A USDOT NATIONAL
UNIVERSITY TRANSPORTATION CENTER

Bridge Avoidance in River-based Drone Autonomy

Mahadev Satyanarayanan
ORCID: 0000-0002-2187-2049
Carnegie Mellon University

FINAL REPORT

July 25, 2023

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated in the interest of information exchange. The report is funded, partially or entirely, under [grant number 69A3551747111 from the U.S. Department of Transportation's University Transportation Centers Program. The U.S. Government assumes no liability for the contents or use thereof.

1 Introduction

Rivers have served as transportation corridors for goods and people since the dawn of civilization. Today, they offer promise as natural safe corridors for autonomous drone flights. By using rivers as corridors, the flight paths of large, load-bearing drones over heavily populated urban areas can be kept to a minimum, thus improving public safety. Rivers interconnect communities and could become an ideal proving ground for drone transportation. We aim for fully autonomous flight: i.e., pre-programmed flight without a remote human pilot, including mission-specific actions in response to runtime observations.

A major challenge to autonomous drone flight over rivers are the numerous bridges. These large metallic structures distort wireless signals, and lead to unreliable GPS-based navigation. There is no human pilot to use his or her acute vision and intelligence to guide the drone. How does an autonomous drone avoid these obstacles reliably and safely?

For task autonomy, greater intelligence correlates with more powerful on-board computing and richer sensing. However, drones can only carry a limited payload. This is unlike an autonomous road vehicle that can easily carry LiDAR, multiple video cameras, and all the necessary compute capability. The weight of these sensing and computing entities reduces the working payload. By reducing weight to a minimum, we can optimize the working payload and hence the economic viability of drone-based transportation.

The key to overcoming on-board sensing and processing limitations is to offload intensive processing to ground-based computing infrastructure over a wireless link. In this research, we explore the use of an ultra-light drone for task autonomy. Our goal is to develop the software and algorithms needed for task autonomy on these lightweight platforms and perfect them, before advancing to heavier drones with substantial payload lift. From the viewpoint of public safety, small and light-weight drones are much more attractive than large drones. In case of catastrophic failure, the kinetic energy of a small drone is much less than that of larger and heavier drones. Inital experimentation with ultra-light drones to develop the software for navigation and obstacle avoidance is thus a prudent approach. In this report, we document the results of our efforts towards the use of vision-based navigation for bridge avoidance on an ultra-light drone platform.

2 Flight Platform

The commercial-off-the-shelf (COTS) drone used in our experiments is the Parrot Anafi (320 g) shown in Figure 1. This drone is progammable via an Android API called Parrot Ground SDK. The API gives full flight control and access to on-board sensors. The only unmet requirement is cellular connectivity. We provide this missing capability by physically attaching a COTS device that supports both Wi-Fi and 4G/5G as payload to the drone. In its simplest form, this device would merely serve as a network relay. A more capable device could also perform some bidirectional on-board processing.

Smartphones are the cheapest and most widely-used COTS devices that support both Wi-Fi and 4G/5G. Unfortunately, even the lightest available Android smartphone (Jelly Pro) weighs 61 g. This is well above the 50 g limit that we have experimentally determined to be the safe payload limit of the Parrot ANAFI drone. Above this limit, the drone's flight

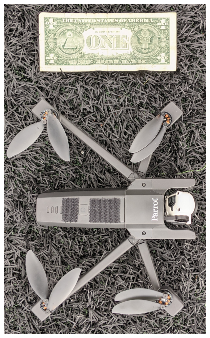


Figure 1: Drone

Figure 2: Watch

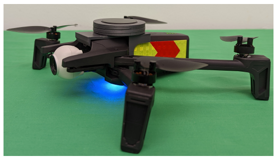


Figure 3: Drone with Watch Payload

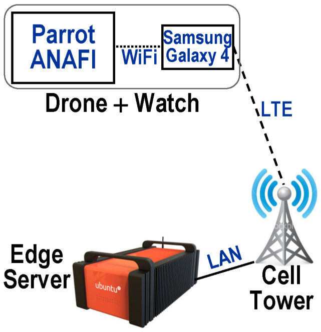


Figure 4: Processing Pipeline

characteristics degrade significantly. Smartwatches are the only other Android devices that support both Wi-Fi and 4G/5G. We selected the Samsung Galaxy Watch 4 (Figure 2) since it weighs only 26 g, and is able to run Android apps using the Parrot Ground SDK.

The drone's Ground SDK API enables an external Android device to control flight and imaging over Wi-Fi. This external Android device is usually a ground-based smartphone controller operated by a human pilot. In our system, the external Android device is the smartwatch that flies with the drone. It is physically attached to the drone using a custom 3D-printed harness weighing 14 g (Figure 3). On this platform, an Android app communicates with the drone over WiFi, and controls it using the Ground SDK API. It also offloads compute-intensive tasks to a ground-based ES over 4G LTE, as shown in Figure 4.

3 End-to-End Pipeline Characterization

3.1 Drone Video Stream

The agility and accuracy of our active vision algorithms are critically dependent on the attributes of the video stream received by the ES. Video from the Parrot Anafi consists of a 720p UDP RTSP stream at 30 FPS [2]. This RTSP stream is produced by hardware on the drone; neither its resolution nor its frame rate are configurable. Although RTSP is an Internet standard, the Anafi does not use standard keyframe-based stream encoding. Instead, it uses a proprietary slice encoding and intra-refresh scheme that disperses keyframe slices across multiple packet transmissions. The motivation for this non-standard design choice is that it reduces the visual impact of packet loss arising from UDP-based RTSP transmission. A negative consequence of this encoding is that software decoding of the proprietary stream format is necessary before individual frames can be selectively dropped to lower frame rate. Such throttling is necessary because of LTE transmission limitations of the watch, as discussed in the next section.

3.2 Thermally-Constrained LTE Transmission

The watch is an austere computing environment with a dual-core 1.18 GHz ARM Cortex-A55 processor, 1.5 GB RAM, and 16 GB flash. Figure 5 compares its attributes to two

	Samsung Galaxy Watch	Unihertz Jelly Pro	Google Pixel 4a
Weight	26 g	61 g	143 g
CPU cores	2	4	8
CPU speed	1.18 GHz	1.45 GHz	2.2 GHz
Memory	1.5 GB	3 GB	6 GB

Figure 5: Austerity of Mobile Hardware

other contemporary mobile devices. The Google Pixel 4a is a widely-used smartphone, while the Unihertz Jelly Pro is the lightest available COTS Android smartphone. We have experimentally determined that their 143 g and 61 g weights are too high to be safe payloads for the ANAFI drone. In contrast, our payload weight of 40 g is light enough for sustained flight without any adverse effects.

As wearable hardware, the watch has stringent thermal protection to shut itself down if its temperature approaches human limits. Both computing and network transmission cause watch temperature to rise significantly. The experiments discussed below show that LTE transmission is the thermal bottleneck.

	Samsung Galaxy Watch	Unihertz Jelly Pro	Google Pixel 4a
Pass-thru @ 30 FPS	68 s	N/A	N/A
Decode and Send @ 2 FPS	240 s	360 s	N/A
Decode and Send @ 0.7 FPS	N/A	N/A	N/A

Input is 720p RTSP video stream at 30 FPS
"N/A" → configuration was unaffected by thermal issues
"Thermal anomaly" → shutdown or severe clock slowdown

Figure 6: Time to Thermal Anomalies

Figure 6 shows the thermal impact of our processing pipeline. As the first row indicates, just receiving the RTSP stream over WiFi and retransmitting it via LTE causes thermal shutdown in 68 seconds. This points to LTE as the source of thermal issues. The bottleneck can be alleviated by slowing down LTE transmission, as shown by the bottom two rows of Figure 6. At 2 FPS, the thermal effects are delayed until 240 s. At 0.7 FPS, the watch is able to sustain transmission indefinitely with no serious thermal effects. Reducing frame rate imposes considerable CPU load, since the input RTSP video stream has to be decoded in software before individual frames can be reconstructed and dropped. Pass-thru, in contrast, involves virtually no CPU load. Yet, thermal shutdown occurs much faster (68 s) with pass-thru. Since WiFi reception and decoding are constant at 30 FPS in all these cases, LTE transmission is strongly implicated in the thermal bottleneck. The situation is inverted with Jelly Pro. Pass-thru involves no thermal issues, but decoding the RTSP stream in order to drop the frame rate to 2 FPS causes thermal anomalies after 360 s. The Pixel 4a is totally unaffected by these thermal issues.

To better understand the watch's thermal sensitivity to LTE transmission, we created a

Sleep Interval	Payload Size	
	35 KB	100 KB
33 ms	✗	✗
100 ms	✗	✗
500 ms	✗	✗
800 ms	✗	✗
1000 ms	✓	✓

✗ thermal shutdown
✓ no thermal shutdown

Figure 7: Effect of Sleeps

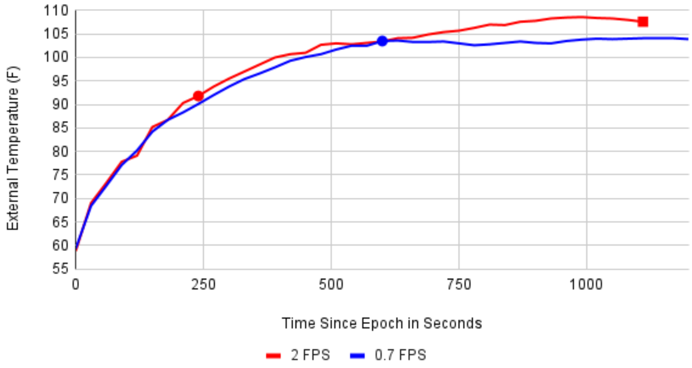


Figure 8: Rise in Watch Temperature

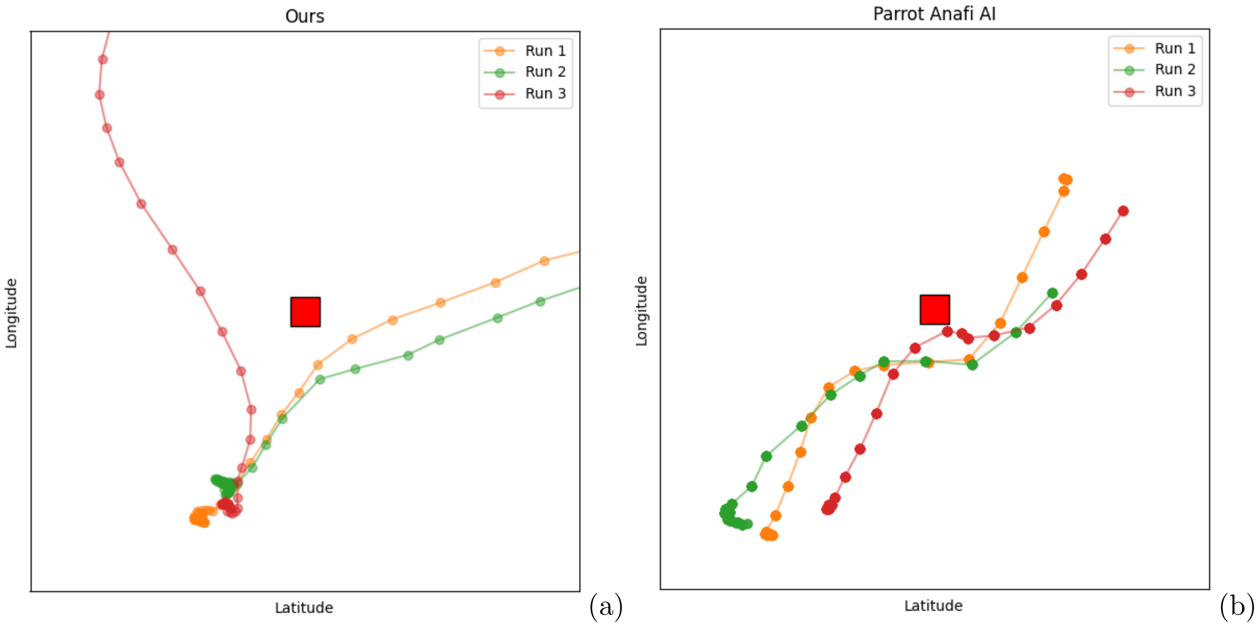
simple stress workload. It consists of a loop in which a fixed-size memory buffer is transmitted via LTE, followed by a sleep of fixed duration. No video stream is involved. Before each run of the experiment, we cool the watch down to 50° F using an ice pack. Figure 7 shows our results for different sleep intervals and two payload sizes that are representative of typical video frames in our experiments. Regardless of payload size, the watch suffers thermal shutdown within a few minutes at sleep intervals below 1000 ms. Above 1000 ms, no thermal shutdown is experienced for the entire flight duration. These results confirm LTE transmission as the thermal bottleneck. With heat produced by decoding factored in, we have found that a frame rate of 0.7 FPS is the highest sustainable.

Figure 8 shows the rise in external temperature of the watch over time under different streaming loads (0.7 FPS and 2 FPS). The circles indicate the points at which the video stream begins to show artifacts, likely from CPU throttling due to heat. The square indicates thermal shutdown. Initially, the watch temperature climbs rapidly at both 0.7 and 2 FPS. Frame degradation begins at around 240 s at 2 FPS and at around 600 s at 0.7 FPS. At 2 FPS, the number of visual artifacts observed is considerable, with several unusable frames. At 0.7 FPS, artifacts are minimal and do not greatly affect frame quality. The 2 FPS stream causes thermal shutdown at 1110 s at a temperature of 108°F (42°C). In contrast, the 0.7 FPS stream plateaus at 104°F (40°C) without ever shutting down.

The extreme austerity and capacity of the watch are thus first-order design considerations in our system. Weight and thermal issues constrain both onboard processing and offloading. Although the ground-based ES can be very powerful, only modest use of its resources can be made while the drone is in flight — at most one 720p frame's worth of image processing per second. This limits the agility of the drone in active vision settings. With improved watch technology or a less thermally constrained offload device, higher LTE transmission rates will become possible, thus improving agility. Our current implementation reflects the constraints of today's COTS hardware.

4 Obstacle Avoidance

Many commercial drones have on-board obstacle avoidance capabilities, typically based on stereoscopic cameras with optical flow algorithms. The Parrot Anafi Ai has such capability, and can move out of the way of obstacles when the drone is moving forwards or backwards.



Our Platform

Anafi Ai

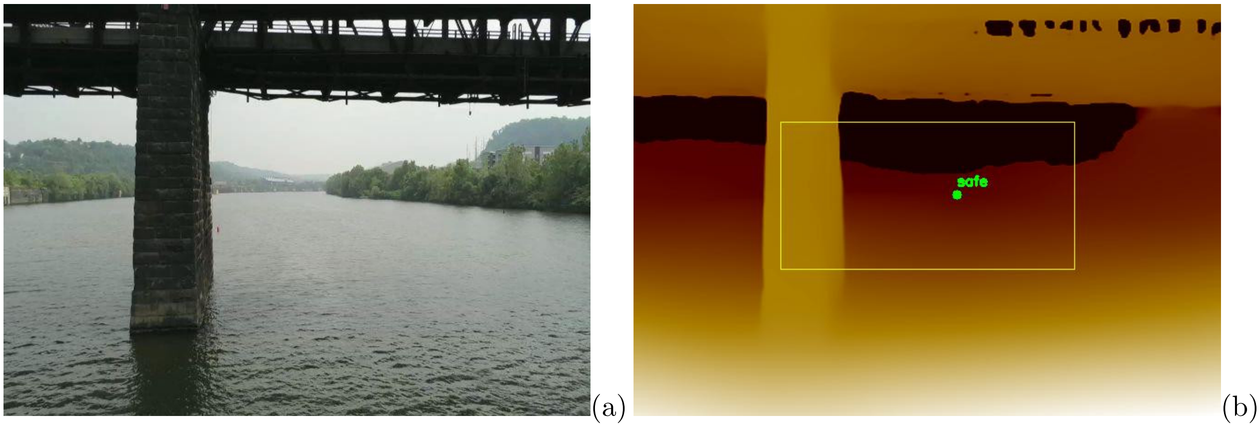
Figure 9: Obstacle Avoidance Experiment Results

Our platform only has a single RGB camera, and must therefore rely upon monocular depth estimation. We use a DNN-based algorithm called MiDaS [1] to provide relative depth estimates. Using MiDaS inference on each frame received by the ES, we construct the inverse relative depth map. Based on the rate of change of relative depth across frames, we identify obstacles in the flight path and actuate away from them.

Our experiments compare our flight platform's monocular obstacle avoidance, using our visual pipeline at 0.7 FPS, with the stereoscopic obstacle avoidance of the Anafi Ai using on-board computing at 30 FPS. We place a 2 m tall by 0.5 m wide foam pillar directly in the drone's path. The drone is instructed to fly at 1 m/s at a fixed altitude of 1 m directly towards the obstacle. We capture a trace of the drone's flight path across 3 different runs.

Figure 9(a) plots the flight path of each run for our platform, along with the position of the pillar. Figure 9(b) plots the flight paths of the Anafi Ai on the same task. Both platforms successfully avoid the obstacle in all cases, but do so using very different tactics. The low frame rate and high end-to-end latency of our pipeline forces our platform to be very conservative, and to give the obstacle a wide berth. Well past the obstacle, the drone has not yet returned to its original flight path. In contrast, the stereoscopic cameras, high frame rate and low end-to-end processing latency of the Anafi Ai together enable it to be much less conservative in obstacle avoidance. The flight paths cluster more tightly around the obstacle, and the drone soon returns to its original flight path.

Using this obstacle avoidance algorithm, we have successfully demonstrated bridge avoidance during autonomous flight by our platform. The specific bridge was the Hot Metal Bridge on the Monongahela river in Pittsburgh. Figure 10(a) shows an input frame from one of our flights, as the drone approaches a pillar on this bridge. Figure 10(b) shows the depth-encoded output of our algorithm on this frame. The green dot labeled "safe" indicates the point towards which the drone actuates to safely avoid the pillar on the left. A live demo of



Raw Input Frame of Hot Metal Bridge

Output of Our Algorithm

Figure 10: Bridge Obstacle Avoidance

this capability was demonstrated to a number of observers (including Stan Caldwell, Executive Director of Traffic21 Institute) on June 6, 2023. The demo worked flawlessly, with the drone returning safely to the takeoff location.

5 Conclusion

Our research has demonstrated that safe vision-based navigation by autonomous drones using edge computing is feasible on an ultra-light drone. Our results show that bridges, which are the main obstacles in drone corridors over rivers, can be safely navigated using this approach. By devoting minimal payload to on-board computing, our approach frees up substantial headroom for working payload. Our work demonstrates that by using rivers as corridors, the flight paths of large, load-bearing drones over heavily populated urban areas can be kept to a minimum, thus improving public safety.

References

[1] Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., and Koltun, V. Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-Shot Cross-Dataset Transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence 44*, 3 (2022), 1623–1637.

[2] Schulzrinne, H., Rao, A., Lanphier, R., Westerlund, M., and Stiemerling, M. Real-Time Streaming Protocol Version 2.0. Tech. Rep. RFC 7826, Internet Engineering Task Force (IETF), December 2016.