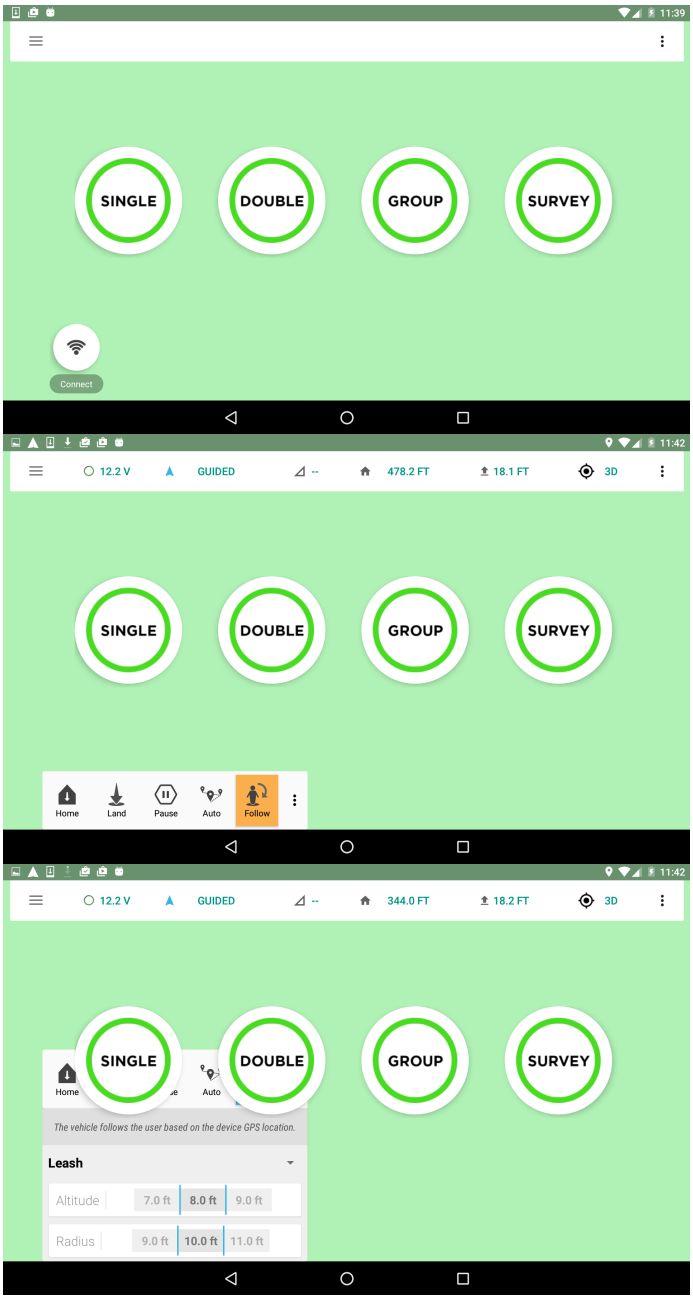


Fig. 16. FlyCast application screenshots

F. *Data Workflow Diagram*

Due to changes in our monitor mount as well as other live streaming concerns we decided to rework our data flow diagram (DFD) multiple times. One of the first changes we made was to send the live video feed from the 3DR Solo to an additional smartphone in place of the monitor. This needed to be done because the GoPro can only output video data to one path at a time. We had not anticipated this and then had to change the workflow. Besides the change in the monitor, the most notable alteration between these diagrams is the addition of a "base station" (figure 17). Due to hardware and software limitations: our original idea of simultaneously controlling the drone and live streaming the footage to YouTube, all from our custom built app, was not feasible. Coupled with a better understanding of the Solo's hardware and data transfer capabilities, the addition of a base station greatly simplified FlyCast's live streaming workflow. The HDMI out port on the 3DR controller, that would otherwise be unused by a FlyCast operator, can be easily hooked up to a monitor for a live video feed. With the use of an El Gato conversion box and software, the feed can be ingested into a laptop on site and live stream the footage directly to YouTube Live and Twitch. Online reports claim that this workflow is achievable, and further experiments have been conducted to confirm this. Finally, slight changes to the audio workflow were made as we were unable to mount an audio receiver directly to the Solo. Testing showed that when mounted the receiver would interfere with the Solo's internal compasses resulting with a "Magnetic Interference" error that would prevent the drone from taking off as a safety feature. Our solution was to add the audio receiver to the base station where it is synced up with the live video stream via the ElGato streaming box (figure 18).

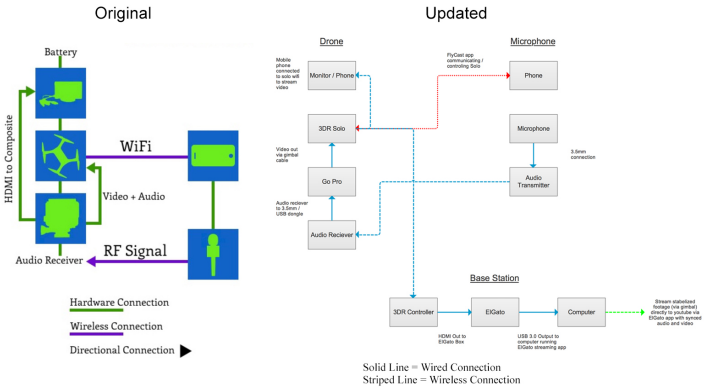


Fig. 17. Data workflow diagram version I and II

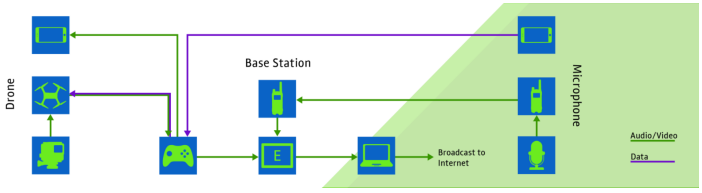


Fig. 18. Data workflow diagram version III

V. CONCLUSION

After 15 months of research and development, we constructed an autonomous aerial workflow, consisting of three connected components that work together to broadcast HD news directly to video platforms YouTube Live and Twitch. The ability to have an autonomous drone mounted camera that can move in three dimensional space adds a unique element never before seen in the broadcast field. Yet, due to various obstacles we've encountered over our research, FlyCast is still very much a proof of concept and may require a few more years for the hardware, software, legal, and public acceptance of drones before the system could replace a professional broadcast crew for everyday reporting. While currently not a full replacement to a professional broadcast workflow, this very portable and sub $2,500 solution is viable alternative to individuals or small groups who are currently reporting breaking news from their mobile devices.

APPENDIX

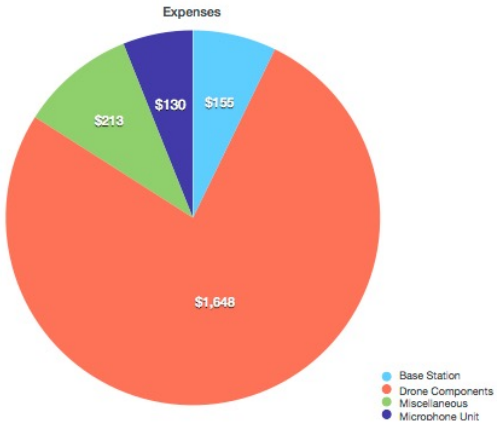
Budget / Equipment List

DRONE COMPONENTS	DESCRIPTION	
3DR Solo	Quadcopter with remote control, one battery, and six propellors	$1,000
3DR Gimbal	Gimbal by 3DR that integrates GoPro control with quadcopter and its SDK. Also automatically stabilizes footage	$360
GoPro Hero 3+ Silver Edition	Main camera that will be used to capture and store media (already owned)	$0
3DR Solo Battery	Extra battery for increased flight / test time	$150
Propellors	Extra propellors required after multiple crashes (2 pack @ $30)	$120
iPhone 5S	Used as mounted reference monitor (already owned)	$0
Mounting System	Various mounts and adhesives to connect iPhone to 3DR	$10
Mini USB to 3.5mm (female)	Cable used to feed audio into GoPro (Not used in final product)	$8
TOTAL		$1,648

BASE STATION	DESCRIPTION	
ElGato Game Capture HD	Streaming device used to sync and steam live video feed from drone to the internet	$149
Laptop with Internet Connection	Used with ElGatos software to steam feed (already owned)	$0
Mini HDMI to HDMI Cable	Cable required to connect the 3DR controller with the ElGato	$6
TOTAL		$155

MICROPHONE UNIT	DESCRIPTION	
Taster SGC Shotgun Microphone	Directional microphone held by broadcaster. Directional in order to not pick up noise from quadcopter (already owned)	$0
Nexus 7 Tablet	Android tablet to run custom app to control broadcast drone (Already Owned)	$0
Transmitter (need info)		$110
Batteries	Multiple sets of batteries to be used by various audio components	$10
3D Printed Microphone Unit	Used to house phone/tablet, microphone, and audio transmitter to be held by broadcaster	$10
TOTAL		$130

MISCELLANEOUS	DESCRIPTION	
Pelican 1560 Hard Case	Case used to carry and protect various FlyCast components	$169
FAA Drone Registration	Legally mandated drone registry	$5
Stickers	Used in promotional video for final presentation	$6
T-Shirts	Custom T-shirts for final presentation	$33
TOTAL		$213



PROJECT TOTAL	
Drone Components	$1,648
Base Station	$155
Microphone Unit	$130
Miscellaneous	$213
TOTAL	$2,146
RESEARCH GRANT	$500
TOTAL AFTER RESEARCH GRANT	$1,646
TOTAL PER PERSON	$823

ACKNOWLEDGMENT

We would like to thank Ricardo Figueroa (rrfppr@rit.edu), our project advisor, for all of the great advice he have given us over the last 15 months. Without his guidance and optimistic outlook we would have never been able to accomplish our goals.

Additionally, we would like to thank Dr. David Long, the Motion Picture Science Program, and the RIT School of Film and Animation for allowing us to conduct this research and make FlyCast become a reality.

This project was made successful via funding through the Chris A. Mondiek Student Support Fund and our parents with whom we pass along our sincerest gratitude.

REFERENCES

1. http://www.goprofanatics.com/forum/gopro-hd-hero3-plus/5876-aperture-control-go-pro-3-a.html
2. https://3dr.com/solo-gopro-drone-specs/
3. https://communityhealthmaps.nlm.nih.gov/2014/07/07/how-accurate-is-the-gps-on-my-smart-phone-part-2/
4. https://3dr.com/kb/solo-gps/
5. http://android.dronekit.io/first_app.html
6. https://discuss.dronekit.io/t/creating-new-vehicle-modes/96
7. http://android.dronekit.io/javadoc/
8. https://github.com/DroidPlanner/Tower/wiki/Work-with-DroneKit-Android
9. https://github.com/benz2012/FlyCast