**The Pennsylvania State University**
**The Graduate School**

# LOCAL TERRAIN MAPPING FOR OBSTACLE AVOIDANCE USING MONOCULAR VISION

A Thesis in
Aerospace Engineering
by
Sean Quinn Marlow



Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

August 2009

The thesis of Sean Quinn Marlow was reviewed and approved* by the following:

Jacob W. Langelaan
Assistant Professor of Aerospace Engineering
Thesis Advisor

Joesph F. Horn
Associate Professor of Aerospace Engineering

George A. Lesieutre
Professor of Aerospace Engineering
Head of the Department of Aerospace Engineering

*Signatures are on file in the Graduate School.

# Abstract

The motivation behind the research described in this thesis is be able to navigate a small unmanned aerial vehicle (UAV) through complex environments. Missions envisioned for small UAVs now require low altitude flights among many obstacles. These obstacles can be simple shapes (telephone poles) or more complex shapes (corners and protrusions of an urban canyon).

This thesis focused on the problem of estimating obstacle locations and safe navigation to a known goal location. The close proximity and complex nature of these environments requires a system of navigation and obstacle avoidance using onboard sensors. However, payload limitations of small UAVs place significant restrictions on sensor size, weight, and power consumption. This thesis describes the development and simulation of an algorithm that enables safe navigation using only a monocular camera and GPS corrected inertial navigation system (GPS/INS).

The obstacle estimation problem poses many challenges. First, the camera measurements (bearings to obstacles and bearing rates of the obstacles due to motion of the vehicle) are heavily corrupted with noise. This greatly reduces the certainty of information obtained from the camera. Second, the equations governing the vehicle motion and vision measurements are highly non-linear. The combination of noisy measurements with non-linear equations leads to significant uncertainties in the estimation problem.

Traditional feature-based mapping techniques (like the Kalman Filter) become intractable in this environment as too many features must be mapped to resolve the complex shaped obstacles. A local occupancy grid is instead used to store estimates of occupied space. The local occupancy grid has a limited size that can be specified depending on the sensor field of view and computational condiserations. The origin of the grid is fixed to the vehicle and in this application the orientation is fixed to an inertial frame. By keeping the grid centered on the vehicle, vehicle motion must be accounted for in the grid with a motion update step. A potential

field trajectory generator provides the means for vehicle navigation and obstacle avoidance. The algorithm performance is examined through simulations in an environment modeled after the McKenna Military Operations in Urban Terrain site at Ft. Benning, GA.

# Table of Contents

# List of Figures

# List of Tables

# Acknowledgments

The work presented in this thesis would not be possible without the insight, direction, and general assistance received from many people. I would first like to thank my family. Throughout my entire academic studies, my parents have been there in every way possible. Without their support, I know that I would not have had the opportunity to pursue so many different avenues. I will forever be in their debt for this. My sister has also always been a part of my life learning experience. From teaching me fractions with Play-Doh as a young kid (and trying to keep me from eating it), Caitlin is forever willing to give me her input on any subject.

My advisor, Dr. Jack Langelaan continuously encourages me and provides the perfect scholastic environment to let ideas build. His ability to break down complex concepts and relate theoretical concepts to real world applications is an incredible skill that only makes me want to learn more. I would also like to thank Dr. Joe Horn for reviewing this thesis and providing beneficial suggestions for improvement.

Many friends have exchanged ideas with me and offered advice, but I would particularly like to thank Sana Sarfraz and Misha Kononov. Sana has been my lab-mate for my entire graduate tenure and has always put up with my often crazy antics. Sana has helped create very relaxed atmosphere in the lab which promotes makes coming in to work everday a fun experience. Her intelligence and insightful input have often helped turn my thought train back onto the correct track. Although Misha claims that a trained monkey could have provided the same help he did, I would often be lost without him. While Misha rarely provided a direct solution to any problem, he always engaged in conversation that helped direct my thoughts **way** outside the box.

Finally, I owe a great deal of gratitude to the Applied Research Lab at Penn State for providing funding for my research through their Exploratory and Foundational Graduate Student Research Program.

# Dedication

To my parents.

Dedication

Chapter 1

# Introduction

This thesis presents the development and simulation of a an algorithm which enables safe autonomous navigation in complex environments using a limited sensor suite. The motivation for this research is to fly small unmanned aerial vehicles (UAVs) through cluttered environments consisting of trees and buildings. The size of the UAVs limits the types of sensors which are available to gather information about its surroundings. A monocular camera is used to obtain bearing and optical flow measurements from obstacles as it fits within both the power and weight restrictions.

The estimation problem poses many challenges. First, the vision measurements are heavily corrupted with noise. This greatly reduces the certainty of information obtained from this sensor. Second, the equations governing the system and measurements are highly non-linear. The combination of noisy measurements with non-linear equations leads to significant uncertainties in the estimation problem which must be accounted for.

This thesis will:

- Provide a framework for navigation using only a monocular camera and GPS corrected inertial navigation system (GPS/INS). This involves preparing mathematical models for vehicle dynamics and the on board sensors, and selecting an appropriate tool to estimate and retain the locations of obstacles.

- Present a solution to the navigation problem based on an occupancy grid and a potential field controller

**Figure 1.1.** The McKenna MOUT Site at Ft. Benning, GA is an example of a complex cluttered environment. It consists of a central town of various shaped buildings surrounded by a forest.

- Present simulation results to show the effectiveness of the designed system in avoiding obstacles through both a forest and urban environment.

## 1.1 Motivation

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 such as search and rescue in forests and surveillance in urban environments (such as the one shown Fig. 1.1). The close proximity and complex nature of these environments requires a system of navigation and obstacle avoidance using on board sensors.

Current technologies for on board detection of obstacles rely heavily on LIDAR and RADAR, large active sensors with great power requirements. The Department of Defense (DoD) specifies passive detection as a goal for all unmanned systems and lists reconnaissance as the number one priority for all classes of unmanned

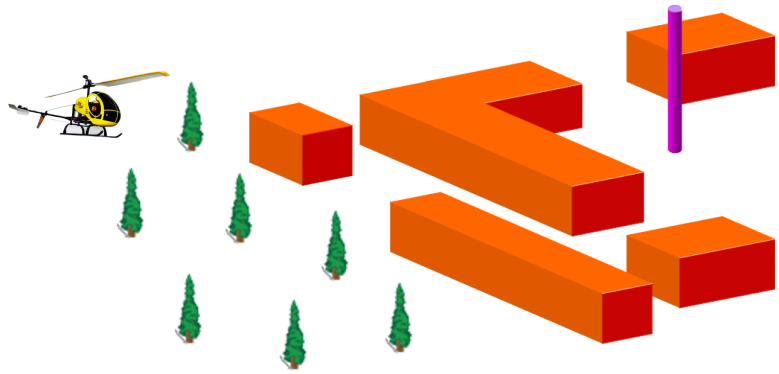


**Figure 1.2.** Schematic of a mission scenario. The UAV must avoid collisions with any trees and buildings and navigate from a known start position to a known goal.

systems [1]. In addition to the restrictions imposed by passive sensing, vehicle size adds significant complications: sensing payloads are restricted in both weight and power requirements and vehicle performance requirements are very strict to complete the necessary maneuvers for obstacle avoidance.

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. By not emitting an observable ping (like LIDAR and SONAR), vision systems lend themselves to stealth applications.

This thesis is concerned with obstacle avoidance for small UAV operating in complex, cluttered, unsurveyed environments (see Fig. 1.2 for a schematic). The primary focus is on generating a local map of the environment which is suitable for use with generic control and planning algorithms.

## 1.2 System Overview

Autonomous flight through an urban environment is a complex problem. Making the problem even more difficult is the noisy measurements of bearing and optical flow from the single camera. The block diagram in Fig. 1.3 shows a system that uses the given sensors (GPS/INS and a monocular camera) to perform obstacle avoidance and safe navigation to the goal.

The system is comprised of three major parts: a stabilized aircraft; an esti-

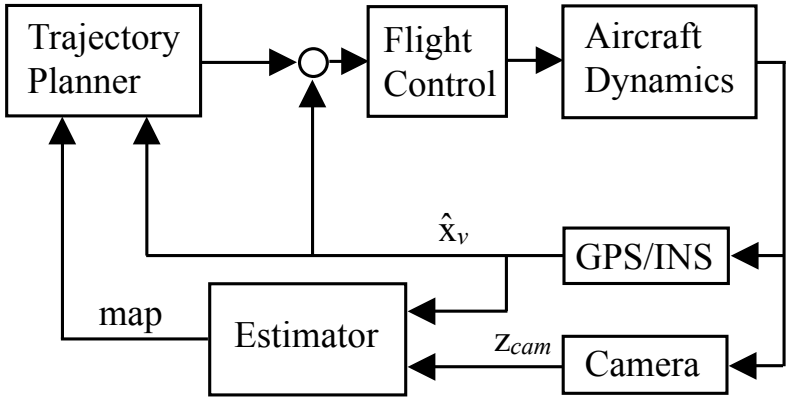



**Figure 1.3.** System block diagram

mator; and a trajectory planner. For this thesis, the stabilized aircraft represents a vehicle platform which can maintain a desired flight condition. As the vehicle moves through the environment, the GPS/INS sensor outputs estimates of vehicle states, $\hat{\mathbf{x}}_v$ (position, velocity, and orientation). A single forward-pointing camera obtains measurements of bearing and optical flow (bearing rates) to obstacles, $\mathbf{z}_{cam}$. The estimator uses the available measurements of bearing and optical flow combined with the estimated vehicle states provided by GPS/INS to compute a map of obstacle position estimates. The trajectory planner uses knowledge of vehicle state and the map of estimated obstacle locations to compute a safe path to the goal.

## 1.3 Problem Description

The critical technology described is the development of a system which can successfully navigate to a goal while safely avoiding obstacles using only measurements of bearing and optical flow and estimate of vehicle states from GPS/INS. This thesis presents solutions to the challenges faced in the navigation problem: managing noisy sensors, estimating obstacle locations in complex terrain, and trajectory planning using a local map.

### 1.3.1 Noisy Sensors

The camera obtains measurements of bearing and optical flow (bearing rate), both very noisy measurements. To get an accurate estimate of obstacle location, a robust estimator is necessary. In addition to the noisy sensor, this system must also handle sensor dropouts (e.g. a brownout while operating over a desert).

This thesis proposes the use of a map to store estimates of obstacle location. The map can continuously be updated with new information on obstacle location taking account for the noise in measurements. Additionally, the map retains estimates of obstacle locations in the case of sensor dropout so that obstacle avoidance is still possible.

### 1.3.2 Estimating Obstacle Locations

Feature-based tracking techniques such as Simultaneous Localization and Mapping (SLAM) (which have the advantage of *not* requiring the availability of camera motion measurements through external means such as GPS) have been successfully used in mapping problems in the past[2]. However, these methods often need prior knowledge of the terrain to initialize the Kalman Filter-based estimators. In cases where no knowledge of the environment is known, they can be initialized using a particle filter approach or estimated based on a terrain map[3, 4].

While these feature-based approaches work well in environments populated with point obstacles (e.g. tree trunks in a forest), they quickly become intractable in large environments or complex environments where obstacles are difficult to define by features.

This thesis proposes the use of an occupancy grid as a map of obstacle locations. The occupancy grid is shown to handle complex objects and portray their location accurately and precisely.

### 1.3.3 Trajectory Planning

The vehicle should not only go to the goal, but also avoid obstacles along the path to the goal. The estimator and map used in this thesis are designed to be compatible with generic trajectory planners. In this thesis a potential field approach is used for its simplicity and speed of computation[5].

## 1.4 Related Work

There has been much research relating to the problem of navigation and obstacle avoidance for mobile robots. The previous section presented references specifically relating to Kalman Filter-based implementations, this section presents a more detailed discussion of research in the related fields of vision based navigation and occupancy grids.

### 1.4.1 Optical Flow

Vision based estimation methods have been popular recently due to low power and weight requirements. 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. [6]) but this is typically formulated as a batch process and is thus not suited for real-time implementation.

Hrabar et al. have fused optical flow and stereo vision measurements on both a tractor and unmanned helicopter to fly in urban canyons in real time[7]. Here the optical flow measurement is used to turn the vehicle away when too large a measurement is obtained. While the fusion system worked well and optical flow measurements could keep the tractor centered in a corridor, the vehicle could not navigate corners using the optical flow techniques.

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[8]. 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. Roberts et. al. uses stereo cameras to determine altitude above the ground [9]. Optical flow measurements from both cameras are then combined with the estimate of relative altitude to determine ground speed.

Chahl and Mizutani propose an optical flow method for ground avoidance[10]. 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[11]. Netter and Franceschini follow ground terrain using optical flow values from a simulated insect eye to estimate the relative ground altitude[12].

Optical flow has been used for obstacle avoidance or ground speed estimation by several researchers. However, direct reliance on measurements of optical flow for obstacle avoidance results in low robustness to noise and sensor dropouts.

### 1.4.2 Occupancy Grids

Scherer et. al. recently successfully flew an autonomous helicopter through the McKenna MOUT site at Ft. Benning, GA[13]. 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 6 mm wire from 38 m away) which greatly assists navigation but comes at the high cost of power, weight, and electromagnetic emissions.

Braillon et al. used stereo and optical flow to populate an occupancy grid representation of the local environment, but their approach required identification of a ground plane [14]. Usher also made use of occupancy grids to fuse data from stereo vision and a scanning laser range-finder[15]. Here, each sensor generated its own occupancy grid which were then combined using a weighted average. The system did identify and avoid obstacles, but the tests were performed on a large tractor capable of handling the heavy LIDAR system.

Badino et. al. used stereo cameras with a Kalman Filter to generate estimates of obstacle locations and store them in a polar occupancy grid[16]. Dynamic programming was used to find a connected line of closest obstacles to determine a region of free space in front of the vehicle.

Recently, occupancy grids have been used with dynamic environments. Two groups have used temporal occupancy grids to look at the change in the environment over time[17, 18]. By noting differences over time, dynamic obstacles can be separated from static ones. Coué et. al. have used a 4 dimensional occupancy grid to estimate obstacle location and velocity across a 2-dimensional space with measurements from a scanning laser range finder[19]. This system both identifies dynamic obstacles and accounts for their movement in prediction steps. However, the computational expense of this approach becomes too great for portable computer systems as the size of the occupancy grid is increased.

Occupancy grids are useful for storing estimate of obstacle locations; however, most current systems make use of LIDAR which is too heavy and power hungry a device to use with small vehicles.

## 1.5 Contributions

The main contributions of this thesis are described below:

- **Method for obstacle avoidance**
  A method for obstacle avoidance using only vision and GPS/INS sensors has been developed. This system fuses estimates of vehicle states from an inertial navigation system correctly by GPS with measurements from a camera to determine relative obstacle locations.

- **Estimator design**
  A map based on the occupancy grid was developed. Obstacle locations relative to the vehicle are estimated. This information was used by a trajectory planner to compute a safe path to the goal through a complex environment.

- **Performance verification through simulations**
  A simulation is run to test the performance of the designed system. Results of simulations show that the occupancy grid based implementation provides a solution to the navigation problem. The locations of obstacles are accurately estimated and the vehicle reaches the goal on 95% of the simulation runs.

## 1.6 Reader's Guide

The remainder of this thesis is organized as follows:

- **Chapter 2** describes the navigation problem mathematically. It defines the models for vehicle kinematics and the sensors. It also provides justification and the derivation of the occupancy grid.

- **Chapter 3** describes the tasks to be accomplished by the navigation system. It then presents the mathematical implementation of the occupancy grid and the controller.

- **Chapter 4** begins with a description of the simulation setup designed to test the navigation system proposed in Chapter 3. It then presents results of the 2 dimensional simulations.

- **Chapter 5** summarizes the results of this research and provides recommendations for future work.

Chapter 2

# The Navigation Problem

The following chapter defines the navigation problem. The major topics discussed are:

1. Problem Statement: The setup of the navigation problem is detailed. A system is described to navigate through a previously unsurveyed obstacle field. The use of an occupancy grid is proposed to handle the complex environment and noisy measurements.

2. System Models: A mathematical model is proposed to simulate vehicle dynamics. A sensor model is proposed to emulate the working of the sensors, a single forward looking camera. The measurements of bearings and bearing rates to obstacles are simulated by this model.

3. Occupancy Grid: The mathematical derivation of the occupancy grid is presented. To make the occupancy grid easier to implement computationally, the estimation equation is transformed to log-odds form.

## 2.1 Problem Statement

The situation considered here is a vehicle moving through an unsurveyed obstacle field consisting of small, convex obstacles (such as tree trunks) and large, potentially non-convex obstacles such as buildings, Fig. 2.1. An on-board camera obtains measurements of bearing and optical flow while GPS/INS provides estimates of velocity and heading.

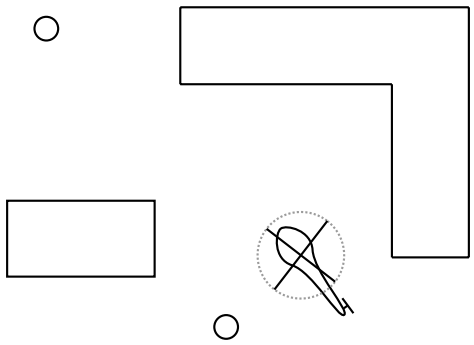


**Figure 2.1.** Navigation/avoidance scenario. The vehicle must fly to a goal (not shown) while avoiding small, convex obstacles (e.g. tree trunks) as well as large, potentially non-convex obstacles such as buildings.

The problem is to navigate safely through the obstacle field to reach the goal. As vehicle state is known through GPS/INS the problem is limited to obstacle avoidance, which requires a means of sensing and then computing relative obstacle position. A robust estimator is needed to handle the noisy measurements associated with bearings and optical flow. The estimator can then be used to generate a map of obstacles and plot a safe trajectory using the map. A map also allows for both continued navigation during sensor dropouts or in areas currently not in the sensor field of view, but previously surveyed.

Given known vehicle state ($\mathbf{x}$) and measurements of bearing and optical flow ($\mathbf{z}$), a map of the environment ($\mathbf{m}$) will be computed. Here,

$$\mathbf{x} = [x, y, \psi, u, v]^T \tag{2.1}$$

where $x$ and $y$ are the location of the vehicle, $\psi$ is the vehicle heading relative to an inertial frame, and $u$ and $v$ are the velocity of the vehicle in the body frame. As optical flow and bearing measurements are noisy, the uncertainty in the map is also needed to determine a trajectory that minimizes the probability of a collision. Thus, it is necessary to compute $p(\mathbf{m}|\mathbf{z}, \mathbf{x})$, the belief in the correctness of the map given vehicle state and measurement history and measurement model $\mathbf{z} = g(\mathbf{x}, \mathbf{m})$, the probability of getting a measurement given the vehicle state and the map.

The rotational freedom of the vehicle introduces a non-linearity to the problem through the kinematic model. Additionally, the projection of the three-dimensional

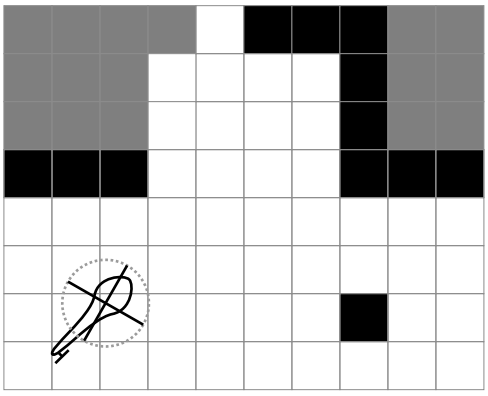


**Figure 2.2.** Schematic of occupancy grid. Cells which are known to be free are white, those which are known to be occupied are black, those which are unknown are grey.

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 (and therefore vehicle motion) is essential. However, obstacles directly in the path of motion (which need to be avoided) generate almost no optical flow and thus no information for a range estimate; transverse motion is required to produce a useful estimate of obstacle location. While transverse motion gives good estimates of obstacles in front of the camera, the vehicle must avoid obstacles which cannot be measured as they are not in the field of view of the camera. As stated above, a map of estimated obstacle locations is useful in this scenario.

An occupancy grid is a mapping algorithm which computes the likelihood that discrete regions of the environment (cells) are occupied by an obstacle. This is shown schematically in Fig. 2.2. As measurements are taken, the likelihood of cells corresponding to the measurements is increased. Occupancy grids can also show areas of low probability of occupancy (i.e. high probability of being free space) by decreasing the likelihood of cells where measurements indicate no obstacles. This makes the occupancy grid very robust to noisy measurements. With several measurements, the estimate location of an obstacle and the uncertainty in the estimate rise out of the grid.

Complex obstacles shapes are easily handled by the occupancy grid, seen in

Fig. 2.2. An occupancy grid is analogous to a greyscale image, where the intensity is proportional to the likelihood that a cell is occupied. Like an image, if finer detail is required for the map a higher resolution grid can be used. As the occupancy grid requires no data association, can represent both occupied and free spaces and the certainty of the belief, is robust to noisy sensors, and can handle complex shapes without being as computationally expensive as a feature based approach, the occupancy grid is chosen to estimate the map of the environment.

After a map of the surrounding environment is generated, a control or planning algorithm can be used to compute a path to the goal which minimizes the likelihood of collision. It is assumed that the flight control system is able to maintain stable, controlled flight.

The techniques described here to address these problems are applicable to full three dimensional, six degree of freedom vehicle. Here motion is considered in a two dimensional environment for algorithm development.

## 2.2 System Models

### 2.2.1 Frames

The vehicle is located at position $x$, $y$ in an Earth-fixed frame (assumed to be an inertial frame). The body frame $B$ (defined by unit vectors $\mathbf{x}_b$ and $\mathbf{y}_b$) stays fixed with the vehicle with its origin at the center of gravity of the vehicle. The orientation is defined by heading $\psi$ with respect to the inertial frame. The occupancy grid coordinate frame $O$ (defined by unit vectors $\mathbf{x}_o$ and $\mathbf{y}_o$) translates with the vehicle sharing a common origin with frame $B$, but the orientation of its axes remain fixed relative to the inertial frame, see Fig. 2.3. Frame $O$ provides a link between the moving vehicle and the occupancy grid. It gives obstacle positions relative to the moving vehicle, but in the axes of the occupancy grid.

For this thesis, a slash in the subscript of a variable will be used to denote the coordinate frame used. For example, $y_{k/o}$ is the y-coordinate of point $k$ with respect to frame $O$.

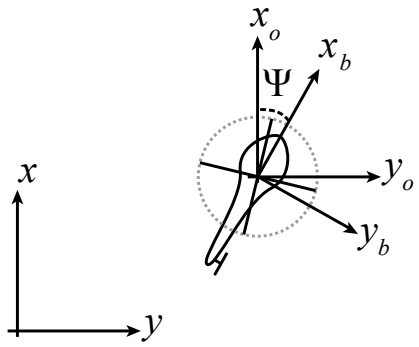



**Figure 2.3.** Coordinate Frames used in the system.

### 2.2.2 Kinematic Model

Velocities $u$ and $v$ are expressed in the body frame $B$. Setting $v = 0$ gives a non-holonomic vehicle (e.g. a tracked ground vehicle or a fixed-wing aircraft). Here the main application is a small autonomous rotorcraft, thus all three degrees of freedom $(u, v, \psi)$ are retained. The control inputs are $a_x$, $a_y$, and $\omega$ (acceleration in $x_b$ and $y_b$ and turn rate).

$$\dot{x} = u\cos\psi - v\sin\psi \tag{2.2}$$

$$\dot{y} = u\sin\psi + v\cos\psi \tag{2.3}$$

$$\dot{\psi} = \omega + \mathcal{N}(0, \sigma_\omega^2) \tag{2.4}$$

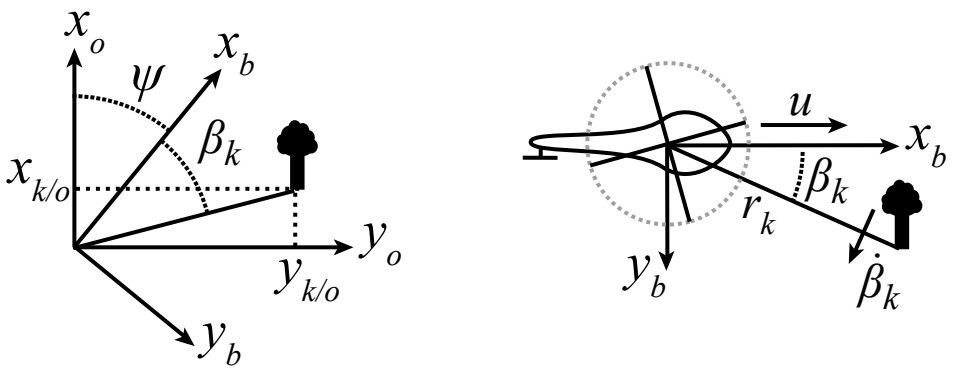
$$\dot{u} = a_x + \mathcal{N}(0, \sigma_{a_x}^2) \tag{2.5}$$

$$\dot{v} = a_y + \mathcal{N}(0, \sigma_{a_y}^2) \tag{2.6}$$

Here $\mathcal{N}(0, \sigma^2)$ denotes a Gaussian random variable with mean 0 and standard deviation $\sigma$. The noise is due to wind, and unmodelled noise in the controller.

### 2.2.3 Sensor Model

The camera is fixed to the vehicle pointed along the $\mathbf{x}_b$ axis. It obtains measurements of bearing and optical flow (bearing rate) generated by objects within the




(a) Computation of $\beta$ (b) Computation of $\dot{\beta}$

**Figure 2.4.** Sensor Model

field of view. To model the camera, bearings to each object are computed

$$\beta_k = \arctan \frac{y_{k/o}}{x_{k/o}} - \psi \tag{2.7}$$

where $x_{k/o}$ and $y_{k/o}$ are the coordinates of the $k^{th}$ obstacle expressed in frame $O$. The bearing rate is computed by taking the time derivative of Eq. 2.7

$$\dot{\beta}_k = \frac{u \sin \beta_k}{r_k} - \frac{v \cos \beta_k}{r_k} - \dot{\psi} \tag{2.8}$$

where $r_k$ is the distance between the camera and the object. This can be seen graphically in Fig. 2.4.

Across the camera field of view, many feature points will be tracked and generate values of optical flow. Often, multiple feature points will be tracked for the same obstacle. Having too many measurements could slow down the estimation process considerably, while not adding much new information if many of the measurements represent the same obstacle. To limit the number of measurements, the field of view of the camera is divided into a number of regions, each of equal angular width $\Delta\beta$. The resolution of $\Delta\beta$ is arbitrary and can vary based on implementation; however, it is limited at the upper end by the pixel resolution of the camera. Figure 2.5 displays an example of optical flow vectors from an image. It also shows how the camera field of view is divided into distinct regions.

Dividing the field of view is equivalent to having an array of optical flow sensors,

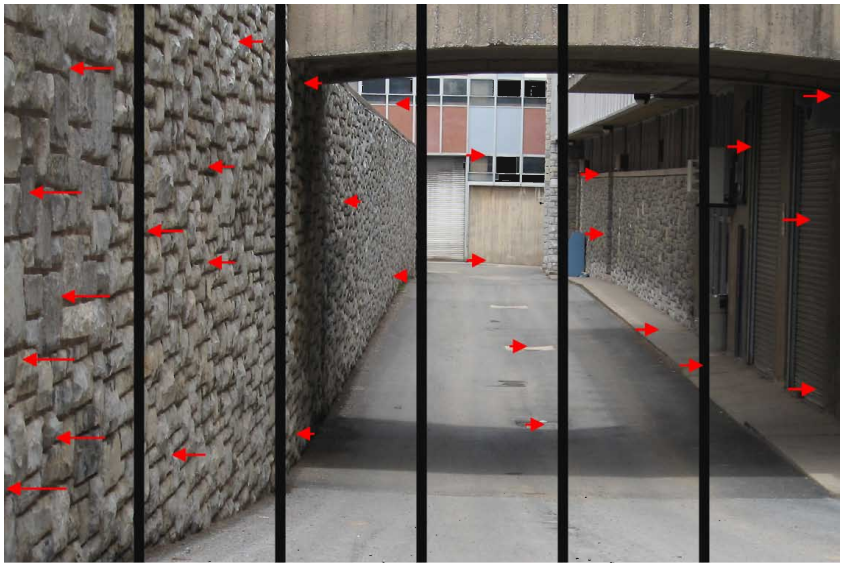


**Figure 2.5.** Discretized camera field of view. The red arrows represent simulated optical flow vectors. As this is only in two dimensions, only the horizontal component of optical flow is shown.

the $n^{th}$ pointing along $\beta_n$ and viewing a region defined by $\beta_n \pm \frac{\Delta\beta}{2}$. As this thesis is concerned only with a static environment, the closer an obstacle is along a given bearing the larger the magnitude of its optical flow vector, as can be seen in Eq. 2.8. Because of this, in each region, the bearing rate with the largest magnitude is taken as the value measured, as this corresponds to the closest obstacle in that region. The optical flow measurement is assumed to be corrupted by Gaussian noise where $\sigma_{\dot{\beta}}$ is the standard deviation of the Gaussian which is a constant across the field of view.

The measurement vector $\mathbf{z}$ is

$$\mathbf{z} = \left[u,\ v,\ \dot{\psi},\ \beta^T,\ \dot{\beta}^T\right]^T + \mathcal{N}(0, \Sigma_{\mathbf{z}}) \tag{2.9}$$

$$\Sigma_{\mathbf{z}} = \begin{bmatrix} \sigma_u^2 & 0 & 0 & 0 & 0 \\ 0 & \sigma_v^2 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{\dot{\psi}}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \Sigma_{\dot{\beta}}^2 \end{bmatrix} \tag{2.10}$$

Where $\beta$ and $\dot{\beta}$ are vectors containing the measurements of bearing and optical

flow for each of the $N$ regions.

$$\begin{aligned} \beta &= [\beta_1,\ \beta_2, \ldots, \beta_n]^T \\ \dot{\beta} &= \left[\dot{\beta}_1,\ \dot{\beta}_2, \ldots, \dot{\beta}_n\right]^T \qquad (2.11) \\ \Sigma_{\dot{\beta}} &= \sigma_{\dot{\beta}}^2 \mathbf{I}_N \qquad (2.12) \end{aligned}$$

where $\mathbf{I}_N$ represents an identity matrix of size $N$. Note that all measurements have Gaussian noise except $\beta$. The $\beta$ measurement does not have noise as it is defined to be the center of the region where the particular measurement of $\dot{\beta}$ is obtained. The width $\Delta\beta$ is captured in the occupancy grid implementation of the sensor model. This is discussed in the following section and in Chapter 3.

### 2.2.4 Range Model

As discussed above, the sensor obtains measurement of optical flow and bearing. To update the map, an estimate of obstacle location is necessary. As bearing to the obstacle is already known, estimating the range to the obstacle will provide the location of the obstacle in polar coordinates. To compute an estimated range $r^*$ for a given measurement $\dot{\beta}$, Eq. 2.8 is solved for $r$

$$\begin{aligned} r &= f(\mathbf{z}) \\ r^* &= \frac{1}{\dot{\beta} + \dot{\psi}}(u \sin\beta - v\cos\beta) \qquad (2.13) \end{aligned}$$

As there is noise in optical flow measurement, uncertainty in estimates of velocity and heading rate, and uncertainty of value of $\beta$ within the measurement region, an uncertainty in the estimate $\sigma_{r^*}$ is also needed. Three possible methods for computing $r^*$ and $\sigma_{r^*}$ are compared:

1. Particle Transform: This method generates many random particles based on the measurement probability distribution and propagates them through the function $f$ in Eq. 2.13 to then compute $r^*$ and $\sigma_{r^*}$. This method preserves the non-linearity of the system but at a high computational cost.

2. Unscented Transform: This method uses a small set of specifically placed particles to represent the measurement probability distribution and propa-

gates them through $f$ to then compute $r^*$ and $\sigma_{r^*}$. This method is much faster to compute than the particle transform[20].

3. Linearization: This method uses the measurements directly to compute $r^*$ and linearizes $f$ to compute $\sigma_{r^*}$. This is the fastest to implement.

Each method will use the measurement vector $\mathbf{z}$ and uncertainty $\Sigma_{\mathbf{z}}$. It is important to note that $\Sigma_{\mathbf{z}}$ has been modified slightly from Section 2.2. In the creation of the measurements, $\sigma_\beta$ is given to be 0, as the $\beta$ value used is that for the center of the region and no noise is added. As the measurement can come from anywhere within the specified region, the distribution of possibilities is approximated as a Gaussian with standard deviation $\sigma_\beta$.

Each method is described in detail below and all three are compared to determine which is the best suited for this application.

### Particle Transform

The particle transform retains the full non-linearity of Eq. 2.13. A series of particles $\mathbf{Z}$ are sampled randomly from the Gaussian distribution $\mathcal{N}(\mathbf{z}, \Sigma_{\mathbf{z}})$. These are then propagated through $f$ such that

$$\mathbf{S} = f(\mathbf{Z}) \tag{2.14}$$

In this step, some particles generate a range less than zero. This implies measuring an obstacle behind the camera, not in the field of view. As the camera can only measure obstacles in the field of view, this computed range is illogical. Thus, any particles with a range less than zero are omitted in the recovery of mean and standard deviation. The computed mean and standard deviation of $\mathbf{S}$ become $r^*$ and $\sigma_{r^*}$, respectively.

### Unscented Transform

For an $n$ state estimation, $2n+1$ Sigma Points are sampled symmetrically about the mean, $\bar{\mathbf{z}}$, with covariance, $\Sigma_{\mathbf{z}}$. For this case, there are 5 states, $\begin{bmatrix} u & v & \dot{\psi} & \beta & \dot{\beta} \end{bmatrix}$,

and therefore 11 Sigma Points.

$$\mathbf{Z} = \left[ \begin{array}{ccc} \bar{\mathbf{z}} & \bar{\mathbf{z}} + \eta\sqrt{\Sigma_{\mathbf{z}}} & \bar{\mathbf{z}} - \eta\sqrt{\Sigma_{\mathbf{z}}} \end{array} \right] \tag{2.15}$$

where $\eta$ is a scale factor that determines the spread of the Sigma Points. The square root of the covariance is computed using Cholesky decomposition such that $\Sigma_{\mathbf{z}} = \sqrt{\Sigma_{\mathbf{z}}}^T \sqrt{\Sigma_{\mathbf{z}}}$. Once the Sigma Points are computed, they are each propagated through $f$ in Eq. 2.13 to produce $\mathbf{S}$, a $2n+1$ vector of ranges. The mean and covariance of the estimated range are then computed.

$$\mathbf{S} = f(\mathbf{Z}) \tag{2.16}$$

$$\bar{\mathbf{s}} = \mathbf{S}\ \mathbf{w}_m \tag{2.17}$$

$$\Sigma_{\mathbf{s}} = [\ \mathbf{S} - \bar{\mathbf{s}}\ ]^T\ \mathbf{W}_c\ [\ \mathbf{S} - \bar{\mathbf{s}}\ ] \tag{2.18}$$

where $\mathbf{w}_m$ and $\mathbf{W}_c$ are weight factors, and $r^* = \bar{\mathbf{s}}$ and $\sigma_{r^*}^2 = \Sigma_{\mathbf{s}}$.

$$\eta = \alpha\sqrt{n} \tag{2.19}$$

$$\mathbf{w}_{m,1} = \tfrac{\alpha^2-1}{\alpha^2} \qquad \mathbf{w}_{m,i} = \tfrac{1}{2n\alpha^2} \tag{2.20}$$

$$\mathbf{W}_{c,1} = \tfrac{\alpha^2-1}{\alpha^2} + (1 - \alpha^2 + \gamma) \qquad \mathbf{W}_{c,i} = \tfrac{1}{2n\alpha^2} \tag{2.21}$$

Typically $\alpha$ is chosen to be 1, and for Gaussian distributions, $\gamma = 2$ is optimal[20].

In this implementation similar to the particle transform, often times at least one Sigma Point per region exists where $\mathbf{S} < 0$. Unlike the particle transform, the negative ranges cannot simply be discarded. For the unscented transform, all of the Sigma Points are needed to represent the probability distribution. This problem is documented by Huster who suggests two ways resolve the issue: to make $\eta$ smaller (by making $\alpha$ smaller) or to alter $\sqrt{\Sigma_{\mathbf{z}}}$ by rotating it by a unitary matrix[21]. To ensure that all values are positive, the optimal $\gamma = 2$ was kept, but $\alpha = 0.09$ was used to reduce the spread of the Sigma Points, as this is the simpler of the two posed solutions. Reducing the spread of the Sigma Points reduces the amount of the uncertainty propagated through the non-linear function, providing a less accurate representation of the non-linearity[21].

### Linearization

In the linearized method, $r^*$ is directly computed from Eq. 2.13 using the measurements, $\mathbf{z}$. The Jacobian of Eq. 2.13 is computed and used to determine the uncertainty in range estimate:

$$\nabla_{r^*} = \left[\frac{\partial r^*}{\partial u} \; \frac{\partial r^*}{\partial v} \; \frac{\partial r^*}{\partial \dot{\psi}} \; \frac{\partial r^*}{\partial \beta} \; \frac{\partial r^*}{\partial \dot{\beta}}\right]^T \tag{2.22}$$

$$\sigma_{r^*}^2 = \nabla_{r^*}^T \; \Sigma_{\mathbf{z}} \; \nabla_{r^*} \tag{2.23}$$

$$\begin{aligned} \sigma_{r^*}^2 \;\; = \;\; & (\dot{\beta}+\dot{\psi})^{-2}\left[\sigma_u^2 \sin^2\beta + \sigma_v^2 \cos^2\beta + \sigma_\beta^2 (u\cos\beta + v\sin\beta)^2\right] + \\ & (\dot{\beta}+\dot{\psi})^{-4}\left[(\sigma_{\dot{\beta}}^2 + \sigma_{\dot{\psi}}^2)(u\sin\beta - v\cos\beta)^2\right] \end{aligned} \tag{2.24}$$

### Results

Figure 2.6 shows a comparison between all three methods. For the comparison, values of $\dot{\beta}$ were generated using Eq. 2.8 with the parameters listed in Table 2.1.

**Table 2.1.** Values used for comparison

| | | | | | |
|---|---|---|---|---|---|
| $u$ | = | $4\,^{\mathrm{m}}/_{\mathrm{s}}$ | $\sigma_u$ | = | $0.2\,^{\mathrm{m}}/_{\mathrm{s}}$ |
| $v$ | = | $0\,^{\mathrm{m}}/_{\mathrm{s}}$ | $\sigma_v$ | = | $0.2\,^{\mathrm{m}}/_{\mathrm{s}}$ |
| $\dot{\psi}$ | = | $40\,^{\circ}/_{\mathrm{s}}$ | $\sigma_{\dot{\psi}}$ | = | $0.06\,^{\circ}/_{\mathrm{s}}$ |
| $\Delta\beta$ | = | $3\,^{\circ}$ | $\sigma_\beta$ | = | $0.5\,^{\circ}$ |
| $r$ | = | $10\,\mathrm{m}$ | $\sigma_{\dot{\beta}}$ | = | $2\,^{\circ}/_{\mathrm{s}}$ |
| $\alpha$ | = | 0.09 | $\gamma$ | = | 2 |

For $\beta > 20°$, all three methods produce similar results, the estimated obstacle location is very close to the actual location, and the uncertainty is also low. For $\beta \leq 20°$, the differences between the linearization, unscented transform, and particle transform are clear. The linearization does not account for the noisy measurements in the estimate of $r^*$ and underestimates the covariance. As the spread of the unscented transform is reduced to make all values non-negative, much of the non-linearity is removed in the computation of covariance (note how it more closely resembles the curve of the linearization).

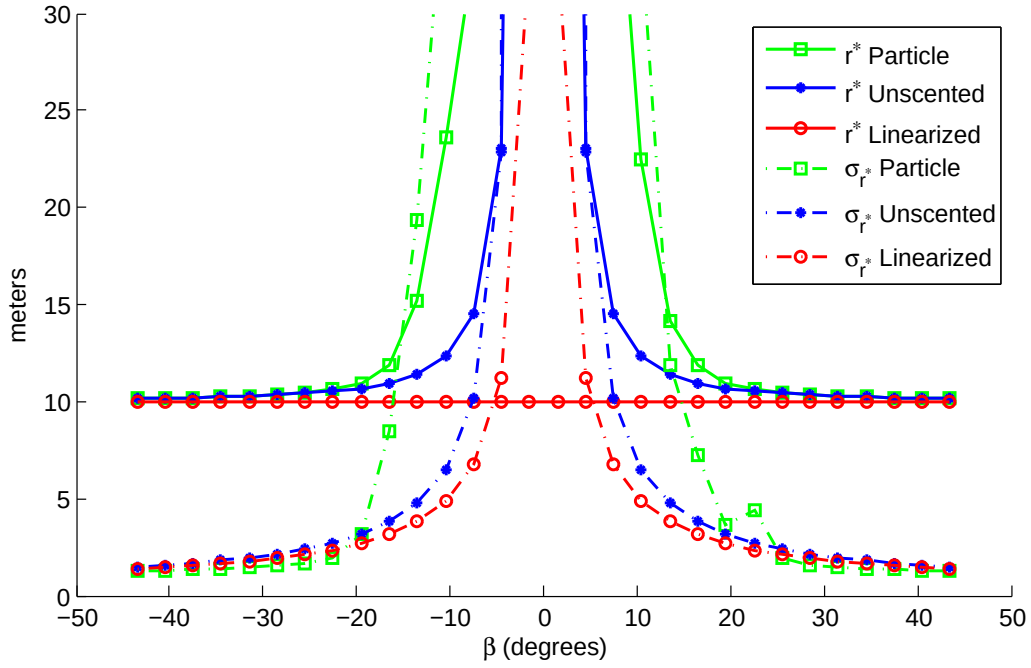



**Figure 2.6.** Comparison between the Particle Transform, Unscented Transform, and Linearization. $\dot{\beta}$ measurements were generated for parameters in Table 2.1.

Ideally, the particle transform would be used to retain the probability distribution through the non-linear function. However, to get an accurate representation of the distribution, 50000 particles were necessary for each angular region. Using fewer particles resulted in large variance in the computed $r^*$ and $\sigma_{r^*}$ on each run. Using this many particles is not feasible as it takes too much time to produce an estimate.

While the unscented transform does compute $r^*$ similar to that of the particle transform, the differences between these methods and the linearization occur for $\beta \leq 20°$. In this region, the value of $\dot{\beta}$ is on the order of $\sigma_{\dot{\beta}}$. Measurements on the same order as sensor noise produce poor estimates. When an estimate is known to not be accurate in advance, the uncertainty in that estimate is more important than the estimate itself. Here, $\sigma_{r^*}$ is nearly the same for both the unscented transform and the linearization; however, the linearization method is much easier to compute. As the linearization provides results similar to the unscented transform for the areas of interest and is not as computationally intensive, Equations 2.13 and 2.24 are used to compute $r^*$ and $\sigma_{r^*}$.

Adapting the linearized range model for use with an occupancy grid will be discussed in Chapter 3.

## 2.3 Occupancy Grid

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. 2.2. While occupancy grids have been well documented (e.g. [22, 23]), for completeness a derivation is presented here. This section follows the derivation given in Thrun [23].

As discussed in Section 2.1, the goal is to estimate a map of the environment, $p(\mathbf{m}|z_{1...t}, x_{1...t})$, where $z_{1...t}$ is the set of all measurements and $x_{1...t}$ is the set of all vehicle states up to time $t$. An occupancy grid is a numerical implementation of a Bayes filter which computes the estimate of $p(\mathbf{m})$ by discretizing the map into a finite number of cells.

$$\mathbf{m} = \{m_{ij}\} \tag{2.25}$$

where $m_{ij}$ is a cell in the map.

The probability that a cell is occupied ($p(m_{ij}) = 1$) or free ($p(m_{ij}) = 0$) is computed for each grid cell[22]. This results in a binary estimation problem over all possible maps, posing a computational problem. An environment with $M$ cells has $2^M$ possible maps. To make the problem tractable, the probability of a particular cell's occupancy is assumed to be independent of all other cells. This is a conservative approximation as possible information about the world is ignored (e.g. continuity between cells along the wall of a building). Because of the assumption of independence of cell occupancy, the problem of estimating $p(\mathbf{m}|z_{1...t}, x_{1...t})$ becomes $p(m_{ij}|z_{1...t}, x_{1...t})$ such that

$$p(\mathbf{m}|z_{1...t}, x_{1...t}) = \prod_{ij} p(m_{ij}|z_{1...t}, x_{1...t}) \tag{2.26}$$

and now the likelihood of each cell's occupancy can be computed independently.

The binary Bayes filter for this discretized map of a static environment is

$$p(m_{ij}|z_{1...t}, x_{1...t}) = \frac{p(z_t|m_{ij}, x_t)\ p(m_{ij}|z_{1...t-1}, x_{1...t-1})}{p(z_t|z_{1...t-1}, x_{1...t})} \tag{2.27}$$

where $p(m_{ij}|z_{1...t}, x_{1...t})$ represents the probability that a grid cell $m_j$ is occupied, given measurement history $z_{1...t}$ and vehicle path $x_{1...t}$. The first term in the numerator is the sensor model, the probability of getting a particular measurement based on the map and current state. The map is not known, and what is desired is the probability of the grid cell being occupied based on the measurements obtained and the current state $p(m_{ij}|z_t, x_t)$. 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. 2.27 to yield Eq. 2.28.

$$p(m_{ij}|z_{1...t}, x_{1...t}) = \frac{p(m_{ij}|z_t, x_t)\ p(z_t|x_t)}{p(m_{ij}|x_t)}\ \frac{p(m_{ij}|z_{1...t-1}, x_{1...t-1})}{p(z_t|z_{1...t-1}, x_{1...t})} \tag{2.28}$$

Both the measurement probability and map are assumed to be independent of state, $p(z_t|x_t) = p(z_t)$ and $p(m_{ij}|x_t) = p(m_{ij})$. These assumptions are founded in that the vehicle's location in the map does not change the likelihood of getting a measurement and that the map does not change based on the location of the vehicle. This simplifies Eq. 2.28 further to:

$$p(m_{ij}|z_{1...t}, x_{1...t}) = \frac{p(m_{ij}|z_t, x_t)\ p(z_t)}{p(m_{ij})}\ \frac{p(m_{ij}|z_{1...t-1}, x_{1...t-1})}{p(z_t|z_{1...t-1}, x_{1...t})} \tag{2.29}$$

To make the problem numerically better conditioned and easier to implement computationally, the occupancy is represented in log-odds form. First the odds form of Eq. 2.29 is computed

$$\begin{aligned} o(m_{ij}|z_{1...t}, x_{1...t}) &= \frac{p(m_{ij}|z_{1...t}, x_{1...t})}{1 - p(m_{ij}|z_{1...t}, x_{1...t})} \\ o(m_{ij}|z_{1...t}, x_{1...t}) &= \frac{p(m_{ij}|z_t, x_t)}{1 - p(m_{ij}|z_t, x_t)} \frac{p(m_{ij}|z_{1...t-1}, x_{1...t-1})}{1 - p(m_{ij}|z_{1...t-1}, x_{1...t-1})} \\ &\times \frac{1 - p(m_{ij})}{p(m_{ij})} \end{aligned} \tag{2.30}$$

which simplifies Eq. 2.29 by canceling the $p(z_t|\dots)$ terms. The factor $p(m_{ij})$ is the initial probability that the cell at location $(i,\ j)$ is occupied. As the environment

is initially unsurveyed, the initial probability of occupancy is $p(m_{ij}) = 0.5$, thus the term on the third line simplifies to 1.

Finally, the log-odds form of the binary Bayes filter is obtained by taking the logarithm of Eq. 2.30:

$$\begin{aligned} l_{t,ij} &= \log o(m_{ij}|z_{1...t}, x_{1...t}) \\ l_{t,ij} &= \log \frac{p(m_{ij}|z_t, x_t)}{1 - p(m_{ij}|z_t, x_t)} + \log \frac{p(m_{ij}|z_{1...t-1}, x_{1...t-1})}{1 - p(m_{ij}|z_{1...t-1}, x_{1...t-1})} \end{aligned} \tag{2.31}$$

The probability of occupancy can be recovered from the log-odds form by solving Eq. 2.31 for $p(m_{ij}|z_{1...t}, x_{1...t})$

$$p(m_{ij}|z_{1...t}, x_{1...t}) = \frac{\exp l_{t,ij}}{1 + \exp l_{t,ij}} \tag{2.32}$$

The second term on the right hand side of Eq. 2.31 is simply the accumulated log-odds of occupancy over all previous time steps:

$$l_{t,ij} = \log \frac{p(m_{ij}|z_t, x_t)}{1 - p(m_{ij}|z_t, x_t)} + l_{t-1,ij} \tag{2.33}$$

The estimation problem is now a recursive equation to compute the occupancy of each grid cell. This can be implemented easily and efficiently.

The first term on the right hand side of Eq. 2.33 is the inverse sensor model (i.e. the change in log-odds occupancy of a grid cell given a sensor measurement $z_t$). Given that there are usually multiple measurements, in this case one for each bearing $\beta_n$, the inverse sensor model for each measurement will be added together

$$l_{t,ij} = \sum_{n}^{N} \log \frac{p(m_{ij}|z_{t,n}, x_t)}{1 - p(m_{ij}|z_{t,n}, x_t)} + l_{t-1,ij} \tag{2.34}$$

Functionally, detecting an obstacle in a grid cell means the log-odds of occupancy in that cell is increased by some amount. The space between the vehicle and the obstacle must be free, thus the log-odds of the occupancy of the cells lying between the occupied cell and the vehicle is decreased. Cells outside the field of view get no new information so the log-odds values of those cells remain unchanged. The amount of the increase and decrease is dependent on the inverse

sensor model, defined in Chapter 3.

## 2.4 Summary

The navigation problem is described in Section 2.1. As vehicle states are assumed to be known, the problem is defined as reaching a goal location while safely avoiding obstacles using noisy measurements of optical flow and bearing from a single camera. To handle the noisy measurements, a map is used to estimate obstacle locations so that the uncertainty in the estimates can be tracked. Due to the complex nature of the environment, Kalman Filter (feature based) approaches are too computationally intensive to execute. An occupancy grid is chosen for its ability to handle both the noisy measurements and the complex environment.

Mathematical models of the vehicle kinematics and sensor are developed in Section 2.2. Motion is modeled as a second order function with control inputs of acceleration ($a_x,\ a_y$) and heading rate ($\dot{\psi}$) expressed in the body frame. Optical flow measurements are simulated by calculating the apparent bearing rate of stationary obstacles due to the translation and rotation of the moving vehicle. To obtain estimates of obstacle location, an estimated range to an obstacle is computed from the optical flow measurement.

A mathematical derivation of the occupancy grid is presented in Section 2.3. The estimation equation is represented in log-odds form to make the problem easier to implement computationally. In log-odds form, the measurement updates can simply be added to the map belief at the previous time step.

A solution of the navigation problem using the occupancy grid is given in Chapter 3 and simulation results are presented in Chapter 4.

Chapter 3

# System Design

The following chapter details the techniques used to solve the safe navigation problem defined in Chapter 2. The design is divided into three sections:

1. Inverse Sensor Model: The inverse sensor model is responsible for updating the occupancy grid with the obtained measurements. The inverse sensor model is divided into two parts, range and direction. Functions are created to approximate the probability distribution in order to save time and computations.

2. Local Occupancy Grid: A global occupancy grid continues to grow as a vehicle explores the area. As computational power is limited, storing and manipulating this large grid will be problematic. A local occupancy grid is used which limits the size of the grid and moves the grid with the vehicle. A motion model for the local occupancy grid is required and the formulation is described.

3. Vehicle Control: A potential field controller is used to create a desired heading. While these are well documented, the particular setup used here is detailed as well as the algorithm to create control inputs from the desired heading.

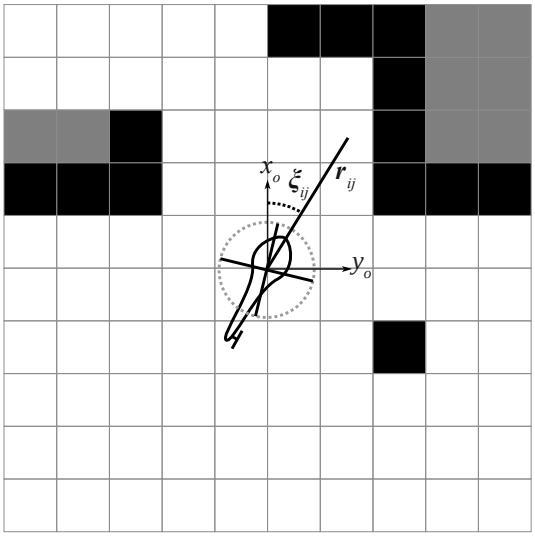



**Figure 3.1.** Schematic of the occupancy grid showing $r_{ij}$ and $\xi_{ij}$

## 3.1 Grid Cell Coordinates

To describe locations in the occupancy grid, both Cartesian and polar coordinates are used. As the coordinate systems defined so far are all Cartesian, knowing the location of cells in the occupancy grid in Cartesian coordinates is a relatively simple task. The coordinates $(x_{ij/o},\ y_{ij/o})$ are location of the center of the cell at $(i,\ j)$ as defined in frame $O$ which translates with the vehicle. However, as the sensor model is defined in a polar coordinate frame, a polar representation of the grid cell locations is beneficial for the inverse sensor model. Polar coordinates are also useful in the vehicle controller as distances to obstacles are used. Thus, $r_{ij}$ is defined as the distance from the vehicle to a grid cell, and $\xi_{ij}$ is the angle from the $\mathbf{x}_o$ axis to the center of the grid cell.

$$r_{ij} = \sqrt{x_{ij/o}^2 + y_{ij/o}^2} \tag{3.1}$$

$$\xi_{ij} = \arctan \frac{y_{ij/o}}{x_{ij/o}} \tag{3.2}$$

Figure 3.1 shows a schematic.

## 3.2 Inverse Sensor Model

The inverse sensor model is used to update the occupancy grid with measurements. As discussed in Section 2.2, the camera field of view is divided into a series of regions of equal angular width. For each region, the maximum magnitude optical flow measured is used as the measurement but the location of that measurement within the region is discarded. From the value of optical flow, an estimated range to the obstacle is computed, thus the obstacle location is known in polar coordinates. As such, the inverse sensor model is comprised of two pieces: first, a function of range to the obstacle; second, a function of angular position to reflect the bearing of the measurement and the angular resolution of the sensor.

### 3.2.1 Range

As discussed in Section 2.2, a linearized range model is used to compute an estimate of range $r^*$ and the uncertainty in the estimate $\sigma_{r^*}$ from the measurements of optical flow $\dot{\beta}$ in each sector $\beta_n$.

In addition to the Gaussian uncertainty described by $\sigma_{r^*}$, 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. Here, a 95%/99.7% rule is used. For a Gaussian distribution, 95% of the values lie within 2 standard deviations from the mean and 99.7% of the values lie within 3 standard deviations from the mean.

$$p \begin{cases} > 0.5, & r^* - 2\sigma_{r^*} \geq r \geq r^* + 3\sigma_{r^*} \\ = 0.5, & r > r^* + 3\sigma_{r^*} \\ < 0.5, & 0 < r < r^* - 2\sigma_{r^*} \end{cases} \tag{3.3}$$

Here, the obstacle is assumed to most likely exist 2 standard deviations in front of and 3 standard deviations behind the estimated range, and this region should have a probability of occupancy greater than 0.5. The region greater than 3 standard deviations behind the estimate is unknown, and thus the probability remains 0.5. Finally, the region greater than 2 standard deviations in front of the estimate is believed to be unoccupied and thus have a probability of occupancy

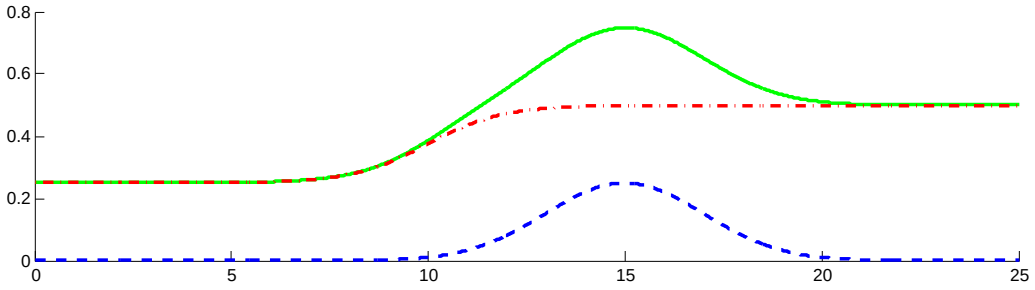



**Figure 3.2.** The range vs. probability (solid green) is constructed by combining a Gaussian (dashed blue) and sigmoid (dash-dotted red). Here $r^* = 15$ and $\sigma_{r^*} = 2$. Note the comparatively low probability of occupancy for $r < r^*$.

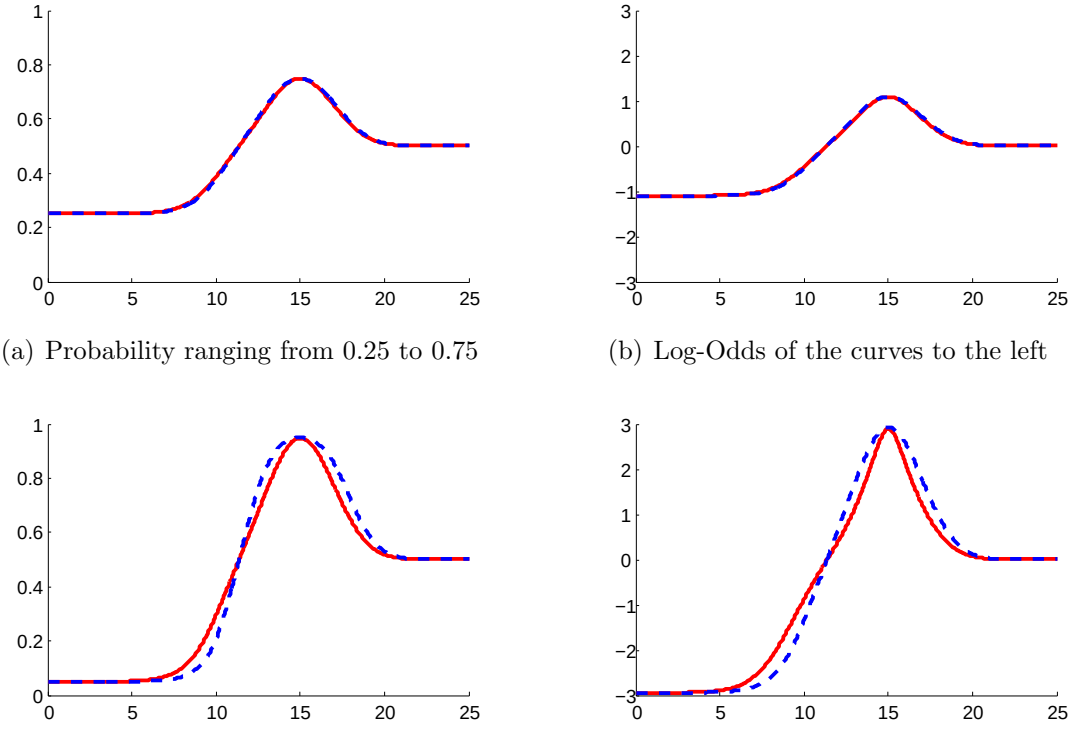
less than 0.5. To model this behavior a sigmoid curve and a Gaussian curve are combined to produce the probability curve of occupancy as a function of range, seen in Fig. 3.2.

One problem with the linearization and this approach is when obstacles are far away or not present. For distant obstacles, $\dot{\beta} \approx -\dot{\psi}$, from Eq. 2.8, and thus $\dot{\beta} + \dot{\psi} \approx 0$. Equation 2.13 will yield $r^* \approx \infty$, which makes sense for a distant object. Unfortunately, Eq. 2.24 will also estimate $\sigma_{r^*} \approx \infty$. While a large estimate uncertainty for a distant object is logical, it is problematic. As can be seen in Eq. 3.3, if $\sigma_{r^*} > 2r^*$, the occupied region extends from the estimated location all the way to the camera. Thus, there is no region to mark as free space between the estimated location and the sensor.

For this reason, a limit is set for the value of $r^*$. When $r^* > r_{max}$, $r^*$ will be set to $r_{max}$ and $\sigma_{r^*}$ will be set to a small value. By doing this, when a very far away obstacle is measured or no obstacle is present, the grid will be updated to reflect the knowledge of free space.

The same problem exists for obstacles located in the direction of motion. As mentioned in Section 2.1, these obstacles generate almost no optical flow separate from the rotation of the vehicle; from Eq. 2.8, again $\dot{\beta} \approx -\dot{\psi}$. This is currently handled as described above, such that often the area directly in the path of motion will be marked as free space. The control section addresses this problem, and implements methods to properly resolve the area in the path of motion.

In the final equation in the derivation of the occupancy grid, Eq. 2.34, the inverse sensor model must be in log-odds form in order to add it to the previous



(a) Probability ranging from 0.25 to 0.75

(b) Log-Odds of the curves to the left

(c) Probability ranging from 0.05 to 0.95

(d) Log-Odds of the curves to the left

**Figure 3.3.** Examples of approximate inverse sensor model curves. Subfigures a and c use a Gaussian and Sigmoid to generate the inverse sensor model probability (solid red lines). The log-odds representation is then calculated and shown in subfigures b and d (solid red lines). Subfigures b and d use a Gaussian and sigmoid to generate an approximate of the log-odds representation (dashed blue lines). These values are then converted back to probability form and are shown in subfigures a and c (dashed blue lines). Here $r^* = 15$ and $\sigma_{r^*} = 2$.

log-odds belief. Thus, the probability form inverse sensor model must be converted to log-odds form. While this is possible, performing this calculation over every grid cell for every measurement is computationally intensive. Thus an approximation for the log-odds form of the probability curve shown in Fig. 3.2 is desired. Figure 3.3 shows the method of approximation.

Figure 3.3(a) shows an inverse sensor model probability curve (solid red line). The log-odds representation of this curve is shown in Fig. 3.3(b) as the solid red line. For this case, the probability and log-odds representation curves are of very similar shape, but shifted and scaled. Thus, an approximated log-odds representation is generated by combining a sigmoid with a Gaussian, shown in Fig. 3.3(b) as

the dashed blue line. The probability representation of this approximated log-odds curve is computed and shown in Fig. 3.3(a) as the dashed blue line.

For this case, the true and approximated probability curves appear nearly identical. The average error between the two curves is $\epsilon_{rms} = 0.0054$ for $5 \leq r \leq 21$, the region where the curves differ. This similarity holds well for $0.1 \leq p(m_{ij}|z_t, x_t) \leq 0.9$. Outside this range, the non-linearities of the log-odds conversion become apparent.

Figures 3.3(c) and 3.3(d) show the same procedure as above, but for a larger range of probabilities. While the approximated curve does not exactly match the true range model, the log-odds form can still be reasonably approximated as a sigmoid and a Gaussian. For $0.05 \leq p(m_{ij}|z_t, x_t) \leq 0.95$ the average error between the two probability curves is $\epsilon_{rms} = 0.0513$ for $5 \leq r \leq 21$.

In log-odds form, the range portion of the inverse sensor model is approximated by the following:

$$\begin{aligned} f_n(r) &= -c_1 \left\{ 1 + \exp\left[ \frac{2\pi(r - r_n^* + 2\sigma_{r_n^*})}{\sigma_{r_n^*}\sqrt{3}} \right] \right\}^{-1} \\ &\quad + \frac{c_2}{\sigma_{r_n^*}\sqrt{2\pi}} \left\{ \exp\left[ -\frac{(r - r_n^*)^2}{2\sigma_{r_n^*}^2} \right] \right\} \end{aligned} \tag{3.4}$$

Here $r_n^*$ is the estimated range obtained from the optical flow measurement in the $n^{\text{th}}$ 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.

Note that in Eq. 3.4, the log-odds value for free space (the amplitude of the sigmoid) is not scaled. This amplitude could be scaled identically to the Gaussian $-c_1(\sigma_{r_n^*}\sqrt{2\pi})^{-1}$. This would incorporate the idea that as the uncertainty in the range estimate increases, the magnitude of the log-odds belief should decrease. This would be the case for a Gaussian distribution; as the standard deviation increases, a lower probability is spread across a wider area. However, a greater uncertainty in obstacle range does not change the belief that the region between the vehicle and the obstacle is free space. Because of this, the log-odds value for believed free space remains a constant, $-c_1$.

### 3.2.2 Direction

A scaling factor is calculated to account for the bearing of the measurement and the uncertainty in vehicle heading. To model the uncertainty in vehicle heading, a Gaussian would ideally be used to scale the value calculated from Eq. 3.4. However, as the range estimate is for the entire angular region because the specific bearing is not kept, the scaling factor over the width of the region should be a constant. To achieve this constant value over the width of a measurement region which tapers off similar to a Gaussian at the edges, a sigmoid function is again used to compute the scaling factor as a function of the difference between $\xi$ and the center of the angular region:

$$g_n(\xi) = \left(1 + \exp \frac{c_3(|\xi \ - \ \beta_n \ - \ \psi| \ - \ 0.5\Delta\beta \ - \ 1.25\sigma_\psi)}{\sigma_\psi}\right)^{-1} \tag{3.5}$$

where $c_3$ is a weighting parameter, $\Delta\beta$ is the angular width of the measurement region, and $\sigma_\psi$ is the uncertainty in the current heading estimate.

For the grid cell located at $(x_{ij},\ y_{ij})$ , the log-odds of occupancy induced by a measurement of optical flow in the $n^{\text{th}}$ region $z_{t,n}$ can be computed by evaluating $r_{ij}$ and $\xi_{ij}$ for that cell and computing

$$\log \frac{p(m_{ij}|z_{t,n}, x_t)}{1 - p(m_{ij}|z_{t,n}, x_t)} = f_n(r_{ij})g_n(\xi_{ij}) \tag{3.6}$$

An example is shown in Fig. 3.4.

The inverse sensor model given in Eq. 3.6 will update the occupancy grid with measurements obtained from the camera. The following section provides details on how the occupancy grid was implemented in this application to store estimates of occupied space.

## 3.3 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. As local obstacle avoidance is the focus of this research, a local occu-

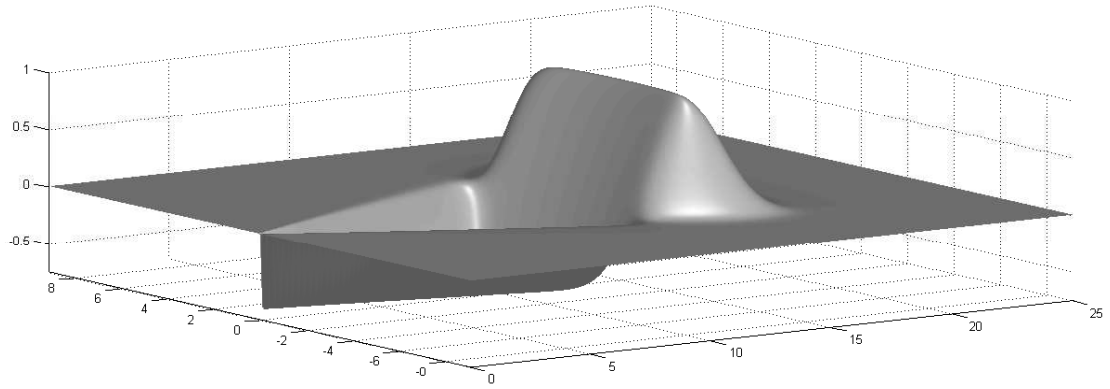



**Figure 3.4.** Example of the Inverse Sensor Model in log-odds form. The example shows an obstacle located at $r^* = 15$, indicated by the peak, with $\sigma_{r^*} = 1.2$. Between the sensor and the obstacle is likely to be unoccupied, indicated by the trench. Regions outside the sensor field of view (or occluded by obstacles) have zero change to their log-odds of occupancy. Here the angular width of the region is $\Delta\beta = 30°$. To create the high resolution, the grid cells are 0.25 m on each side.

pancy grid (fixed to the vehicle) is used. The 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. For the local occupancy grid, the grid origin remains fixed to the vehicle, Fig. 3.5. Then, the grid can either rotate with the vehicle, or the vehicle can rotate within the grid.

Grid orientation can be fixed to any convenient frame. Vehicle (and thus grid) motion is computed using a motion update step. The complexity of this motion update depends in part on the choice of grid orientation. If the grid remains fixed with the vehicle, both a rotation and translation are performed as part of the motion update. If the orientation of the local occupancy grid remains fixed to an inertial frame, only a translation is necessary.

Computationally the occupancy grid representation of the environment can be treated as an image, and techniques developed for image processing (e.g. blurring, convolution, rotation) and libraries used for image processing (e.g. OpenCV, MATLAB Image Processing Toolbox) can be used to translate and rotate the local occupancy grid as it moves with the vehicle. In any update where one grid cell does not map directly to another, some blurring (loss of certainty in the estimates) will occur. For a rotation, the grid cells will only match if the rotation is a mul-

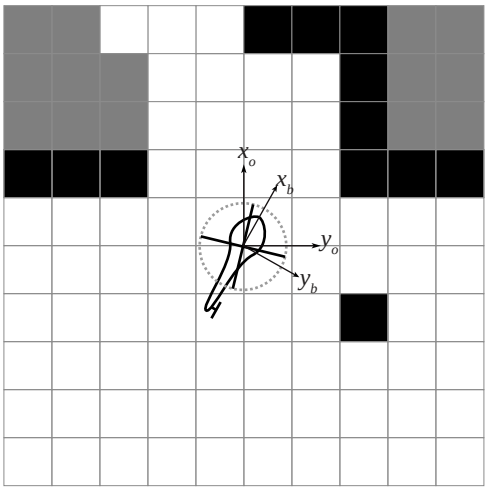



**Figure 3.5.** Schematic of local occupancy grid. Cells which are known to be free are white, those which are known to be occupied are black, those which are unknown are grey. Note that the center of the occupancy grid is at the origin of the $O$ frame.

tiple 90°. In order to keep the grid correctly matched to the vehicle orientation many small rotations will take place, thus creating a large loss of precision over time. Additionally, a rotation is a time consuming task. For these reasons, the orientation of the grid is fixed in frame $O$.

By fixing the orientation in frame $O$, only a translation of the grid is necessary for the motion update. Here this is performed using a convolution. The convolution kernel is generated based on the translated area of a grid cell, see Fig. 3.6. The percentage of the original cell area overlapping a surrounding cell determines the value of the convolution integral.

$$L_{t+1} = \mathbf{K} * L_t \tag{3.7}$$

$$\begin{aligned} \mathbf{K} &= \mathbf{K}_y * \mathbf{K}_x \\ \mathbf{K}_y &= \left[ \max\left(\frac{-\dot{y}\Delta_t}{\Delta_{grid}}, 0\right) \quad 1 - \left|\frac{\dot{y}\Delta_t}{\Delta_{grid}}\right| \quad \max\left(\frac{+\dot{y}\Delta_t}{\Delta_{grid}}, 0\right) \right] \\ \mathbf{K}_x &= \left[ \max\left(\frac{+\dot{x}\Delta_t}{\Delta_{grid}}, 0\right) \quad 1 - \left|\frac{\dot{x}\Delta_t}{\Delta_{grid}}\right| \quad \max\left(\frac{-\dot{x}\Delta_t}{\Delta_{grid}}, 0\right) \right]^T \end{aligned} \tag{3.8}$$

where $\Delta_{grid}$ is the size of a grid cell, the $*$ operator indicates a convolution, and $\max(n, 0)$ returns $n$ if $n > 0$.

The rate of motion update is in principle arbitrary. In practice is it affected

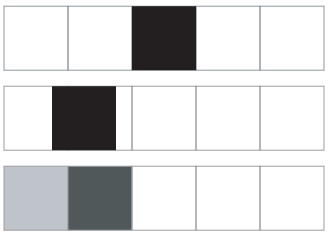


**Figure 3.6.** Example translation of occupancy grid in the $y_o$ direction

by grid size, vehicle motion, and computational considerations. Performing the motion update at every time step poses two problems. First, a convolution takes computational time. Performing a two-dimensional convolution at 50Hz slows the operation of the algorithm. Second, performing the convolution introduces blur when the translation does not map one cell exactly to another (Fig. 3.6). 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 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 are only performed when the translation is greater than or equal to the length of a grid cell. Additionally, the motion updates are 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.

To model the reduction in certainty of the grid as the vehicle moves, the kernel does not sum to one. By making the sum of the kernel less than one, the magnitude of each cell goes down at each iteration, and thus the certainty in the estimate will also be reduced. Thus the motion update kernel becomes:

$$\begin{aligned}
\mathbf{K}_y &= w_c \times \left[\max\left(\frac{-\Delta_y}{\Delta_{grid}} - 1, 0\right) \quad \max\left(1 - \left|\frac{-\Delta_y}{\Delta_{grid}} - 1\right|, 0\right) \quad 0 \right. \\
&\qquad \left. \max\left(1 - \left|\frac{+\Delta_y}{\Delta_{grid}} - 1\right|, 0\right) \quad \max\left(\frac{+\Delta_y}{\Delta_{grid}} - 1, 0\right)\right] \qquad (3.9) \\
\mathbf{K}_x &= w_c \times \left[\max\left(\frac{+\Delta_x}{\Delta_{grid}} - 1, 0\right) \quad \max\left(1 - \left|\frac{+\Delta_x}{\Delta_{grid}} - 1\right|, 0\right) \quad 0 \right.
\end{aligned}$$

$$\max\left(1-\left|\frac{-\Delta_x}{\Delta_{grid}}-1\right|,0\right) \quad \max\left(\frac{-\Delta_x}{\Delta_{grid}}-1,0\right)\Bigg]^T \tag{3.10}$$

where $\Delta_y = \dot{y}\Delta_t$, $\Delta_x = \dot{x}\Delta_t$, and the kernels are evaluated when $\Delta_y \geq \Delta_{grid}$ or $\Delta_x \geq \Delta_{grid}$.

The value of $w_c$ is the overall magnitude of the kernel. For $w_c = 1$, there is no loss of information over the motion update. If $w_c = 0$, the log-odds of the occupancy grid will go to 0 after the motion update (i.e. no knowledge of the surroundings). For $0 < w_c < 1$, the magnitude of the log-odds of the occupancy grid will be reduced after the motion update, and thus the certainty of the occupancy grid will be reduced.

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.

The local occupancy grid is implemented as follows:

1. Initialize the log-odds occupancy grid to all zeros (0.5 probability).
2. Update the grid with measurements using the inverse sensor model.
3. Determine how far the vehicle has moved since the last motion update.
4. If the distance is greater than the length of a grid cell, use the convolution to perform a motion update.
5. Repeat from Step 2.

## 3.4 Vehicle Control

While the methods described are applicable for a vehicle with three degrees of freedom $[u\ v\ \psi]$, a non-holonomic vehicle is used to simplify vehicle control (i.e. $v$ and $\dot{v}$ are set to be 0). Thus the trajectory planner consists of a heading generator and the controller uses the desired heading to compute control inputs, $a_x$ and $\omega$.

The heading generator uses a potential field type approach to avoid obstacles and reach the goal[5, 24, 25]. The particular implementation in this thesis is similar to the virtual force field and vector field histogram methods used by Borenstein and

Koren[26, 27]. Regions of high probability of occupancy represent high potential (to be avoided) and the goal is represented by a sink. The desired heading is determined by a gradient. This gradient is computed at the vehicle (i.e. the origin of the occupancy grid) as

$$\nabla = (1 - w_g)\nabla_{grid} + w_g\nabla_{goal} \tag{3.11}$$

where $w_g$ is a factor to weight goal seeking vs. obstacle avoidance, $\nabla_{grid}$ is the gradient term from the occupancy grid and $\nabla_{goal}$ is the gradient term resulting from the sink representing the goal. The desired heading is computed from $\nabla$ as

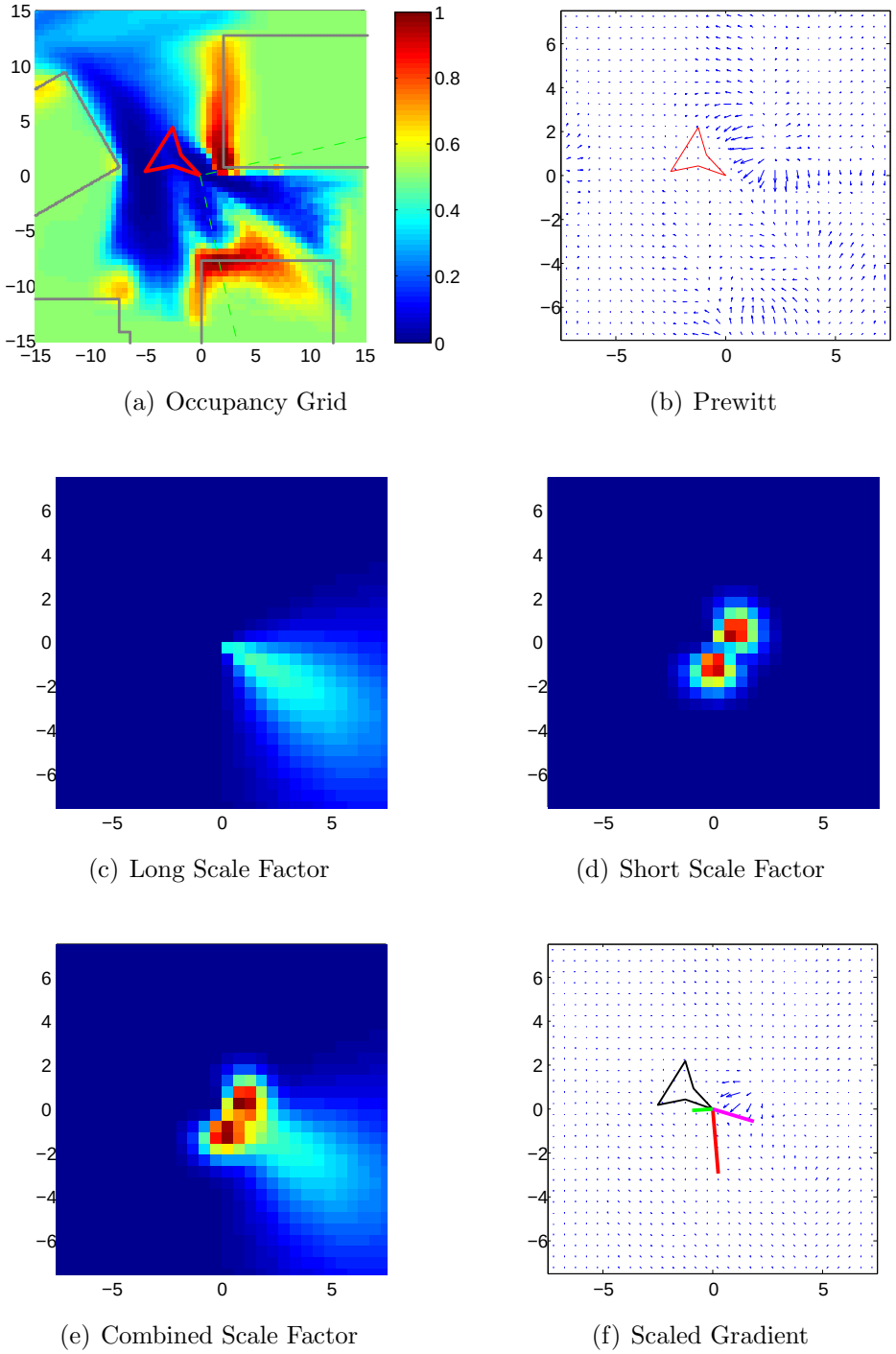
$$\psi_{des} = \arctan \frac{\nabla_y}{\nabla_x} \tag{3.12}$$

where $\nabla_y$ and $\nabla_x$ are the $y$ and $x$ components of the gradient $\nabla$, respectively. Put simply, the controller will attempt to steer the vehicle in the direction of steepest descent towards the goal.

Figure 3.7 shows the steps in creating $\nabla_{grid}$ (the gradient vector due to the occupancy grid) at a particular point in time. Only a region in the vicinity of the vehicle (here, a 30 by 30 cell box centered on the vehicle) is considered. The gradient of the occupancy grid in probability form is computed, $\nabla^*_{grid}$. This is then scaled by $K$ ($\nabla_{grid} = K \times \nabla^*_{grid}$) so that the location of an obstacle relative to the vehicle is included in the trajectory planner. Figure 3.7(a) shows the probability values of the local occupancy grid with colors representing $p(m_{ij} = \text{occupied})$. The probabilities are computed from the log-odds representation using Eq. 2.32.

### 3.4.1 Grid Gradients

First a Prewitt edge detector (seen in Fig. 3.7(b)) is used to find gradients in both the $x_o$ and $y_o$ directions, $\nabla^*_{grid}$. This will give large values only at edges (i.e. at boundaries between occupied and unoccupied space) and small values in regions of similar occupancy probability.




(a) Occupancy Grid

(b) Prewitt

(c) Long Scale Factor

(d) Short Scale Factor

(e) Combined Scale Factor

(f) Scaled Gradient

**Figure 3.7.** Creating the potential field controller. The vehicle is shown by the cranked triangle. Subfigure (f) shows the gradient vectors with non-uniformly scaled magnitudes: short green = $\nabla_{grid}$, medium magenta = $\nabla_{goal}$, long red = $\psi_{des}$

### 3.4.2 Scaling

Second, scaling factors $k_{ij}$ are introduced to allow weighting to the occupancy grid gradient term, $\nabla^*_{grid}$. This factor is created in two parts. The first part to the scale factor is a long term planner. It looks ahead and determines which direction is better in terms of avoiding obstacles. The second part looks closer to the vehicle and in a wider span to be sure that the vehicle is not headed towards or will turn into an area of high probability of occupancy.

#### Long-term

For the first part of the scale factor, values near the vehicle should still be weighted more heavily than values further away to avoid the more immediate obstacles. However, values at a distance are included to cause the vehicle to turn before it strikes them. A Gaussian is used to scale the gradient field with the standard deviation being a function of vehicle speed. This makes the occupancy grid gradient influence to be a function of time, reacting to obstacles that are a certain time away. The standard deviation of the Gaussian is

$$\sigma_{dist} = \frac{c_4\sqrt{u^2 + v^2} + c_5}{\Delta_{grid}} \tag{3.13}$$

where $\Delta_{grid}$ is the length of a grid cell and $c_4$ and $c_5$ are parameters which allow further tuning of the scaling factor in its dimension and standard deviation.

Values in the direction of motion should have greater influence on desired heading than those from values to the sides and those already passed. A second Gaussian was used to weight the occupancy grid gradients based on vehicle orientation.

$$\sigma_{dir} = \frac{\pi}{c_6\sqrt{u^2 + v^2} + c_7} \tag{3.14}$$

where $c_6$ and $c_7$ are more tuning parameters. Again the standard deviation is a function of vehicle speed. At high speeds, obstacles in the direction of motion are of a larger concern than obstacles to the sides. At slow speeds, the vehicle can more quickly change directions, thus there is a wider field to consider to determine the best possible path.

These two Gaussians were combined to create the scaling factor $k_{ij}^{\text{long}}$

$$k_{ij}^{\text{long}} = \exp\left(-\frac{r_{ij}^2}{2\sigma_{dist}^2}\right) \times \exp\left(-\frac{(\xi_{ij}-\psi)^2}{2\sigma_{dir}^2}\right) \tag{3.15}$$

where $r_{ij}$ is the distance from a grid cell to the vehicle, and $\xi_{ij} - \psi$ is the angle between the unit vector to the grid cell and the direction of motion. This is shown in Fig. 3.7(c).

**Short-term**

For the second part, the construction of the scale factor is similar to that of the first, using a Gaussian for both distance and direction. However, the parameters are different to reflect the different objective. Additionally, the areas off center are weighted more heavily to keep the vehicle from turning into any obstacles. This is accomplished by using an absolute value on the directional Gaussian:

$$k_{ij}^{\text{short}} = c_8 \times \exp\left(-\frac{(r_{ij}-c_9)^2}{2c_{10}^2}\right) \times \exp\left(-\frac{(c_{11}-|\xi_{ij}-\psi|)^2}{2c_{12}^2}\right) \tag{3.16}$$

This is shown in Fig. 3.7(d).

The scale factor is simply the combination of the two parts:

$$k_{ij} = k_{ij}^{\text{long}} + k_{ij}^{\text{short}} \tag{3.17}$$

The term $\nabla_{grid}$ is generated by taking the sum of the scaled occupancy grid gradient.

$$\nabla_{grid} = \sum_{ij} \left(k_{ij} \times \nabla_{grid,ij}^{*}\right) \tag{3.18}$$

Figure 3.7(d) shows the scale factor due to the close avoidance, Fig. 3.7(c) shows the scale factor from the longer trajectory planning and Fig. 3.7(e) shows the combination of these, $k_{ij}$. Figure 3.7(f) shows the result of applying the scale factors to the initial gradient, shown in Fig. 3.7(b).

### 3.4.3 Goal

The goal gradient $\nabla_{goal}$ is simply a unit vector in the direction of the goal.

$$\nabla_{goal} = \left[ \frac{x_{goal} - x}{\sqrt{(x_{goal} - x)^2 + (y_{goal} - y)^2}} \ , \ \frac{y_{goal} - y}{\sqrt{(x_{goal} - x)^2 + (y_{goal} - y)^2}} \right]^T \tag{3.19}$$

### 3.4.4 Control Inputs

Once a desired heading $\psi_{des}$ is determine from Eq. 3.12, the turn rate command is computed as a function of the difference between the desired current heading.

As described earlier, the estimator provides poor estimates of obstacle location in the region in front of the camera. A dither term is incorporated which causes the vehicle to sweep back and forth, resolving the area that would otherwise be unknown.

$$\omega = c_{13}(\psi_{des} - \psi)^{c_{14}} + c_{15} \cos\left(\frac{2\pi \times t}{c_{16}}\right) \tag{3.20}$$

where $\omega$ is in units of $^{\text{rad}}/_{\text{s}}$ and $\psi$ and $\psi_{des}$ are in radians.

To allow smaller radius turns (which may be necessary in environments with densely packed obstacles) a speed controller is implemented. This allows vehicle speed to vary (within a set range) with commanded turn rate, which permits smaller radius turns.

$$a_x = c_{17} - c_{18}\omega \tag{3.21}$$

Here $c_{17}$ and $c_{18}$ are the acceleration control parameters.

## 3.5 Summary

This chapter has presented a method for navigation through an unsurveyed environment. The occupancy grid serves as a map to store the locations of free space and obstacles. By making a local occupancy grid, the system is tractable and less computationally intensive while still retaining the information necessary to navigate through the surrounding environment. A convolution is used to translate the occupancy grid with the vehicle motion. The inverse sensor model updates the grid with the measurements of bearing and bearing rate obtained from the cam-

era. A potential field method controller is used to create a desired heading based on the local occupancy grid and the desired goal location. Lastly, a simple difference controller generates control inputs based on the current and desired headings. The next chapter will provide simulation results showing the performance of this method.

Chapter 4

# Simulation Results

The following chapter presents the results obtained from a simulation designed to test the system proposed in Chapter 3. The simulation consists of navigation between 100 different starting locations and 3 different goal locations. The simulation is setup to represent a typical environment for navigation given the sensors described in Chapter 2.

The description of the simulated environment and the definition of parameters and constants presented earlier are described in Section 4.1. Results are presented in Section 4.2.

## 4.1 Setup

To demonstrate the utility of the proposed approach, simulations of flight through an environment modeled after the McKenna Military Operations in Urban Terrain (MOUT) site at Ft. Benning, GA (shown in Fig. 4.1) are conducted..

The urban town is made up of a series of walls, each represented by a line of points. Points are used for convenience in computing bearings and bearing rates to obstacles. Instead of having to ray cast across a simulated surface, measurements are computed for each point. The town limits are defined by an ellipse 87 m by 57 m rotated 20.° For the perimeter forest, tree trunks are represent by a circle of 8 equally spaced points. To be sure the algorithm works for all sizes of trees, diameters vary between 0.5 m and 1.5 m. One hundred and fifty trees are randomly located outside of the town, but no closer than 8 m to another tree. In addition to

**Figure 4.1.** McKenna MOUT Site at Ft. Benning, GA

convenience in implementing the simulation, the use of points to represent objects corresponds well to the actual computation of optical flow where feature points are found for each object and tracked across the field of view.

Three goal locations are selected in different parts of the urban setting. One is close to the center of the town, the other two are between buildings near the edge of the town. One hundred starting locations are randomly generated. These locations are either within or outside of the forest. The locations are no closer than 4 m to a tree and no closer than 10 m to one another. This provides an even spread across the environment while not starting the vehicle in an immediate avoidance scenario before it has a chance to map the environment.

The simulation is called a success if the vehicle reaches within 2 m of the goal. If the vehicle comes within 1 m of any obstacle, the simulation is marked as a failure. Additionally, due to memory constraints, the simulation is allowed to run for 60 s. As the maximum distance between any starting point and goal is 150 m, this means that the vehicle must travel faster than 2.5 $^{m}/_{s}$ on the longest route (ignoring the extra distance required to avoid obstacles).

The local occupancy grid is a square of 30 m per side. As the smallest obstacle to detect is no smaller than 0.5 m, each grid cell of the occupancy grid is also a square 0.5 m per side. As stated in Section 2.3, the initial belief of the environment is uncertainty. Thus, the occupancy grid is initially set with a probability of 0.5 and the log-odds representation of the grid is initialized to zeros.

As stated in Section 3.4, vehicle kinematics are assumed to be non-holonomic

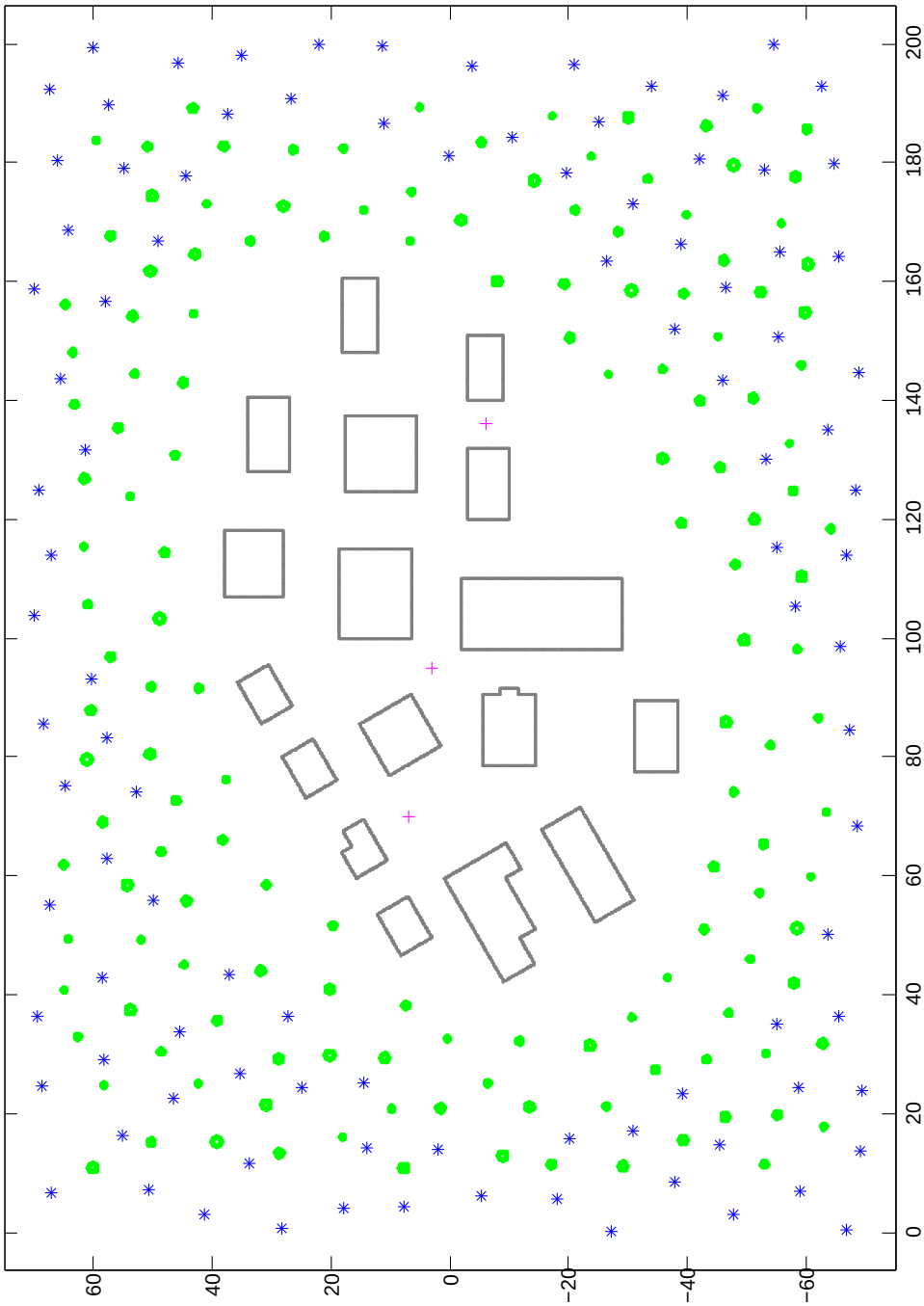



**Figure 4.2.** Simulated MOUT site surrounded by forest. Grey lines represent buildings, green dots represent trees. The blue stars are the 100 different starting locations and the magenta crosses are the goal locations.

(i.e. $v$ and $\dot{v}$ are 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. The vehicle is initialized with a speed of $4\,{}^{\mathrm{m}}/_{\mathrm{s}}$ and pointed in the direction of the goal. This sometimes positions the vehicle so that it is initially heading towards a tree. This is one reason for creating the starting locations such that they are no closer than 4 m to any obstacle. Limits are set on the maximum and minimum vehicle speed and turn rate control input:

$$\begin{aligned} 2.5\,{}^{\mathrm{m}}/_{\mathrm{s}} &\leq u \leq 5.5\,{}^{\mathrm{m}}/_{\mathrm{s}} \\ -1\,{}^{\mathrm{rad}}/_{\mathrm{s}} &\leq \dot{\psi} \leq 1\,{}^{\mathrm{rad}}/_{\mathrm{s}} \end{aligned}$$

Vehicle kinematics are computed at a rate of 50 Hz. The kinematics are also updated using a second order model (e.g. $x_{t+1} = x_t + \dot{x}\Delta t + \frac{1}{2}\ddot{x}\Delta t$). Measurements of optical flow are sampled at a rate of 10 Hz. Because of this, the controller only produces control commands at 10 Hz. If the controller were to run more frequently, there would not be any new information regarding obstacles, only a possible slight change in the positions of obstacles due to the distance traveled by the vehicle. As the occupancy grid cells are 0.5 m in size and the vehicle maximum velocity is $5.5\,{}^{\mathrm{m}}/_{\mathrm{s}}$, the maximum distance the vehicle could travel in the 0.1 s between controller updates, is slightly larger than the length of a grid cell. As the occupancy grid is only translated in a motion update when the vehicle has traveled the length of a grid cell, the occupancy grid could displace at maximum one cell between control updates.

For this simulation, the camera model has a field of view of 90° which is divided equally into 24 measurement regions, each 3.75° wide. As discussed in Section 3.2, there is a maximum range $r_{max}$ where values of $r^*$ beyond that return no useful information. For the parameters used in this simulation, $r_{max} = 27$ m. As the local occupancy grid is 30 m on a side, the furthest distance from the vehicle is 22 m to the corners of the grid. With $r_{max} = 27$ m, this means that all measurements can be represented in the local occupancy grid.

The standard deviation for parameters which contain Gaussian noise are given in Table 4.1.

**Table 4.1.** Standard Deviations

| | | | |
|---|---|---|---|
| $\sigma_\beta = 0.625°$ | | $\sigma_{\dot{\beta}} = 1.25\,^\circ/\mathrm{s}$ | |
| $\sigma_u = 0.2\,\mathrm{m/s}$ | | $\sigma_{a_x} = 0.05\,\mathrm{m/s^2}$ | |
| $\sigma_\psi = 1°$ | | $\sigma_\omega = 2\,^\circ/\mathrm{s}$ | |
| $\sigma_{\dot{\psi}} = 0.25\,^\circ/\mathrm{s}$ | | | |

**Table 4.2.** Constants

| | | | |
|---|---|---|---|
| $c_1 = 0.3$ | $c_2 = 1$ | $c_3 = 15$ | $c_4 = 1$ |
| $c_5 = 1.25$ | $c_6 = 0.99$ | $c_7 = 1.95$ | $c_8 = 0.0035$ |
| $c_9 = 1$ | $c_{10} = 0.75$ | $c_{11} = \frac{5\pi}{16}$ | $c_{12} = \frac{\pi}{6}$ |
| $c_{13} = 1.1$ | $c_{14} = 0.85$ | $c_{15} = 0.45$ | $c_{16} = 3$ |
| $c_{17} = 0.9$ | $c_{18} = 1.2$ | $w_g = 0.0385$ | $w_c = 0.9772$ |

There are 18 constants and 1 weighting parameter used in the system design. The values of these are provided in Table 4.2. Values $c_1 - c_3$ are for the inverse sensor model and are dependent upon the sensor used. The control weighting parameter $w_g$ and control constants $c_4 - c_{18}$ dictate the path followed by the vehicle. Limited tuning was performed on these values, and they are not necessarily optimal.

The magnitude of the motion update kernel $w_c$ is determined to be 0.9772 based on the speed of the vehicle and resolution of the local occupancy grid. It is decided that after the vehicle travels half the length of the grid, the certainty of a grid cell should be halved. For a a grid that is 30 m on a side, this should occur every 15 m. For a grid cell size of 0.5 m on a side, there will be 30 motion updates over that span.

$$w_c = 0.5^{\frac{1}{30}} \approx 0.9772 \tag{4.1}$$

## 4.2 Results

As previously stated, one run is executed from each starting location to each goal location for a total of 300 runs. Table 4.3 displays the raw data from these 300 runs. Initially, there appears to be a success rate of 89.33%, with 29 instances where the vehicle comes closer than 1 m to an obstacle. Upon looking at the data, in 19 of the 29 runs marked as crashes, the collision occured within the first 2 s of

the simulation.

**Table 4.3.** Results

| | Goal 1 | Goal 2 | Goal 3 | Total |
|---|---|---|---|---|
| Success | 89 | 89 | 90 | 268 |
| Crash | 8 | 11 | 10 | 29 |
| DNF | 3 | 0 | 0 | 3 |
| Total | 100 | 100 | 100 | 300 |

These collisions are most likely due to the vehicle being pointed directly at an obstacle. In an actual application, the vehicle would be aimed so that it is heading towards the goal, but initially away from any close obstacles. Most likely, these collisions could be avoided with better initial conditions. Table 4.4 shows the data if these collisions inside of 2 s are thrown out. The simulation now has a success rate of 95.37%.

**Table 4.4.** Adjusted Results

| | Goal 1 | Goal 2 | Goal 3 | Total |
|---|---|---|---|---|
| Success | 89 | 89 | 90 | 268 |
| Crash $> 2$ s | 2 | 4 | 4 | 10 |
| DNF | 3 | 0 | 0 | 3 |
| Total | 94 | 93 | 94 | 281 |

The reason for the three runs which did not finish is unknown. The data for these runs is not saved so one can only speculate on what occurred. One possibility is a simple lack of time to complete the course. Two of the 3 were starting at a distance greater than 130 m from the goal. Looking at the data, they both traveled over 250 m. The other was initially under 50 m from the goal. It would appear from this that there was ample time to reach the goal for all three of these runs, if the vehicle could take a direct line.

Due to the complexity of the environment and the potential field controller, it is possible that the shortest path to the goal had a lower potential than a longer path, due to the noisy measurements. In this case, the vehicle could have turned away from the optimal path to pursue one with a lower potential.

A second possibility is the local minima associated with potential field controllers. While overall the trend is to navigate towards the goal, there possibly exist spots in the environment where the draw of the goal and the repulsion from

the obstacles are equal to each other. One such example would be a corridor ending in a dead end towards the goal. The vehicle is drawn down the corridor to reach the goal, but is forced to turn away at the end. At some point leaving the corridor, the draw of the goal turns the vehicle back down towards the dead end, and the vehicle is stuck in an endless loop. Both of these possibilities could be solved with a more intelligent control architecture.

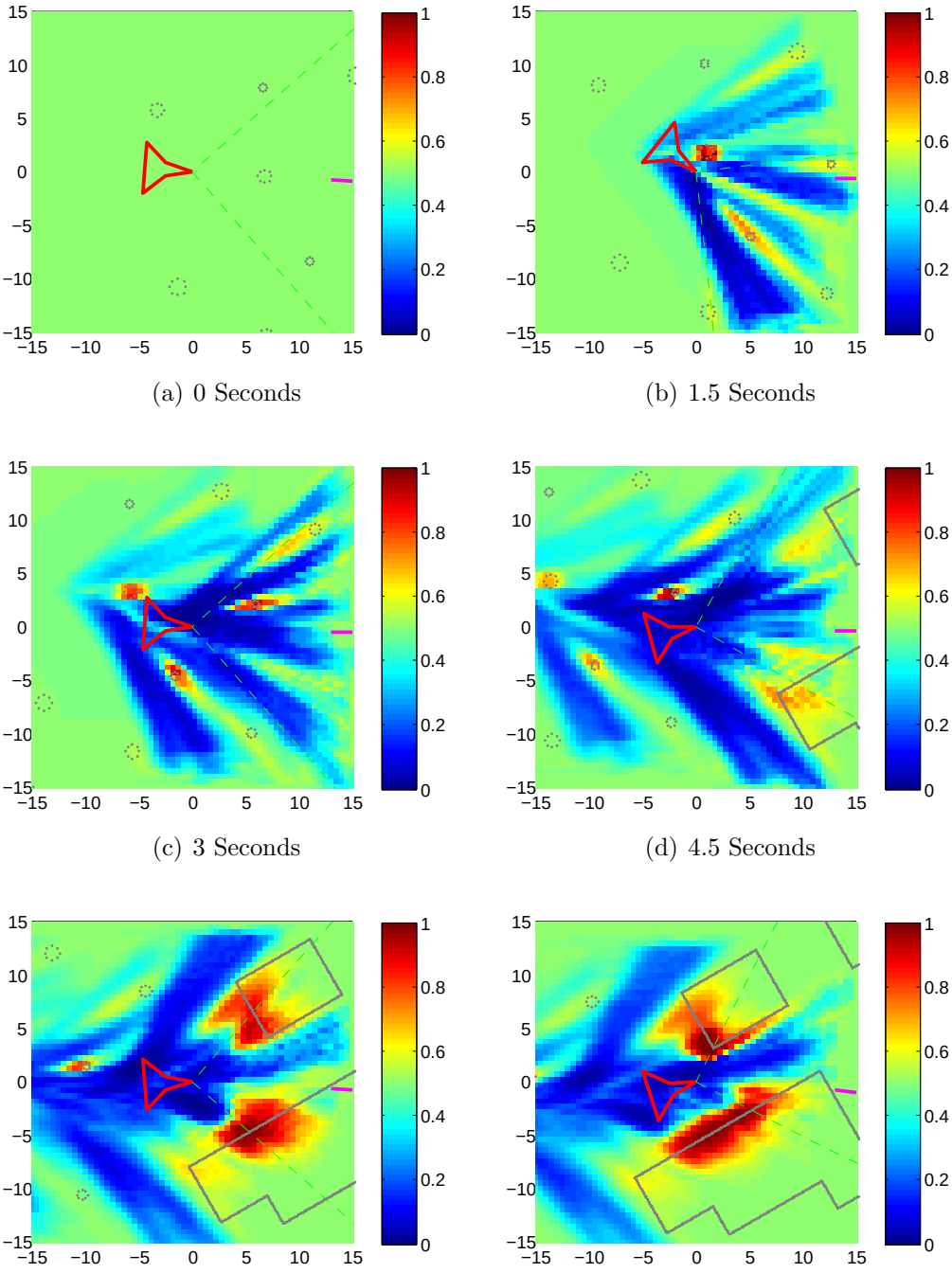
For the 273 runs which did not result in a collision, the average distance of closest approach is 1.786 m with a standard deviation of 0.4262 m. Over all of the 300 runs, the average velocity of the vehicle is 5.1069 $^{m}/_{s}$.

Figures 4.3 and 4.4 display the results from a representative run of the simulation. Obstacles (shown in gray) are 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 space with unknown occupancy, thus causing the vehicle to turn to improve its knowledge of the area directly in front. The added dither also aids in the resolution of the unknown area directly to the front.

Figure 4.5 shows the vehicle flight paths from a sample of 30 successful runs of the simulation, 10 to each goal. The effect of the added dither control can be observed when the vehicle operates in open space. When no obstacles are present to force an avoidance maneuver, the dither control causes a sinusoidal shape to appear in the vehicle path.

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. The model handles both small point obstacles and large complex shaped obstacles equally well.

The occupancy grid also maintains its memory of space which has left the field of view. This aids in the navigation around corners and trees, as the controller can use the information of what is directly to the sides of the vehicle even thought it is not currently in view. The convolution successfully translates the occupancy grid beliefs such that the grid memory is possible. The scaling of the convolution to less than one does cause the confidence of the cells to fade over time while not blurring the grid.




(a) 0 Seconds

(b) 1.5 Seconds

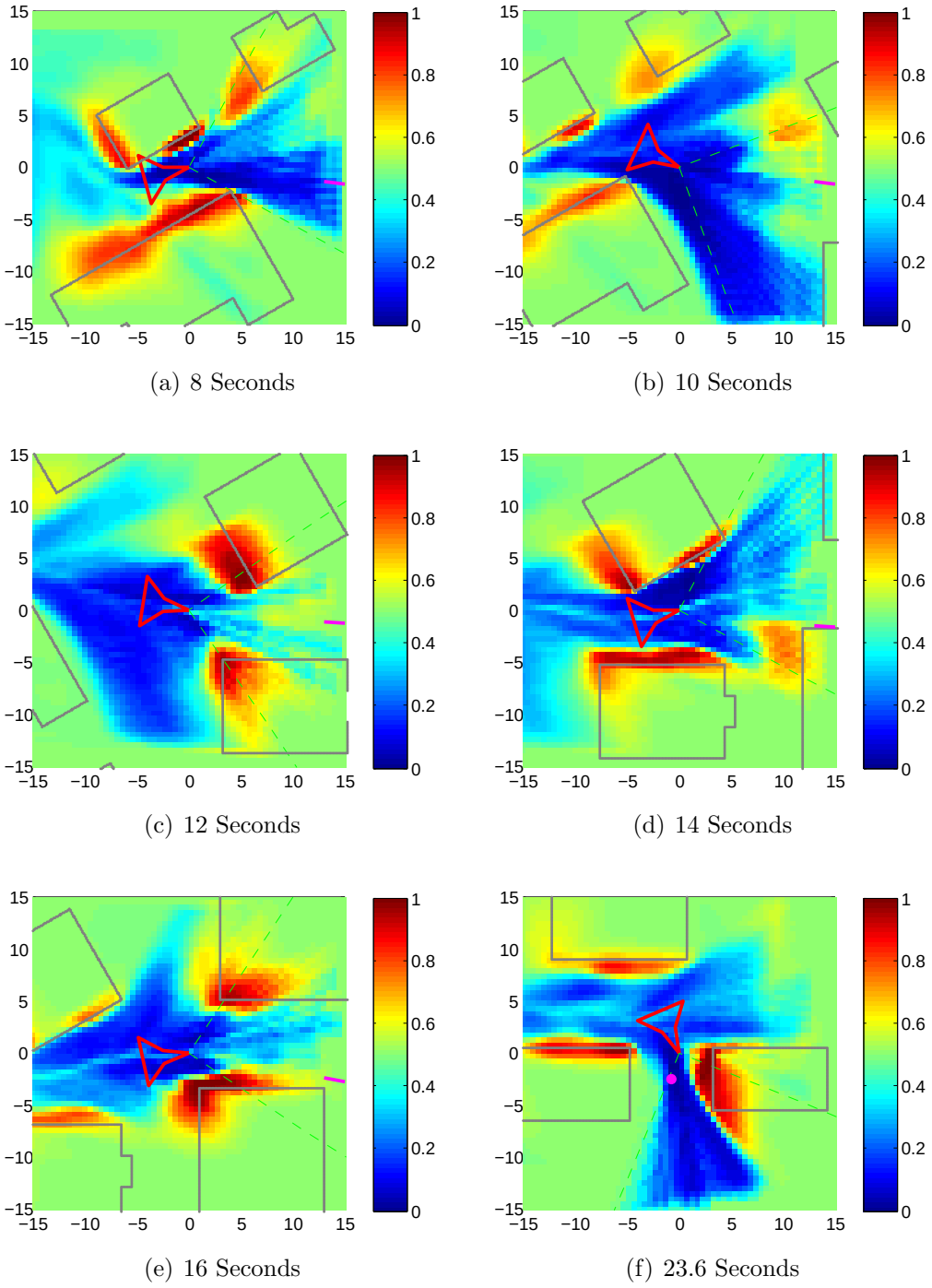
(c) 3 Seconds

(d) 4.5 Seconds

(e) 6 Seconds

(f) 7 Seconds

**Figure 4.3.** 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. Cell color represents occupancy probability: red = high, blue = low, green = 50% (uncertainty). Obstacles are shown as gray lines, where visible the magenta dot is the goal, otherwise the magenta line points towards the goal.



(a) 8 Seconds

(b) 10 Seconds

(c) 12 Seconds

(d) 14 Seconds

(e) 16 Seconds

(f) 23.6 Seconds

**Figure 4.4.** Continuation of the sequence of images from representative run from Fig. 4.3. Again, the vehicle is shown as a red cranked triangle, camera field of view is shown as green dashed lines. Cell color represents occupancy probability: red = high, blue = low, green = 50% (uncertainty). Obstacles are shown as gray lines, where visible the magenta dot is the goal, otherwise the magenta line points towards the goal.

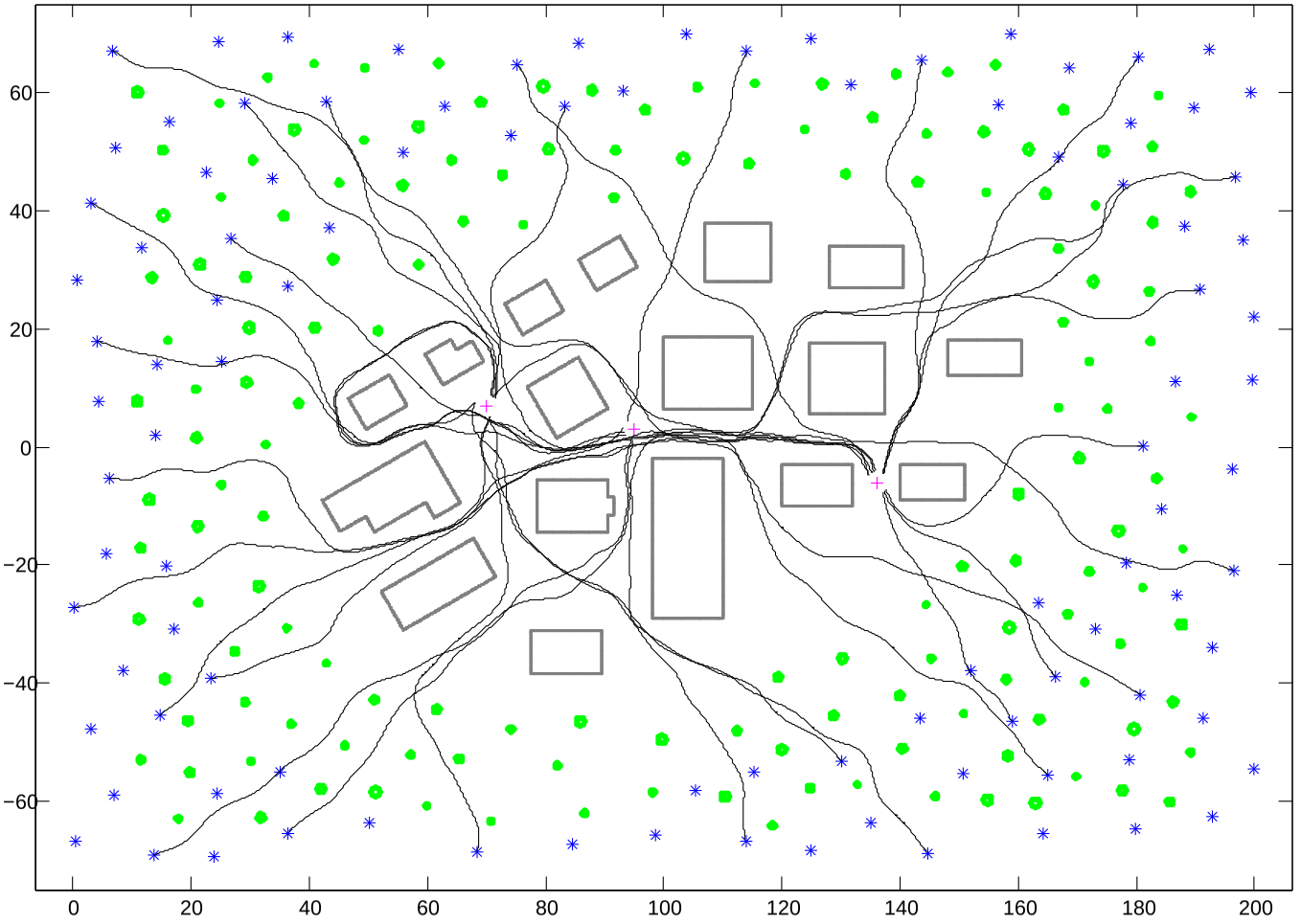



**Figure 4.5.** Flight paths from select simulations. The black lines here show the paths of 30 simulations which successfully navigate to the goal. The effect of the dither in the controller is apparent when the vehicle operates in open space.

Chapter 5

# Conclusion

Flying a small unmanned aerial vehicle (UAV) through cluttered environments has been the motivation for the obstacle avoidance algorithm presented in this thesis. Missions envisioned for small UAVs require low altitude flights among many obstacles such as search and rescue in forests and surveillance in urban environments. The close proximity and complex nature of these environments requires a system of navigation and obstacle avoidance using onboard sensors. This thesis focused on the problem of estimating obstacle locations and safe navigation to a known goal location using a monocular camera and GPS/INS.

The current technologies for obstacle avoidance rely heavily on LIDAR and RADAR, large sensors with great power requirements. However, payload limitations of small UAVs places significant restrictions on sensor size, weight, and power consumption. 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 consumption and light weight, vision sensors are passive, reducing the probability of detection.

The obstacle estimation problem poses many challenges. First, the camera provides measurements of bearings to obstacles and bearing rates of the obstacles due to motion of the vehicle (and thus camera). Both of these measurements are heavily corrupted with noise. This greatly reduces the certainty of information obtained from the camera. Second, the equations governing the vehicle motion and vision measurements are highly non-linear. The combination of noisy measurements with non-linear equations leads to significant uncertainties in the estimation problem

which must be accounted for.

To handle the noisy camera measurements and produce an accurate estimate of obstacle location, a robust estimator is necessary. In addition to the noisy sensor, the estimator must also handle sensor dropouts (e.g. a brownout while operating over a desert). A map is used to store estimates of obstacle locations. The map can continuously be updated with new information on obstacle location and take into account the noise in measurements. Additionally, the map retains the estimates of obstacle locations in the case of sensor dropout so that obstacle avoidance is still possible.

Traditional mapping techniques based on the Kalman Filter fail in this particular application. While the feature based approaches work well in environments populated with point obstacles (e.g. tree trunks in a forest), they quickly become intractable in large environments or complex environments where obstacles are difficult to define by features.

An occupancy grid approach is used to map the estimated locations of obstacles. As local obstacle avoidance is the focus of this research, a local occupancy grid is used. The local occupancy grid has a limited size governed by sensor field of view and computational considerations. The origin of the grid is fixed to the vehicle and in this application the orientation is fixed to an inertial frame. By keeping the grid centered on the vehicle, vehicle motion must be accounted for in the grid with a motion update step.

To make the problem numerically better conditioned and easier to implement computationally, the occupancy grid is represented in log-odds form. This allows for measurement updates to simply be added to the belief at the previous timestep. In addition, the motion updates can be performed using a convolution kernel.

Estimates of range to obstacles are produced from the bearing and bearing rate measurements obtained from the camera and estimates of vehicle state from an inertial navigation system corrected by GPS. The inverse sensor model takes the estimates of obstacle locations and increases the log-odds of cells corresponding to these estimates, and decreases the log-odds of cells between the vehicle and the estimated obstacle.

Vehicle navigation and obstacle avoidance is handled by a potential field trajectory generator. Regions of high occupancy represent high potential (to be avoided)

and the goal is represented by a sink. The trajectory planner attempts to steer the vehicle in the direction of steepest decent towards the goal.

A two-dimensional simulation of the described system was performed in an environment modeled after the McKenna Military Operations in Urban Terrain site at Ft. Benning, GA. The vehicle was started at various points in a forest surrounding the town with goal locations inside the town. Out of 281 simulation runs, the vehicle had a success rate of 95%, colliding with an obstacle 10 times and not reaching the goal due to time constraints 3 times. The occupancy grid provided both precise and accurate displays the estimates of obstacle and free space locations. The longer an obstacle was in the field of view of the camera, the more certain and more precise the estimated location became. The grid approach was able to handle both small point obstacles and large complex shaped obstacles equally well.

## 5.1 Summary of Contributions

### 5.1.1 Method for obstacle avoidance

A method for obstacle avoidance using only vision and GPS/INS sensors has been developed. This system fuses estimates of vehicle states from an inertial navigation system correctly by GPS with measurements from a camera to determine relative obstacle locations.

### 5.1.2 Estimator design

A map based on the occupancy grid was developed. Obstacle locations relative to the vehicle are estimated. This information was used by a trajectory planner to compute a safe path to the goal through a complex environment.

### 5.1.3 Performance verification through simulations

A simulation is run to test the performance of the designed system. Results of simulations show that the occupancy grid based implementation provides a solution

to the navigation problem. The locations of obstacles are accurately estimated and the vehicle reaches the goal on 95% of the simulation runs.

## 5.2 Recommendations for Future Work

### 5.2.1 Trajectory Planners and Controllers

While the gradient-based control algorithm implemented here worked well with an overall success rate of 95%, it (like all potential field approaches to path planning) is subject to local minima and the possibility of navigating into dead ends[28]. Additionally, the vehicle did have several collisions which would not be acceptable in a hardware implementation. A more intelligent trajectory planner should be used to minimize collisions and produce a closer to optimal path to the goal.

Combinations of global and local path planners has been successful in the past[13]. Using a path planning routine (such as A* or D*) could provide a route that leads to the goal, while a local potential field approach could ensure that the vehicle still steers away when navigating too close to perceived obstacles.

The control algorithm used here also assumed a non-holonomic vehicle ($v = 0$. Rotorcraft have the unique maneuvering capability of sidewards flight. By translating laterally, optical flow measurements could be obtained for objects in front of the vehicle. While a controller to utilize this capability would be more complicated than the one presented here, this would greatly improve the ability to resolve the area in front of the camera.

### 5.2.2 Three Dimensions

The work presented in this thesis accounts for only two dimensions, though the world is three-dimensional. While the two-dimensional environment used here is similar in many ways (avoiding tree trunks and building walls), in practice these small UAVs will have to navigate through doorways, under branches, and other obstacles with vertical components.

While the occupancy grid method used here can work in a third dimension (becomes a cube-like occupancy grid), the required computer memory and computational effort will grow exponentially. While three-dimensional occupancy grids

have been implemented, they used large vehicles with powerful computers[13]. For the smaller vehicles discussed in this thesis, new techniques might need to be researched to meet the computational requirements.

### 5.2.3 Hardware Implementation

The simulation results prove that the concept works in theory; however, an actual hardware implementation will prove if the system can function in real-time. While the models and equations designed in this thesis can be used in hardware, more development is necessary. An actual method for calculation of optical flow from a camera feed needs to be developed. Also, the vehicle will operate in a three-dimensional environment and thus have 6 degrees of freedom (DOF) (in contrast the the 3 DOF in this thesis). While a two-dimensional occupancy grid can still be used to represent the environment, models of vehicle (and thus camera) motion need to be derived for the 6 DOF.

# Bibliography


[1] United States Department of Defense, "Office of the Secretary of Defense Unmanned Systems Roadmap (2007–2032)," .

[2] Kim, J. and S. Sukkarieh (2007) "Real-time implementation of airborne inertial-SLAM," *Robotics and Autonomous Systems*, **55**, pp. 62–71.

[3] Davison, A. (2003) "Real-time simultaneous localisation and mapping with a single camera," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, vol. 2, pp. 1403–1410.

[4] Langelaan, J. W. (2007) "State Estimation for Autonomous Flight in Cluttered Environments," *Journal of Guidance, Control and Dynamics*, **30**(5), pp. 1414–1426.

[5] Khatib, O. (1986) "Real-time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotics Research*, **5**(1), pp. 90–98.

[6] Pollefeys, M., L. V. Gool, M. Vergauwen, et al. (2003) "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, pp. 262–267.

[7] Hrabar, S., G. Sukhatme, P. Corke, et al. (2005) "Combined optic-flow and stereo-based navigation of urban canyons for a UAV," in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pp. 3309–3316.

[8] Kim, J. and G. Brambley (2007) "Dual Optic-flow Integrated Navigation for Small-scale Flying Robots," *Australasian Conference on Robotics and Automation, 2007*.

[9] Roberts, J. M., P. I. Corke, and G. Buskey (2002) "Low-Cost Flight Control System for a Small Autonomous Helicopter," in *2002 Australasian Conference on Robotics and Automation*, ARAA, Auckland, New Zealand.

[10] Chahl, J. and A. Mizutani (2006) "An algorithm for terrain avoidance using optical flow," *American Control Conference, 2006*.

[11] Zufferey, J.-C. and D. Floreano (2005) "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.

[12] Netter, T. and N. Franceschini (2002) "A Robotic Aircraft that Follows Terrain Using a Neuromorphic Eye," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Lausanne, Switzerland.

[13] Scherer, S., S. Singh, L. Chamberlain, and M. Elgersma (2008) "Flying Fast and Low Among Obstacles: Methodology and Experiments," *The International Journal of Robotics Research*, **27**(5), pp. 549–574.

[14] Braillon, C., C. Pradalier, K. Usher, et al. (2006) "Fusion of stereo and optical flow data using occupancy grids," in *Proc. IEEE Intelligent Transportation Systems (ITSC'06)*, Toronto, Ont., pp. 1240–1245.

[15] Usher, K. (2006) "Obstacle Avoidance for a Non-holonomic Vehicle using Occupancy Grids," in *2006 Australasian Conference on Robotics and Automation*.

[16] Badino, H., U. Franke, and R. Mester (2007) "Free Space Computation Using Stochastic Occupancy Grids and Dynamic Programming," in *Workshop on Dynamical Vision, ICCV*, Rio de Janeiro, Brazil.

[17] Arbuckle, D., A. Howard, and M. Mataric (2002) "Temporal occupancy grids: a method for classifying the spatio-temporal properties of the environment," in *Intelligent Robots and System, 2002. IEEE/RSJ International Conference on*, vol. 1, pp. 409–414.

[18] Mitsou, N. and C. Tzafestas (2007) "Temporal Occupancy Grid for mobile robot dynamic environment mapping," in *Control & Automation, 2007. MED '07. Mediterranean Conference on*, pp. 1–8.

[19] Coué, C., C. Pradalier, C. Laugier, T. Fraichard, and P. Bessière (2006) "Bayesian Occupancy Filtering for Multitarget Tracking: An Automotive Application," *The International Journal of Robotics Research*, **25**(1), pp. 19–30.

[20] van der Merwe, R. and E. A. Wan (2001) "The Square-Root Unscented Kalman Filter for State and Parameter-Eestimation," in *International Conference on Acoustics, Speech, and Signal Processing*, pp. 3461–3464.

[21] Huster, A. (2003) *Relative Position Sensing by Fusing Monocular Vision and Inertial Rate Sensors*, Ph.D. thesis, Stanford University.

[22] Elfes, A. (1989) "Using occupancy grids for mobile robot perception and navigation," *IEEE Trans. Comput.*, **22**(6), pp. 46–57.

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

[24] Latombe, J.-C. (1991) *Robot Motion Planning*, Kluwer Academic Publishers.

[25] Choset, H., K. M. Lynch, S. Hutchison, et al. (2005) *Principles of Robot Motion: Theory, Algorithms, and Implementations*, The MIT Press.

[26] Borenstein, J. and Y. Koren (1989) "Real-time obstacle avoidance for fast mobile robots," *Systems, Man and Cybernetics, IEEE Transactions on*, **19**(5), pp. 1179–1187.

[27] ——— (1991) "The vector field histogram-fast obstacle avoidance for mobile robots," *Robotics and Automation, IEEE Transactions on*, **7**(3), pp. 278–288.

[28] Koren, Y. and J. Borenstein (1991) "Potential field methods and their inherent limitations for mobile robot navigation," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pp. 1398–1404 vol.2.