Team 5 - QDMC - Land on the Bullseye Using 1 late day

Vishnu Sashank Dorbala
University of Maryland
vdorbala@umd.edu

Tim Kurtiak
University of Maryland
tkurtiak@umd.edu

Ilya Semenov
University of Maryland
isemenov@umd.edu

Surabhi Verma
University of Maryland
sverma96@umd.edu

***Abstract*—This report presents the implementation and results of a computer vision landing with the Parrot Bebop drone on a concentric circle bullseye. We present methods for identifying projected ellipses and estimating camera position based off of the projection of the circle.**

I. Problem Statement

In this project, we are given an estimate and covariance of coordinates where a circular bullseyse exists. Given these inputs, the drone must takeoff, identify the circular bullseye in its camera, and land at the center of the bullseye. It is recommended but not required to use homography to estimate pose relative to the marker in order to track to the center of the bullseye. The radius of the bullseye circles are known.

II. Ellipse Identification

Finding ellipses in an image of the bullseye tag proved to be more complicated that using a built in function. $Houghcircles()$ returns inaccurate radius information, which is necessary for pose estimation, and $fitellipse()$ simply fits an ellipse around some points. If a good mask is aquired and the contours are a full closed ellipse, then these methods may work. An example of such a situation is shown in figures 1 and 2.

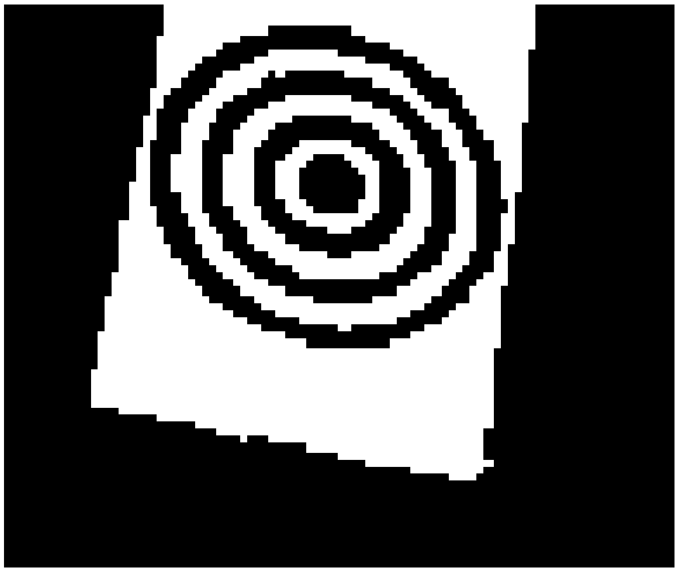

Figure 1. A good mask taken from a distance

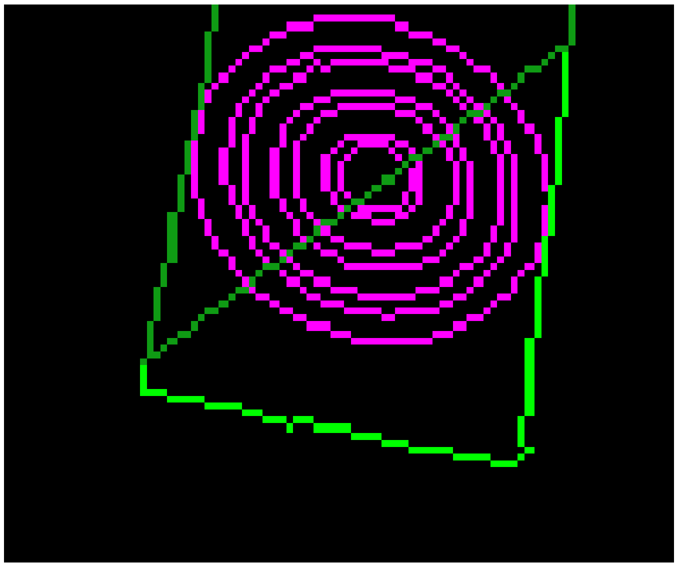

Figure 2. Contours that work well with $fitellipse()$ are shown in pink here

However, due to glare the user is quite likely to need to process an image such as figure 3.

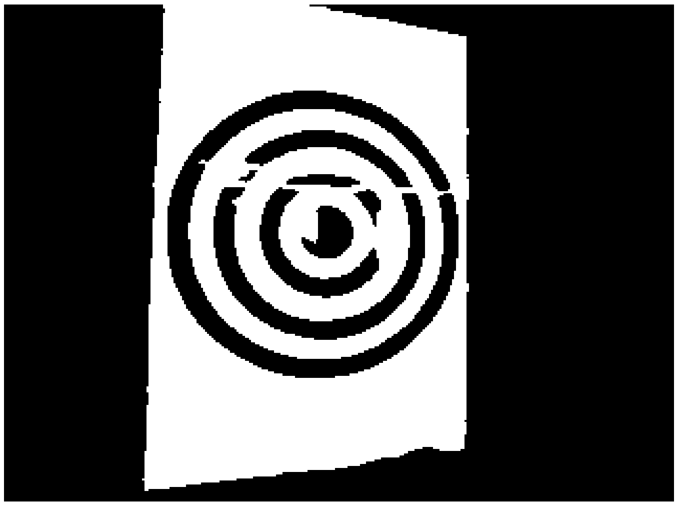

Figure 3. A mask that arises when glare is present

In this case the contours do not for full ellipses, instead they cut through the rings and create ”sausage” shapes. $Fitellipse()$ with this contour would not follow the curvature, but rather try to fit all the points in an ellipse. Refer to figure 4 for a depiction of these problematic contours.

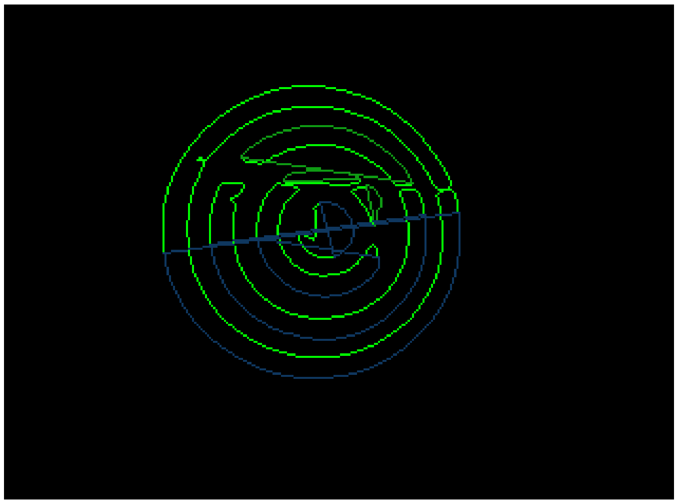

Figure 4. Contours that arise when glare is present are shown in green, selective treatment is overlaid in dark blue

Instead a custom implementation is used based on [2]. Using a least squares approach on the error between an ellipse and a collection of points allows one to fit an ellipse to a collection of points describing a curve.

An general conic section is described as $ax^2 + bxy + cy^2 + dx + ey + f = 0$. Then let $a = [a, b, c, d, e, f, g]^T$ and $x = [x^2, xy, y^2, x, y, 1]^T$ with x_i corresponding to input point i of N.

The ellipse constraint $a^T Ca$ is:

$$a^T \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} = 1 \tag{1}$$

Define the design matrix $D = [x_1, x_2, ..., x_n]^T$ and the least squares problem resolves into the system of equations:

$$\begin{aligned} 2D^T Da - 2\lambda Ca = 0 \\ a^T Ca = 1 \end{aligned} \tag{2}$$

Where λ is the Lagrange multiplier. An eigen analysis finds eigenvector u_i which can be scaled by a μ_i such that:

$$\mu_i = \sqrt{\frac{1}{\mu_i^T C \mu_i}} \tag{3}$$

Which gives a solution $a_i = \mu_i u_i$. The solution with the lowest residual is taken, which is proven to offer a unique solution as per [2].

This will fit an ellipse to the curvature formed by some points in an arc, rather than around these points. However, this method is not appropriate for direct implementation on the sausage shapes due to the existence of two edges that define an ellipse, and additional noise on the ends.

Instead a segmentation method is used to find curves that serve well for ellipse fitting. See above in figure 4 the dark blue contour represents a closed shape formed by an open section of another contour. If the contour centroid sits above or below the center of the marker it is split top-bottom, otherwise it is left-right. Then the half of the split that returns the best ellipse is used for fitting. The result is shown below in figure 5.

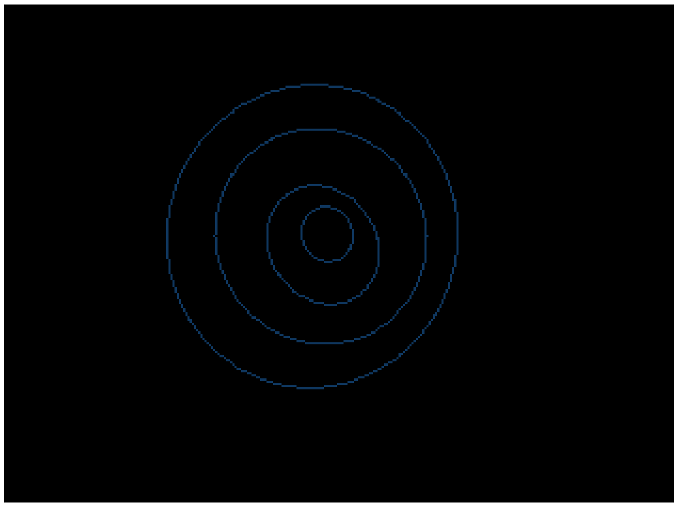

Figure 5. Ellipses from the contour portions shown in dark blue

This same implementation is adequate for full contours as well, as shown in figure 6.

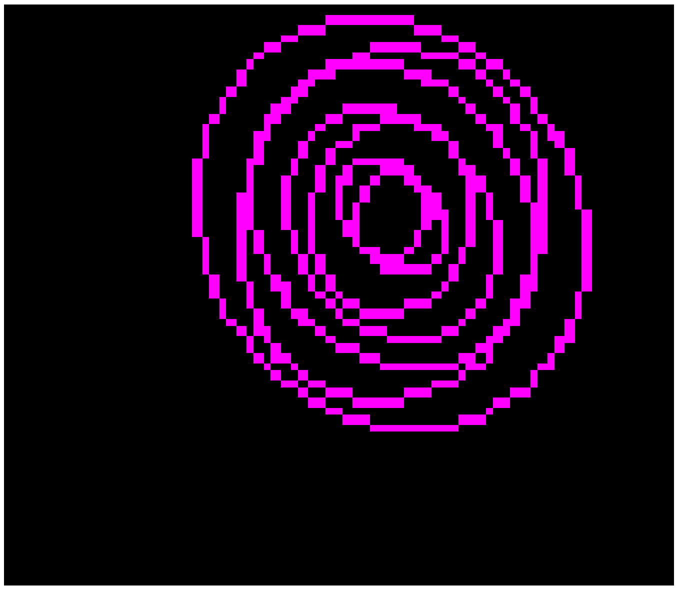

Figure 6. Ellipses from the pink contours in figure 2

The more pixels that the contour represents the better the ellipse fit will be. This method also directly returns ellipse coefficients that can be used in position estimation.

III. Position Estimate

A full method for estimating pose from the matrix equation for an ellipse is covered in our team's submission for Quiz 5. However this method has significant implementation challenges. As such, a simpler and more straightforward method of position estimation using homography is discussed here using reference [1].

Homography is a form of prospective transform where the known 3D points are all coplanar. Homography relates a set of known 2D projections of the 3D world points via a corresponding rotation and translation. The homography transformation is described by equation 4 below.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = HX_{3D} \tag{4}$$

Where H is the homography matrix defining an unknown protective transform shown in equation 5

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{bmatrix} = \begin{bmatrix} \vec{r_1} & \vec{r_2} & \vec{t} \end{bmatrix} \tag{5}$$

Homography requires a set of known 2D image points and their cooresponding 3D world points. In practice, we choose points on the ellipse that correspond to the center, the intersection of the ellipse with the semimajor axis, and the intersection of the ellipse with the semiminor axis. This yields a set of five 2D points per ellipse. If the radius of the corresponding circle in world coordinates is known, the corresponding 3D world points can be constructed as the set:

$$X_{3D} = \begin{bmatrix} 0 & \rho & -\rho & 0 & 0 \\ 0 & 0 & 0 & \rho & -\rho \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{6}$$

Where ρ is the known radius of the circle. The exact yaw orientation of the world points is not important because the yaw rotation of the circle will be ill defined due to the Z-axis symmetry of the circle. Therefore the set of points described in equation6 above can be used for all concentric circles with ρ substituted with the known radius. The definition of world points above sets the world coordinate frame with the origin at the center of the bullseye. The Homography matrix can be solved using the set of known points reorganized according to Equation III below where the example shown 4 known points. Additional points will augment the A matrix below in the same pattern shown. For the implimentation of concentric circles, there can be up to $5g = n$ points input into the homography matrix where g is the number of ellipsies detected.

$$\text{AH} = \begin{bmatrix} -x_1 & -y_1 & -1 & 0 & 0 & 0 & x_1x_1' & y_1x_1' & x_1' \\ 0 & 0 & 0 & -x_1 & -y_1 & -1 & x_1y_1' & y_1y_1' & y_1' \\ -x_2 & -y_2 & -1 & 0 & 0 & 0 & x_2x_2' & y_2x_2' & x_2' \\ 0 & 0 & 0 & -x_2 & -y_2 & -1 & x_2y_2' & y_2y_2' & y_2' \\ -x_3 & -y_3 & -1 & 0 & 0 & 0 & x_3x_3' & y_3x_3' & x_3' \\ 0 & 0 & 0 & -x_3 & -y_3 & -1 & x_3y_3' & y_3y_3' & y_3' \\ -x_4 & -y_4 & -1 & 0 & 0 & 0 & x_4x_4' & y_4x_4' & x_4' \\ 0 & 0 & 0 & -x_4 & -y_4 & -1 & x_4y_4' & y_4y_4' & y_4' \end{bmatrix} \begin{bmatrix} h1 \\ h2 \\ h3 \\ h4 \\ h5 \\ h6 \\ h7 \\ h8 \\ h9 \end{bmatrix} = 0$$

The singular value decomposition (SVD) is performed on matrix A which generates an optimized solution for the vector form of the homography matrix H. The last resulting eigenvector of the singular value decomposition yields the coefficients of the homography matrix. Lastly, we normalize the vector of homography coefficients by the last value h_9 and rearrange in matrix form. In order to extract pose from the resulting homography matrix, we must know the camera calibration matrix K. With this, the camera calibration can be applied as such:

$$K^{-1}H = \begin{bmatrix} h_1' & h_2' & h_3' \end{bmatrix} \tag{7}$$

From this corrected homography matrix, the position is found via equation 8

$$T = \frac{h_3'}{||h_1'||} \tag{8}$$

On the other hand, the resulting homography rotation elements are not guaranteed to be a rotation matrix or orthonormal. In order to convert the output to the closest possible rotation matrix, we first generate an orthonormal set with h_1' and h_2' and then compute the singular value decomposition.

$$\begin{bmatrix} h_1' & h_2' & h_1' \times h_3' \end{bmatrix} = U\Sigma V^T \tag{9}$$

The singular value decomposition is used to minimize the error between the homography matrix and a rotation matrix such that the result is a rotation matrix representing the homography result. The rotation matrix R is then conducted using equation10 below.

$$R = U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & detUV^T \end{bmatrix} V^T \tag{10}$$

The resulting rotation matrix R represents the rotation of the camera with respect to the world frame, and the translation T is the distance from camera center to the center of the bullseye. It should be noted that the yaw result in the rotation matrix above should not be trusted. Instead, one should use on board odometer or magnetometer readings for yaw estimates.

IV. Implementation

Our approach towards implementing the bullseye tracker used a simple estimate of the aircraft position with the down facing camera based on the center of the tag. If we assume that the Bebop estimate of altitude is accurate, the relative position of the center of the circle can be attained through similar triangles. The FOV in degrees of the camera in the x and y directions is known, the distance in pixels between the center of the image and the center of the tag is found, and converted into degrees using linear interpolation and the FOV found above. This gives the angle between the camera (assumed to be at the center of the drone) and the target. The altitude reading from the odometry is used then to convert the angle to a world x,y distance to the target. The following heuristic is used to guide the drone onto the bullseye using the position estimate explained above.

- If drone is > 0.1 m away from the center horizontally, execute a movement to center the aircraft.
- If drone is < 0.1 m away from the center, descend by 0.5 meters.
- If drone is centered on the bullseye and < 1.5 meters in altitude, execute landing.

V. Results

The bullseye landing video is included here: https://youtu.be/xtOOoGBjHto

An RVIZ visualization of the landing is shown attached to the submission in Bullseye-rviz.mp4.

A video of pose estimate is also included as Bullseye.mp4. Note that the estimated position switches between distance from the takeoff point when the circles are not visible to distance from the estimated tag position whern the tag is in view.

VI. Lessons Learned

While our approach to this problem worked well, we would like to add additional features to the solution in future iterations. For example, the solution demonstrated used only the down facing camera to see the bullseye because of the large field of view of the down facing camera. However, it would be beneficial to also use the front facing camera to estimate position for future flights. In some cases, the bullseye may not be in view of the down facing camera which could prevent the drone from landing quickly. Additionally, the homography estimate was shown to be a robust solution, however a PnP implementation may be better.

Lastly, improvements in outer loop controller could be implemented in the future to improve run time. The landing algorithm proved to be reliable and robust, but it was slow. Additional gains tuning and a faster frequency waypoint controller could help speed up the tracking solution.

VII. Conclusion

The implementation of methods described above resulted in a robust solution for finding the bullseye and executing a landing. While functional, vehicle speed could be improved in future iterations. Additionally, more complex and accurate position finding algorithms like full blown PnP could be implemented to further improve the solution. Overall, the drone performance resulted in a successful tracking and landing on the target.

References

[1] Sanket, Nitin; https://cmsc426.github.io/gtsam/p4
[2] Andrew W. Fitzgibbon and Maurizio Pilu and Robert B. Fisher;Direct Least Squares Fitting of Ellipses, 1996 http://cseweb.ucsd.edu/ mdailey/Face-Coord/ellipse-specific-fitting.pdf