# Autonomous Quadrotor Landing on a Moving Platform

Stan Brown & Chris Choi

## 1 Layout for Report

- Introduction
- Related work
- Hardware
- Software libraries used
- Methods
  - camera methods (image processing)
  - PID controllers
- Results and Discussion
  - Image Processing Results
  - PID Results and Testing
  - Hardware difficulties?
- Future work
- Conclusion

## 2 Introduction

The idea of quadrotor has been explored by several companies and researchers since quadrotors became a popular alternative to fixed wing and single rotor aircraft (DJI auto landing, Pixhawk, make a reference). However most of these methods usually assume that the landing zone is static and that there is very little wind or disturbances. In cases of high wind or a rapidly moving platform, which from the quadrotor's reference frame can be be though of as nearly equivalent cases, normal autonomous landing procedures will likely fail.

Over the years there has been been at least some interest in the problem of landing a quadrotor on a moving platform and there have been at least 5 academic publications on the topic [1, 2, 3, 4, 5, 6]. The problem has been explored from a fairly wide range of approaches.

Possible applications for this work include autonomous landing on a boat in high wind conditions, landing on a charging platform and more.

The major challenge with autonomous precision landing is the difficulty of obtaining a reliable estimate of the landing target relative to the quadrotor, the landing target can often go out of the camera's field of view causing the quadrotor to lose track of the target, other vision based artifacts that include lighting, lens distortion, color, etc.

## 3 Related Work

The challenging of autonomous landing can be categorized into three distinct parts. First, the quadrotor must sense or detect the landing target to obtain positional estimates relative to the quadrotor. Secondly, the quadrotor must plan its rendezvous trajectory with the landing target. Third and finally, the quadrotor must apply a set of controller commands to approach the platform, matching both speed and heading on the approach and land on the moving target.

### 3.1 Perception and State Estimation

For perception existing solutions use a variety of simple to complex techniques. In [2] a basic color threshold technique was to identify the landing target, while [6] used optical flow in images captured on board the quadrotor to obtain necessary relative information for control.

### 3.2 Planning

In the literature we reviewed, rendezvous planning was neglected as it was assumed that quadrotor has a clear view of its landing target. Most authors tend to focus on the detection and estimation of the quadrotor's pose relative to the target and maintaining an accurate estimation throughout the landing procedure which is mostly handled by a series of control loops.

### 3.3 Control

A comparison between a PID and Linear Quadratic Control (LQC) was explored in [4]. However the authors admit, even though LQC performed slightly better than the PID controller the difference was minor and may be attributed to the additional time spent tuning the LQC. In [6] a PID controller was developed to land on a oscillating platform in the vertical direction with no lateral movement. While interesting, the solution took over 1 minute to transition from hovering over the landing pad to landing. Additionally the experiments did not seem to account for pitch or roll of the platform.

## 4 Methodology

## 5 Hardware

### 5.1 Quadrotor

For the hardware, we have chosen to reuse equipment that is readily available to us in the Wave Lab. For the quadrotor we have been using the DJI F450 quadrotor frame, Emax 2213-935KV motors with complimentary 1045R propellers, Thunder Power lipo 11.1V 4200mah battery, DJI E310 420S 20A electronic speed controller and Pixhawk v2.4 as our flight controller.

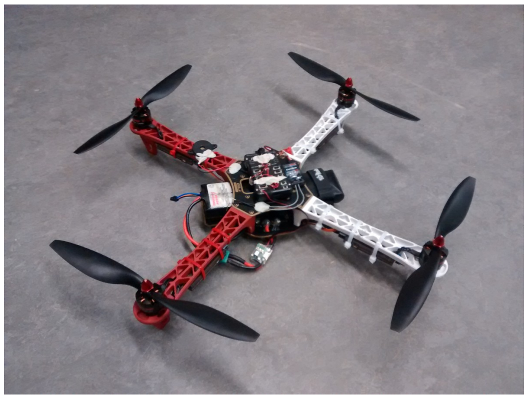


Figure 1: DJI F450 with Pixhawk v1.5

#### 5.1.1 Onboard Autonomous Control System

In terms of onboard autonoumos control, we have decided to use the Odriod XU4 due to its size and processing power available, and a Logitech C270 HD webcam as our vision sensor. One problem we may face very soon is the narrow field of view and relatively low frame rate (25Hz) the logitech webcam has. We will be keeping our options open, and plan on replacing the logitech webcam with a Ximia xiQ USB3 camera with a wide angle lens in the near future.

### 5.2 Vision Capture

Firefly camera

#### 5.2.1 Pixhawk Firmware - PX4

As mentioned in the hardware section, we have had numerous issues with regards to using the software suite for the Pixhawk Flight Controller, as well as the PX4 firmware itself. There are many instances where we could not explain the instability of the quadrotor during flight, many of which lead to costly crashes, but we believe through these experiences we have developed a procedure to lower the risk for such instances from happening often.

#### 5.2.2 Odriod to Pixhawk Communication

For our autonomous system to control the quadrotor directly, the Odriod has to be able to communicate with the Pixhawk, this is where `mavros` (ROS package) comes into play. Mavlink is a popular UAV communication protocol between UAV and ground control softwares, `mavros` creates a mavlink ROS node to enable developers to monitor and issue mavlink commands with ease. We have successfully used mavros to arm and disarm Pixhawk (tested), in the near future we should be able to send velocity/attitude commands to control the quadrotor.

#### 5.2.3 AprilTag

On the landing target from we have successfully calibrated the Logitech webcam and used the Apriltag library to obtain a pose estimate, and have manually verified the estimates are accurate, for distance the error is $\pm$1cm. We have also successfully implemented the ability to identify two different apriltags of different sizes (see Fig **??**), pose estimates from both apriltags are equal.

Next is to workout how to transform the measurements from the pose estimates of the apriltags in relation to the quadrotor's body frame.

### 5.3 Quadrotor Control

In reviewing prior work, a reoccurring problem lies in the difficulty in maintaining an accurate pose estimation between the quadrotor and the landing target throughout the entire manoeuvre. As the quadrotor approaches the landing target, the target often goes beyond the camera's field of view, causing camera based navigation techniques to fail. Furthermore, in certain conditions the tilt angle of the quadrotor could cause difficulty in maintaining a visual on the landing pad for a downward facing or fixed camera.

To address both of these issues, we propose a novel solution utilizing an nested apriltags which we term inception apriltag. We will replicate the experiment performed by [5], where we assume the quadrotor already has a visual of the landing platform. We aim to implement a series of controllers for the approach and landing of the quadrotor to the moving landing platform.

## 5.4 Measurement and State Estimation

One of the major challenges when using AprilTags on a robot that has limited computational power, such as a quadrotor, is the rate at which the AprilTag library can compute the state. This problem is exacerbated when using larger image sizes as the computational time of the AprilTag software appears to grow at least exponentially as shown in (add a figure here). This problem was also noted in the work of [5] and he addressed the issue by reducing the brightness of the image so that majority of the image is black except for the april tag. An example of this implementation is shown in figure (Kevins image). This allowed Ling [1] to calculate states at a rate of approximatly 10 - 15 fps but it came at the cost of requiring the brightness parameters to be set ahead of time and also removed a lot of the robustification that is provided in the standard AprilTag library.

In this work, the slow rate of state estimation was addressed using a novel method where the image size and calibration parameters are set depending on the current distance between the quadrotors position relative to the landing pad.

## 5.5 Adaptive Image Preprocessing

As discussed previously, the low rate of the AprilTag library when processing images of 640 by 480 pixels results in an update rate of approximately 3 to 5 fps on a Snapdragon 8 core processor, which is to low to be used effectively in a PID control loop. Building on the work of Ling, who noted that higher update rates were possible using AprilTags when the majority of the image is filled with black pixels, a set of image preprocessing methods were developed to maximize the update rate of the AprilTag library.

### 5.5.1 Adaptive Image Windowing

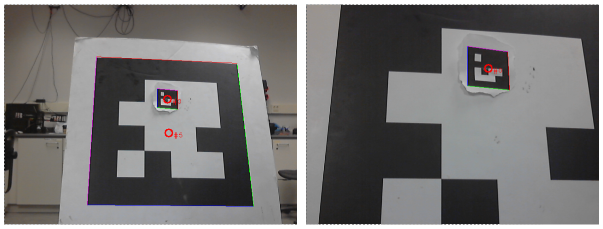
If one assumes that the quadrotor does not move to fast, there is relativity low rotation between image captures, and that the image update rate is quite fast (60 fps in this implementation), then the location the AprilTag in the following image can be estimated based off of the location it was last observed in the previous image. Therefore whenever an AprilTag is measured in an image, a bounding box around the AprilTag is calculated and then used in the following image to black out the protions of the image where the AprilTag is unlikely to be. An example of this implementation is highlighted in Figure (Adptive windowing).

While the adaptive windowing procedure works well for when the AprilTag observed at a distance, where they do not take up a significant portion of the image, it begins to fail when the quadrotor approach the landing pad as the observed size of AprilTag begins to become larger and larger in the image. This leads to the processing time of the AprilTags to increase as the quadrotor gets closer, which is a major issue as in these cases a higher update rate is actually required. In cases where the AprilTag is not observed in the expected location (bounding box), the size of the bounding box is set to the size of the image and the entire image is processed.

### 5.5.2 AprilTag Inception

In order to be used for landing, the AprilTag attached to the landing pad must be relatively larger so that it can be detected from a distance of at least 15 meters which from our experience means the AprilTag must be at least 50 cm by 50 cm in size. The large size of the AprilTag becomes a major issue during the final portion of the landing procedure as the AprilTag image will both appear quite large in the captured image which causes to the Adaptive Windowing Procedure outlined above to become ineffective. Furthermore close proximity to the AprilTag also grealy increases the chance that a portion of the AprilTag will be located outside of the captured image, causing no state estimation to be provided at the critical stages of landing.

To address both the decreasing state update rate as a function of proximity and reduce the probablity of losing site of the AprilTag during landing a secondary AprilTag is embedded in the larger AprilTag. This secondary AprilTag is assigned different family id, and is placed in the center of a primary AprilTag. Whenever the secondary AprilTag is captured in the image, the adaptive windowing method is set to track only the secondary AprilTag rather than the larger one, which reduces the portion of the image that must be processed in with each image capture. An example of this procedure is highlighted in Figure 2.



(a) 2 Apriltags Detected (b) 1 Apriltag Detected

Figure 2: Apriltag Inception - To mitigate the FOV problem assocaited with

#### 5.5.3 Adaptive Image Down-sampling

It was also noted that the image could be greatly down sampled as well if the quadrotor approaches the landing pad as not as many pixels are required to calculated the location and state of the AprilTag in the image. Therefore 3 image sampling sizes were also introduced into the code which sub-sample the image to 160 by 140 pixels and 320 by 280 (half and quarter resolution) when the distance between the camera and the AprilTag is less than 1.5 and 3 meters respectively. If the camera is further than 3 meters from the AprilTag the native resolution of 640 by 480 is used. This change only affected the image processing time when the quadrotor is within 3 meters of the quad and has the desired effect of increasing the rate of AprilTag process.

In order to use this down sampling effectively without corrupting the AprilTag estimations, 3 camera calibration files were also computed at the normal, half and quarter camera resolutions. The calibration file passed to the AprilTag library during the state estimation procedure is selected based on what the current camera resolution is set at.

## 6 Results

#### 6.0.4 Detection Delay

$$t_{\text{capture}} = t - \delta t_{\text{camera}} + \delta t_{\text{detection}} \tag{1}$$

## References

[1] D. Lee, T. Ryan, and H. Kim, "Autonomous landing of a vtol uav on a moving platform using image-based visual servoing," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 971–976, May 2012.

[2] J. Kim, Y. Jung, D. Lee, and D. Shim, "Outdoor autonomous landing on a moving platform for quadrotors using an omnidirectional camera," in *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*, pp. 1243–1252, May 2014.

[3] H. Voos and H. Bou-Ammar, "Nonlinear tracking and landing controller for quadrotor aerial robots," in *Control Applications (CCA), 2010 IEEE International Conference on*, pp. 2136–2141, Sept 2010.

[4] J. Friis, E. Nielsen, R. F. Andersen, J. Boending, A. Jochumsen, and A. Friis, "Autonomous landing on a moving platform," *Control Engineering, 8th Semester Project, Aalborg University, Denmark*, 2009.

[5] K. Ling, U. of Waterloo. Department of Mechanical, and M. Engineering, *Precision Landing of a Quadrotor UAV on a Moving Target Using Low-cost Sensors.* University of Waterloo, 2014.

[6] B. Herisse, T. Hamel, R. Mahony, and F.-X. Russotto, "Landing a vtol unmanned aerial vehicle on a moving platform using optical flow," *Robotics, IEEE Transactions on*, vol. 28, pp. 77–89, Feb 2012.