VISION-BASED AUTONOMOUS LANDING 
AND NIGHTTIME TRAJECTORY 
ESTIMATION OF QUADROTORS

A thesis presented
By
Yang Liu
to

The Department of Electrical and Computer Engineering
in partial fulfillment of the requirements
for the degree of

Master of Science
in the field of
Computer Engineering
at
Northeastern University
Boston, Massachusetts

December, 2019
Abstract

The agility and aerial view of drones can bring huge benefits to military, industrial and commercial fields. In recent years, huge improvements have been made in drone navigation technology. However, little effort has been put in drone navigation at night. This thesis is a cornerstone towards autonomous navigation for quadrotors at night. In the first part of the thesis, an vision-based autonomous landing program is implemented. It allows drones to land on a movable charging platform for longer battery life. This process provides a reliable solution with small error. It can also be transferred from grayscale camera to thermal camera in our future research, making long term autonomous task possible under poor light conditions. In the second part of this thesis, an optical flow based trajectory estimation program is implemented on thermal camera. This method can stabilize a drone’s position with more accuracy than GPS can in dark environment.
Acknowledgements

I would like to express my sincere gratitude to my mentor, Professor Hanumant Singh. Thanks you for allowing me to pursue my own research ideas, for providing numerous opportunities for me to develop a wider perspective. Your knowledge, patience and enthusiasm helped me through my hardest time in the project.

To all my teammates from the Field Robotics Lab - Stephen Hagen, Zhiyong Zhang, Vikrant Shah, Pushyami Kaveti and Jagatpreet Singh: I could not have asked for a better group of people to work with. Specially, I would like to thank Mithun Diddi for supporting me throughout drone projects this whole time. If it were not for his hard work setting up the hardwares, I wouldn’t have made this far in my thesis.

Lastly, I would like to thank my family and friends for supporting me emotionally throughout the thesis. In particular to my girlfriend Yi Bu for helping with my thesis writing and cheering me up to overcome all the obstacles in this entire journey.
# TABLE OF CONTENTS

1 Introduction 1

2 Software Environment, Robot Platform and Sensors 2

3 Autonomous Landing 4
   3.1 GPS Navigation 4
   3.2 Apriltag Navigation 5
   3.3 Apriltag target for thermal cameras 7

4 Optical-Flow Based Position Estimation with Thermal Camera 7
   4.1 Thermal Camera Calibration 7
   4.2 Optical Flow on Thermal Camera 8
   4.3 Trajectory Estimation 11

5 Experiment 12
   5.1 Autonomous Landing 12
   5.2 Opticalflow on Thermal Camera 13

6 Conclusion and Future Works 15
LIST OF FIGURES

1  Hardware settings for the drone ........................................ 3
2  The landing platform ...................................................... 4
3  Apriltag used for the landing system ................................. 5
4  The control logic for tag searching and landing ................. 6
5  Special Apriltag target built for nighttime landing .......... 7
6  Thermal checkerboard and image seen by the thermal camera .. 8
7  Undistortion of thermal image .......................................... 8
8  Projection of point P in image plane ................................ 9
9  Optical flow tracking with thermal camera ......................... 11
10 Control command during a landing process ....................... 12
11 Landing trajectory regarding the Apriltags ...................... 12
12 Lidar reading before and after filtering .............................. 13
13 Thermal optical flow at noon ........................................... 14
14 Trajectory estimated by thermal camera at noon ............... 14
15 Thermal image and RGB image at night ............................ 15
16 Trajectory estimated by thermal camera at night ............... 15
1 Introduction

Today’s drone navigation depends heavily on GPS system. When GPS signal is available, drones can finish some tasks autonomously but only within a limited range because of short battery life. However, the accuracy of GPS positioning is not good enough to handle delicate control tasks like precise hovering or landing. Hence, in the first part of this thesis, we implement an vision-based autonomous landing system through which the drone lands onto a wireless charging platform, in the purpose of extending its working time in a single mission. Several kinds of vision-based autonomous landing approaches have been developed in the past decade. Bartk et al. proposed a color and shape based landing system[7], Wenzel et al. presented a landing system based on alphabatic pattern[29], De Croon et al. applied a second-order approximation to the optic flow field to land a micro air vehicle(MAV) onto a slope[9]. These methods are computationally efficient, but tend to fail due to false target detections. We adopt Apriltag as a more reliable landing target. Apriltag is a visual fiducial system invented by Edwin Olson[24][28]. It uses a 2D barcode-like ”tag” which allows a 6-DOF localization of features from a single image. In the later version of apriltag, a small tag can be incorporated in a large one, so that the drone can acquire the big tag from a high altitude, while the small tag provides localization information when the drone comes closer[19]. We have further created a special Apriltag target for thermal camera, so that the landing process can be done even when no visible light is available.

Compound eyes have lower resolution, fixed focus and smaller binocular overlap than monocular eyes[27]. To adapt to the cluttered flying environment, insects rely on image motion, also known as optical flow, with which 3D movement is projected to a 2D image plane[13]. Many bio-inspired algorithms have been invented to calculate optical flow. It can be calculated in dense manners like Frneback method[15] and Horn-Schunck method[18], or in sparse manners by tracking features on images. Optical flow has been used for stabilizing the drone’s position for more than a decade. Honegger et al. presented an open source and open hardware design of an optical
flow sensor for velocity estimation[17]. It is known as PX4Flow, which is a module for PX4 open source autopilot[22]. Fantoni et al. applied optic flow in path planning[14]. Kimberly McGuire et al. implemented optical flow on a 40g pocket drone[21]. T.I. Braber et al. implemented an optical flow based stabilization algorithm for MAV without a range detection sensor[8]. However, most researches focused on scenarios where there is sufficient light for RGB cameras. In the second part of this thesis, we propose an optical flow based trajectory estimation solution using a thermal camera. We implement the optical flow on thermal camera using Lucas-Kanade tracker[20] with Shi-Tomasi feature detector[30]. Using a proper control system with the trajectory estimator, the drone can hover stably at night.

2 Software Environment, Robot Platform and Sensors

Our software runs on Ubuntu 16.04. Ubuntu is a complete Linux operating system, freely available with both community and professional support[3]. We use ROS Kinetic (Robotic Operating System) as our software platform. ROS is a flexible framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms[2]. In our case, ROS is used for sending high level command to the drone and exchanging data between our ground control station (laptops) and the drone. A Wi-Fi router enables wireless communication between our laptops and the drone via SSH. The remote access makes our field tests easier, in that we can send the high level controls and monitor the feedback from the drone directly from our laptops.

A DJI Matrice-100[12] (hereafter referred as M100) quadrotor is used as our test platform. DJI also provides a control software interface called DJI-Onboard-SDK to handle the lower level control for M100[10]. The Onboard-SDK has a ROS interface[11] which takes the velocity control and angular control as input, and con-
verts this high-level control input into throttle control as output.

<table>
<thead>
<tr>
<th>Sensor</th>
<th>Manufacturer</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grayscale camera</td>
<td>Flir Systems</td>
<td>Blackfly S USB3</td>
</tr>
<tr>
<td>Infrared thermal camera</td>
<td>Flir Systems</td>
<td>FLIR ADK</td>
</tr>
<tr>
<td>1D lidar</td>
<td>Garmin</td>
<td>LIDAR-Lite v3</td>
</tr>
</tbody>
</table>

Table 1: Onboard sensors list

![Sensors used in the project](image1.png) ![Sensors on the drone](image2.png)

(a) Sensors used in the project  (b) Sensors on the drone

Figure 1: Hardware settings for the drone

We use Nvidia Jetson TX2 as our onboard computer. TX2 is a fast, power-efficient embedded computing device\[16\]. It has a Nvidia Pascal-family GPU and loaded with 8GB of memory, which is sufficient for the onboard computation. It is connected to the M100 through an Orbitty Carrier. Orbitty Carrier is a product from Connect Tech which is designed to match the Nvidia Jetson TX2/TX1 module form factor. The Orbittys design includes 1x USB 3.0, 1x USB 2.0 OTG, 1x HDMI, 1x GbE, 1x microSD, 2x 3.3V UART, I2C, and 4x GPIO. Ideal for robotics and unmanned applications or any small form factor rugged environment\[1\]. All our onboard sensors are connected to the TX2 and uses ROS as the middleware communication system to exchange data. The control commands are sent to M100 through ROS as well. Besides the onboard IMU and GPS, the main sensors we use include: a downward
grayscale camera[5], a thermal camera[4] and a one-dimensional lidar[25]. Models of the sensors that we use in this project is listed in Table 1. The grayscale camera has a 30° field of view and 1.3-million-pixel resolution, which provides the drone with sufficient image details in the field. The infrared thermal camera provides the drone with a unique view under low light conditions and makes night time navigation possible. The one-dimensional lidar is a cheap and light-weighted solution for height detection. Fig.1 shows the basic settings and customized components of our drone.

The landing platform is fixed on the top of a Humvee jeep, so when the battery goes lower than 30%, the drone can always fly to their home position and land on the charging platform. Fig.2 shows the charging platform used for this project. The wireless charging units are attached to the legs of the drone.

Figure 2: The landing platform

3 Autonomous Landing

3.1 GPS Navigation

The autonomous landing mission happens in an open area, and the drone can head back to the charging platform under GPS waypoint navigation. Although GPS can provide a decent estimation of the drone’s position, it is not precise enough to lead the drone onto the charging platform. In an ideal scenario where GPS signal is highly available, the position error given by GPS can be more than 0.4 meters[23], while in
our experiments, the error can be as large as 1 meter or more. The wireless charging platform is only 1*1 meter in size, so a more accurate navigation system is needed in our case. Hence, we introduce apriltag for the navigation system.

### 3.2 Apriltag Navigation

Apriltag is one of the state-of-the-art solutions for pose estimation among all existing visual fiducial systems. We use apriltag to provide the drone with a better estimation of the landing platform than the GPS does. An apriltag is put at the center of the charging platform so that the drone can follow the tag and land accordingly. We generate a customized Apriltag where a small tag lies inside a big tag so that the tags are visible from multiple altitude. As shown in Fig.3, the drone sees only the large tag when it is more than 8 meters high. On the other hand, the big tag becomes too large to fit into the camera frame when the drone flies closer to the landing platform and only the small tag can be recognized. The two tags make up for each other for a wider navigation range.

![Apriltag seen by the drone](image1)

![A small apriltag inside a big apriltag](image2)

Figure 3: Apriltag used for the landing system

When an Apriltag appears in the camera, a pose estimation is published to TF topic. Each tag has a unique ID, so the information provided by the big tag and the small tag are unique to each other. DJI-Onboard-SDK allows users to send velocity control when GPS is available, so we use the estimated position of the apriltags as
input and drone’s velocity as output for our high-level controller. Tags are fixed at the center of the landing platform, and a PD velocity controller is implemented for horizontal control to keep the drone right above the tag. A proportional controller is used for vertical velocity control of the drone. In other words, the drone goes down fast when it’s far above the landing platform and descends more slowly as it gets closer to the platform. This behavior stabilizes the drone for landing and therefore reduces the error when it actually lands.

In order to make sure that the Apriltag is always in sight, the drone is given commands to go down only when it is inside a cone area of the landing platform. In some rare case if the drone gets blown away and the tag is no longer detected, the drone will ascend 1 meter to search for the tag again. The drone can ascend 12 meters at most for a thorough search at the given GPS point, because the tag detection above 12 meters is not guaranteed. The field of view of the camera that we use is 53, so the searching area at 12m high is 

$$12 \times \tan \frac{53\times\pi}{180\times2} \times 2 \approx 11.9m.$$ 

In our experiments, an 80cm * 80cm Apriltag is printed to be seen within 12 meters, and that the GPS error is almost never larger than 10 meters. Hence, this logic works reliably. The illustration of the control logic is shown in Fig.4.

![Diagram](image_url)

**Figure 4:** The control logic for tag searching and landing
3.3 Apriltag target for thermal cameras

Based on the customized Apriltag we built for RGB camera, we make a special Apriltag with hardboard and copper foil. We use the copper foil to cover the black grids of the Apriltag and leave the rest uncovered, appearing as white grids in the image. Fig.5 shows the special tag we made and how it looks like in an infrared thermal camera under the sunlight.

![Apriltag target in an RGB image](image1.png) ![Apriltag target in a thermal image](image2.png)

(a) Apriltag target in an RGB image (b) Apriltag target in a thermal image

Figure 5: Special Apriltag target built for nighttime landing

This method works fine when the sun heats up the whole target. However the tag is less likely to be recognized when it is cooled down at night because the contrast between adjacent grids decreases. To tackle this issue, we could either introduce an active heat source for the tag, or design the tag with a different material. Future research needs to be done to test out these methods on field.

4 Optical-Flow Based Position Estimation with Thermal Camera

4.1 Thermal Camera Calibration

Just like RGB cameras, a thermal camera also needs calibration. Thermal cameras are insensitive to the even texture on a normal checkerboard, so we need to create a
special checkerboard target for calibration. Fig.6 shows the checkerboard we made for the thermal camera. It uses wood as base material and half of the grids are covered with copper foil. The resulting image appears as black and white grids. Each time before we take an image, we use a heater to heat up the thermal checkerboard in order to make the adjacent grids more distinguishable in the thermal frame. Fig7 shows the undistorted image after calibration.

![Figure 6: Thermal checkerboard and image seen by the thermal camera](image1)

(a) Raw image  
(b) Undistorted image

![Figure 7: Undistortion of thermal image](image2)

4.2 Optical Flow on Thermal Camera

Motion field is the 2D projection of the 3D motion of surfaces in the environment, whereas optical flow is the apparent motion of the brightness patterns. These two
motions are not always the same and, in practice, the goal of 2D motion estimation is application dependent[6]. The optic flow is formulated as a vector field over two dimensions, where the domain is either a two-dimensional visual field or a focal plane, and the range is a vector describing how fast and in which direction the texture is moving. Optic flow is created by the translational and rotational movements of a point P (X, Y, Z) in the camera reference frame. Considering the projection of this point in the image plane p(x, y) (Fig.8), we can define it according to point P and focal length f.

![Figure 8: Projection of point P in image plane](image)

Lucas-Kanade feature tracking method, which assumes that the displacement of features between two image frames is small and approximately constant, is used in this project to track the flow. With this assumption, the optical flow can be written as:

\[
I(x, y, t) = I(x + dx, y + dy, t + dt) \tag{1}
\]

I stands for the intensity at point \((x, y)\) at time \(t\). We can therefore derive that:

\[
\Delta I(x, y)(v_x, v_y) + I_t = 0 \tag{2}
\]

Where \(\Delta I(x, y)\) is the spatial gradient, \((v_x, v_y)\) is the image velocity and \(I_t\) is the temporal derivative of image intensity. By solving Eq.4 based on a weighted least squares minimization of the intensity conservation constraint, we can estimate the optic flow at the given moment by calculating the mean flow of each feature detected in the frame.

\[
OF^t_i = \min_{\mathbf{v}} \sum (x_i, y_i) [\Delta I(x_i, y_i, t)\mathbf{v} + I_t(x_i, y_i, t)]^2 \tag{3}
\]

\[
OF^t = \frac{\sum OF^t_i}{n} \tag{4}
\]
This equation assumes that all features inside the view are static and stable, which is not always the case in application. Let’s denote the corresponding feature from \( t_i \) moment to \( t_{i+1} \) moment as \( P_{i+1} = f_{ik}(P_i) \). This correspondence is valid only when:

\[
f_{ik}(P_i) = P_{i+1} \tag{5}
\]

\[
f_{ik}^{-1}(P_{i+1}) - P_i = <\epsilon \tag{6}
\]

Also, the non-static features should not be used for calculating optical flow, so flow from point \( P_i \) is only used when:

\[
OF_{t_{mean}}^t = \frac{\Sigma OF_{t_i}^t}{n} \tag{7}
\]

\[
OF_{t_{stderr}}^t = \sqrt{\frac{\Sigma (OF_{t_i}^t - OF_{t_{mean}}^t)^2}{n}} \tag{8}
\]

A detected feature is valid if:

\[
|OF_i^t - OF_{t_{mean}}^t| < 1.5 \ast OF_{t_{stderr}}^t \tag{9}
\]

Where 1.5 is the confidence coefficient. The relationship between drone’s movement and optical flow can be expressed as:

\[
\begin{bmatrix}
OF_x \\
OF_y
\end{bmatrix} = T_{OF} + R_{OF} \tag{10}
\]

\[
T_{OF} = \frac{1}{Z} \begin{bmatrix}
-f & 0 & x \\
0 & -f & y
\end{bmatrix} \begin{bmatrix}
v_x \\
v_y \\
v_z
\end{bmatrix} \tag{11}
\]

\[
R_{OF} = \begin{bmatrix}
\frac{xy}{f} & -(f + x^2) & y \\
(f + y^2) & -\frac{xy}{f} & -x
\end{bmatrix} \begin{bmatrix}
\omega_x \\
\omega_y \\
\omega_z
\end{bmatrix} \tag{12}
\]

\( OF \) stands for optical flow; \( v_i \) and \( \omega_i \) are the translation velocity and rotation rate, respectively. We use the data from a gyroscope to compensate the rotation of the drone, so that we can estimate the translational movement of the drone as:

\[
\int T_{OF} dt = \int (OF - R_{OF}) dt \tag{13}
\]
Fig. 9 shows an example of optical flow result in IR image. The green lines are the tracked flows of static features, while the red lines are tracked flows on a pedestrian which has been filtered out.

4.3 Trajectory Estimation

With optical flow at hand, we just need the altitude from the ground to estimate drone’s trajectory. A 1D lidar is used to detect the altitude. Since pitch and roll angles could affect the data coming from the lidar, we need to offset these affect for data to reflect real altitude. Recall Fig.8, The translational movement between two image frames can be calculated as:

\[ d = OF \frac{h}{f} \ast \cos(\theta) \ast \cos(\phi) \] (14)

where \( d \) is the displacement of the drone, \( h \) is the height, \( f \) is the focal length and \( \theta \) and \( \phi \) are pitch and roll angles of the drone, respectively. Finally, data from the compass reflects drone’s heading angle. The trajectory can therefore be calculated as:

\[
\begin{bmatrix}
x \\
y
\end{bmatrix} = d \ast \begin{bmatrix}
\cos(\gamma) \\
\sin(\gamma)
\end{bmatrix}
\] (15)

where \( \gamma \) is the heading angle according to the magnetic north.
5 Experiment

5.1 Autonomous Landing

The autonomous landing program is tested on field for many times and is robust in different light condition. Fig.10 shows an example of the velocity control during the landing process from one of our tests. In the given plot, the landing process was triggered at 10 meters’ height. As shown in Fig.10, the velocity input on z axis reduces as the drone gets closer to the platform. In figure 10, the control in X and Y axis have some overshooting due to a strong $K_p$. In our experiments, we keep this value high so that the drone holds position well even under strong wind condition. Fig.11 is the corresponding trajectory of the drone.

Figure 10: Control command during a landing process

Figure 11: Landing trajectory regarding the Apriltags
5.2 Opticalflow on Thermal Camera

In the tests for thermal opticalflow, the range data from the lidar is not always accurate. The data given by lidar occasionally causes big error because of multi-path error and weak reflection[26]. For example, the detected height can suddenly drop to 0.1 meter or bounce to 50 meters when the drone is actually hovering at 10 meters. To avoid such problems from the control algorithm and trajectory estimation, a zero-order hold reaction is applied to eliminate these error data points. We also implement an exponential moving average on the rest of the data to smooth the height curve. Fig.12 shows the raw and filtered lidar data. Comparing to raw data, the error data points are ignored after the filtering and the curve for height data is smoother. We also notice that the filtered data does not align with the raw data around mid-point. This is because data from the lidar is affected by the pitch and roll angles of the drone and the filtered data avoids these noise.

![Figure 12: Lidar reading before and after filtering](image)

Our testing field is an empty outdoor parking plot with cement pavement that is typical for city environment. Fig.13 shows the view from an RGB camera and thermal camera respectively. With the daytime sunlight and heating, both the thermal image and RGB image have good contrast and distinct textures to track. We manually flew the drone by a square pattern. The trajectory estimated by the thermal camera is shown in Fig.14. In this plot, the red line reflects data from GPS and it is used as ground truth in the test; the blue line is the estimated trajectory given by thermal optical flow. From taking off to landing, the entire trajectory is more than 100 meters
and the overall accumulated error is around 4 meters, which means that the error is less than $\frac{4}{100} = 4\%$.

Figure 13: Thermal optical flow at noon

Figure 14: Trajectory estimated by thermal camera at noon

Our nighttime test was conducted 1 hour after sunset. The acquired images in Fig.15 shows that the RGB camera hardly picked up any signal, while the thermal image still preserved important details. Comparing the result in Fig.13, the contrast in the latter image is lower than the former image taken at noon. It is probably because the ground cools down after sunset and different materials tend to share the same temperature. Although the clarity of the image was decreased, we still managed to retain error within certain range in the trajectory estimation (Fig.16).
6 Conclusion and Future Works

This thesis is part of our lab’s research project towards autonomous drone navigation at night. Here we present an Apriltag-based autonomous landing system on a grayscale camera. Then, we make a special Apriltag for thermal cameras and show that it is detectable in thermal images. Currently, the landing system works reliably in grayscale camera, so our next step will be applying the system on a thermal camera. We also explored the application of optical flow based trajectory estimation on a thermal camera. The movement of the drone was tracked at night with a suitably
small accumulated error. Therefore, we plan to integrate this method into the drone’s control system to stabilize the drone at night. Further research will be done to combine the optical flow stabilization program with the autonomous landing program so that in low light, GPS denied environment, the drone will also be able to search for the landing target and trigger the landing process whenever the landing target is in sight.

References


