Local Terrain Mapping for Obstacle Avoidance using Monocular Vision

Sean Quinn Marlow

Graduate Research Assistant, Aerospace Engineering

Jack W. Langelaan

Assistant Professor, Aerospace Engineering

The Pennsylvania State University

University Park, PA 16802

Abstract

We present a method for navigation of a small unmanned rotorcraft through an unsurveyed environment consisting of forest and urban canyons. Optical flow measurements obtained from a vision system are fused with measurements of vehicle velocity to compute estimates of range to obstacles. These estimates are used to populate a local occupancy grid which is fixed to the vehicle. This local occupancy grid allows modeling of complex environments and is suitable for use by generic trajectory planners. Results of simulations in a two-dimensional environment using a potential field obstacle avoidance routine are presented.

Introduction

Currently, many unmanned aerial vehicles (UAVs) operate at high altitudes where the region is free of obstacles. However, this limits the tasks which can be performed. Missions envisioned for small UAVs now require low altitude flights among many obstacles (e.g. search and rescue in forests or surveillance in urban canyons). The Department of Defense (DoD) lists reconnaissance as the number one priority for

Presented at the 2009 AHS Unmanned Vehicles Specialist's Forum

all classes of unmanned systems and specifies passive detection as a goal for all unmanned systems [15]. Current technologies for detecting obstacles rely heavily on LIDAR and RADAR, large active sensors. In addition to the restrictions imposed by passive sensing, navigating small vehicles in confined areas adds significant complications: vehicle performance requirements are very stringent due to requirements imposed by obstacle avoidance; and sensing payloads are restricted in both weight and power requirements.

This paper is concerned with obstacle avoidance for small autonomous rotorcraft operating in complex, cluttered, unsurveyed environments. The primary focus is on generating a local map of the environment which is suitable for use with generic control and planning algorithms.

With the advent of low-cost, light-weight and low power CCD cameras, the use of vision systems for obstacle avoidance has become an active field of research. In addition to low power requirements and light weight, vision sensors are passive, reducing the probability of detection. Vision based techniques such as structure from motion seek to build a three-dimensional model of the surrounding environment using known motion of a monocular camera (e.g. [10]) but this is typically formulated as a batch process and is thus not suited for real-time implementation. Feature-based techniques such as Simultaneous Localization and Mapping (which have the advantage of *not* requiring the availability of camera motion measurements through external means such as GPS) quickly become intractable in large environments or environments where obstacles are difficult to define by features [7, 8].

Optical flow has been used for obstacle avoidance or ground speed estimation by several researchers (e.g. [9, 11]). However, direct reliance on measurements of optical flow for obstacle avoidance results in low robustness to noise and sensor dropouts. Using measurements to generate a map of the environment can greatly improve performance. Further, this map can be shared if a flock of UAVs conducts a cooperative mission. Here we generate a local map by fusing measurements of optical flow obtained from a vision system with measurements of vehicle velocity from GPS using an occupancy grid [3].

This paper describes the procedure for generating the local map and combines the local map with a control algorithm based on a potential field approach [5]. To demonstrate the effectiveness of this approach we present results of simulations of navigation through a two-dimensional environment consisting of a forest and an urban area based on the McKenna Military Operations in Urban Terrain (MOUT) site at Fort Benning, Georgia.

Related Work

Most unmanned vehicle systems which map the surrounding terrain use LIDAR and RADAR to detect obstacles. Stanley, the vehicle that won the DARPA Grand Challenge, used the measurements of five scanning laser range finders to create an occupancy grid. The use of cameras was limited to color and texture matching (e.g. finding the color and texture of a dirt road instead of vegetation) [14]. Scherer et. al. recently successfully flew an autonomous helicopter through the McKenna MOUT site at Ft. Benning, GA [12]. Their helicopter used a LIDAR system to create a map of the surroundings and IMU and differential GPS measurements to estimate the helicopter state. The use of LIDAR provides a near perfect map of the surroundings (able to detect a 6mm wire from 38m away) which greatly assists navigation but comes at the high cost of power, weight, and electromagnetic emissions.

Vision based estimation methods have been popular recently due to low power and weight requirements. Hrabar et al. have fused optical flow and stereo vision measurements on both a tractor and unmanned helicopter to fly in urban canyons [4]. While the fusion system worked well, optical flow measurements could keep the tractor centered in a corridor but was less effective in navigating turns

Kim and Brambley proposed a system to hold a constant altitude by fusing two optical flow measurements from optical mouse sensors in an extended Kalman filter [6]. With dual optical flow, they are able to estimate both velocity and distance to ground. However, they make use of a terrain map to predict optical flow measurements. Chahl and Mizutani also propose an optical flow method for ground avoidance [2]. Using one camera to measure optical flow at each pixel, they generate an elevation map of the terrain ahead. Zufferey and Floreano also use 1-D cameras for optical flow measurements to turn away from textured walls [16].

These approaches did not seek to generate a local map, rather, they used optical flow directly to compute a control input. Braillon et al. use stereo and optical flow to populate an occupancy grid representation of the local environment, but their approach requires identification of a ground plane [1].

Problem Statement

The situation considered here is an aircraft flying through an unsurveyed obstacle field consisting of small, convex obstacles (such as tree trunks) and large, potentially non-convex obstacles such as buildings (Fig. 1). An on-board camera obtains measurements of bearing and optical flow while GPS provides measurements of velocity and heading.

The aircraft is located at position x, y in an Earth-fixed frame. Coordinate frame O (defined by x_o and y_o) translates with the aircraft (keeping the CG of the aircraft at the origin of frame O), but holds a constant North-East orientation. The orientation of the body frame B (defined by x_b and y_b) is defined by the heading angle ψ, and body-frame velocities are defined by u and v.

The problem is to estimate the location of obstacles and reach the goal while avoiding collisions with obstacles. As vehicle states are obtain using GPS the problem essentially becomes that of mapping and obstacle avoidance. As this problem involves navigating through any unsurveyed terrain, obstacles could be large or small.

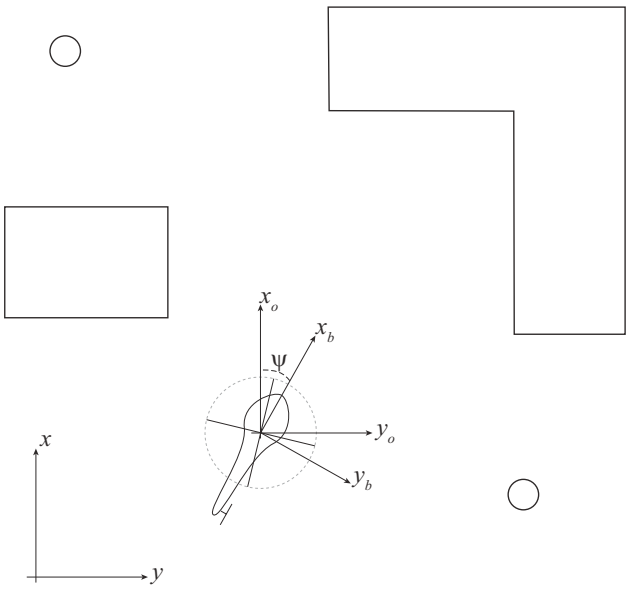


Fig. 1 Navigation/avoidance scenario. The autonomous rotor craft must fly to a goal (not shown) while avoiding small, convex obstacles (e.g. tree trunks) as well as large, potentially non-convex obstacles such as buildings.

The rotational freedom of the vehicle introduces a non-linearity to the problem through the kinematic model. Additionally, the projection of the three-dimensional world onto the two-dimensional image plane and then conversion to bearings and optic flow also creates non-linearities in the sensor model. These both complicate the problem of mapping.

Information about obstacles is only available from measurements of bearing and optical flow, thus camera motion (aircraft motion) is essential. Unfortunately, obstacles directly in the path of motion (which

we would like to avoid) generate almost no optical flow and no information for a range estimate; transverse motion is required to produce a useful estimate of obstacle location. This transverse motion does provide the benefit of ensuring that collision is avoided.

The techniques described here to address these problems are applicable to full three dimensional, six degree of freedom flight. Here we consider flight in a two dimensional environment.

System Description

The block diagram in Fig. 2 shows a system that uses the given sensors (GPS and a monocular camera) to perform obstacle avoidance. The GPS sensor outputs estimates of vehicle states, $\hat{\mathbf{x}}_v$ (velocities and heading of the vehicle). The vehicle state estimates and the camera measurements $\mathbf{z}_{cam}$ (bearings and bearing rates to obstacles) are fused in the estimator which computes estimates of obstacle ranges along with the associated covariances. These estimates are then integrated into the occupancy grid which is given to the heading generator, which computes a desired heading ψ_{des} (steering away from obstacles and trying to get to the goal). The difference between the desired heading and the current estimated heading is used by the flight controller to generate control inputs.

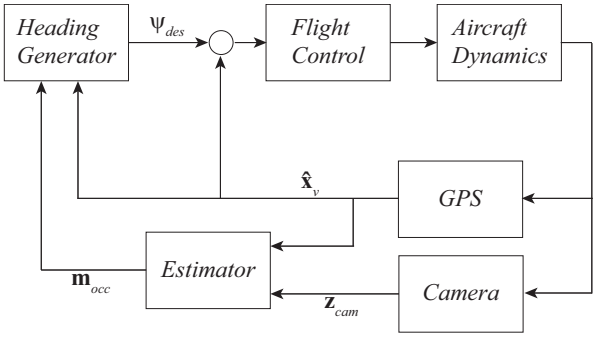


Fig. 2 System block diagram

It is assumed that the flight control system is able to maintain stable, controlled flight.

Kinematic Model

Vehicle rotation is expressed as angle ψ relative to frame O. Velocities u and v are expressed in the body frame b. The control inputs are $\dot{u}$, $\dot{v}$, and $\dot{\psi}$.

$$\dot{x} = u\cos\psi - v\sin\psi \tag{1}$$

$$\dot{y} = u\sin\psi + v\cos\psi \tag{2}$$

$$\dot{\psi} = \omega + \mathcal{N}(0, \sigma_\omega^2) \tag{3}$$

$$\dot{u} = a_x + \mathcal{N}(0, \sigma_{a_x}^2) \tag{4}$$

$$\dot{v} = a_y + \mathcal{N}(0, \sigma_{a_y}^2) \tag{5}$$

Here $\mathcal{N}(0, \sigma^2)$ denotes a Gaussian random variable with covariance σ^2.

Sensor Model

The camera is fixed to the aircraft looking towards the front of the vehicle (along the x_b axis). The camera obtains measurement of optical flow (bearing rates) along a set of bearings $[\beta_1, \beta_2, ...\beta_n]$, so that the optical flow along the i^{th} bearing is modeled by Eq. (7). This is equivalent to having an array of optical flow sensors, the i^{th} pointing along β_i.

$$\beta_i = \arctan\frac{y_{i,O}}{x_{i,O}} - \psi \tag{6}$$

$$\dot{\beta}_i = \frac{u\sin\beta_i}{r_i} - \frac{v\cos\beta_i}{r_i} - \dot{\psi} + \mathcal{N}(0, \sigma^2) \tag{7}$$

Where $x_{i,O}$ and $y_{i,O}$ are the coordinates of the obstacle nearest the vehicle in the i^{th} "bin" (defined by $\beta_i \pm \frac{\Delta\beta}{2}$), expressed in frame O and r_i is the distance between the camera and the obstacle.

In addition to range, vehicle velocity $[u,\ v]$ and turn rate $\dot{\psi}$ affect the optical flow measurement. If these are known, then range to the obstacle can be computed.

Estimator Design

Once estimating obstacle location, a method of mapping the obstacles is necessary. A Kalman filter based approach will work well for an environment with scattered, point obstacles (like the forest). How-

ever, it does not lend itself to a more complex environment (e.g. urban canyons or interior corridors) as the number of states to estimate grows increasingly large and data association grows increasingly difficult. A different approach is necessary: here we use an occupancy grid, which is a numerical implementation of a Bayes filter for a static environment.

Qualitatively, an occupancy grid is a mapping algorithm which computes the likelihood that discrete regions of the environment (cells) are occupied by an obstacles. This is shown schematically in Fig. 3.

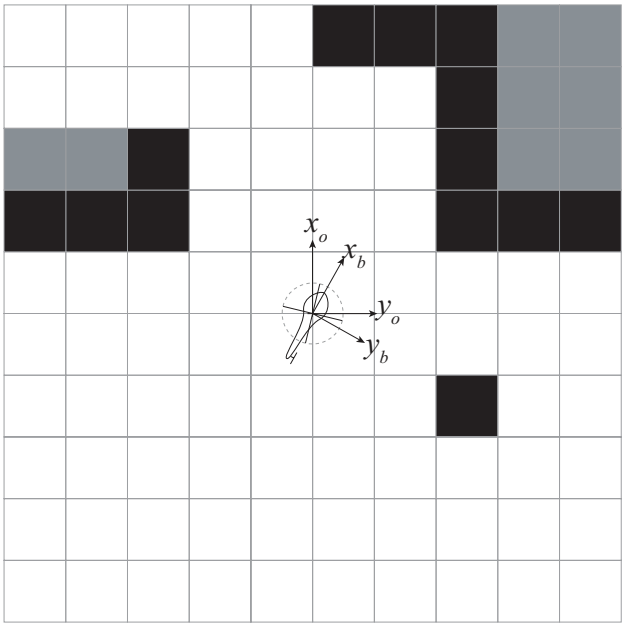


Fig. 3 Schematic of occupancy grid. Cells which are known to be free are white, those which are known to be occupied are black, those which are unknown are grey. Coordinate frames and the vehicle are shown at the origin of the grid.

Once the probabilities of occupancy for cells surrounding the vehicle are known, a control or planning algorithm can be used to compute a path to the goal which minimizes the likelihood of collision.

While occupancy grids have been well documented (e.g. [3,13]), for completeness a derivation is presented here. This section follows the derivation given in Thrun [13].

The Bayes filter for a map of a static environment is

$$p(m|z_{1\ldots t}, x_{1\ldots t}) = \frac{p(z_t|m, x_t)\; p(m|z_{1\ldots t-1}, x_{1\ldots t-1})}{p(z_t|z_{1\ldots t-1}, x_{1\ldots t})} \tag{8}$$

where m represents a particular map, z denotes measurements, x denotes vehicle state, t represents the current time step and $p(m|z_{1\ldots t}, x_{1\ldots t})$ represents the probability that a particular map m is correct, given measurement history $z_{1\ldots t}$ and vehicle path $x_{1\ldots t}$ (i.e. our belief in the correctness of m).

The first term in the numerator is the sensor model, the probability of getting a measurement based on the map and current state. The map is not known, and what is desired is the probability of the map based on the measurements obtained. To achieve this, Bayes' Rule $\left(p(A|B) = \frac{p(B|A)p(A)}{p(B)}\right)$ is used for the

sensor model of Eq. (8) to yield Eq. (9).

$$p(m|z_{1\ldots t}, x_{1\ldots t}) = \frac{p(m|z_t, x_t)\, p(z_t|x_t)}{p(m|x_t)} \frac{p(m|z_{1\ldots t-1}, x_{1\ldots t-1})}{p(z_t|z_{1\ldots t-1}, x_{1\ldots t})} \tag{9}$$

We then assume that the measurement probability is independent of state, i.e. $p(z_t|x_t) = p(z_t)$, and that the map is independent of the state, i.e. $p(m|x_t) = p(m)$. This simplifies Eq. (9) further to:

$$p(m|z_{1\ldots t}, x_{1\ldots t}) = \frac{p(m|z_t, x_t)\, p(z_t)}{p(m)} \frac{p(m|z_{1\ldots t-1}, x_{1\ldots t-1})}{p(z_t|z_{1\ldots t-1}, x_{1\ldots t})} \tag{10}$$

An occupancy grid is a numerical implementation which divides the environment into a finite number of cells, and we compute the probability that each cell is occupied [3]. This results in a binary estimation problem over all possible maps. However this can also pose a computational problem. An environment with M cells has 2^M possible maps. To make the problem tractable, we assume that the probability of a particular cell's occupancy is independent of all other cells. We now estimate $p(m_j|z_{1\ldots t}, x_{1\ldots t})$ for $j = 1 \ldots M$. For each cell, $p(m_j) = 1$ means that the cell is occupied.

To make the problem numerically better conditioned and easier to implement computationally, we represent the occupancy in log-odds form. First we compute the odds form of Eq. (10)

$$o(m_j|z_{1\ldots t}, x_{1\ldots t}) = \frac{p(m_j|z_{1\ldots t}, x_{1\ldots t})}{1 - p(m_j|z_{1\ldots t}, x_{1\ldots t})} = \frac{p(m_j|z_t, x_t)}{1 - p(m_j|z_t, x_t)} \frac{p(m_j|z_{1\ldots t-1}, x_{1\ldots t-1})}{1 - p(m_j|z_{1\ldots t-1}, x_{1\ldots t-1})} \frac{1 - p(m_j)}{p(m_j)} \tag{11}$$

which simplifies Eq. (10) by canceling the $p(z_t|\ldots)$ terms. The factor $p(m_j)$ is the initial probability that the j^{th} cell is occupied. Since the environment is initially unsurveyed, the initial probability of occupancy is $p(m_j) = 0.5$, thus the last term on the right simplifies to 1.

Finally, the log-odds form of the occupancy grid is obtained by taking the logarithm of Eq. (11):

$$\begin{aligned} l_{t,j} &= \log o(m_j|z_{1\ldots t}, x_{1\ldots t}) \\ l_{t,j} &= \log \frac{p(m_j|z_t, x_t)}{1 - p(m_j|z_t, x_t)} + \log \frac{p(m_j|z_{1\ldots t-1}, x_{1\ldots t-1})}{1 - p(m_j|z_{1\ldots t-1}, x_{1\ldots t-1})} \end{aligned} \tag{12}$$

The second term on the right hand side of Eq. (12) is simply the accumulated log-odds of occupancy over all previous time steps:

$$l_{t,j} = \log \frac{p(m_j|z_t, x_t)}{1 - p(m_j|z_t, x_t)} + l_{t-1,j} \tag{13}$$

Hence we have a recursive equation to compute the occupancy of the j^{th} grid cell. This can be implemented easily and efficiently.

The first term on the right hand side of Eq. (13) is the inverse sensor model (i.e. the increment in log-odds occupancy of the j^{th} grid cell given a sensor measurement z_t. Given that there are usually multiple measurements, in this case one for each bearing β_i, inverse sensor model for each measurement can be added together. Supposing there are k measurements

$$l_{t,j} = \sum_{i=1}^{k} \log \frac{p(m_j|z_{t,i}, x_t)}{1 - p(m_j|z_{t,i}, x_t)} + l_{t-1,j} \tag{14}$$

Functionally, detecting an obstacle in a grid cell means we increment the log-odds of occupancy in that cell by some amount while we decrement the log-odds of occupancy of cells lying between the occupied cell and the vehicle. Cells outside the sensor field of view remain unchanged. The amount of the increment and decrement is dependent on the inverse sensor model.

Inverse Sensor Model

The sensor model divides the camera's field of view into bins of equal angular width, and the maximum optical flow measured in each bin is used as the measurement. This can then be used to estimate the distance to the closest object in that bin by solving Eq. (7) for r_i.

$$r_i = \frac{1}{\dot{\beta}_i + \dot{\psi}}(u \sin \beta_i - v \cos \beta_i) \tag{15}$$

To compute the uncertainty in the range estimate (caused by noise in bearing and optical flow measurements and by uncertainty in heading rate and velocity), the Jacobian of the equation for r_i is computed and used to compute the range uncertainty:

$$\nabla_{r_i} = \begin{bmatrix} \frac{\partial r_i}{\partial u} & \frac{\partial r_i}{\partial v} & \frac{\partial r_i}{\partial \beta} & \frac{\partial r_i}{\partial \dot{\beta}} & \frac{\partial r_i}{\partial \dot{\psi}} \end{bmatrix}^T \tag{16}$$

$$\sigma_{r_i}^2 = \nabla_{r_i}^T \begin{bmatrix} \sigma_u^2 & 0 & 0 & 0 & 0 \\ 0 & \sigma_v^2 & 0 & 0 & 0 \\ 0 & 0 & \sigma_\beta^2 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{\dot{\beta}}^2 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\dot{\psi}}^2 \end{bmatrix} \nabla_{r_i} \tag{17}$$

where $\sigma_{(\cdot)}$ represents the standard deviation of noise associated with the relevant state or measurement.

In addition to the Gaussian uncertainty described by σ_{r_i}, the sensor model must reflect the intuition that the space between the camera and the detected obstacle is unoccupied and the space behind the detected obstacle remains unknown. To model this behavior a sigmoid curve and the Gaussian described above were combined to produce an occupance weight as a function of range:

$$f_i(r) = \frac{-c_1}{\sigma_{r_i}\sqrt{2\pi}} \left\{ 1 + \exp\left[\frac{2\pi(r - r_i^* + 2\sigma_{r_i})}{\sigma_{r_i}\sqrt{3}} \right] \right\}^{-1} + \frac{c_2}{\sigma_{r_i}\sqrt{2\pi}} \left\{ \exp\left[-\frac{(r - r_i^*)^2}{2\sigma_{r_i}^2} \right] \right\} \tag{18}$$

Here r_i^* is the estimated range obtained from the optical flow measurement in the i^{th} bearing. The factors c_1 and c_2 are used to scale the two contributions to the probability of occupancy and to account for normalization.

Since the total camera field of view is divided into angular bins, the probability of occupancy must be computed over the width of a bin. A sigmoid function is used to compute a value as a function of angle ξ from the bin centerline:

$$g_i(\xi) = \left(1 + \exp\left| \frac{c_3(\xi - \beta_i - w_\beta)}{w_\beta} \right| \right)^{-1} \tag{19}$$

where c_3 is a weighting parameter.

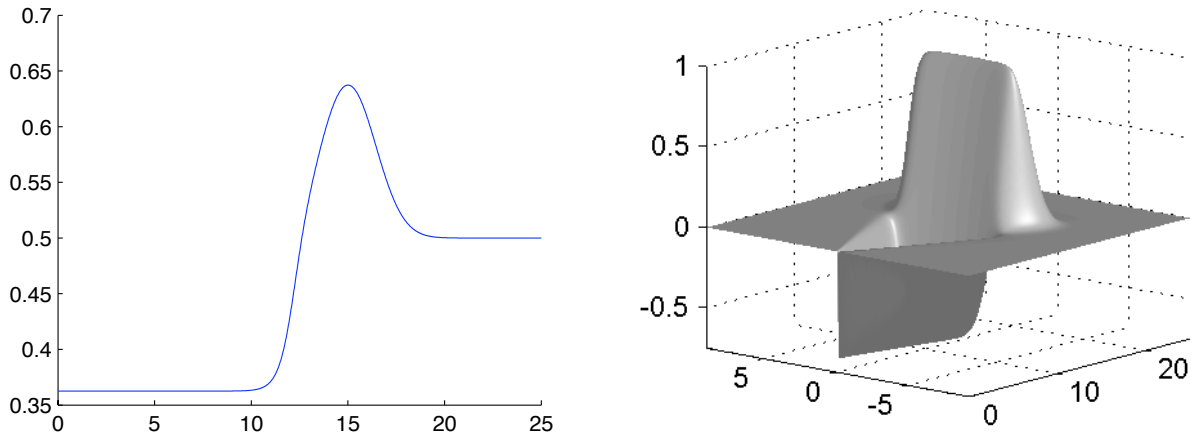
For the j^{th} grid cell located at $x_j,\ y_j$ with respect to the vehicle, the probability of occupancy induced by a measurement of optical flow in the i^{th} bin can be computed by evaluating r and ξ for that cell and computing

$$p(m_j | z_t, x_t) = f_i(r_{x,y}) g_i(\xi_{x,y}) \tag{20}$$

for a measurement z_t. Computing the log-odds and summing over all measurements completes the inverse sensor model. An example is shown in Fig. 4.

Local Occupancy Grid

Typically, occupancy grids remain globally fixed while the vehicle moves through the grid. This allows for a simple implementation; however, as the vehicle explores the environment, the map quickly grows large and becomes computationally intractable. Since we are interested in local obstacle avoidance a local occupancy grid (fixed to the vehicle) is used. A local occupancy grid has a limited size governed by sensor field of view and computational considerations, and can thus be adapted to the specific hardware available on a particular vehicle.



(a) Range vs. probability for $r^* = 15$ and $\sigma_r^2 = 2$. Note the comparatively low probability of occupancy for $r < r^*$.

(b) 3-D view of sensor model in log-odds form

Fig. 4 Example of the Inverse Sensor Model. Regions outside the sensor field of view (or occluded by obstacles) have zero change to their log-odds of occupancy.

Computationally the occupancy grid representation of the environment can be treated as an image (Fig. 3), hence techniques developed for image processing (e.g. blurring, convolution, rotation) and libraries used for image processing (e.g. OpenCV) can be used to translate and rotate the local occupancy grid as it moves with the vehicle. In this implementation the grid translates with the vehicle but its orientation remains fixed in frame O. Keeping the orientation fixed eliminates the need to create a rotation convolution, a time consuming task. Additionally, in a grid rotation, some information is lost due to blurring as rotated cells may map to multiple cells.

Translating the occupancy grid with the vehicle means that a motion update must be performed on the occupancy grid. Here this is performed using a convolution. However, this introduces blur when the translation does not map one cell exactly to another (Fig. 5). The top image shows the initial occupancy grid, with the center cell known to be occupied. Translation by a fraction of a grid cell (middle image) means that the probability

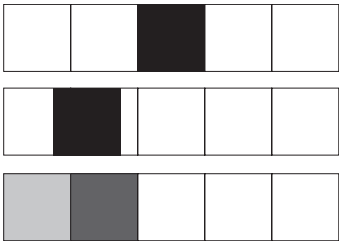


Fig. 5 Translation of occupancy grid.

of occupancy must be split over the neighboring cells (bottom image), introducing a blur. This blur artificially decreases the belief in occupancy (or freedom) of a grid cell. To reduce the artificial blurring motion updates were only performed when the translation was greater than or equal to the length of a grid cell. Additionally, the motion updates were performed separately in the x_o and y_o directions. Not only does this reduce some of the blurring, but also decreases computational time as performing two one-dimensional convolutions is less computationally intensive than one two-dimensional convolution.

Note that the artificial blurring can also be reduced by increasing the resolution of the occupancy grid. This comes at the cost of increased computation requirements.

Vehicle Control

In this work, the desired heading is computed using a potential field generated using the occupancy grid combined with a term to represent the goal. Regions of high probability of occupancy represent high potential (to be avoided) and the goal is represented by a sink. The vehicle is then commanded to steer in the direction of the gradient. This gradient is computed at the vehicle (i.e. the origin of the occupancy grid) as

$$\nabla = (1 - w_g)\mathbf{K} * \nabla_{grid} + w_g \nabla_{goal} \tag{21}$$

where w_g is a factor to weight goal seeking vs. obstacle avoidance, $\mathbf{K}$ is a blurring kernel (to be discussed later), ∇_{grid} is the gradient of the occupancy grid and ∇_{goal} is the gradient induced by the goal. The desired heading is computed from ∇ as

$$\psi_{des} = \arctan \frac{\nabla_y}{\nabla_x} \tag{22}$$

where ∇_y and ∇_x are the y and x components of the gradient ∇, respectively.

Finally the vehicle turn rate command is

$$\omega = c_6(\psi_{des} - \psi)^{c_7} \tag{23}$$

where ψ is the vehicle's current heading.

Figure 6 shows the steps in creating ∇_{grid} (the gradient due to the occupancy grid). Only a region in the vicinity of the vehicle (here, any cell more than 9 grid cells away from the vehicle is ignored) is

considered when computing the gradient. First a Prewitt edge detector is used to find gradients in both the x_o and y_o directions. Since this will give large values only at edges (i.e. at boundaries between occupied and unoccupied space) the blurring kernel $\mathbf{K}$ is introduced to allow the effect of edges to be "felt" earlier. This kernel was created in two parts: first, gradients near the vehicle should be weighted heavily, but values at a distance should still be included to cause the vehicle to turn before it strikes them; second, gradients due to obstacles in the direction of motion should have greater influence on desired heading than those from obstacles already passed.

First, a Gaussian was used to blur the gradient field in all directions with the standard deviation being a function of vehicle speed. This allows the gradient influence to be a function of time, 'sensing' obstacles that are a certain time away, allowing time to avoid them.

$$\sigma_{blur} = \frac{c_4(\sqrt{u^2 + v^2} + c_5)}{\Delta_{grid}} \tag{24}$$

where Δ_{grid} is the length of a grid cell and c_4 and c_5 are parameters which allow further tuning of the blurring kernel in its dimension and standard deviation. Second, a Gaussian was used to more heavily weight the obstacles which are in the direction of motion. These two factors were combined to create the kernel $\mathbf{K}$

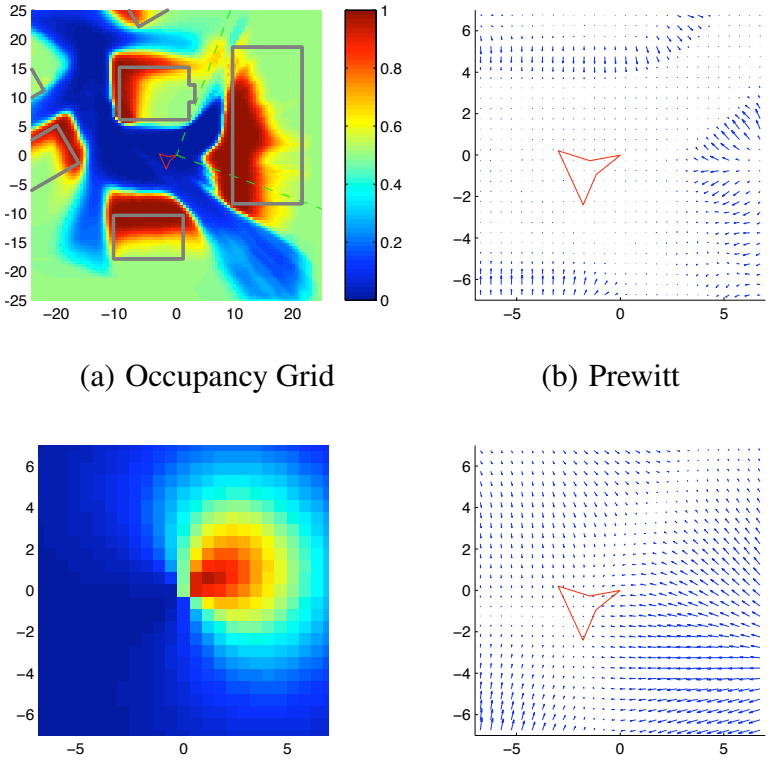
$$k_{x,y} = \exp\left(-\frac{r_{x,y}^2}{2\sigma_{blur}^2}\right) \ * \ \exp\left(-\frac{\xi_{x,y}^2}{2\sigma_{dir}^2}\right) \tag{25}$$

where $k_{x,y}$ are the components of $\mathbf{K}$, $r_{x,y}$ is the distance from a grid cell to the vehicle, $\xi_{x,y}$ is the angle between the unit vector to the grid cell and the direction of motion and σ_{dir} is the width of the directional blur. Figure 6(c) shows the resulting kernel for motion diagonally upwards and to the right; Fig. 6(d) shows the result of applying the directional blurring kernel to the initial gradient, shown in Fig. 6(b).

The goal gradient ∇_{goal} is simply a unit vector in the direction of the goal.

The vehicle is located at the origin of the occupancy grid, which is at the corner of four grid cells. Thus the gradient ∇ is evaluated at the centers of the four nearest grid cells and the average value is used to compute the desired heading direction.

To allow smaller radius turns (which may be necessary in environments with densely packed obstacles) a speed controller was implemented. This allows the vehicle to accelerate (up to a maximum speed) when



(a) Occupancy Grid (b) Prewitt

(c) Blurring Kernel (d) Blurred Gradient

Fig. 6 Creating the potential field due to the occupancy grid at 31.1 seconds. Subfigures (a) and (c) use the same color bar. The vehicle is shown by the cranked triangle. Motion is upwards to the right.

flying straight and decelerate (to a minimum speed) when large turn rates are commanded.

$$a_x = c_8 - c_9\omega \tag{26}$$

Here c_8 and c_9 are the acceleration control parameters.

Simulation

A simulation is run to demonstrate the algorithm. The unmanned rotorcraft starts at the edge of a forest, represented by round tree trunks. After navigating through the forest, it enters an urban setting modeled after the McKenna MOUT Site in Ft. Benning, GA, as seen in Fig. 7. A goal location is placed within the town limits. This environment combines both small convex obstacles (tree trunks) and large possibly concave obstacles (buildings). For these simulations vehicle kinematics were assumed to be non-holonomic (i.e. v and $\dot{v}$ were set to be 0). This is a more challenging example of kinematics than that provided by a holonomic vehicle both from the standpoint of map generation and obstacle avoidance: turns are required both to avoid obstacles and to ensure that obstacles directly ahead of the vehicle are accurately resolved.

Fig. 7 McKenna MOUT Site at Ft. Benning, GA

The tree locations were randomly generated each run and the dimension of the trees (axes of the ellipses representing the trunks) were also randomly generated. To minimize 'clumping' of the forest no two trees were allowed to be closer than 7 meters apart and the trees ranged from 0.5 meters to 1.5 meters in diameter.

For the results specified below, the camera was assumed to have a 90° field of view and there were 24 measurements bins, each 3.75° wide. Camera range was 25 meters, so the size of the occupancy grid was 50 meters on a side. As the smallest obstacle to identify was no smaller than 0.5 meters, the grid cells

were chosen to also be 0.5 meters on a side. The rest of the parameters and constants are listed below.

$$\begin{aligned}
\sigma_{u_{est}} &= 0.4\ m/s & \sigma_{a_x} &= 0.05\ m/s \\
\sigma_{\psi_{est}} &= 1^\circ & \sigma_\omega &= 2^\circ/s \\
\sigma_\beta &= 1.875^\circ & \sigma_{\dot{\beta}} &= 0.1^\circ/s \\
c_1 &= 2 & w_g &= 0.93 \\
c_2 &= 2 & c_6 &= 1.1 \\
c_3 &= 15 & c_7 &= 0.85 \\
c_4 &= 0.5 & c_8 &= 0.9 \\
c_5 &= 4 & c_9 &= 1.2
\end{aligned}$$

Limited tuning was performed on control law parameters (w_g and c_6, c_7, c_8, c_9

Additionally, limits were set on the helicopter speed and turn rate.

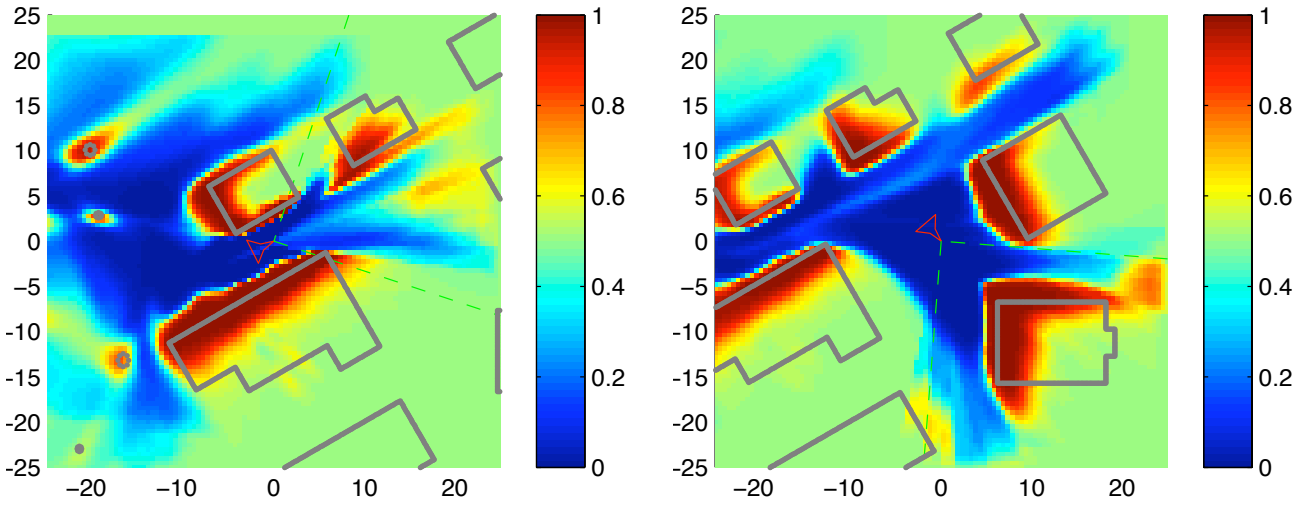
$$1.5\ m/s \leq u \leq 5.5\ m/s$$
$$-70^\circ/s \leq \dot{\psi} \leq 70^\circ/s$$

Results

Figure 8 displays the results from a representative run of the simulation. The vehicle successfully navigates through narrow corridors until it reaches the goal. Obstacles (shown in gray) were precisely and accurately identified, except for those directly in front of the camera (as expected). Note that the gradient-based approach to control causes the vehicle to steer away from "unknown" space, thus causing the vehicle to turn to improve its knowledge of the area directly in front.

As a particular obstacle remains within the field of view it is localized with greater accuracy. Further, the confidence that space thought to be unoccupied actually is free space increases with time in the field of view. Further, the occupancy grid maintains its 'memory' of space which has left the field of view.

Note that the gradient-based control algorithm implemented here only accounts for the area within 7 meters of the vehicle. While it worked very well for both obstacle avoidance and navigation to the goal, it (like all potential field approaches to path planning) is subject to local minima and the possibility of flying into dead ends. This can be avoided by implementing a global planner and treating obstacle avoidance as a local perturbation (this is done in [12]) or by using a longer-range planner to compute a path to the



(a) 19 Seconds

(b) 22.9 Seconds

(c) 28 Seconds

(d) 40 Seconds

Fig. 8 Sequence of images from representative run. The vehicle is shown as a red cranked triangle, camera field of view is shown as green dashed lines. Red represents a high probability that a cell is occupied; blue represents a low probability that a cell is occupied (i.e. high probability that the space is free). Green represents a 50% probability (uncertainty) that a cell is occupied. Obstacles are shown as gray lines, the magenta dot is the goal.

edge of the occupancy grid. This will improve trajectories flown at the cost of increased computation requirements.

Conclusion

This paper has presented an algorithm for generating a map useful for navigating a small unmanned rotor craft through a previously unknown static environment using only a monocular camera and measurements of vehicle speed.

The camera's field of view is divided into bins of equal angular width. The maximum optical flow in each bin is used along with measurements of vehicle speed to compute an estimate of range to the nearest obstacle in that bin. These range measurements are used to generate an occupancy grid representation of the environment in the vicinity of the vehicle. This occupancy grid representation allows modeling of both small and large obstacles and can be used by trajectory planning algorithms to find a safe, feasible path to the goal.

Simulation results show a very accurate and precise map can be generated from measurements of optical flow. Further, this map has been combined with a potential field-based obstacle avoidance and navigation algorithm to demonstrate safe flight to a goal in a complex, cluttered environment which was previously unsurveyed.

Acknowledgements

The support of the Applied Research Laboratory's Exploratory and Foundational Research Program is gratefully acknowledged.

References

[1]C. Braillon, C. Pradalier, K. Usher, J. L. Crowley, and C. Laugier, *Experimental Robotics*, ser. Springer Tracts in Advanced Robotics. Springer Berlin/Heidelberg, 2008, vol. 39, ch. Occupancy Grids for Stereo and Optical Flow Data, pp. 367–376.

[2]J. Chahl and A. Mizutani, "An algorithm for terrain avoidance using optical flow," *American Control Conference, 2006*, pp. 6 pp.–, June 2006.

[3]A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *IEEE Trans. Comput.*, vol. 22, no. 6, pp. 46–57, June 1989.

[4]S. Hrabar, G. Sukhatme, P. Corke, K. Usher, and J. Roberts, "Combined optic-flow and stereo-based navigation of urban canyons for a uav," *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pp. 3309–3316, Aug. 2005.

[5]O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[6]J. Kim and G. Brambley, "Dual optic-flow integrated navigation for small-scale flying robots," *Australasian Conference on Robotics and Automation, 2007*, pp. 7 pp.–, December 2007.

[7]J. Kim and S. Sukkarieh, "Real-time implementation of airborne inertial-slam," *Robotics and Autonomous Systems*, vol. 55, pp. 62–71, 2007.

[8]J. W. Langelaan, "State estimation for autonomous flight in cluttered environments," *Journal of Guidance, Control and Dynamics*, vol. 30, no. 5, pp. 1414–1426, September-October 2007.

[9]T. Netter and N. Franceschini, "A robotic aircraft that follows terrain using a neuromorphic eye," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Lausanne, Switzerland, 2002.

[10]M. Pollefeys, L. V. Gool, M. Vergauwen, K. Cornelis, F. Verbiest, and J. Tops, "3D capture of archaeology and architecture with a hand-held camera," in *ISPRS workshop on Vision Techniques for Digital Architectural and Archaeological Archives*, Ancona, Italy, July 2003, pp. 262–267.

[11]J. M. Roberts, P. I. Corke, and G. Buskey, "Low-cost flight control system for a small autonomous helicopter," in *2002 Australasian Conference on Robotics and Automation*. Auckland, New Zealand: ARAA, November 2002.

[12]S. Scherer, S. Singh, L. Chamberlain, and M. Elgersma, "Flying Fast and Low Among Obstacles: Methodology and Experiments," *The International Journal of Robotics Research*, vol. 27, no. 5, pp. 549–574, 2008. [Online]. Available: http://ijr.sagepub.com/cgi/content/abstract/27/5/549

[13]S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, 2005.

[14]S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney, "Stanley: The robot that won the darpa grand challenge," *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, June 2006.

[15]United States Department of Defense, "Office of the secretary of defense unmanned systems roadmap (2007–2032)."

[16]J.-C. Zufferey and D. Floreano, "Toward 30-gram autonomous indoor aircraft: Vision-based obstacle avoidance and altitude control," *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pp. 2594–2599, April 2005.