Vision-based Navigation Frame Mapping and Planning for Collision Avoidance for Miniature Air Vehicles

Huili Yu*,[a], Randy Beard[a], Jeffrey Byrne[b]

[a]*Department of Electrical and Computer Engineering, Brigham Young University, Provo, Utah, 84602*
[b]*Scientific Systems Company, Inc., 500 W Cummings Park, Suite 3000, Woburn, MA 01801*

Abstract

This paper presents a vision-based navigation frame mapping and path planning technique for collision avoidance for Miniature Air Vehicles. A depth map that represents the range and bearing to obstacles is obtained by computer vision. Based on the depth map, an extended Kalman Filter is used to estimate the range and bearing. Using this information, a map, constructed in polar coordinates, is created in the navigation frame of the MAV. The Rapidly-Exploring Random Tree algorithm is employed to find a collision-free path in the navigation frame. The proposed algorithm was successfully implemented in both simulation and flight tests.

Key words: Computer vision, Mapping and path planning, Miniature Air Vehicle, Collision avoidance

*Corresponding author. Tel.:+18014223230
Email addresses: huiliyu.yhl@gmail.com (Huili Yu), beard@byu.edu (Randy Beard), jbyrne@ssci.com (Jeffrey Byrne)

1. Introduction

Miniature Air Vehicles (MAVs) have the potential to perform tasks that are too difficult or dangerous for human pilots. For example, they can monitor critical infrastructure and real-time disasters, perform search and rescue, and perform in-storm weather measurements [1]. For many of these applications, MAVs are required to navigate in urban or unknown terrains where obstacles of various types and sizes may hinder the success of the mission. MAVs must have the capability to autonomously plan paths that do not collide with buildings, trees, or other obstacles. Therefore, the path planning problem for MAVs has received significant attention [1][2][3] [4][5].

The general framework for the path planning problem can be described as follows: *given a description of the environment, find a feasible path between two configurations in the environment that does not result in a collision with any obstacles*. Three common solutions to the path planning and obstacle avoidance problems are probability roadmap, potential fields, and cell decomposition [6]. The probability roadmap planner produces fast paths to the goal with computation measured in seconds [7][8] [9][10]. It consists of two phases: preprocessing and query processing. The preprocessing phase constructs a roadmap by taking random samples from a configuration space and connecting these configurations to other nearby configurations. The query phase connects the start and goal configurations to the roadmap. While the original probability roadmap planners were designed for holonomic vehicles, Reference [11] extends the probability roadmap to nonholonomic vehicles. The potential field planner was originally described in

[6][12][13] and its basic idea is to construct an attractive force at the goal and a repulsive force on the obstacles, which are functions of the current state. Potential fields cannot guarantee collision-free paths since they often produce local minima. Cell decomposition methods partition a configuration space into cells, which are connected to generate a graph. Search algorithms like A* or Dijkstra's algorithm are applied to the graph for finding a path from the initial to goal configurations [14][15].

All path planning methods require some computation time to generate a new trajectory after detecting obstacles on the current trajectory, adding time constraints to the path planning problem. It is possible that the MAV will be required to avoid new obstacles while the path planning algorithm is in the middle of execution. This observation motivates the use of a multi-level path planning algorithm. Reference [16] presents a hierarchical path planning scheme using wavelets for agents operating in partially known environments so that the computational cost can be managed. Wavelets are used to obtain an approximation of the environment at different level of fidelity, causing a cell decomposition of the environment with variable resolution. Based on the cell decomposition, Dijkstra's algorithm is used to plan paths at each time globally. Reference [17] designs a local replanning algorithm that extends the results of Ref. [16] using the localization property of the wavelet transform so that the path is planned globally once and is replanned locally at each time step. Reference [18] improves the performance of the local path planning strategy in Ref. [17] using the sensor data directly to construct a sector-like multi-resolution decomposition of the agent's immediate environment. The

cell decomposition is constructed in polar coordinates compatible to the on-board sensor data and a path is planned locally based on the cell decomposition.

The contribution of this paper is that we design a vision-based local path planner within a multi-tiered path planning system. The path planning system consists of three tiers: global, local and reactive path planners. The global path planner generates an inertial path from the current configuration to the goal. The local path planner (the focus of this paper) modifies portions of the global path to maneuver around obstacles that were not in the original global map, but that have been detected by the vision sensor. The reactive planner responds to pop-up threats where collisions are imminent. Each tier acts at a different time scale and has a decreasing computation time. The global path planner reacts to obstacles where the predicted time-to-collision is greater than T_{ttc1} seconds and requires the longest computation time. The local path planner reacts to obstacles where the predicted time-to-collision is T_{ttc2}-T_{ttc1} seconds and requires less computation time. The reactive path planner reacts to obstacles where the predicted time-to-collision is 0-T_{ttc2} seconds and requires very little computational resources.

This paper explores a vision-based navigation frame mapping and path planning technique for designing the local path planner. This technique is motivated by the observations that only local information is important for the local path planning problem and it is much easier to create maps and plan paths directly in the navigation frame of the MAV using the camera data without transforming to the inertial frame. A single strap-down camera is employed to measure the range and bearing to obstacles since it is cheap and lightweight. Using computer vision

based time-to-collision estimation, we obtain a depth map at each time step. Based on the depth map, we use the extended Kalman Filter (EKF) to estimate range and bearing and the nearest neighbor (NN) approach to address the data association problem in the navigation frame. We create a map in the navigation frame using polar coordinates. We then apply the Rapidly-Exploring Random Tree (RRT) algorithm to plan collision-free paths in the navigation frame.

The proposed algorithm can solve the path planning and multiple obstacles avoidance problems for a MAV operating in an unknown environment and has the following unique features:

- The world map is constructed using polar coordinates in the navigation frame of the MAV without transforming the camera data to the inertial frame.
- The EKF is used to estimate the range and bearing to obstacles in the navigation frame and to address measurement uncertainties produced by the camera.
- The nearest neighbor approach is used for addressing the data association problem in the navigation frame.
- The RRT algorithm is applied to the navigation frame map for planning collision-free paths.

The proposed path planning algorithm is separated into three categories: computer vision, mapping and guidance. Figure 1 shows the information flow of the

the path planning algorithm. The video stream sent by the camera is processed by time-to-collision estimation to generate range and bearing measurements to obstacles. Using these measurements and IMU measurements from the MAV, the localizer employs the EKF to estimate the range and bearing. Based on the locations of the obstacles, the map builder creates a map in the navigation frame using polar coordinates. The path planner then applies the RRT algorithm to plan a collision-free path. The path is smoothed by the path smoother and sent to the autopilot.

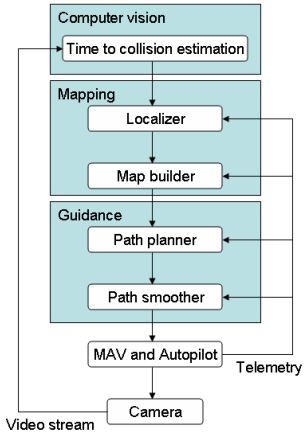


Figure 1: The information flow of the proposed path planning algorithm. The video stream is processed by time-to-collision estimation to generate a depth map that provides noisy range and bearing measurements to obstacles. The localizer filters the measurements to estimate the range and bearing to obstacles. The map builder creates a map based on the locations of the obstacles in the navigation frame. A collision-free path is planned by the path planner and smoothed by the path smoother. The autopilot maneuvers the MAV to follow the smoothed path.

This paper is organized as follows. Time-to-collision estimation used to obtain the depth map is described in Section 2. Section 3 describes the vision-based mapping in the navigation frame using polar coordinates. In Section 4, the RRT algorithm is applied to the map for finding a collision-free path. Sections 5 and 6

present simulation and flight test results for the proposed algorithm.

2. Time-to-collision estimation

Time-to-collision estimation is an approach to visual collision detection where an image sequence from a forward looking camera is processed to measure the seconds to collision for surfaces in the scene. In this section, we review the dominant approaches in the literature for monocular visual collision detection and estimation. These methods can be summarized in four categories: structure from motion, ground plane methods, flow divergence and insect inspired methods.

Structure from motion (SFM) is the problem of recovering the motion of the camera and the structure of the scene from images generated by a moving camera. SFM techniques [19] provide a sparse or dense 3D reconstruction of the scene up to an unknown scale and rigid transformation, which can be used for obstacle detection when combined with an independent scale estimate for metric reconstruction, such as from inertial navigation to provide camera motion or from a known scene scale. Modern structure from motion techniques generate impressive results for both online sequential and offline batch large scale outdoor reconstruction. Recent applications include online sparse reconstruction during MAV flight for downward looking cameras [20], and visual landing of helicopters [21][22]. However, SFM techniques consider motion along the camera's optical axis, as found in a collision scenario, to be degenerate due to the small baseline, which results in significant triangulation uncertainty near the focus of expansion which must be modelled appropriately for usable measurements.

Ground plane methods [23][24], also known as horopter stereo, stereo homography, ground plane stereo or inverse perspective mapping use an homography induced by a known ground plane, such that any deviation from the ground plane assumption in an image sequence is detected as an obstacle. A known ground plane forms a constant disparity surface or *horopter* in an image sequence after compensating for perspective effects. Using an estimate of the camera motion from odometry, sparse feature tracking or dense optical flow, a homography can be computed resulting in a registration of the ground plane in an image sequence. The registered images from the homography induced by the ground plane are then differenced, and any non-zero residual is defined to be an obstacle not on the ground plane. This approach has been widely used in environments that exhibit a dominant ground plane, such as in the highway or indoor ground vehicle community, however the ground plane assumption is not relevant for aerial vehicles.

Flow divergence methods rely on the observation that objects on a collision course with a monocular image sensor exhibit expansion or *looming*, such that an obstacle projection grows larger on the sensor as the collision distance closes [25][26]. This expansion is reflected in differential properties of the optical flow field, and is centered at the focus of expansion (FOE). The FOE is a stationary point in the image such that expansion rate from the FOE or *positive divergence* is proportional to the time-to-collision. Flow divergence estimation can be noisy due to local flow correspondence errors and the amplifying effect of differentiation, so techniques rely on various assumptions to improve estimation accuracy. These include assuming a linear flow field due to narrow field of view during terminal

approach [25][27][26][28], assuming known camera motion and positioning the FOE at the image center [29][30][31][32], or known obstacle boundaries for measurement integration [28][29][33]. These strong assumptions limit the operational envelope, which have lead some researchers to consider the qualitative properties of the motion field rather than metric properties from full 3D reconstruction as sufficient for collision detection [34][26]. However, this does not provide a measurement of time-to-collision and does not localize collision obstacles in the field of view.

Insect vision research on the locust, fly, and honeybee show that these insects use differential patterns in the optical flow field to navigate in the world. Specifically, research has shown that locusts use expansion of the flow field or "looming cue" to detect collisions and trigger a jumping response [35]. This research has focused on biophysical models of the Lobula Giant Movement Detector (LGMD), a wide-field visual neuron that responds preferentially to the looming visual stimuli that is present in impending collisions. Models of the LGMD neuron have been proposed [36] which rely on a "critical race" in an array of photoreceptors between excitation due to changing illumination on photoreceptors, lateral inhibition and feedforward inhibition, to generate a response increasing with photoreceptor edge velocity. Analysis of the mathematical model underlying this neural network shows that the computation being performed is visual field integration of divergence for collision detection, which is tightly coupled with motor neurons to trigger a flight response. This shows that insects perform collision detection, not reconstruction. This model has been implemented on ground robots for experi-

mental validation [37][38][39], however the biophysical LGMD neural network model has been criticized for lack of experimental validation [40], and robotic experiments have shown results that do not currently live up to the robustness of insect vision, requiring significant parameter optimization and additional flow aggregation schemes for false alarm reduction [41][42]. While insect inspired vision is promising [43], experimental validation in ground robotics has shown that there are missing pieces. Specifically, Graham argues "[this model] ignores integration over the the visual field...how do inputs (to LGMD) become related to angular size and velocity [40]?" This *aggregation* or grouping of flow consistent with collision has shown to be a critical requirement to a successful model.

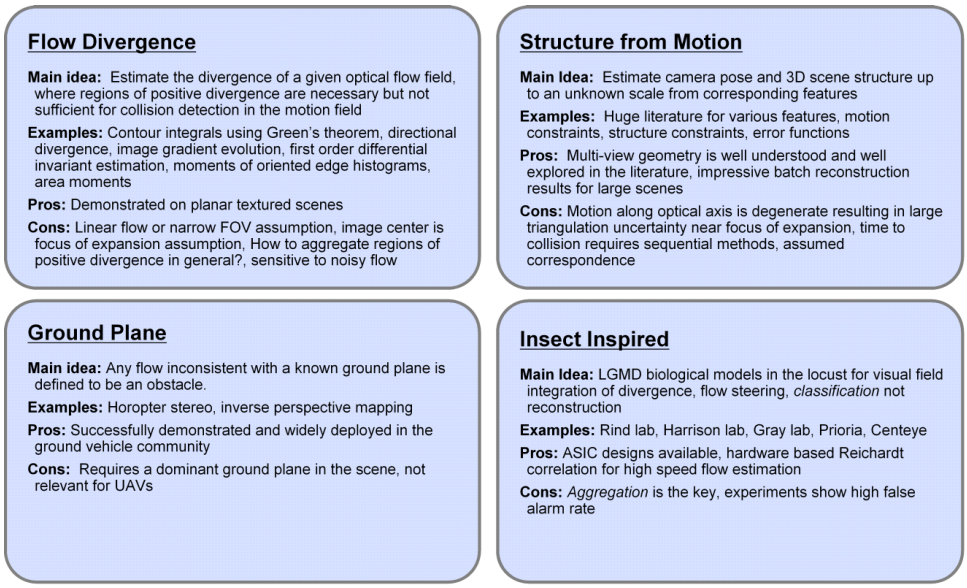


Figure 2: Time-to-collision estimation survey.

Figure 2 shows a summary of the different approaches to time-to-collision. This review shows that time to collision estimation methods for small UAVs re-

quire aggregation to determine obstacle boundaries and to handle false alarms from correspondence errors, while modelling the uncertainty in time to collision measurements from the degenerate motion along the optical axis.

One such approach to time-to-collision estimation which addresses these observations is *expansion segmentation* [44]. Expansion segmentation provides (i) a detection of significant "collision danger regions" of significant positive divergence in inertial aided video, and (ii) an estimate of maximum likelihood time-to-collision within the danger regions. Using the inertial velocity of the vehicle and the camera frame rate, the time to collision estimate can be used to recover range. Formally, expansion segmentation is the labeling of collision and non-collision nodes in a conditional Markov random field. The minimum energy binary labeling is determined in an expectation-maximization framework to iteratively estimate labeling using the mincut of an appropriately constructed affinity graph, and the parameterization of the joint probability distribution for time to collision and appearance. This joint probability provides a global model of the collision region, which can be used to estimate maximum likelihood time-to-collision over optical flow likelihoods, which is used to aid with local motion correspondence ambiguity [44].

3. Vision-based mapping for MAVs in the navigation frame

A depth map can be obtained from time-to-collision estimation and inertial measurements. In this section, we first use the depth map to obtain the range and bearing measurements in the navigation frame. Based on those measurements,

we estimate the range and bearing using the EKF and address the data association problem in the navigation frame using the nearest neighbor approach. We then create a map in the navigation frame using polar coordinates because this representation is ideally matched to camera information.

3.1. Range and bearing measurements

Time-to-collision estimation approach gives the range measurements to obstacles by multiplying time-to-collision by the groundspeed of the MAV, which is given by $v = V + v_w$, where V is the airspeed, v is the groundspeed, and v_w is the wind velocity relative to the inertial frame. Accordingly, a depth map, which represents the range and bearing to obstacles in the navigation frame, can be obtained. Figure 3 shows a 20×20 (in units of pixels) depth map of a simulated obstacle. In the depth map, the red pixels provide range measurements and their corresponding positions provide bearing measurements. For instance, the pixel at the jth row and the ith column provides the bearing to the obstacle given by

$$\begin{aligned} \eta &= tan^{-1}\left(\frac{i-\frac{r_x+1}{2}}{f}\right), \\ \theta &= tan^{-1}\left(\frac{j-\frac{r_y+1}{2}}{\sqrt{f^2+(i-\frac{r_x+1}{2})^2}}\right), \end{aligned} \tag{1}$$

where η and θ are the azimuth and elevation of pixel (i, j), r_x and r_y represent the number of pixels along the horizontal and vertical directions in the image plane, and f is the focal length in units of pixels. Note that Eq. (1) computes η and θ in the body frame. In order to use them in this paper, they need to be transformed to

the navigation frame which is the body frame with MAV roll and pitch removed.

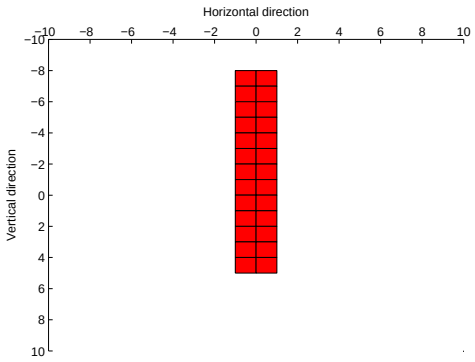


Figure 3: A 20×20 (in units of pixels) depth map. The red pixels give the range to the obstacle and their positions give the bearing to the obstacle in the body frame.

Since the measurement uncertainties produced by the camera increase as the distance from the MAV increases, we use the *sweet spot* measurement model with Gaussian distribution [45] to represent the time-to-collision uncertainties [44]. The observation made by the camera at time step k is given by the linear measurement model

$$\mathbf{z}_k = \mathbf{x}_k + \mathbf{v}_k, \tag{2}$$

where $\mathbf{x}_k = [r_k, \eta_k]^T$ is the state and the measurement noise $\mathbf{v}_k$ is a Gaussian random vector with zero mean. We assume that the covariance matrix $\mathbf{R}_k$ of $\mathbf{v}_k$ has a diagonal structure

$$\mathbf{R}_k = \begin{bmatrix} \sigma_{r_k}^2 & 0 \\ 0 & \sigma_{\eta_k}^2 \end{bmatrix}. \tag{3}$$

The variance of the range measurement $\sigma_{r_k}^2$ is a function $f_r(r_k)$ of the range r_k from the obstacle to the camera. The variance of the bearing measurement $\sigma_{\eta_k}^2$ is also a function $f_\eta(r_k)$ of the range. Accordingly, we use the *sweet spot* model

$$\begin{aligned} \sigma_{r_k}^2 &= f_r(r_k) = a_1(r_k - a_2)^2 + a_0, \\ \sigma_{\eta_k}^2 &= f_\eta(r_k) = \alpha f_r(r_k), \end{aligned} \tag{4}$$

where a_0, a_1, a_2 and α are model parameters. In this paper, we let $a_2 = 0$, which implies that the noise is at its minimum value at the origin of the navigation frame.

3.2. Range and bearing estimation using the EKF

Based on the measurements, we employ the EKF to estimate the range and bearing. Since the obstacle map is in the navigation frame of the MAV, which is located at the origin, we need to derive the equations of motion of each obstacle relative to the MAV. Figure 4 shows the motion of an obstacle relative to the MAV in the navigation frame, where r and η are the range and bearing to the obstacle and **B** is the location of the obstacle in the navigation frame. Based on Fig. 4,

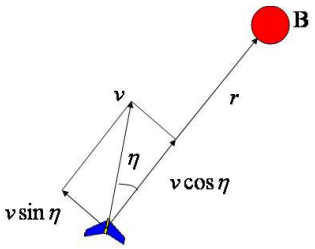


Figure 4: The motion of an obstacle relative to the MAV. The range and bearing to the obstacle are represented by r and η. The location of the obstacle is represented by **B**.

the equation of motion of the obstacle relative to the MAV in terms of range and

bearing is given by

$$\begin{aligned} \dot{r} &= -v\cos\eta, \\ \dot{\eta} &= \frac{v}{r}\sin\eta - \dot{\psi}, \end{aligned} \tag{5}$$

Let $\mathbf{x} = [r, \eta]^T$ be the state and let $\mathbf{f}(\mathbf{x})$ represent the right hand side of Eq. (5) and $\mathbf{h}(\mathbf{x}_k) = \mathbf{x}_k$. We rewrite Eq. (5) and Eq. (2) as the standard form of the state transition and observation models for the EKF

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}) + \mathbf{w}, \\ \mathbf{z}_k &= \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k, \end{aligned} \tag{6}$$

where $\mathbf{w}$ is the process noise that is assumed to be a Gaussian zero mean white noise with the covariance matrix $\mathbf{Q}$ and $\mathbf{v}_k$ is the measurement noise that is assumed to be a Gaussian zero mean white noise with the covariance matrix $\mathbf{R}_k$. Based on Eq. (6), the range and bearing are estimated using the standard continuous-discrete time EKF algorithm [46]. When measurements for a specific obstacle are not available, we only use the prediction model.

3.3. Data association

The camera observations must be associated with the existing obstacles in the map during the flight in order for the EKF to converge. In addition, the MAV may revisit obstacles that already exist in the map after an extended period of time. The challenge is to associate the new camera measurements with those obstacles.

These two problems motivate data association algorithms that relate sensor observations with the features included in the map. A data association algorithm is composed of two elements: a test to determine the compatibility between a sensor observation and a map feature, and a selection criterion to choose the best match among the set of possible matches [47]. Since the nearest neighbor algorithm has minimal computational complexity, we use it to solve the data association problem directly in the navigation frame, which saves the computational cost of transforming the data to the inertial frame.

The nearest neighbor algorithm is a classical technique in tracking problems [48]. The normalized squared innovation test is employed to determine compatibility, and the smallest Mahalanobis distance is used to select the best matchings. Suppose that a set of n obstacles $\{O_1, \cdots, O_n\}$ exist in the map at time step k. Let $\mathbf{x}_{ki} = [r_{ki}, \eta_{ki}]^T$ and $\hat{\mathbf{x}}_{ki} = [\hat{r}_{ki}, \hat{\eta}_{ki}]^T$, $i = 1, \cdots, n$ represent the actual and the estimated states of the obstacles respectively and let $\mathbf{P}_{ki}$ represent the covariance of the estimation error for the ith obstacle. Similarly, let $\mathbf{z}'_{kj} = \mathbf{h}(\mathbf{x}_{kj})$ and $\mathbf{z}_{kj} = \mathbf{h}(\mathbf{x}_{kj}) + \mathbf{v}_{kj}$, $j = 1, \cdots, m$ represent the theoretical and actual observations of a set of m measurements $\{M_1, \cdots, M_m\}$ and let $\mathbf{R}_{kj}$ be the covariance matrix of $\mathbf{v}_{kj}$. Let O_{i_j} represent the ith existing obstacle that corresponds to the jth measurement. The jth measurement M_j and its corresponding obstacle O_{i_j} at time step k are then related by the implicit measurement function [47]

$$\mathbf{g}(\mathbf{x}_{ki_j}, \mathbf{z}'_{kj}) = \mathbf{z}'_{kj} - \mathbf{h}(\mathbf{x}_{ki_j}) = \mathbf{z}'_{kj} - \mathbf{x}_{ki_j} = \mathbf{0}. \tag{7}$$

Note that the second equality of Eq. (7) results from the fact that $\mathbf{h}(\mathbf{x}_{ki_j}) = \mathbf{x}_{ki_j}$. Equation (7) states that the relative location between the theoretical observation of the measurement and the corresponding obstacle must be zero.

The nearest neighbor algorithm simply pairs each measurement with the obstacle considered most compatible according to Eq. (7). We rewrite the right hand side of Eq. (7) as

$$\begin{aligned} \mathbf{g}(\mathbf{x}_{ki_j}, \mathbf{z}_{kj}) &= \mathbf{z}_{kj} - \hat{\mathbf{x}}_{ki_j} + (\mathbf{z}'_{kj} - \mathbf{z}_{kj}) - (\mathbf{x}_{ki_j} - \hat{\mathbf{x}}_{ki_j}) \\ &= \lambda_{ki_jj} + (\mathbf{z}'_{kj} - \mathbf{z}_{kj}) - (\mathbf{x}_{ki_j} - \hat{\mathbf{x}}_{ki_j}), \end{aligned} \tag{8}$$

where $\lambda_{ki_jj} \stackrel{\triangle}{=} \mathbf{g}(\hat{\mathbf{x}}_{ki_j}, \mathbf{z}_{kj}) = \mathbf{z}_{kj} - \hat{\mathbf{x}}_{ki_j}$ represents the innovation of the pairing between the jth measurement M_j and its corresponding obstacle O_{i_j} at time step k and the covariance of λ_{ki_jj} can be obtained as

$$\begin{aligned} \mathbf{S}_{ki_jj} &= cov(\mathbf{z}'_{kj} - \mathbf{z}_{kj}) + cov(\mathbf{x}_{ki_j} - \hat{\mathbf{x}}_{ki_j}) \\ &= \mathbf{R}_{kj} + \mathbf{P}_{ki_j}. \end{aligned} \tag{9}$$

The compatibility between M_j and O_{i_j} is determined using the innovation test that measures the Mahalanobis distance as

$$D^2_{ki_jj} \stackrel{\triangle}{=} \lambda^T_{ki_jj} \mathbf{S}^{-1}_{ki_jj} \lambda_{ki_jj} < \chi^2_\beta, \tag{10}$$

where χ^2_β is the chi-square distribution with β degrees of freedom and β is the desired confidence level. This test determines a subset of the existing obstacles

that are compatible with the measurement M_j. The nearest neighbor selection criterion chooses the obstacle with the smallest Mahalanobis distance.

3.4. Mapping in the navigation frame using polar coordinates

For the local path planning problem, only the information about the local areas around the MAV is important. This observation motivates the use of on-board camera data to build a map directly in the navigation frame instead of the inertial frame. Accordingly, we save the computational resources of transforming the camera data from the navigation frame to the inertial frame, at the expense of updating the map from body motion. Since the information about the environment is obtained by on-board cameras, Cartesian coordinates are not the most efficient method of representing the information. We create the map using polar coordinates, which are more compatible with the range and bearing information obtained by the camera and allow the data to be processed more efficiently. Figure 5 shows the map in the navigation frame using polar coordinates. The origin of the map is the current location of the MAV. The circles represent the obstacles and the ellipses around them represent the range and bearing uncertainties. The map only stores obstacles at the range from the MAV less than $v * T_{ttc1}$, where v is the groundspeed. Obstacles at the range greater than $v * T_{ttc1}$ will be removed from the map and will be considered as new obstacles if they are observed again.

4. Path planning in the navigation frame

Given the map, a path can be planned for the MAV to maneuver to its goal location while avoiding the obstacles. In this section, we use the Rapidly-Exploring

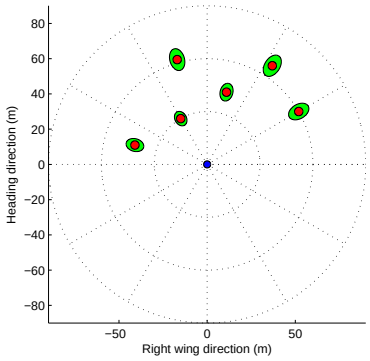


Figure 5: The map in the navigation frame of the MAV using polar coordinates. The origin of the map is the current location of the MAV. The circles represent the obstacles and the ellipses around them represent the range and bearing uncertainties.

Random Tree (RRT) algorithm to plan a collision-free path.

The RRT algorithm was initially developed in [49] and has been widely used in robot path planning. The RRT algorithm can quickly search the space of possible solutions by extending a tree in random directions in the configuration space as described in Algorithm 1. When the RRT algorithm is employed to plan a path for the MAV, the nodes of the tree are potential MAV waypoints and the branches are paths to the waypoints. The tree initially consists of the MAV's location as a single node. A random configuration is generated and the tree is extended toward this configuration, creating a new branch and node as outlined in Algorithm 1. When a path is found or a maximum number of iterations is reached, the RRT algorithm terminates.

The path generated by the RRT algorithm may include numerous extraneous configurations that may cause unnecessary maneuvering, and should therefore be smoothed. We design a fast and effective configuration elimination algorithm to

Algorithm 1: Rapidly-Exploring Random Tree algorithm

```
1 Initialize a tree containing one node - the current configuration of the MAV;
2 while A path has not been found and the maximum number of iterations is
  not reached do
3 |   rand ← A random MAV configuration;
4 |   near ← The configuration in the tree that is closest to rand;
5 |   if the tree can be connected from near to rand without collision then
6 |   |   Extend the tree from near to rand;
7 |   end
8 end
```

smooth the path. This algorithm walks through the path from the beginning to end nodes, eliminating unnecessary nodes along the way. It initially looks at the first node and tries to find the last node in the path to which the first node can be directly connected without collision. It is guaranteed to at least connect to one node in the graph. If the only node that can be connected is the next node in the path, the algorithm moves forward one step and tries to connect the second node to the last possible node. If the first node can be connected to any other node, all intermediate nodes are eliminated and the algorithm moves forward to the connecting node and repeats the process until a connection to the end node is found. Using this algorithm, all intermediate nodes that can be skipped without causing any collision are eliminated from the path.

The RRT algorithm requires computation time to generate paths. The MAV may still collide with new obstacles while the RRT algorithm is in the middle of execution. We use the following scheme to address this problem. Suppose that the objective of the proposed algorithm is to find a collision-free path from x_{init} to

x_{goal}, where x_{init} and x_{goal} represent the initial and goal configurations. Define the function $W_t = waypointRRT\left(x_t, x_{goal}, M_t\right)$ to represent the RRT algorithm, where x_t is the initial configuration for the RRT algorithm at time step t, M_t is the navigation frame map, and $W_t = \left[\begin{array}{cccc} W_{t,1}, & W_{t,2}, & \cdots, & W_{t,n_t} \end{array}\right]$ is the waypoint path planned by the RRT algorithm, where n_t is the number of the waypoints. Let the length of each path segment for the RRT algorithm $\|W_{t,i+1} - W_{t,i}\|_2 = l, (i = 1, \cdots, n_t - 1)$, where l is chosen such that $2v*T > l \geq v*T$, where v is the groundspeed of the MAV and T is the computation time of the RRT algorithm. At time step $t = 0$, choose a configuration x_0 such that collisions do not exist from x_{init} to x_0 and $\|x_{init} - x_0\|_2 = l$. The function $waypointRRT\left(x_0, x_{goal}, M_0\right)$ is then executed based on the map M_0 and the MAV is maneuvered to x_0 simultaneously. Since $l \geq v*T$, the execution of the function $waypointRRT\left(x_0, x_{goal}, M_0\right)$ will finish and a waypoint path W_0 will be generated before the MAV reaches x_0. When the MAV reaches x_0 at time step $t = t_1$, let $x_{t_1} = W_{0,2}$. The MAV is maneuvered to x_{t_1}. The function $waypointRRT\left(x_{t_1}, x_{goal}, M_{t_1}\right)$ is simultaneously executed to generate a waypoint path W_{t_1} based on the map M_{t_1} if collisions exist. Otherwise, let $W_{t_1} = \left[\begin{array}{cccc} W_{0,2}, & W_{0,3}, & \cdots, & W_{0,n_0} \end{array}\right]$. Repeat this process until the MAV reaches the goal. Finally, a collision-free waypoint path $\left[\begin{array}{ccccccc} x_{init}, & x_0, & x_{t_1}, & x_{t_2}, & \cdots, & x_{goal} \end{array}\right]$ from x_{init} to x_{goal} is generated.

5. Simulation

The feasibility of the mapping and collision avoidance algorithms was tested using a simulation environment developed in MATLAB/SIMULINK. This sim-

ulator uses a six degree-of-freedom model for the aircraft, where a North-East-Down (NED) coordinate system is used. The algorithm is tested on two scenarios as described below.

5.1. Scenario I

In the first scenario, the MAV was commanded to maneuver through twenty-five $20m \times 20m \times 100m$ obstacles between waypoint **S** (0,100,-40) and waypoint **E** (600,700,-40). Figure 6 shows the terrain for the first scenario, where the square sign represents waypoint **S** and the plus sign represents waypoint **E**. A 20×20 pixel depth map was used. The parameters for the sweet spot measurement model were set at $a_0 = 0.1528$, $a_1 = 0.004$, $a_2 = 0$ and $\alpha = 0.00002$. The covariance matrix of the process noise was $Q = \begin{bmatrix} 10 & 0 \\ 0 & 0.0076 \end{bmatrix}$ and the groundspeed was $v = 10m/s$.

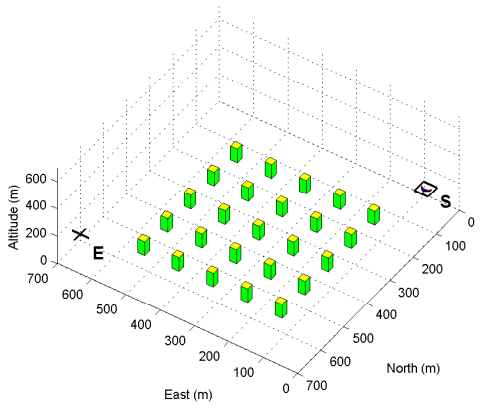
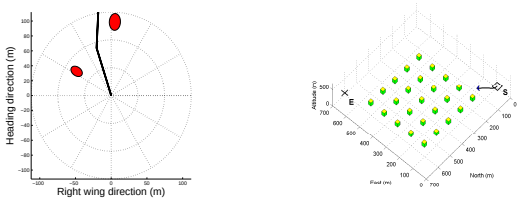


Figure 6: The terrain for the first simulation scenario. The MAV is maneuvered through twenty-five obstacles between waypoint **S** (0,100,-40) and waypoint **E** (600,700,-40), which are represented by the square and plus signs respectively.

Figure 7 shows the update of the map and the evolution of the corresponding path as obstacles are observed. Subfigures on the left show the navigation frame maps and paths based on the available information about the obstacles at different time. The ellipses represent the estimated locations and sizes of the obstacles. The plus sign in subfigure (i) represents the waypoint **E** in the navigation frame. Subfigures on the right show the inertial paths. The final path followed by the MAV between the two waypoints is shown in Figure 8. Figure 9 shows the tracking error for the range and bearing to the obstacle at (150,250,-40) using the EKF, where the solid lines represent the tracking error and the dashed lines represent the positive and negative square roots of the error covariance.

5.2. Scenario II

In the second scenario, the MAV was commanded to maneuver through a simulated city called "Megacity" for which ground truth range was available. This path followed waypoint **S** (-250,-400,-60) to waypoint **E** (250,150,-60) as shown in Figure 10. A 640×480 pixel depth map was used. The parameters for the sweet spot measurement model were set at $a_0 = 0.1528$, $a_1 = 0.001$, $a_2 = 0$ and $\alpha = 0.00002$. The covariance matrix of the process noise was $Q = \begin{bmatrix} 10 & 0 \\ 0 & 0.0076 \end{bmatrix}$ and the groundspeed was $v = 10m/s$. Figure 11 shows the maps and the corresponding collision-free paths as the MAV maneuvered through the megacity terrain. Subfigures in the first and second columns show the camera views and the depth maps at different time steps. Subfigures in the third column show the update of the map and the evolution of the path in the navigation frame. Subfigures in the



(a) $t = 9s$ (b) $t = 9s$

(c) $t = 18s$ (d) $t = 18s$

(e) $t = 25s$ (f) $t = 25s$

(g) $t = 38s$ (h) $t = 38s$

(i) $t = 62s$ (j) $t = 62s$

Figure 7: The update of the map and the evolution of the path in the first simulation scenario. Subfigures on the left show the navigation frame maps and paths based on the available information about the obstacles at different time. The ellipses represent the estimated locations and sizes of the obstacles and the plus sign in subfigure (i) represents the waypoint **E** in the navigation frame. Subfigures on the right show the inertial paths followed by the MAV.

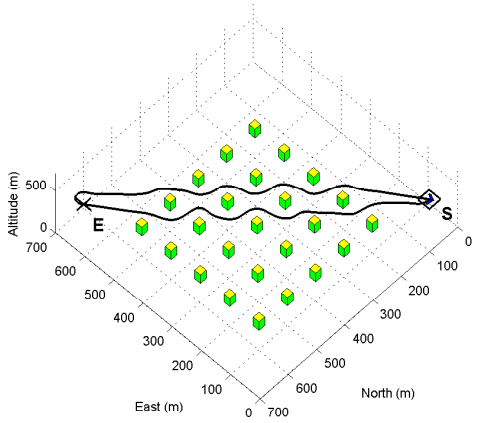


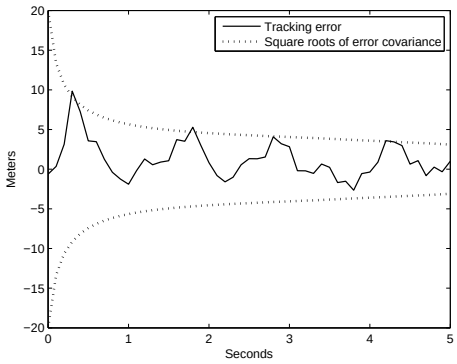
Figure 8: The final path followed by the MAV between the waypoints **S** and **E**.

fourth column show the inertial path followed by the MAV.

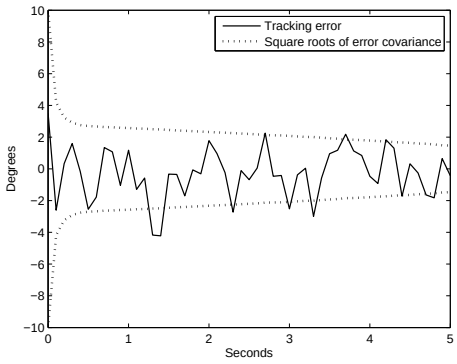
6. Flight tests

6.1. Experimental testbed

Figure 12 shows the main components of the experimental testbed. Figure 12(a) shows the Kestrel autopilot designed by Brigham Young University (BYU) and commercialized by Procerus Technologies [50]. It is equipped with 3400 29-MHz microprocessor, rate gyroscopes, accelerometers, and absolute and differential pressure sensors. The autopilot measures $2.0 \times 1.37 \times 0.47$ inches and weighs 16.65 grams, making it ideal for miniature air vehicles. Figure 12(b) shows the airframe used in the flight test. This airframe is a 48 inches wingspan XS EPP foam flying wing selected for its durability, ease of component installation, and flying characteristics. Embedded in the airframe are the Kestrel autopilot, bat-

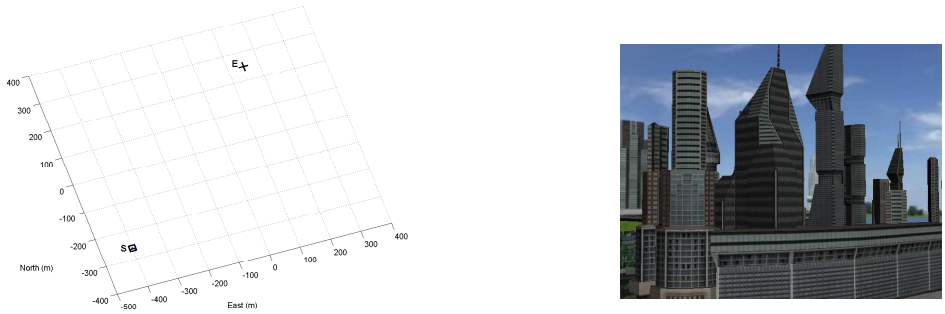


(a) The range tracking error



(b) The bearing tracking error

Figure 9: The tracking error for the range and bearing to the obstacle at (150,250,-40) using the EKF. The solid lines represent the tracking error and the dashed lines represent the positive and negative square roots of the error covariance.



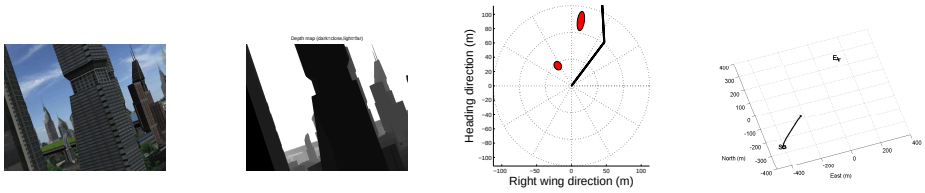
(a) (b)

Figure 10: The megacity terrain for the second simulation scenario. The square and cross signs in subfigure (a) represent waypoints **S** and **E** respectively. Subfigure (b) shows the megacity terrain.

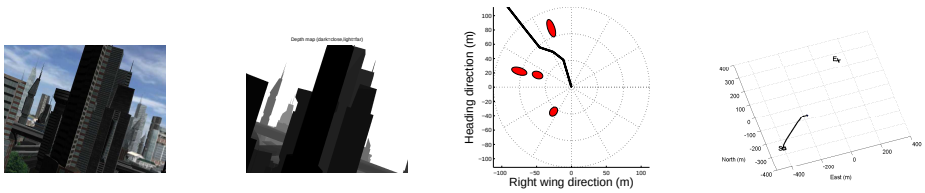
teries, a 1000 mW 900 MHz radio modern, a GPS receiver, a video transmitter, and a small analog camera. Figure 12(c) shows the ground station components. A laptop runs the Virtual Cockpit ground-control software and interfaces with the MAV through a communication box. Video is transmitted to ground via a 2.4 GHz analog transmitter and captured on the laptop using an Imperx VCE-PRO PCMCIA frame grabber, which provides 640×480 images at 30 frames per second. An RC transmitter is used as a stand-by fail-safe mechanism to facilitate safe operation. Figure 13 shows the flowchart describing the interaction of the hardware and software components used in the flight test.

6.2. Flight test results

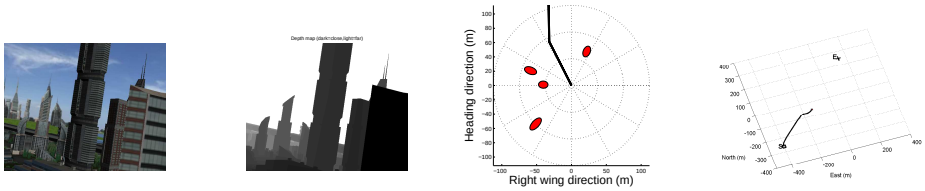
We conducted flight tests to verify the feasibility of the proposed vision based collision avoidance algorithm, which was used to avoid two large red targets, as shown in Figure 14. Information about the locations and dimensions of the targets was not provided to the MAV, rather color segmentation was used as a tempo-



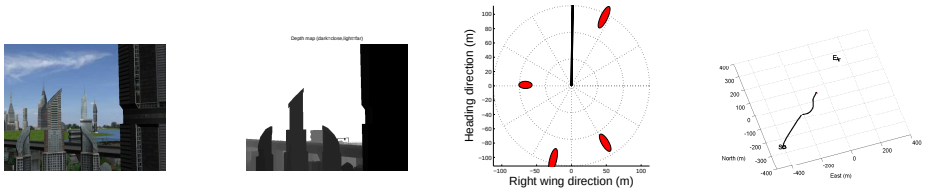
(a) t=24s



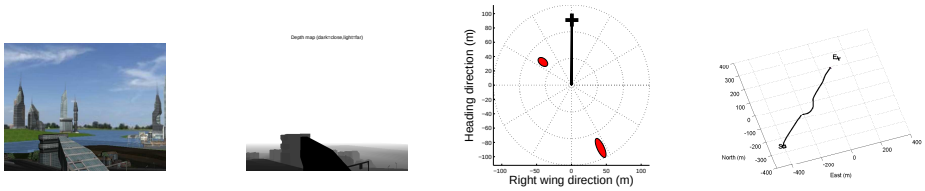
(b) t=28s



(c) t=32s

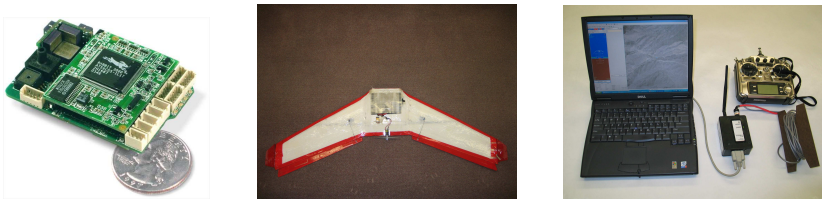


(d) t=44s



(e) t=62s

Figure 11: The update of the map and the evolution of the path in the second simulation scenario. Subfigures in the first and second columns show the camera views and the depth maps at different time steps. Subfigures in the third column show the update of the map and the evolution of the path in the navigation frame. Subfigures in the fourth column show the inertial path followed by the MAV.



(a) (b) (c)

Figure 12: The hardware testbed. Subfigure (a) shows the Kestrel autopilot, subfigure (b) shows the airframe, and subfigure (c) shows the ground-station components.

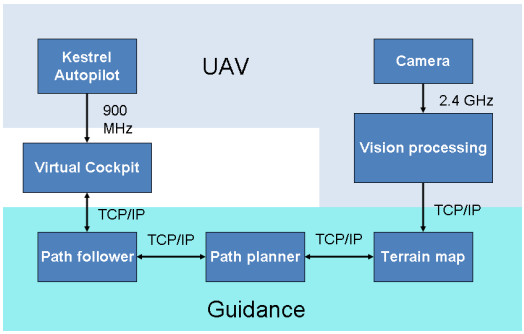


Figure 13: The flowchart describing the layout of the basic hardware and software components used in the flight test.

rary surrogate for time-to-collision estimation to test the mapping and avoidance algorithms. The groundspeed was $v = 11m/s$. During the flight, the guidance strategy generated roll commands on the ground station and transmitted them to the autopilot. Telemetry information was recorded on the ground station.

Figure 15 shows the telemetry plot of the MAV avoiding the two targets, where the rectangles represent the targets, the dotted line represents the flight path and the solid line represents the originally commanded but conflicted waypoint path. As the MAV approached the targets, the proposed path planning algorithm generated a path around the targets and the MAV began to track the generated path. As the MAV passed the targets, it once again began to track the original waypoint

Figure 14: The two large red targets used to verify the vision based collision avoidance algorithm described in this paper.

path. As shown in Figure 15, the MAV successfully avoided the targets without human intervention. Figure 16 shows the images of the targets as captured by the MAV as it executes the avoidance maneuver. The range and azimuth tracking errors using the EKF during the time that the targets are in the field of view of the camera are shown in Figure 17. These tracking errors converge to small steady errors as the MAV approaches the targets.

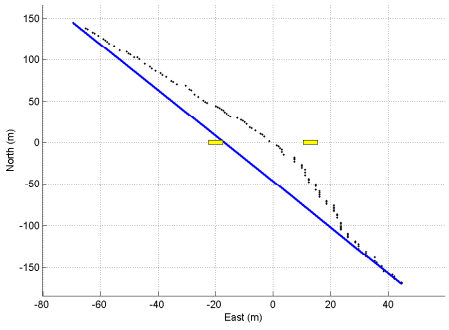
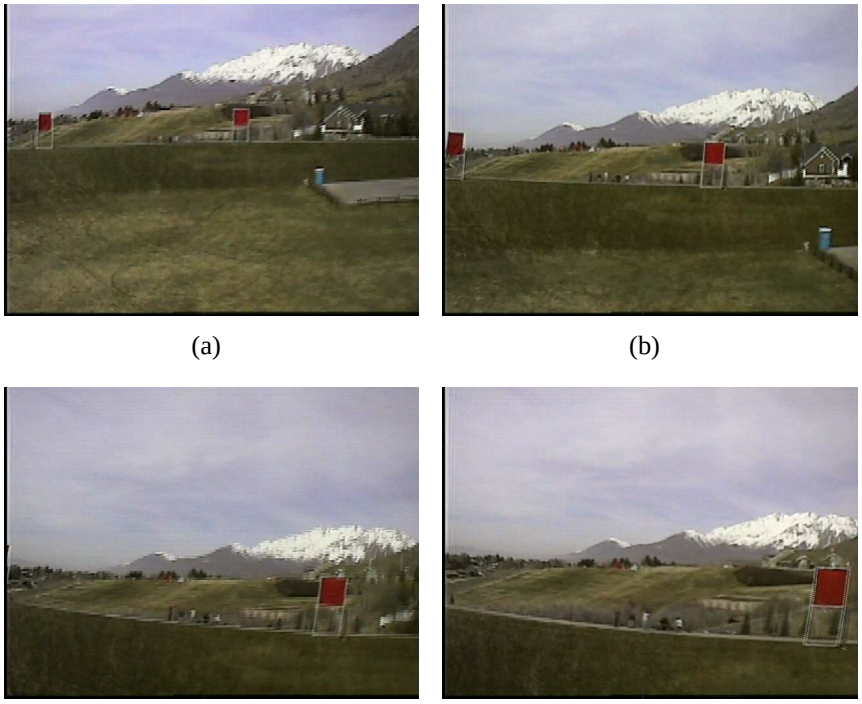


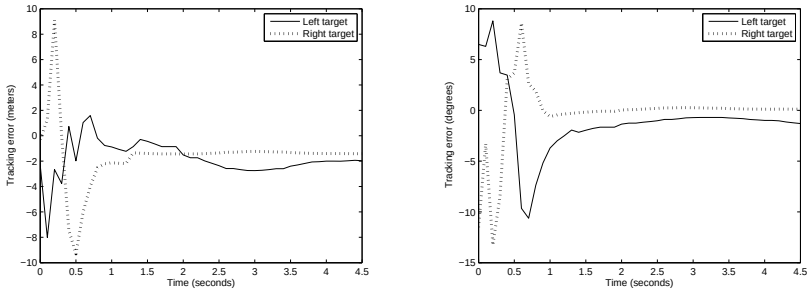
Figure 15: The telemetry plot of the MAV avoiding two targets. The rectangles represent the targets. The dotted line represents the flight path and the solid line represents the commanded but conflicted waypoint path.



(a) (b)

(c) (d)

Figure 16: The images of the targets from the MAV as it executes the avoidance maneuver.



(a) Range tracking error (b) Azimuth tracking error

Figure 17: The range and azimuth tracking errors using the EKF during the time that the targets are in the field of view of the camera.

7. Conclusions and future work

In this paper we present a vision-based navigation frame mapping and path planning technique for MAVs operating in unknown environments. We create a map in the navigation frame using polar coordinates without transforming the camera data to the inertial frame. The EKF is used to estimate the range and bearing to obstacles and to address the measurement uncertainties. The data association problem is solved in the navigation frame using the nearest neighbor approach and the path is planned in the navigation frame using the RRT algorithm. The simulation and flight test results show that the proposed technique is successful in solving path planning and multiple obstacles avoidance problems for MAVs operating in unknown environments.

In this work, polar coordinates are used to create the map for addressing the two dimensional path planning problem. In the future, we will use spherical coordinates to create maps that address the three dimensional path planning problem. In addition, we will validate the expansion segmentation methods for vision based

time-to-collision estimation with the mapping and planning algorithms.

8. Acknowledgement

This research was supported by the Air Force Research Laboratory, Munitions Directorate under SBIR contract No. FA 8651-07-c-0094 to Scientific Systems Company, Inc. and Brigham Young University. We acknowledge the assistance of Jeffrey Saunder in obtaining the flight test results.

References

[1] Brandon R. Call. Obstacle avoidance for unmanned air vehicle using computer vision. Master's thesis, Brigham Young University, December, 2006.

[2] Andrew Curtis. Path planning for unmanned air and ground vehicles in urban environments. Master's thesis, Brigham Young University, 2008.

[3] E. Frazzoli, M. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. *Journal of Guidance, Control and Dynamics*, 25:116–129, Jan.-Feb. 2002.

[4] Y. Watanabe, E.N. Johnson, and A.J. Calise. Vision-based approach to obstacle avoidance. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, August 2005.

[5] A. Pongpunwattana and R. Rysdyk. Real-time planning for multiple autonomous vehicles in dynamics uncertain environments. *AIAA Journal of*

Aerospace Computing, Information, and Communication, 1:580–604, December 2004.

[6] J.C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[7] Jeffrey Saunders, Randal Beard, and Jeffrey Byrne. Vision-based reactive multiple obstacle avoidance for micro air vehicles. In *Proceedings of American Control Conference*, June 2009.

[8] N.M. Amato and Y. Wu. A randomized roadmap method for path and manipulation planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 113–120, Minneapolis, MN, 1996.

[9] L.E. Kavraki, P. Svestka, J. C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transaction on Robotics and Automation*, 12(4):566–580, 1996.

[10] L.E. Kavraki, M.N. Koloumtzakis, and J.C. Latombe. Analysis of probabilistic roadmaps for path planning. *IEEE Transaction on Robotics and Automation*, 14(1):166–171, 1998.

[11] Peng Cheng, Zuojun Shen, and Steven M. LaValle. Using randomization to find and optimize trajectories for nonlinear systems. In *Proceedings of Annual Allerton Conference on Communications, Control, Computing*, 2000.

[12] J. Barraquand and J.C. Latombe. Robot motion planning: a distributed

representation approach. *The International Journal of Robotics Research*, 10(6):628–649, 1991.

[13] Jerome Barraquand, Bruno Langlois, and Jean Claude Latombe. Numerical potential field techniques for robot path planning. *IEEE Transaction On Systems, Man, and Cybernetics*, 22(2):224–241, March/April 1992.

[14] T. Lozano-Perez. Automation planning of manipulator transfer movements. *IEEE Transaction Systems, Man and Cybernetics*, 11(10):681–698, 1981.

[15] David Zhu and Jean Claude Latombe. New heuristic algorithms for efficient hierarchical path planning. *IEEE Transaction on Robotics and Automation*, 7(1):9–20, 1991.

[16] P. Tsiotras and E. Bakolas. A hierarchical on-line path-planning scheme using wavelets. In *European Control Conference*, Kos, Greece, July 2-5 2007.

[17] R. Cowlagi and P. Tsiotras. Multiresolution path planning with wavelets: a local replanning approach. In *IEEE American Control Conference*, June 2008.

[18] E. Bakolas and P. Tsiotras. Multiresolution path planning via sector decompositions compatible to on-board sensor data. In *Proceedings of AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, Hawaii, August 2008.

[19] J. Oliensis. A critique of structure-from-motion algorithms. *Computer Vision and Image Understanding (CVIU)*, 80(2):172–214, 2000.

[20] Omead Amidi Takeo Kanade and Qifa Ke. Real-time and 3D vision for autonomous small and micro air vehicles. In *IEEE Conf. on Decision and Control (CDC 2004)*, pages 1655–1662, December 2004.

[21] James F. Montgomery Srikanth Saripalli and Gaurav Sukhatme. Vision-based autonomous landing of an unmanned aerial vehicle. In *IEEE International Conference on Robotics and Automation*, pages 2799–2804, 2002.

[22] Omid Shakernia, Rene Vidal, Courtney S. Sharpy, Yi Ma, and Shankar Sastry. Multiple view motion estimation and control for landing an unmanned aerial vehicle. In *IEEE Conference on Robotics and Automation*, 2002.

[23] J. Santos-Victor and G. Sandini. Uncalibrated obstacle detection using normal flow. *Machine Vision and Applications*, 9(3):130–137, 1996.

[24] R.A. Brooks L.M. Lorigo and W.E.L. Grimson. Visually-guided obstacle avoidance in unstructured environments. In *Proceedings of IROS '97*, page 373379, Grenoble, France, September 1997.

[25] R. Nelson and Y. Aloimonos. Obstacle avoidance using flow field divergence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(10):1102–1106, October 1989.

[26] T. Poggio, A. Verri, and V. Torre. Green theorems and qualitative properties of the optical flow. Technical Report 1289, MIT, 1991.

[27] N. Ancona and T. Poggio. Optical flow from 1-D correlation: Application to a simple time-to-crash detector. *Int. J. Computer Vision*, 14(2), 1995.

[28] A. Rosenfeld Z. Duric and J. Duncan. The applicability of green's theorem to computation of rate of approach. *International Journal of Computer Vision*, 31(1):83–98, 1999.

[29] S.J. Maybank. Apparent area of a rigid moving body. *Image and Vision Computing*, 5(2):111–113, 1987.

[30] T. Camus. Calculating time-to-contact using real-time quantized optical flow. Technical Report NISTIR 5609, National Institute of Standards and Technology, March 1995.

[31] T.A. Camus, D. Coombs, M. Herman, and T.H. Hong. Real-time single-workstation obstacle avoidance using only wide-field flow divergence. *Videre: A Journal of Computer Vision Research*, 1(3), 1999.

[32] H. Liu, M. Herman, R. Chellappa, and T.H. Hong. Image gradient evolution: A visual cue for collision avoidance. In *Proceedings of the International Conference on Pattern Recognition*, 1996.

[33] R. Cipolla and A. Blake. Image divergence and deformation from closed curves. *International Journal of Robotics Research*, 16(1):77–96, 1997.

[34] A. Verri and T. Poggio. Motion field and optical flow: Qualitative properties. *Pattern Analysis and Machine Intelligence*, 11(5):490–498, 1989.

[35] N. Hatsopoulos, F. Gabbiani, and G. Laurent. Elementary computation of object approach by a wide-field visual neuron. *Science*, 270:1000–1003, November 1995.

[36] F.C. Rind and D.I. Bramwell. Neural network based on the input organization of an identified neuron signaling impending collision. *Journal of Neurophysiology*, 75:967 985, 1996.

[37] Sergi Bermudez i Badia. A fly-locust based neuronal control system applied to an unmanned aerial vehicle: the invertebrate neuronal principles for course stabilization, altitude control and collision avoidance. *The International Journal of Robotics Research*, 26(7):759–772, 2007.

[38] Shigang Yue and F.C. Rind. A collision detection system for a mobile robot inspired by the locust visual system. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA05)*, pages 3832–3837, April 2005.

[39] Hirotsugu Okuno and Tetsuya Yagi. Real-time robot vision for collision avoidance inspired by neuronal circuits of insects. In *IROS'07*, pages 1302–1307, San Diego, CA, October 2007.

[40] Lyle Graham. How not to get caught. *Nature Neuroscience*, 5:1256 – 1257, 2002.

[41] Shigang Yue and F.C. Rind. Collision detection in complex dynamic scenes

using an LGMD-based visual neural network with feature enhancement. *IEEE transactions on neural networks*, 17(3):705–716, 2006.

[42] F.C. Rind Shigang Yue. Visual motion pattern extraction and fusion for collision detection in complex dynamic scenes. *Computer Vision and Image Understanding*, 104(1):48–60, October 2006.

[43] Egelhaaf and Kern. Vision in flying insects. *Curr. Opin. Neurobiol.*, 12:699–706, 2002.

[44] J. Byrne and C.J. Taylor. Expansion segmentation for visual collision detection and estimation. In *IEEE International Conference on Robotics and Automation (ICRA'09)*, 2009.

[45] R. A. Freeman P. Yang and K. M. Lynch. Multi-agent coordination by decentralized estimation and control. *IEEE Transaction on Automatic Control*, 2008.

[46] Frank L. Lewis. *Optimal Estimation: With An Introduction To Stochastic*. New York: Wiley, 1986.

[47] Jose Neira and Juan D. Tardos. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17, no.6:890–897, December 2001.

[48] Y. Bar-Shalom and T.E. Fortmann. *Tracking and Data Association*. Boston, MA: Academic, 1988.

[49] S. M. LaValle. Rapidly-exploring random trees: A new tool for path planning". Technical report, Computer Science Dept, Iowa State University, Tech. Rep. TR 98-11, August 1998.

[50] Procerus technologies: Fly light with the world's smallest uav autopilot. http://www.procerusuav.com/, August 2006.