CSCI5561 Project: Moving Object Following With An Aerial Robotic Platform And Its Onboard Camera

Dario Canelon, Jiannan ”Eric” Zhang
dario@cs.umn.edu, zhan2001@umn.edu

Abstract— **Object tracking and following is important for many applications. Different methods have been developed to track specific objects based on their unique features. Our objective is to utitilize a flying platform with onboard cameras to detect objects and then do some predetermined action. The actions contemplated are landing on a marked location, landing on a helipad, detecting a desired arbitrary object, and following arbitrary objects. This progress report gives an overview of the progress made towards the aforementioned goals to date.**

I. Introduction

Tracking and following objects is one of the most important topics in computer vision. It requires different technologies to achieve accuracy and robustness. Different methods can be applied according to different applications, from early days when mechanical trackers and magnetic trackers were widely used to recent years that sensors and image-based vision systems are getting more and more powerful because of the growth computers. Among all of this, vision system has a great advantage because it is not invasive and less costly [1]. Based on the surveys [1] [2], tracking methods we can use in our project is classified into 3 categories: Edge-based tracking, template matching, interest-point-based approaches. Most of the methods use some natural features of the specific objects, and they have different computational payloads. In this instantiation we will be testing our tracking system on an the Parrot AR Drone to track objects while in flight. This task can be accomplished job by a combination of object recognition, tracking, and following.

II. Problem

Our story is simple and useful: a helicopter in a battle field wants to find a tank on the ground which is in need of help (maybe it need transportation because a mountain is in front of it, or it is surrounded by enemies and needs air force to save it), then it sends an SOS signal. Immediately, the AR Drone will be aware of this and fly to the object, this requires 3 steps: taking off and find the object, fly towards it while tracking the position of the object, finally stop at a certain point from the object. In our application, we want the AR Drone to stop at the top.

III. Requirements

- Controlled movements. Firstly, the robot must be controlled in a stable manner in order to effectively more towards a target or track an object.
- Object recognition. Once the robot is airborne, a predetermined object or feature is found in order to provide feedback to the control system that handles the navigation.
- Tracking. Once an object or marker is found, it must be tracked in the image while the drone, the object, or both are in movement.
- Following. The tracking phase gives way to the following phase whose main objectives is to navigate the drone towards the target.

IV. Related Work

1) RAPiD tracker [**?**]. Edge-based tracking, low computational load.
2) Lucas-Kanade algorithm for global region recognition.[**?**].
3) Feature detection and matching. SIFT [**?**], SURF[**?**], BRIEF [**?**]

V. State Machine

State machine flow:

1) Controlled takeoff. The drone implements this indepentdently with a function call.
2) Receive gamepad commands. This allows the user to move the drone independently of the control and vision algorithms and also allows for positioning of the drone sufficiently close to the object to be tracked.
3) When a prescribed button on the gamepad is pressed, the robot transitions into control mode.
4) While in the control mode, the robot begins to execute control and vision algorithms to obtain a desired outcome.
5) Once the control mode switch is pressed, the drone reverts back to manual control.
6) Repeat.

VI. Control - Simple helipad

The control loop uses visual features as feedback during navigation. In the current stage, the control waits for two or four points, that describe the endpoints of one or two detected lines, respectively. These points are used to calculate the navigation inputs to the drone.

1) If two points are given (P_1, P_2), go to step 2, if four points are given go to step 7
2) Angular and linear velocity inputs are determined.
3) First, the drone is aligned so that the line is perpendicular, with in some threshold, with the bottom edge of the image.

4) Then the drone is aligned so that the center of the line is within some distance from the center of the image.
5) Loop until threshold requirements are met and proceed to step 6
6) Advance along the vertical axis of the image plane by some amount and go to step 1
7) Because four points are given we know that we have found the two lines that mark the landing location, therefore we align the drone along the x and y axis.
8) Once we are within some threshold we land the rotorcraft.
9) End.

While the above describes the control algorithm for landing on two perpendicular lines, it only serves a temporary purpose as the real objective is to land on the helipad. The following sections describe the vision algorithms for detecting the blue lines, used as the simple helipad, and the detection of the helipad itself.

VII. Color Tracking

For an easier but intuitive application, we chose to use the AR Drones bottom camera to help the drone park by itself. We have crossed blue lines on the ground, the AR Drone starts from a point further from the crossed point. It first detect the straight lines and find the center (average) of the blue pixels, it provides feed back to the AR Drone control system which moves the AR Drone in real-time.

Algorithm and processing flow:

1) Receive image from AR Drones bottom camera
2) Do smoothing (Gaussian 3*3)
3) Find light blue pixels (pixels with color values in a threshold with respect to R/G/B)
4) Calculate average of the X-coordinates of all pixels
5) Calculate deviation of X-coordinates of all pixels
6) If result of 5 is larger than a threshold, go to 1, otherwise, go to step 7
7) Move the AR Drone according to the difference between average and image center (X)
8) If the deviation is below the threshold go to step 9
9) Publish the points message for the control algorithm
10) End.

VIII. Helipad detection

One simple application of our project is to let the AR Drone detect its helipad in real-time. There are two challenges for this job: one is the detection in an arbitrary noisy environment, the other is the real-time processing/ controlling.

Among the edges, we are interested in the couples of ellipses (the helipad will be detected as ellipses due to perspective projection) which have following characteristics:

1) They are all closed.
2) One is totally inside the other.
3) dx1dx2, dy1dy2, as shown in figure.

Algorithm:

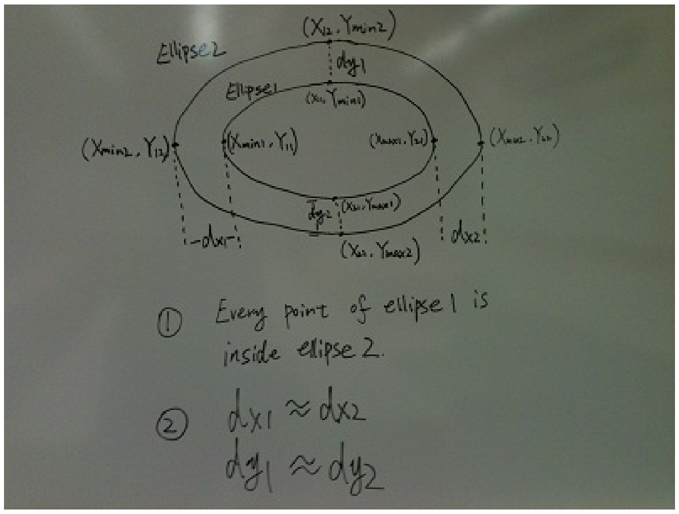

Fig. 1: Ellipse detection illustration

1) For all lines from Canny edges, store it in a 2D linked list L (in edge tech blog)
2) Discard all lines which are too short to process (set threshold less than 10 pixels is good in practice)
3) Discard all lines which are not closed (set threshold 2-4 pixels is good in proctice)
4) Find every pair of lines L1 and L2 from L, find four points:
    a) If L1 is inside L2 or L2 is inside L1, go to 5
    b) Otherwise, find another pair, go back to 4
5) If distance ($|X_{min1} - X_{min2}| + |X_{max2} - X_{max1}| < threshold1$) and ($|Y_{min1} - Y_{min2}| + |Y_{max1} - Y_{max2}| < threshold2$), they are the ellipses we need.
6) End.

Detection flow (in a loop):

1) Receive AR Drone front camera images
2) Do smoothing (Gaussian 5*5), edge detecting (canny, threshold 100, 50)
3) Edge thinning
4) Edge following
5) Lines processing (discard useless lines)
6) Find helipad center from lines using the algorithm described above
7) Go back to 1.

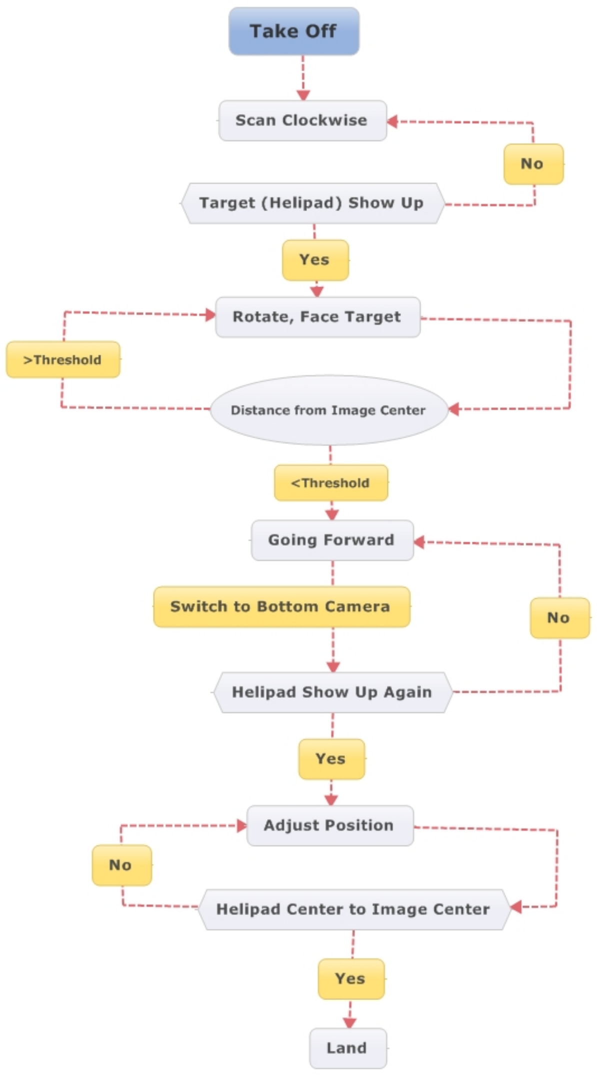

Fig. 2: Ellipse detection illustration

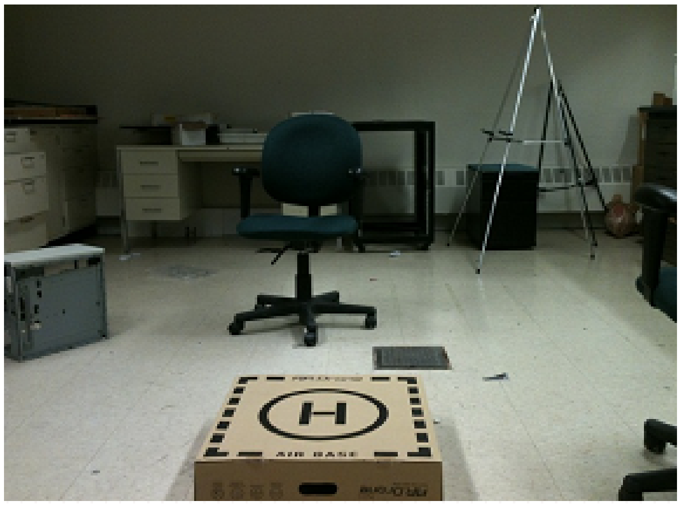

Fig. 3: Helipad image 1

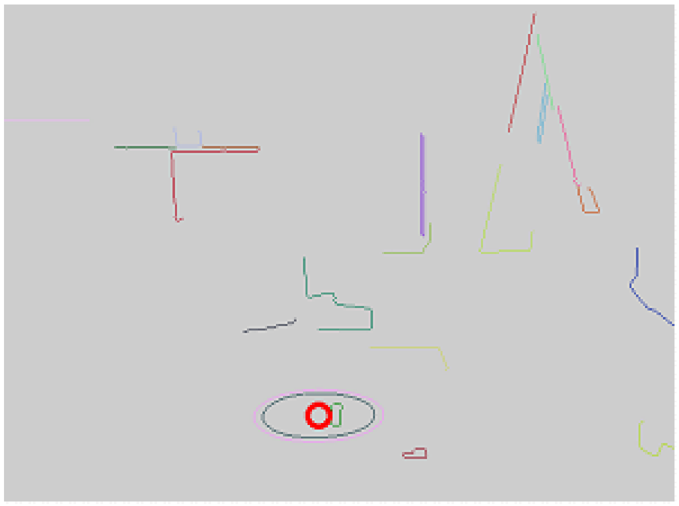

Fig. 4: Helipad image 1 with ellipse detection

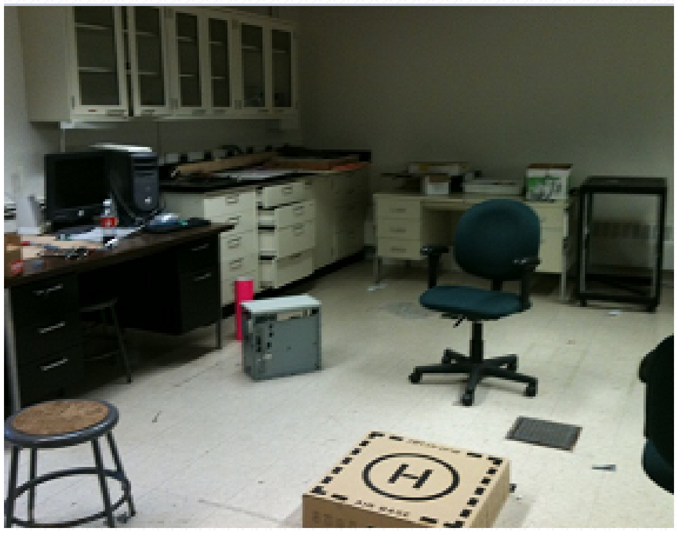

Fig. 5: Helipad image 2

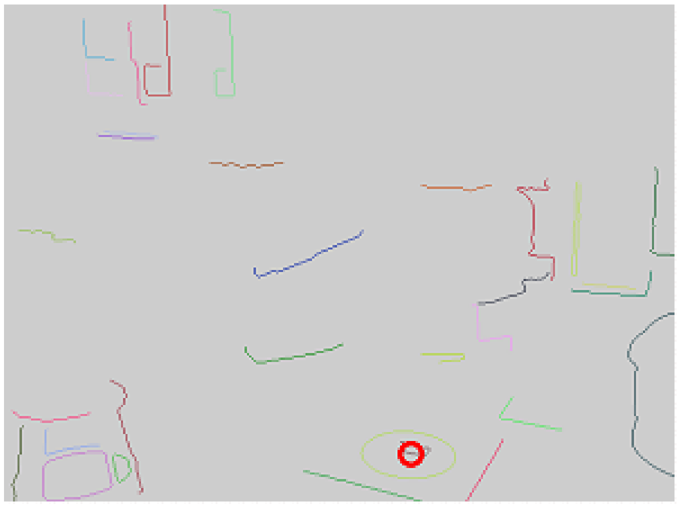

Fig. 6: Helipad image 2 with ellipse detection