AIAA Guidance, Navigation, and Control Conference and Exhibit
21 - 24 August 2006, Keystone, Colorado

AIAA 2006-6541

Obstacle Avoidance For Unmanned Air Vehicles Using Image Feature Tracking

Brandon Call[*], Randy Beard[†] and Clark Taylor[‡]

Department of Electrical and Computer Engineering, Brigham Young University, Provo, Utah, 84602, USA

Blake Barber[§]

Department of Mechanical Engineering, Brigham Young University, Provo, Utah, 84602, USA

This paper discusses a computer vision algorithm and a control law for obstacle avoidance for small unmanned air vehicles using a video camera as the primary sensor. Small UAVs are used for low altitude surveillance flights where unknown obstacles can be encountered. Small UAVs can be given the capability to navigate in uncertain environments if obstacles are identified. This paper presents an obstacle detection methodology using feature tracking in a forward looking, onboard camera. Features are found using the Harris Corner Detector and tracked through multiple video frames which provides three dimensional localization of the salient features. A sparse three dimensional map of features provides a rough estimate of obstacle locations. The features are grouped into potentially problematic areas using agglomerative clustering. The small UAV then employs a sliding mode control law in the autopilot to avoid obstacles.

I. Introduction

Small unmanned air vehicles are primarily used for low altitude surveillance tasks such as forest fire tracking, civilian search and rescue, military reconnaissance, convoy support and other military operations in urban terrain. Paths for small UAVs are planned by human operators or by automated path planners. Human operators can inadvertently create infeasible paths. Automated path planners often use terrain data or environment maps that include interpolation errors and omit relatively small but important features like trees and buildings. In order to effectively complete objectives in the presence of unknown obstacles, small surveillance UAVs must have reactive navigation abilities.

A variety of sensors could potentially be used to navigate around unknown obstacles, however, small UAVs require lightweight, mechanically simple solutions. Additionally, passive sensors are preferable in military operations because signals from active sensors can be detected and jammed or may interfere with similar sensor signals on other nearby vehicles. Video cameras provide a solution that is lightweight, simple, passive and inexpensive, and are therefore an attractive sensor for small UAVs. However, a video camera's limitations include reduced efficiency under low-visibility conditions such as night, fog, and smoke. Additionally, significant computer processing of sensor information is required in order to be useful.

A. Previous Work

Many different sensors have been used for obstacle avoidance on autonomous systems. Zelenka and Almsted developed a 35-gigahertz radar designed for obstacle avoidance for aircraft in Ref. 1. Earlier, Zelenka, Clark and Zirkler[2] developed a forward looking laser radar to assist a helicopter pilot in obstacle detection and

[*]Graduate Research Assistant, Electrical and Computer Engineering, Brigham Young University, 459 Clyde Building, Provo, Utah, 84602

[†]Associate Professor, Electrical and Computer Engineering, Brigham Young University, 459 Clyde Building, Provo, Utah, 84602

[‡]Assistant Professor, Electrical and Computer Engineering, Brigham Young University, Provo, Utah, 84602

[§]Graduate Research Assistant, Mechanical Engineering, Brigham Young University, Provo, Utah, 84604 avoidance. Laser ranging was also used by Saunders et al.[3] for small fixed wing aircraft. Sonar is used primarily for underwater autonomous systems.[4] Many of these sensors are too heavy to be used on a small UAV making video cameras an attractive alternative.

Developers of many different types of autonomous vehicles employ video sensors to gain information about the surrounding environment. Lorusso and Michelli[5] used optical flow computed from video cameras to autonomously steer ground vehicles on city roads, country roads or motorways. Song and Huang[6] use video and real time optical flow to aid a guide-dog robot. Even on larger aircraft, video is an attractive sensor. Gandhi et al[7] used video to assist pilot detection of mid-air collisions. This implementation processed only one frame every seven seconds but was still useful enough to detect potential collisions.

Computing optical flow and performing three dimensional reconstruction are very similar mathematically. Both problems can be classified as structure from motion. Barron et al.[8] provides a survey of optical flow computation algorithms. Shahraray and Brown[9] locate stationary objects from a moving camera. They used a second order model of camera motion and spline smoothing to robustly estimate the depth of the image pixels.

B. Overview

Three dimensional world information must be acquired from the video in order to successfully avoid an obstacle that is on the UAVs desired path. Obtaining three dimensional data from video requires stereo vision or multiple frames from a single camera. The relative location of an object from camera to camera or the movement of the objects in multiple frames provides enough information to compute an estimate of the three dimensional location of the object in view. For the experiments presented in this paper, a single camera on a moving UAV was used, however the algorithms are applicable to a stereo camera configuration. Many different ways to compute optical flow have been outlined by Barron et al.[8] Optical flow for a full image can be difficult to compute quickly enough to use as an input to a flight controller. Therefore, in this paper, we restrict the scope of optical flow computation by only computing optical flow at a limited number of pixels. This provides a sparse three dimensional map of the world which can then be used for flight controller input. The Harris Corner Detector[10] provides a measure of information contained near each pixel. Then, image correlation is used to locate that pixel in a subsequent frame. Next, the features are clustered into objects which can then be avoided using an agglomerative clustering algorithm.[11] The agglomerative clustering algorithm combines groups of features into new groups until all groups are separated by a distance larger than a threshold distance. The nearest cluster that intersects the path of the UAV is then avoided using a sliding mode control law.

To implement obstacle avoidance in this way, we assume that obstacles will be feature rich by having a texture or multiple corners as seen at the pixel level. Most features encountered in the real world will have these types of features. To violate this assumption, an obstacle would have to have either no texture or parallel lines with no visible endpoints. Examples of obstacles that violate this constraint are flag poles, cement walls or possibly wires. Additionally, optical flow calculations will potentially fail when viewing a reflective glass surface, as the motion seen in reflection will not be representative of the motion of the structure.

II. Feature Tracking

Instead of computing an optical flow vector for each pixel or for each region of pixels, feature tracking is a method of computing optical flow for visually interesting pixels. Each pixel is assigned a value based on how easily it can be tracked and then a small window around the pixel is correlated in subsequent frames to relocate the same image feature. The velocity of the image feature and the movement of the camera allows the three dimensional location of the image feature to be calculated and mapped.

A. Measuring Pixel Information

Corners are the best image feature for optical flow computations because they provide accurate movement information in two dimensions. A textureless image region provides no information and an edge in the image only gives information perpendicular to the edge as seen in Figure 1.

The determination of the feature value is a computationally expensive step in the algorithm. The algorithm can be made to run on computers with slower processors by reducing the size of the image portion that will be used for obstacle detection. When using a forward-facing video camera, the center of the image provides the most immediate information for obstacle detection.

B. Corner Detector

The feature strength is calculated using the Harris Edge and Corner Detector described in Ref. 10. This corner detector uses the moment matrix

$$M_{ij} = \begin{bmatrix} \bar{I}_x^2 & \bar{I}_{xy} \\ \bar{I}_{yx} & \bar{I}_y^2 \end{bmatrix} \tag{1}$$

of the vertical and horizontal image gradients at each pixel to determine if the pixel is an edge, corner or untextured region.

I_x and I_y are the image gradients in the horizontal and vertical directions respectively of the grayscale image I_1. $\bar{I}$ is the average over a square window of the image. The moment matrix is then used in

$$R_{ij} = det|M_{ij}| - 0.04(I_x^2 + I_y^2)^2 \tag{2}$$

to calculate feature strength R. Small values for R are untextured areas, while large negative numbers are edges and large positive numbers are corners.

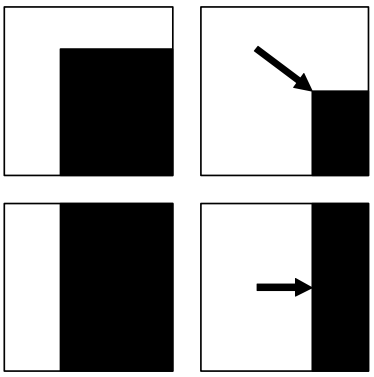


Figure 1. Corners are the best image feature for optical flow computations because they provide accurate movement information in two dimensions (top) while edges provide information along one dimension (bottom).

C. Feature Selection

The pixels with the largest feature strength are selected for tracking. However, based on the movement of the camera, there will potentially be one pixel that will not provide any optical flow information even if it has a high feature value because the object will not move in the image coordinates. This pixel is called the epipole and is the pixel that is inline with the translational motion of the camera. When a UAV is flying straight with a forward facing camera the epipole is in the center of the image. An optional linear weighting was used on the feature strength values to bias the feature selection away from the epipole when necessary. Additionally, one good image corner produces multiple neighboring pixels with high feature strength. The pixel with the highest feature strength is selected and the feature strength of neighboring pixels is set to zero so that the same image feature isn't selected multiple times.

When a feature is selected, a window of the grayscale image is saved for image correlation. A ray, represented in vehicle frame, originating at the center of the camera and pointing in the direction of the pixel is also recorded for object localization.

There are multiple coordinate transformations that must be done to convert from pixel coordinates to camera, body or vehicle coordinates. These transformation matrices depend on the attitude of the UAV and the location of the camera on the plane and the camera calibration matrix. We use the matrices outlined by Redding[12] to calculate transformations. When transforming the two dimensional pixel data into three dimensional coordinates only a ray originating at the camera center extending in the direction of the pixel can be obtained.

D. Image Correlation

Image correlation is performed when a second video frame (I_2) is available for processing. The feature window saved from I_1 is convolved using the pseudo-normalized correlation function

$$\frac{2 * \sum I_1 * I_2}{\sum I_1^2 + \sum I_2^2} \tag{3}$$

as derived by Moravec in Ref. 13. The pixel with the highest correlation value is then used to perform a biquadratic fit to a 3x3 neighborhood of correlation scores to achieve sub pixel accuracy. The fit function is

$$q = a * x^2 + b * y^2 + c * x * y + d * x + e * y + f \tag{4}$$

where q is a correlation score, x is the column index, and y is the row index. The coefficients a-f are fit by least squares. The subpixel maximum of the fitted surface is then found by partial differentiation, which leads to the following solution for x and y:

$$x = (-2 * b * d + c * e)/(4 * a * b - c^2) \tag{5}$$

$$y = (-2 * a * e + c * d)/(4 * a * b - c^2). \tag{6}$$

The implementation first verifies that the pixel-resolution maximum occurs at an interior location, then computes the subpixel maximum, then verifies that the subpixel maximum is within one pixel of the pixel-resolution maximum.

The covariance of the fit is used as a metric of how good the fit is. The number of incorrect correlations that are accepted is reduced significantly by using a relatively high threshold for this fit value.

E. Ray Intersection

The two rays are generated by the location of the camera and the direction of the feature located in I_1 and the location of the camera and the direction of the feature found in I_2 using image correlation. The position of the feature is estimated by finding the point where the two rays intersect or the point where they are the closest. In implementation, the Global Positioning System (GPS) is the easiest way to provide a relative location for the UAV. The vision algorithm will still work when GPS is denied if a relative position can be determined by other means. Alternatively, a stereo camera pair will have a fixed, known distance between the cameras, but on a small UAV the distance will be small and the ray intersection calculations will be ill conditioned.

The intersection or closest points on two rays can be computed as follows. A line in three dimensional space is represented by parametric equations of the form $(p + kv)$ where p_i is a point on the line i, v_i is the vector indicating the direction of the line i, k_i is the scalar used to specify a distance from p_i in the direction of v_i.

To find the point which is closest to both lines (p_s), we minimize the error (e) from both lines, giving

$$e \quad = \quad p_1 + k_1 v_1 - (p_2 + k_2 v_2). \tag{7}$$

Because e is a vector, we need to minimize its magnitude, leading to

$$< e, e > \quad = \quad < p_1 - p_2 + k_1 v_1 - k_2 v_2, p_1 - p_2 + k_1 v_1 - k_2 v_2 > . \tag{8}$$

Writing out each component of the vector individually (via subscripts x, y, and z), and defining T as $p_1 - p_2$, we obtain:

$$< e, e > \quad = \quad (T_x + k_1 v_{1x} - k_2 v_{2x})^2 + (T_y + k_1 v_{1y} - k_2 v_{2y})^2 + (T_z + k_1 v_{1z} - k_2 v_{2z})^2. \tag{9}$$

Taking the derivative of the above equation in terms of k_1 and k_2 to find the minimum error leads to:

$$\frac{\delta < e, e >}{\delta k_1} \quad = \quad 2(T_x v_{1x} + T_y v_{1y} + T_z v_{1z}) + 2k_1(v_{1x}^2 + v_{1y}^2 + v_{1z}^2) - 2k_2(v_{1x}v_{2x} + v_{1y}v_{2y} + v_{1z}v_{2z}) \tag{10}$$

$$\frac{\delta < e, e >}{\delta k_2} \quad = \quad 2(T_x v_{2x} + T_y v_{2y} + T_z v_{2z}) + 2k_2(v_{2x}^2 + v_{2y}^2 + v_{2z}^2) - 2k_1(v_{1x}v_{2x} + v_{1y}v_{2y} + v_{1z}v_{2z}). \tag{11}$$

Setting these derivatives equal to zero, we can rewrite the formulas in matrix form as

$$\begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} \tag{12}$$

where

$$\begin{aligned} s_1 &= - < T_v \cdot v_1 >, \\ s_2 &= < T_v \cdot v_2 >, \\ m_1 &= < v_1 \cdot v_1 >, \\ m_2 &= - < v_1 \cdot v_2 >, \\ m_3 &= - < v_1 \cdot v_2 >, \end{aligned}$$

and

$$m_4 = < v_2 \cdot v_2 > .$$

Note that all of the quantities except for k_1 and k_2 are known. The unknowns k_1 and k_2 can be solved for by inverting the M matrix. Evaluating the lines at the respective k's gives two points, where the closest point of intersection of the two lines will be half way between those two points.

III. Feature Clustering

Feature tracking with ray intersection provides a sparse map of three dimensional locations of some world features. At each frame, the features are clustered using an agglomerative diversity clustering technique.[11] Initially each group is put into it's own cluster. Then the two closest clusters are combined creating a new group. This is repeated until the two closest groups are farther away than a threshold distance. This clustering algorithm works well for feature points because there are an unknown number of obstacles in any given image and there is a minimum distance at which it would be dangerous for the UAV to attempt to navigate between.

After the clusters are created, a bounding cylinder is created around the cluster of features. The bottom of the bounding cylinders extend downward to a known ground surface or indefinitely if the height above ground is unknown. A bounding cylinder is chosen over bounding box or sphere to facilitate the obstacle avoidance control law.

IV. Obstacle Avoidance Control Law

Because vision data can only reconstruct obstacles that lie nearest to the camera inside of its field of view, global obstacle information is not available. For this reason, it was determined to use a reactive avoidance algorithm rather than a global path planning algorithm such as a rapidly exploring random tree.[14]

Once obstacles have been grouped and mapped from pixel to world coordinates, each of the obstacles is checked for intersection with the waypoint path. The position and radius of the cylinder representing the object that intersects the waypoint path at the point closest to the UAV is then passed as an input to the reactive avoidance controller. When the reactive avoidance controller receives an input it overrides the waypoint navigation controller with course commands from the sliding mode controller described by

$$\chi^c = \chi + \frac{1}{\alpha}\left(\frac{V_g}{\delta}\sin(\chi-\gamma) + c\frac{k}{(1+k(\delta-r))^2}V_g\cos(\chi-\gamma) + \frac{\kappa}{\alpha}\mathrm{sat}\left(\frac{\tilde{\chi}}{\epsilon}\right)\right), \tag{13}$$

where $\tilde{\chi}$ is defined as $\chi^d - \chi$, k is a tunable parameter that controls the convergence rate of the vector field, κ is a tunable parameter that controls the reactiveness of the controller to tracking errors in course, and ϵ is a tunable parameter that controls the width of the sliding mode around the nominal radius r. Other terms in the control equation are defined as shown in Figure 2.

Nelson et al.[15] showed that this control law would cause asymptotic convergence of the UAV course to the vector field described by

$$\chi^d = \gamma + c\left(\frac{\pi}{2} - \tan^{-1}(k(\delta-r))\right). \tag{14}$$

An example vector field based on (14) is shown in Figure 2.

The parameter c in equation (13) represents the direction of the orbit with $c = 1$ corresponding to a clockwise orbit and $c = -1$ corresponding to a counterclockwise orbit. The value of c passed to the controller is 1 if the object is found to lie on the right hand side of the path or directly on the path and -1 otherwise. To avoid switching orbit directions after the avoidance algorithm has begun, the value of c is only calculated on the initial iteration of the avoidance algorithm for each new obstacle.

The radius argument r passed into the control law (13) is the radius of the intersecting cylinder plus some additional safety threshold. The avoidance controller will cause the UAV to orbit the center of the obstacle at some offset radius greater than the radius of the cylinder which represents the obstacle. As the dimensions and location of the cylinder representing the obstacle are continually updated, the arguments passed to the controller are updated to reflect these changes. This continues until the object has been determined to be clear of the path from the UAV to the waypoint. At that point normal waypoint navigation continues along the path from the UAV to the next waypoint.

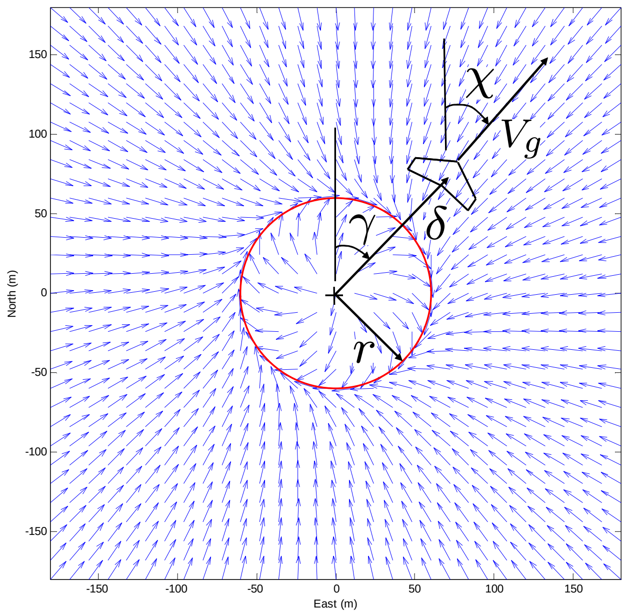


Figure 2. This figure shows an example vector field for flying a circular orbit. The desired course of the UAV is specified by the direction of the vector field.

V. Results

Flight experiments on this algorithm were performed in simulation using the open-source UAV simulator Aviones developed at Brigham Young University. This simulator has a six degree of freedom physics model, real world terrain data and simulated obstacles. The UAV was commanded to fly a waypoint path through a square obstacle with 80m sides as seen in Figure 3. The simulator also provides synthetic video streaming for image processing.

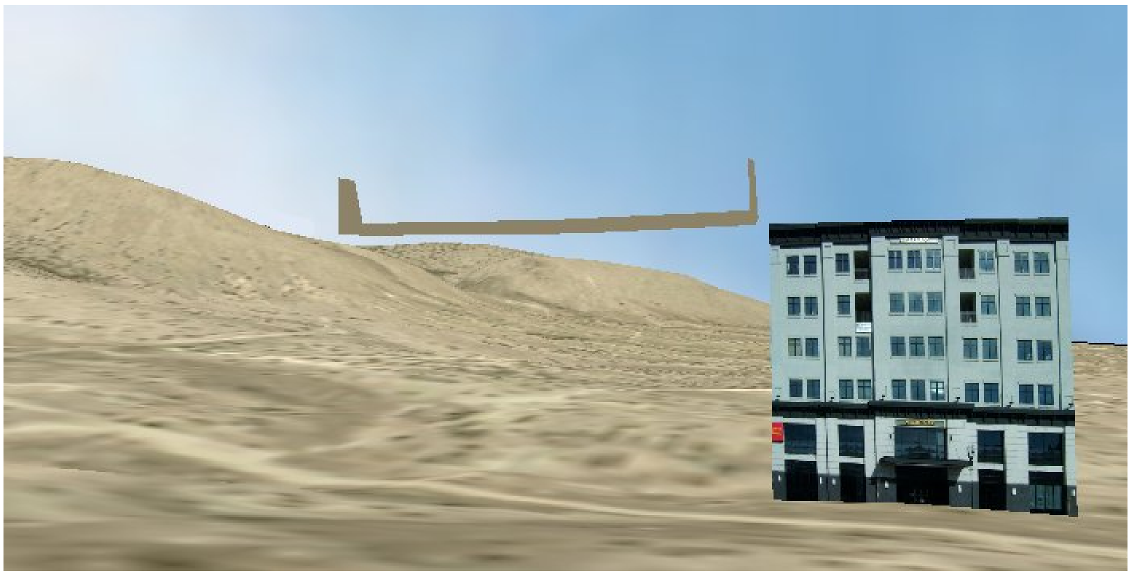


Figure 3. A screen shot from the simulator as the UAV navigates around an obstacle.

A. Feature Tracking Results

Figure 4 shows the three dimensional locations produced by one pass at the rectangular obstacle located at $40m$ indicated by the square. In the image, there is a large group of pixels near the wall and a smaller group more than $150m$ from the wall. The group of features correctly located near the obstacle have mean of $41.08m$ and standard deviation of $10m$. The smaller incorrect group of features were caused by incorrect image correlation which leads to largely incorrect three dimensional feature locations. This smaller group comprises approximately 9% of the image features.

Figure 5 is a plot of the accuracy of the feature localization vs distance from the feature. In Figure 5a there is a clear division between small errors and large errors. The large errors are caused by incorrect image correlation. The large errors are filtered using the clustering algorithms described in Section III. The subset enclosed by the rectangle in Figure 5a is shown in Figure 5b. As seen in Figure 5b, the accuracy of obstacle detection is greatly increased when the distance to the obstacle is decreased. The algorithm is accurate enough at $130m$ to successfully locate the obstacle. The estimated obstacle location can be further refined using subsequent video frames until a critical turning distance is achieved. Interestingly, no data was gathered at distances closer than $60m$. When the UAV was closer than $60m$, it was too close to the texture that was mapped onto the obstacle and the texture was blurred. This created no significant image features for the algorithm. In hardware flight tests, this problem will not be encountered.

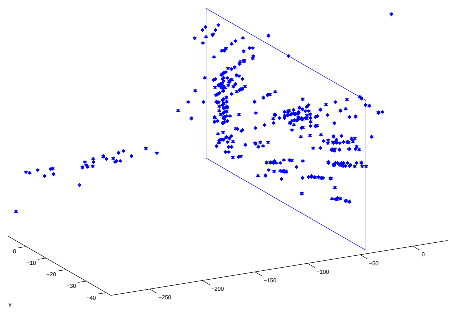
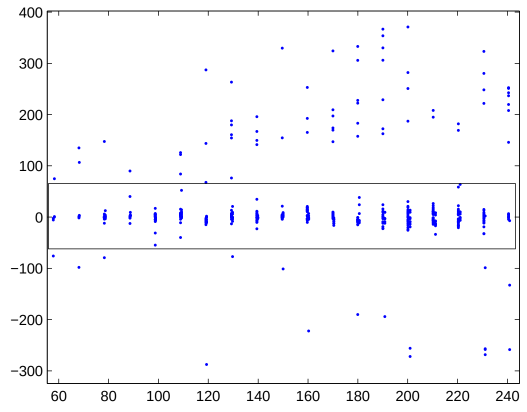
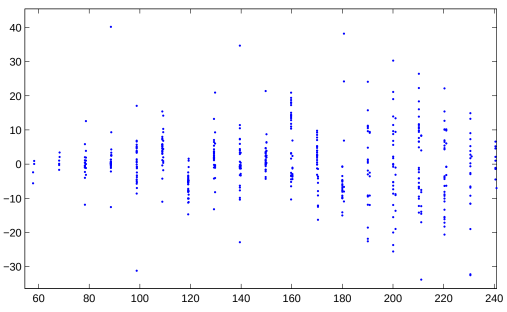


Figure 4. Feature location results from one pass at a textured wall.



(a) A plot of the error in feature tracking relative to distance. Large errors are caused by incorrect image correlation and comprise 9% of the features.



(b) The a subsection of the plot in (a) focusing on the section where the image correlation was accurate

Figure 5. Plots of obstacle detection accuracy vs distance from obstacle

B. Feature Clustering Results

Clusters of features located in three dimensional coordinates were clustered using the agglomerative clustering technique described in Section III. Clusters are represented using bounding cylinders. Figure 6 shows an overhead view of the cylindrical clusters created from one pass at a rectangular obstacle. The UAV approached the obstacle from the left, and only the left face of the obstacle was visible in the video frames. The dark circle represents the average center and radius. The average center is at $(-42.8, 3.0)$ with an average radius of $34.0m$. A perfect cylindrical representation of the front face of the obstacle would be centered at $(-40, 0)$ and have a radius of $40m$.

C. Flight Simulation Results

Figure 7 shows screen shots of flight simulation. The building texture had repetitive features which caused some image correlation error. The white crosses indicate image features that were correlated with high confidence, and located in world coordinates. The figure also contains a projection of the cluster cylinder projected into the image frame. The projected cylinder appears as a white rectangle in the image. Figure 8

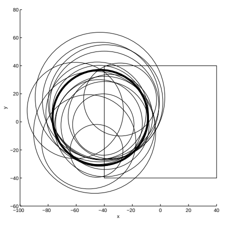
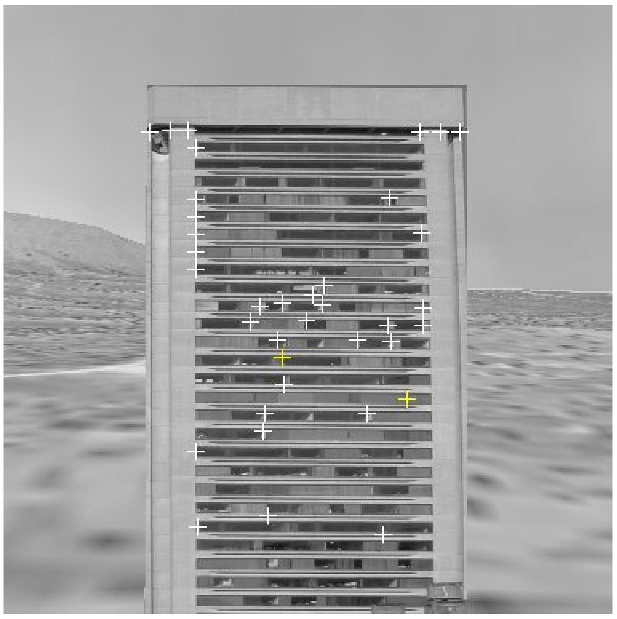


Figure 6. An overhead view of the cylindrical clusters created during one pass at the rectangular obstacle

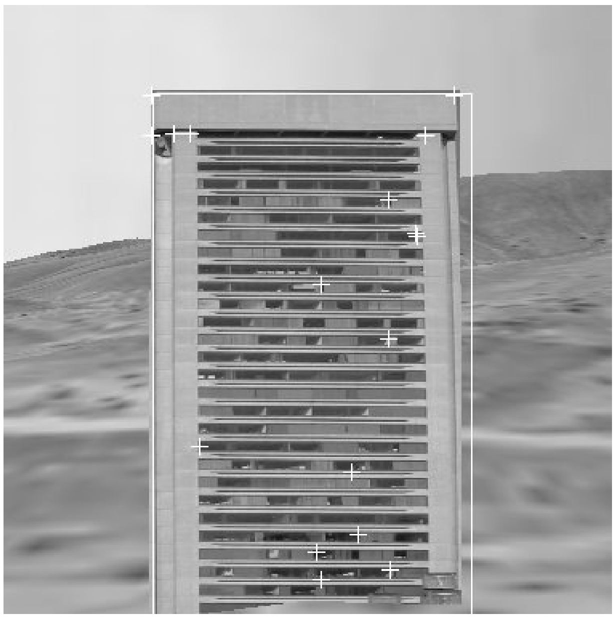
VI. Conclusion

This paper presents a method for detecting and avoiding obstacles for a small UAV using feature tracking from an onboard camera. Features that are well suited for tracking were located using the Harris Corner Detector. Image regions corresponding to the features were then correlated with subsequent video frames to determine feature movement (ie. optical flow). A world coordinate was determined for each feature using ray intersection and then the features were clustered into obstacles using their proximity in world location. The clusters were represented by bounding cylinders which were avoided using a sliding mode UAV controller. It was found that a sparse depth map from a single forward facing camera provides enough information to avoid obstacles in the path of the UAV.

Future work on this project may include improving the robustness of the feature tracking algorithm using statistical analysis and experimenting with dense optical flow calculations to create depth maps. Additionally, the algorithm may be improved by evaluating additional clustering techniques. Finally, the algorithm will be tested in hardware and may be combined with side-looking sensors and height above ground for low altitude urban navigation.



(a) Imagery from a simulated UAV approaching an obstacle. White crosses indicate features that have been located in three dimensional coordinates and projected back into the image frame.



(b) A screenshot of simulation where the UAV correctly detects the obstacle and encloses it with a bounding cylinder

Figure 7. Obstacle Avoidance using UAV simulation

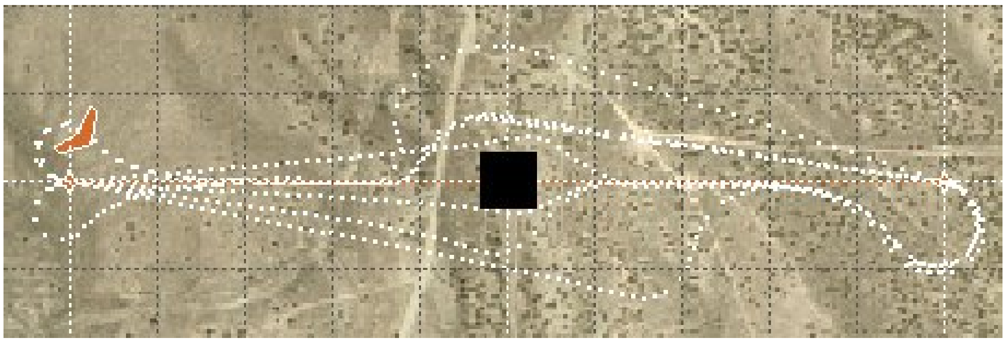


Figure 8. An overhead view of the flight path of the UAV as it avoids an obstacle indicated by the black box. The path of the UAV is indicated by the white dashed line.

Acknowledgments

This research was supported by AFOSR grant FA9550-04-C-0032. Some of this work was performed while the first author was a summer intern at the Munitions Directorate of the Air Force Research Laboratory at Eglin Air Force Base under the supervision of Robert Murphy and Kevin O'Neal.

References

[1]Zelenka, R. E. and Almsted, L. D., "Design and Flight Test of 35-Gigahertz radar for terrain and obstalce avoidance," *Journal of Aircraft*, Vol. 34, March/April 1997, pp. 261–263.

[2]Selenka, R. E., Clark, R. F., and Zirkler, A., "Development and flight test of terrain-referenced guidance with ladar forward sensor," *Journal of Guidance Control and Dynamics*, Vol. 19, July/August 1996, pp. 823–828.

[3]Saunders, J. B., Call, B., Curtis, A., Beard, R. W., and McLain, T. W., "Static and Dynamic Obstacle Avoidance in Miniature Air Vehicles," *AIAA Infotech at Aerospace*, 2005, Paper no. AIAA-2005-6950.

[4]Petillot, Y., Ruiz, I. T., and Lane, D. M., "Underwater Vehicle Obstacle Avoidance and Path Planning Using a Multi-Beam Forward Looking Sonar," *IEEE Journal of Oceanic Engineering*, Vol. 26, No. 2, April 2001, pp. 240–251.

[5]Lorusso, A. and Micheli, E. D., "Approach to Obstacle Detection and Steering Control From Optical Flow," *IEEE Intelligent Vehicle Symposium, Proceedings*, 1996, pp. 357–362.

[6]Song, K. T. and Huang, J.-H., "Fast Optical Flow Estimation and its Application to Real-Time Obstacle Avoidance," *TRA*, Vol. 3, May 2001, pp. 2891–2896.

[7]Gandhi, T., Yang, M.-T., Kasturi, R., Camps, O., Coraor, L., and McCandles, J., "Detection of Obstacles in the Flight Path of an Aircraft," *TAES*, Vol. 39, No. 1, 2003, pp. 179–191.

[8]Barron, J., Fleet, D., and Beauchemin, S., "Performance of optical flow techniques," *International Journal of Computer Vision*, Vol. 12, No. 1, Feb 1994, pp. 43–77.

[9]Shahraray, B. and Brown, M. K., "Robust Depth Estimation From Optical Flow," *IEEE Second International Conference on Computer Vision*, December 1988, pp. 641–650.

[10]Harris, C. J. and Stephens, M., "A combined corner and edge detector," *In Proc. 4th Alvey Vision Conf. , Manchester*, 1988, pp. 147–151.

[11]Gascuel, O. and McKenzie, A., "Performance Analysis of Hierarchical Clustering Algorithms," *Journal of Classification*, Vol. 21, No. 1, 2004, pp. 3–18.

[12]Redding, J., *Vision-based Target Localization from a Small, Fixed-wing Unmanned Air Vehicle*, Master's thesis, Brigham Young Univeristy, 459 Clyde Building, Provo, UT 84604, 2005.

[13]Moravec, H., *Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover*, Ph.D. thesis, Stanford Univerisity, May 1980.

[14]Cheng, P., Shen, Z., and LaValle, S. M., "RRT-Based trajectory design for autonomous automobiles and spacecraft," *Archives of Control Sciences*, Vol. 11, No. 3-4, 2001, pp. 167–194.

[15]Nelson, D. R., Barber, D. B., McLain, T. W., and Beard, R., "Vector Field Path Following for Small Unmanned Air Vehicles," *Proceedings of the American Control Conference*, June 2006, pp. 5788–5794.