2007-04-12

Applications of Search Theory to Coordinated Searching by Unmanned Aerial Vehicles

Steven R. Hansen
Brigham Young University - Provo

APPLICATIONS OF SEARCH THEORY TO COORDINATED SEARCHING BY UNMANNED AERIAL VEHICLES

by

Steven R. Hansen

A thesis submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Mechanical Engineering

Brigham Young University

Thesis Completed
April 2007

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a thesis submitted by

Steven R. Hansen

This thesis has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

Date	Timothy W. McLain, Chair
Date	Craig C. Smith
Date	Michael Goodrich

BRIGHAM YOUNG UNIVERSITY

As chair of the candidate's graduate committee, I have read the thesis of Steven R. Hansen in its final form and have found that (1) its format, citations, and bibliographical style are consistent and acceptable and fulfill university and department style requirements; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the graduate committee and is ready for submission to the university library.

Date

Timothy W. McLain
Chair, Graduate Committee

Accepted for the Department

Matthew R. Jones
Graduate Coordinator

Accepted for the College

Alan R. Parkinson
Dean, Ira A. Fulton College of
Engineering and Technology

ABSTRACT

APPLICATIONS OF SEARCH THEORY TO COORDINATED SEARCHING BY UNMANNED AERIAL VEHICLES

Steven R. Hansen

Department of Mechanical Engineering

Master of Science

Concepts in optimal search theory have been used in human-based aerial search since World War II. This thesis addresses the technical and theoretical issues necessary to apply this crucial theory to search path planning for Small Unmanned Aerial Vehicles (SUAVs).

A typical search often requires that more than one target be located. Accordingly, a method is presented to locate multiple targets in three dimensions, as well as to differentiate between them. However, significant error can be present when locating targets from an airborne platform, and the idea of target quality is also introduced as a way to describe the reliability of target estimates.

Flight test results are presented to validate the target differentiation algorithm. In this test, five out of six targets as close as 6.1 m apart are located and differentiated with less than four meters of error. This flight test also provides color information that is useful in generating artificial target images and understanding the target detection probability. Image skew is then incorporated into the detection probability model, and a function is derived that predicts target detection as a function of distance.

In order to measure the effectiveness of search algorithms with this model, the concept of a probability map is introduced. This map can be updated as the search progresses, and is

stored on a probability grid containing nodes that keep track of the probable target locations and the probability of detection. Using this tool, a search width is developed for a given airborne agent. The search width is then used to derive optimal search performance based on a given probability map and SUAV.

Finally, the concepts of efficiency and completeness are given specific definitions in the context of discrete search. These metrics are used to develop a search plan that focuses on efficiency, and one that focuses on completeness. Example simulations are used to illustrate the conditions under which each plan might be desirable, and a composite search strategy is presented that combines both plans.

ACKNOWLEDGMENTS

Thank you, Angelina, for loving me and being patient with me. Thank you, Dr. McLain, for trusting and teaching me. Thank you, Dr. Goodrich, for guiding the search and rescue efforts at BYU. And thank you, AFOSR, for funding this research (contract FA9550-04-1-0209).

Table of Contents

List of Tables

List of Figures

Nomenclature

α, β	vertical and horizontal viewing angles
α_{az}, α_{el}	camera azimuth and elevation angles
α_T	portion of a pixel's PSF overlapping the target projection
γ	target contact
$\mathbf{\Gamma}$	target contact set
$\mathbf{\Gamma_I}$	*image set* containing contacts from a single image
$\mathbf{\Gamma_j}$	*target set* containing contacts assigned to a single target
δ_g	node spacing in the probability grid
δ_m	distance in front of q_m along the sliding segment
δ_{pt}	distance between the PSF origin and the target boundary, in pixels
$\zeta(t^*)$	search completeness at t^*
$\eta(t^*)$	search efficiency at t^*
θ_o	observation angle, between the observation direction and straight down
λ	Lagrange multiplier
Λ	area detection probability
μ_*	mean detection time using plan $*$
ξ	search efficiency
ρ_*	probability density in region $*$
σ	standard deviation
σ_{PSF}	standard deviation of the test camera's point spread function
τ	time constant
Φ	generalized search effort, which could be in terms of t, L, A, etc.
ϕ, θ, ψ	UAV body roll, pitch, and heading angles
ϕ_*	search effort density in region $*$
χ	cluster of target contacts

χ_P	pixel color, which contains hue, saturation and value
χ_R	road color, which contains hue, saturation and value
χ_T	target color, which contains hue, saturation and value
ψ_c	commanded heading
ψ_g	the current greedy heading
$\psi_{g,last}$	the last greedy heading
ψ_p	test angle, used to determine the benefit of a given direction
$\psi_{p,max}$	the ψ_p corresponding to the maximum $\bar{p}'_{\psi_p}$
$\mathbf{A}$	camera intrinsic matrix
a_1, a_2	detection probability parameters
a, b	height and width of an ocean vessel wake
A, B	some area
$\mathbf{A_C}$	test point at a distance of $L_t/2$ from $\mathbf{C}$ in the $\hat{\mathbf{t}}$ direction
A_{diode}	Mag-Lite® diode projected area (m^2) on a plane one meter away
A_k	area represented by node group G_k
A_{NU}	the non-uniform probability region of the variable map
A_{obj}	cross-sectional area of a target object in m^2
A_p	the projected area of one pixel onto a plane one meter away, in m^2
A_T	the total area in the variable map
A_U	the uniform probability region of the variable map
$\mathbf{B_C}$	test point at a distance of $-L_t/2$ from $\mathbf{C}$ in the $\hat{\mathbf{t}}$ direction
$\mathbf{C}$	target center, which is a specific test point
$C_@$	denotes that the target is contained in the area where detection occurs
C_*	denotes that the target is contained in area $*$
C_B	battery storage in Coulombs
$\mathbf{d}$	vector from a test point to a contact origin
d	absolute distance
D	target detection event
d_f	distance traveled in a single flight
d_G	threshold distance from a UAV to a node group, used in group assignment

d_{jk}	closest distance from u_j to G_k
$D(t)$	target detection by time t
$\tilde{D}(t)$	target detection before time t
$D^*(t)$	target detection at time t
$d_{\gamma m}$	maximum contact assignment distance, used in target differentiation
d_χ	distance from cluster point χ
d_{jk}	distance from UAV u_j to group G_k
d_{un}	distance of some other UAV, u_n, to point q_{ψ_p}
$\mathbf{e}(\mathbf{T}, \gamma)$	error vector between $\mathbf{T}$ and γ
$\mathbf{E}(\mathbf{T}, \Gamma)$	sum of all $\mathbf{e}(\mathbf{T}, \gamma)$ for $\mathbf{T}$ and Γ
f	focal length in meters
f_p	focal length, measured in average pixels (average between f_x and f_y)
f_x	focal length, measured in pixel widths
f_y	focal length, measured in pixel heights
G_k	node group k
h	observer altitude
$I(*)$	instantaneous glimpse of a target at *
k	constant used in M_{IC}
k_1, k_2	radial distortion coefficients
k_3, k_4	radial distortion approximation coefficients
k_{un}	probability scale factor based on proximity to other UAVs
k_s	image skew constant
L	search length, which is the total distance traveled while searching
L_{NU}	edge length for the non-uniform area of the variable map
L_s	length of the sliding segment
L_t	test length along the target axis
L_T	edge length for the total area of the variable map
m	number of existing contact sets
M	some detection model
M_{CC}	connected component detection model

M_{DR}	definite range detection model
M_{IC}	inverse cube detection model
n	number of contacts in a contact set
N	the total number of nodes
$\dot{N}$	node search rate
N_{ψ_p}	number of selected nodes around point q_{ψ_p}
n_B	number of batteries
N_B	total number of nodes contained in all bounded groups
N_C	node threshold in the contour search
N_G	node threshold in the greedy search
N_k	the number of nodes in group G_k
$n_{\scriptscriptstyle L}$	number of divisions in L, used in defining the random search law
N_q	contact quantity threshold used in target quality calculation
N_{UAV}	the total number of UAVs
p	true image pixel (distortion removed)
$\breve{p}$	an image pixel (appears distorted in original image)
$\dot{p}_{jk}$	a probability rate estimate for u_j searching G_k
$p(*)$	probability of *
$\mathbf{P}_{\chi}$	cluster point
$\bar{p}_{\psi_p}$	average probability in the vicinity of point q_{ψ_p}
$\bar{p}'_{\psi_p}$	some $\bar{p}_{\psi_p}$ for a UAV, modified by its proximity to every other UAV
p_1, p_2	tangential distortion coefficients
$P_*(L)$	some path of length L in area $*$
P_r	power required
P_s	image area occupied by the target projection, in pixels
P_T	probability threshold, used to determine node groups
$P(x)$	set of target glimpses from a long, straight pass at lateral range x
q	location point on a plane at zero altitude
Q	target quality
q_{ψ_p}	a point at distance R_p from the UAV and angle psi_p relative to X_S

q_b	point at the back of the sliding segment
q_c	UAV progress point along the contour
q_f	point at the front of the sliding segment
q_m	midpoint of the sliding segment
r	range, which is distance along the ground
R	definite range, used in the definite range detection model
R_{cb}	rotation matrix from the camera frame to the body frame
$R_*(L)$	a random path of length L in area $*$
r_m	distance from the image center in meters (at $Z_C = 1$ m)
$\breve{r}_m$	distorted distance from the image center in meters (at $Z_C = 1$ m)
R_{NT}	the ratio of non-uniform area to total area in the variable map
r_{obj}	radius of a spherical target in meters
r_{objp}	projected radius of a spherical target in pixels
r_p	distance of a given pixel from the target center, in pixels
R_p	range from the UAV, used in finding local probable benefits
r_t	radius of the target projection in pixels
s	scalar used in calculating q
s_1	detection probability parameter
s_2	detection probability parameter
s_q	quantity scale factor used in target quality calculation
$S(t)$	some specified set of target glimpses occurring from time zero to time t
S_x, S_y	pixel scale factors
t	time
t^*	time at the end of a search
$\hat{\mathbf{t}}$	normalized target axis direction vector in the world frame
$\mathbf{T}$	a test point
t_A	time between node group assignments
t_C	time threshold for the contour search
t_f	time (duration) of a single flight
t_g	time on the ground in-between flights

t_G	time threshold for the greedy search
U	the target remaining un-detected
$\hat{\mathbf{u}}$	normalized direction vector in the world frame
$\mathbf{u}_c$	direction vector in the camera frame
$\hat{\mathbf{u}}_c$	normalized direction vector in the camera frame
$\breve{\mathbf{u}}_c$	distorted direction vector in the camera frame
$\hat{\mathbf{u}}_b$	normalized direction vector in the UAV body frame
u_j	UAV number j
v	observer velocity
V_B	battery voltage
V_g	groundspeed
$\mathbf{w}$	UAV world position and/or contact origin location
W	search width
x	lateral range
(x_0, y_0)	image center point in the pixel frame
X, Y, Z	axes of the world coordinate frame
X_b, Y_b, Z_b	axes of the UAV body coordinate frame
X_c, Y_c, Z_c	axes of the camera coordinate frame
X_P, Y_P	axes of the pixel coordinate frame
X_{PSF}, Y_{PSF}	axes of the point spread function coordinate frame
X_S, Y_S, Z_S	axes of the UAV stability coordinate frame
(x_{jk}, y_{jk}, z_{jk})	relative vector from u_j to the nearest point of G_k
(x_{tp}, y_{tp})	target center point in the pixel frame
$[x_{u_c}, y_{u_c}, z_{u_c}]^T$	direction vector in the camera frame
$[x_{\breve{u}_c}, y_{\breve{u}_c}, y_{\breve{u}_c}]^T$	distorted direction vector in the camera frame
$Z_*(L)$	a parallel path of length L in area $*$

Chapter 1

Introduction

With the recent miniaturization of autopilot technology [1], small autonomous aircraft have become feasible platforms for complex aerial missions. Some of the evolving possibilities include search and rescue, fire monitoring, area surveillance, terrain mapping and military use. Each mission requires these aircraft to scan an area, differentiate between multiple targets and locate them.

To enhance the search capabilities of Small Unmanned Aerial Vehicles (SUAVs), this thesis draws on the practical experience of others. Concepts in optimal search theory have been used to improve the effectiveness of human-based aerial search since the end of World War II [2]. This thesis asserts that these concepts can be used to improve SUAV search performance.

1.1 Motivation

The MAGICC[1]Lab at Brigham Young University has been extending the capabilities of SUAVs for a number of years. Some current capabilities include simultaneous arrival, localization of a single target, curved path following, canyon following, and many more. The time has come to begin combining these capabilities into more complex missions. Of particular interest is the ability to search for and locate multiple targets.

At the beginning of a search, a target's location consists of an assumed probability distribution that has been constructed from items such as last-seen location, prominent landmarks, tracking clues, probable target behavior, etc. The question is how to apply search effort to the probability distribution (often referred to as a "probability map") in a way that appropriately considers both efficiency and completeness. An understanding of optimal

[1]MAGICC stands for Multiple AGent Intelligent Coordination and Control.

search theory is an excellent starting point, and offers helpful guiding principles. However, optimal search theory is based on the assumptions that search effort is both infinitely divisible and arbitrarily allocatable [3]. These assumptions imply that a search must be characterized by long distances, large aircraft and long-range sensors.

Search planning for SUAVs typically represents a violation of the assumptions used in optimal search theory. The relatively short-range sensor sweeps cannot be assumed to apply anywhere, and in any quantity. They must be applied here and now—wherever and whenever the SUAV happens to be.

1.2 Literature Review

1.2.1 Target Differentiation

Target differentiation is built upon the foundation of target localization. In this field, Joshua Redding determines a method for estimating the x-y location of a target based on video and telemetry information from a single UAV [4]. Redding's work indicates that projecting a line in the direction of the seen target, and finding where this line intercepts a zero-altitude plane can be used to locate a target. One limitation of this method is that distinguishing between multiple targets can become difficult when the information is an overlay of several target distributions on the same plane. In addition, errors in UAV height an orientation can cause a bias in the estimated target position when they are projected onto an incorrectly-assumed plane.

More useful techniques in 3D localization of a single point are presented by Hartley and Zisserman in [5]. This method focuses on arranging data from multiple target sightings into an $\mathbf{Ax} = \mathbf{0}$ representation, then inverting $\mathbf{A}$. This yields a minimum least-squares target estimate, which is desirable if the exact camera position and intrinsic parameters are precisely known. However, if there is significant error in the known camera (or UAV) position data, then the least-squares estimate will be influenced by outliers in a squared sense. In addition, if a large number of sightings is to be considered in this estimate, then inverting the large $\mathbf{A}$ matrix may be difficult.

1.2.2 Detection Probability and Search Patterns

Some of the best search theory comes from war history. Initial progress was made by Koopman and his colleagues in the Anti-Submarine Warfare Operations Research Group (ASWORG) of the U.S. Navy during World War II [6, p.1]. ASWORG became the Operations Evaluation Group (OEG) at the end of the war, and was assigned to publish three volumes of general scientific and historical interest. One of these volumes was called *Search and Screening*. The volume was originally confidential, but was later declassified [2, p.1]. It represents the efforts of many collaborators, and its principles are those that have been boiled in the crucible of war experience. Valuable information can be found in its pages that should be understood and considered before formulating any search strategy. *Search and Screening, General Principles with Historical Applications* [2] is an updated version of this book. Years later, the Office of Naval Research began supporting theoretical work related to the Scorpion search, which involves considerations of uncertain search width and false targets [6, p.xiii-xiv]. Lawrence D. Stone composed an additional volume called *Theory of Optimal Search* based on these findings. The ideas of these individuals, and others, is the theoretical foundation for many military and civilian searches today.

Applicable concepts from both works will be reviewed more completely in the chapter on optimal search theory in this thesis. Search theory answers the question of where and when search effort should be applied, but offers few suggestions in actual path planning.

Many have suggested search path generation methods. Wollan presents a heuristic approach to search path generation based on Zamboni polygons that minimizes the path length while completely covering a user-defined area [7]. This method promotes completeness when searching a uniform probability distribution within a Zamboni polygon, but does not consider the possibility that some regions may have higher probability than others. Another paper presents a dynamic decision model where each agent chooses its own path to avoid other agents while searching a variable-probability environment [8]. Here agents are drawn to high-probability areas in a way that could more closely align with Koopman's optimal search criterion. However, as time progresses the proposed solution suffers from comparatively disorganized paths, and eventually leads to a condition where a Zamboni pattern exhibits

better performance. In both cases, search path evaluation would benefit from comparing the results to an optimal search allocation.

Zhijun Tang, from Ohio State University, assumes that very accurate sensors warrant a detection probability of unity within the sensor footprint of each mobile sensor agent (MSA). Under this assumption, Tang develops a path for multiple MSAs to surround the survival zone of a target and execute a non-escape search [3]. This approach has useful applications, but does not fare well under metrics of minimum time to detection or maximum detection probability when detection within the sensor footprint is not certain. Tang's work focuses on tactics to track a moving target, and he assumes that the sensor footprint remains constant relative to the position and path of the MSA. However, these results are not fully applicable to a fixed-camera platform, whose sensor footprint will change with agent orientation, and whose detecting capability is less than perfect.

1.3 Contributions

This thesis particularly considers path planning on a scale where SUAVs are expected to carry out the search. The simplicity, reliability and low cost of a fixed-wing, fixed-camera platform make it the platform of choice for many SUAV missions. As such, this thesis focuses on extending the search-and-locate capabilities of this platform. Inexpensive wireless video systems that are designed for SUAVs [9] typically include a Charge-Coupled Device (CCD) camera. Accordingly, a detection probability model is developed for this camera.

Many written texts focus on either search theory or path planning, but this thesis combines and compares the two areas. In addition, diverse subjects are considered in this thesis such as target identification, target localization, target differentiation, search theory, path generation and multi-agent coordination. It is often difficult to appropriately model a realistic search due to the myriad of obstacles that must be overcome in so many areas.

Despite these obstacles, this thesis offers the following contributions:

- A practical numerical method for locating targets in three dimensions.
- A way to count and separate multiple targets as seen from multiple images.
- The concept of target quality.

- A target detection probability model based on CCD camera sensing.
- Discrete methods for measuring search rate using a probability grid.
- A method for determining search width using a probability grid.
- An optimal detection function for any given probability map.
- The metrics of search efficiency and search completeness.
- A search strategy that can be applied to any given probability map.

1.4 Document Organization

This thesis not only serves to further the work of search theory, but also serves to give practical instruction to those who will further this work in the MAGICC Lab at Brigham Young University (BYU). As such, there may be included detail that seems obvious to one audience, but is helpful to another. This detail is intended to simplify, not overwhelm, the issue. Hopefully the format is beneficial to all.

This thesis begins in chapter two by discussing the topic of target differentiation. The chapter focuses on the computational methods which allow a UAV to locate multiple targets and differentiate between them. The chapter is included in this thesis because target differentiation concepts are considered a necessary first-step in developing SUAV search capabilities. Target differentiation deals with subjects in target localization, image skew, and target quality. Chapter two concludes with a field test, and a discussion of results.

Following target differentiation, the third chapter lays the foundation for a discussion about search planning. A review of optimal search theory is considered invaluable to perpetuating this work. Thus, chapter three is devoted entirely to the applicable aspects of this theory, and describes some principles that are frequently referenced in state-of-the-art search planning today. The entire chapter can be thought of as an in-depth literature review on search theory.

Chapters four and five represent original contributions, where discrete methods for measuring search progress are offered based on optimal search theory. Chapter four develops

a probabilistic model for target detection using a CCD camera, and image skew considerations are added to the model. Discrete approximation methods are then used to obtain agent search width, turn performance, and an optimal detection function for a probability map.

Chapter five begins by offering the metrics of search efficiency and search completeness. Path planning strategies are also presented that focus on these metrics, and example simulations are given to help the reader understand when it might be best to use each strategy.

The final chapter discusses conclusions to the presented research, as well as possible future work. This work could relate to target detection, a coordinated search effort, other probability map uses, and search algorithm improvements.

Chapter 2

Target Differentiation

One aim of this thesis is to enable simple SUAVs to search for and locate multiple targets efficiently. However, before the searching can begin, the capability of distinguishing between and locating multiple targets must exist. This chapter establishes this capability by first reviewing target localization mathematics and describing the camera calibration procedure. After a method for locating a single target has been presented, the original contributions of 3D target localization and target quality are developed, and guidelines are established for differentiating between multiple targets. Finally, a target differentiation field test is presented and the results are discussed.

2.1 Localization Mathematics

When determining a target's direction from an SUAV, it is first necessary to determine the target's direction in the camera frame. (See Figure 2.1.) It is important to note that the focal length, f, of an image is rarely known, but is a theoretical standard for comparison. For this reason it is typically set to unity, as will be done in the following calculations. A camera may be calibrated to find its intrinsic matrix, $\mathbf{A}$, which describes how the pixel coordinate system relates to the camera coordinate system. A simplified intrinsic matrix is

$$\mathbf{A} = \begin{bmatrix} f_x & 0 & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.1}$$

Ignoring image skew for the moment, each of the terms in Eq.(2.1) has physical meaning. For example, f_x represents the number of pixels that could be lined up side by side to equal the focal length. Similarly, f_y represents the number of pixels that could be stacked

on top of each other to equal one focal length. The parameters x_0 and y_0 indicate the pixel coordinates where the camera z-axis intersects the pixel frame's x-y plane. (See Figure 2.1.)

There are also scale factors, S_x and S_y, having the relationships

$$S_x = \frac{f}{f_x} \quad \text{and} \quad S_y = \frac{f}{f_y}. \tag{2.2}$$

S_x is the width of one pixel that has been projected onto a perpendicular plane one focal length away, and S_y is the height of one pixel at the same distance.

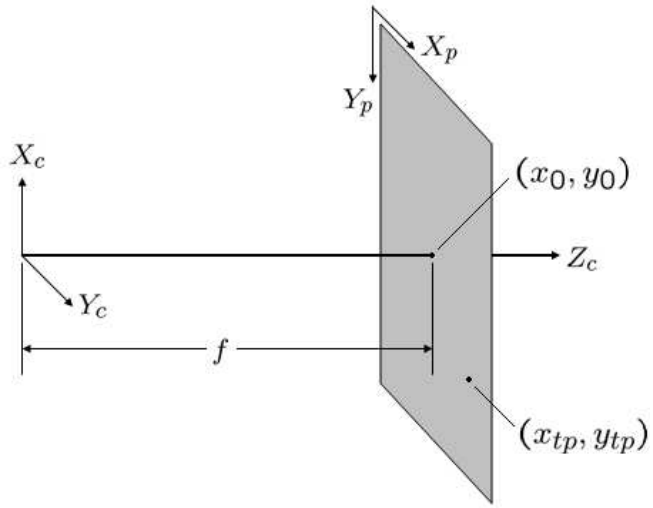


Figure 2.1: The relationship between the camera and pixel coordinate frames. The X_c, Y_c and Z_c axes originate from the camera center and represent the camera coordinate frame. The X_p and Y_p axes originate from the upper-left hand corner of the image, and represent the pixel coordinate frame.

With this understanding, one can easily construct a vector in the camera frame that points in the direction of a target pixel, (x_{tp}, y_{tp}). This vector is

$$\mathbf{u_c} = \begin{bmatrix} S_y\,(y_0 - y_{tp}) \\ S_x\,(x_{tp} - x_0) \\ f \end{bmatrix}. \tag{2.3}$$

Then for future calculations it is desired to normalize $\mathbf{u_c}$ to obtain

$$\hat{\mathbf{u}}_\mathbf{c} = \frac{\mathbf{u_c}}{\|\mathbf{u_c}\|}. \tag{2.4}$$

If $\hat{\mathbf{u}}$ can be found in the world coordinate frame, then it can be compared with other target direction vectors. To do this, $\hat{\mathbf{u}}_\mathbf{c}$ must first be rotated to the body frame as shown in Figure 2.2 using

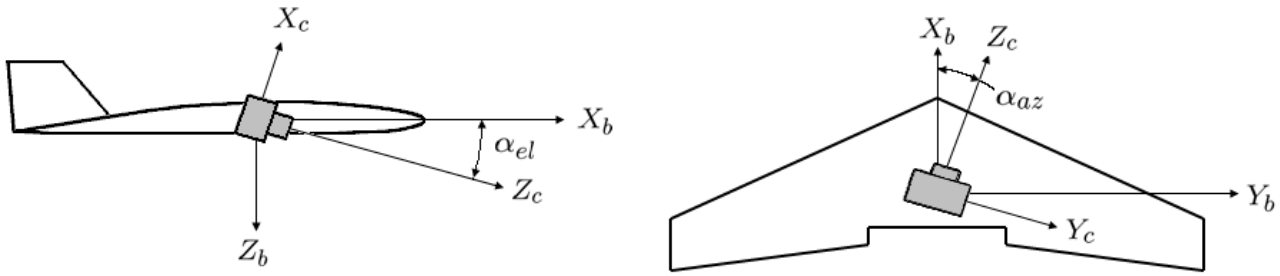
$$\hat{\mathbf{u}}_\mathbf{b} = \mathbf{R_{cb}}\hat{\mathbf{u}}_\mathbf{c} \tag{2.5}$$

where

$$\mathbf{R_{cb}} = \begin{bmatrix} c_{az}c_{el} & -s_{az} & c_{az}s_{el} \\ s_{az}c_{el} & c_{az} & s_{az}s_{el} \\ -s_{el} & 0 & c_{el} \end{bmatrix} \tag{2.6}$$

$$s_{el} = \sin\left(\frac{\pi}{2} - \alpha_{el}\right) \quad c_{el} = \cos\left(\frac{\pi}{2} - \alpha_{el}\right)$$

$$s_{az} = \sin\left(\alpha_{az}\right) \quad c_{az} = \cos\left(\alpha_{az}\right).$$



(a) Elevation Rotation (b) Azimuth Rotation

Figure 2.2: Rotations from camera to body frame.

Once the body-frame target direction vector, $\hat{\mathbf{u}}_\mathbf{b}$, has been obtained, the final rotation to the world coordinate frame can be made. The angles ϕ, θ and ψ are rotations about the X_b, Y_b and Z_b axes respectively. (See Figure 2.3.) This rotation can be done using

$$\hat{\mathbf{u}} = \mathbf{R_{bw}}\hat{\mathbf{u}}_\mathbf{b} \tag{2.7}$$

where

$$\mathbf{R_{bw}} = \begin{bmatrix} c_\psi c_\theta & c_\psi s_\theta s_\phi - s_\psi c_\phi & c_\psi s_\theta c_\phi + s_\psi s_\phi \\ s_\psi c_\theta & s_\psi s_\theta s_\phi + c_\psi c_\phi & s_\psi s_\theta c_\phi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix} \tag{2.8}$$

$$s_* = \sin(*)$$

$$c_* = \cos(*)$$

$$\phi = \text{UAV roll angle}$$

$$\theta = \text{UAV pitch angle}$$

$$\psi = \text{UAV heading angle.}$$

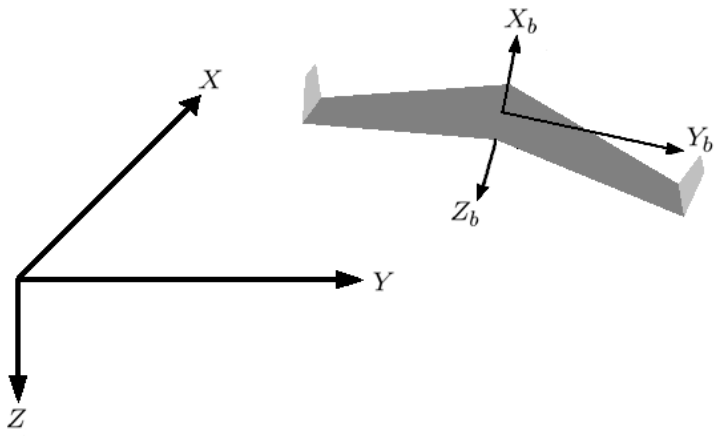


Figure 2.3: The body and world coordinate frames.

After all the above rotations are complete, the UAV world position, $\mathbf{w}$, can be considered the starting point of a ray that points in the $\hat{\mathbf{u}}$ direction toward the target. This assumes that the distance between the body frame and the camera frame origins is negligible compared to the effects of UAV attitude and position error, as is often the case for SUAVs.

The ray now originates from a point in the world coordinate frame, and points in a direction that is known in the world coordinate frame.

Supposedly, this ray was generated when a potential target was identified in an image, and the target center was designated as pixel (x_{tp}, y_{tp}) in that image. UAV telemetry information was then used to perform the above rotations, as well as provide the ray origin. This information is called a *target contact*, γ. As referred to by Stone, this term denotes any discovered information that could indicate the presence of a target [6, p.137]. In the case of target localization, it takes the form

$$\gamma = \left\{ \mathbf{w} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad \hat{\mathbf{u}} = \begin{bmatrix} x_{\hat{u}} \\ y_{\hat{u}} \\ z_{\hat{u}} \end{bmatrix} \right\}. \tag{2.9}$$

If this is the only contact so far, then the target's position may be estimated by assuming that the target is on a plane at zero altitude, as shown in Figure 2.4. Under this assumption there is some scalar, s, that describes the distance from the UAV to the target. Thus, the target's estimated position is

$$\mathbf{q} = \mathbf{w} + s\hat{\mathbf{u}}. \tag{2.10}$$

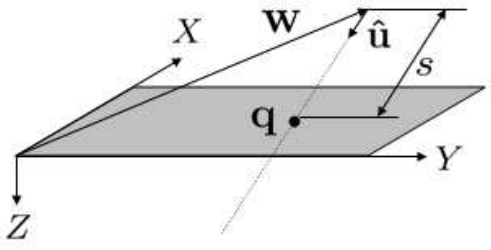


Figure 2.4: Projection of a ray onto the zero-altitude plane.

The planar constraint requires that the inner product $\langle \mathbf{w} + s\hat{\mathbf{u}}, \hat{\mathbf{z}} \rangle$ be equal to zero. This can be used to obtain

$$s = -\frac{z}{z_{\hat{u}}} \tag{2.11}$$

from

$$s = -\frac{\langle \mathbf{w}, \hat{\mathbf{z}} \rangle}{\langle \hat{\mathbf{u}}, \hat{\mathbf{z}} \rangle}$$

and

$$\hat{\mathbf{z}} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T.$$

This value for s may now substituted into Eq.(2.10) to give

$$\mathbf{q} = \begin{bmatrix} x - \frac{z}{z_{\hat{u}}} x_{\hat{u}} \\ y - \frac{z}{z_{\hat{u}}} y_{\hat{u}} \\ 0 \end{bmatrix}. \tag{2.12}$$

Repetition of this method for all data points is the basis for Redding's target localization technique [4]. However, if UAV height-above-ground (HAG) altitude is uncertain, then the zero-altitude assumption can cause the projected points to give a scattered and inaccurate estimate of the target's location. This is particularly true in mountainous or changing terrain.

To solve this problem, original contributions in 3D target localization and target quality are presented below. The quality describes how well a target has been distinguished from nearby targets and other noise, and this quality may be more fully assessed with the original contact information. Thus, it is desired to store the complete γ for later comparison, rather than keeping the assumed $\mathbf{q}$ only.

Now consider a test point $\mathbf{T}$ as shown in Figure 2.5. The minimum error vector between the test point and the target contact may be found using one step of Gram-Schmidt orthogonalization [10, p.119]. The scalar, s, is equal to $-\langle \hat{\mathbf{u}}, \mathbf{d} \rangle$, and the magnitude of the error vector is a function of both the chosen test point and the target contact:

$$|\mathbf{e}(\mathbf{T}, \gamma)| = |\mathbf{d} - \langle \hat{\mathbf{u}}, \mathbf{d} \rangle \hat{\mathbf{u}}| \tag{2.13}$$

where

$$\mathbf{d} = \mathbf{w} - \mathbf{T}.$$

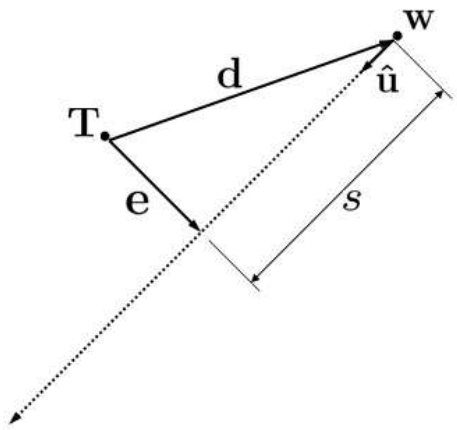


Figure 2.5: This is a graphical representation of the minimum vector **e** between test point **T** and target contact γ. As shown in Eq.(2.9), γ consists of an origin **w** and a direction $\hat{\mathbf{u}}$.

Now let $\mathbf{\Gamma} = \{\gamma_1, \gamma_2, \ldots, \gamma_n\}$ be the set of contacts generated by viewing a single target, where n is the number of contacts in set $\mathbf{\Gamma}$. Also let the *error sum* between $\mathbf{\Gamma}$ and **T** be defined as

$$\mathbf{E}(\mathbf{T}, \mathbf{\Gamma}) = \sum_{i=1}^{n} |\mathbf{e}(\mathbf{T}, \gamma_\mathbf{i})|. \tag{2.14}$$

For each $\mathbf{\Gamma}$ there exists some specialized test point **C**, or *target center*, which estimates the target's location. This point minimizes $\mathbf{E}(\mathbf{T}, \mathbf{\Gamma})$ as shown in Figure 2.6 such that

$$\mathbf{E}(\mathbf{C}, \mathbf{\Gamma}) \leq \mathbf{E}(\mathbf{T}, \mathbf{\Gamma}) \; \forall \; \mathbf{T} \neq \mathbf{C}. \tag{2.15}$$

Arriving at point **C** can be accomplished by a variety of numerical methods. For this application, it is preferred to use a gradient-descent method that begins with a large step size and then reduces the step size after all of the neighboring options exhibit a greater error sum. As each contact is added, a starting point for the algorithm should be obtained from the assumed **q** estimates (See Eq.(2.12).) for at least the first two steps, and thereafter from

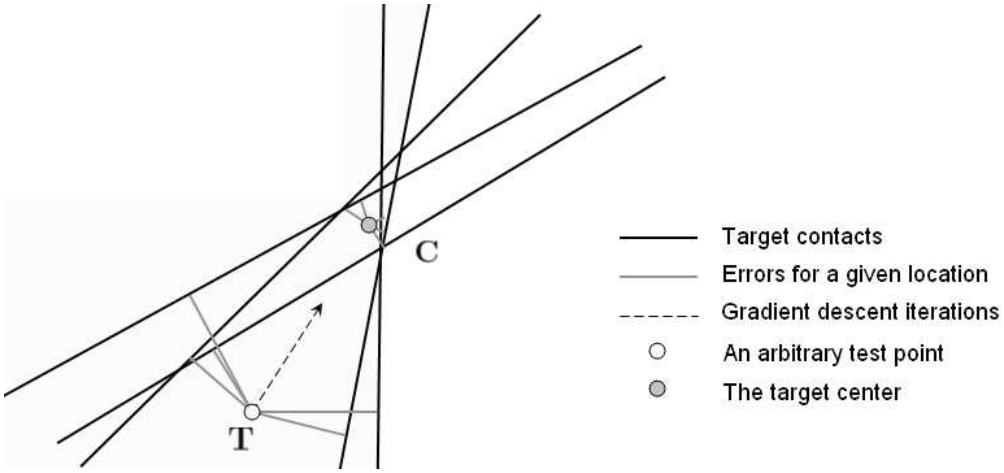


Figure 2.6: Once a test point is chosen, steps can be taken in the direction that minimizes the error sum to arrive at the target center.

each preceding step. This is because a unique solution to Eq.(2.15) does not exist for one or two contacts. Code for finding the target center point is included in Appendix A.1.

It is interesting to note that for an arbitrary set of contacts there is no guarantee that only one $\mathbf{C}$ exists. However, if the contacts in a contact set are all generated by viewing a single, stationary target, then the rays often gather near the target location. This typically allows the algorithm to follow the gradient to a unique, absolute minimum in the vicinity of target's actual position. Of course, the degree of contact convergence depends on the inherent error in the sensing method.

One redeeming feature of the error minimization method is that the computed target center is attracted to the largest *cluster* of contacts, χ. A cluster of contacts is defined as all of the contacts that pass within a specified distance, d_χ, of a *cluster point*, $\mathbf{P}_\chi$, where d_χ and $\mathbf{P}_\chi$ may be arbitrarily chosen. The size of the cluster is determined by the number of contacts in the cluster, n_χ.

Consider Figure 2.7, where the contact set contains two cluster points, $\mathbf{P}_{\chi\mathbf{1}}$ and $\mathbf{P}_{\chi\mathbf{2}}$. For this example, it is assumed that $d_{\chi 1} = d_{\chi 2}$ as shown, and that $d_{\chi 12}$ is the distance between $\mathbf{P}_{\chi\mathbf{1}}$ and $\mathbf{P}_{\chi\mathbf{2}}$. Assuming that $d_{\chi 1}, d_{\chi 2} \ll d_{\chi 12}$, the error sums associated with test points $\mathbf{P}_{\chi\mathbf{1}}$ and $\mathbf{P}_{\chi\mathbf{2}}$ are

$$\mathbf{E}\left(\mathbf{P}_{\chi\mathbf{1}}, \mathbf{\Gamma}\right) \approx 3d_{\chi 12} \quad \text{and} \quad \mathbf{E}\left(\mathbf{P}_{\chi\mathbf{2}}, \mathbf{\Gamma}\right) \approx 2d_{\chi 12}.$$

Thus, the error minimization method will select the largest cluster point when the distance between cluster points is large compared to $d_{\chi 1}$ and $d_{\chi 2}$. In the case shown, $\mathbf{P_{\chi 2}}$ is selected as the target center. This effect causes outliers to be rejected when a large cluster exists, as is typical near an actual target location.

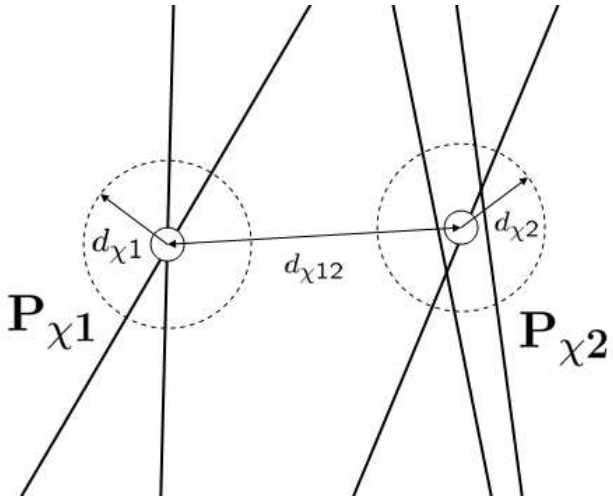


Figure 2.7: Two cluster points in a set of contacts.

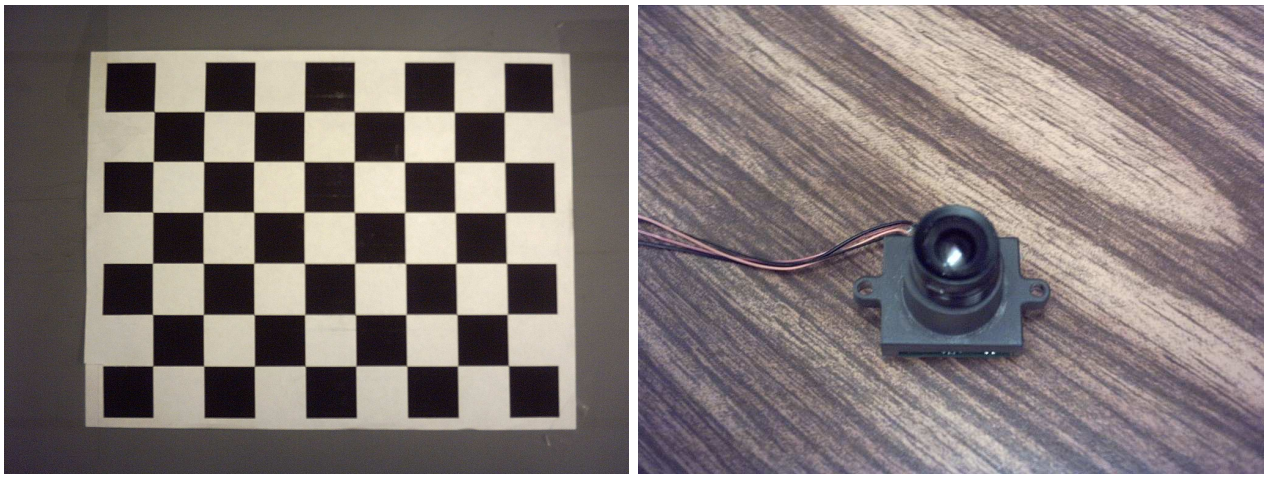
The error minimization method is preferred over that presented in [5] for two reasons. First, it is a numerically feasible way to handle a large number of contacts, compared to the task of inverting a large $\mathbf{A}$ matrix. Second, it is not influenced by each contact through a squared metric, and prevents outliers from overly attracting the solution. Experience indicates that this added stability typically keeps the target center closer to the true target location. This makes it more likely for new contacts to be assigned to the correct target if they are closest to the target that generated them. In practice, the suggested iterative method is a dependable and real-time[1] approach to locating a target in three dimensions.

[1]There is no noticeable delay in contact processing when a new image set of five contacts is processed once every half second, and there are hundreds of contacts already. Calculations are performed on a Dell Inspiron 600m notebook PC, with an Intel Pentium M 1.80 GHz processor and 512 MB of RAM.

2.2 Camera Calibration and Image Skew

New techniques in camera calibration have been surfacing as CCD cameras have become widely used in the home and professional communities. A flexible technique is presented in [11], and similar techniques have been developed in Matlab [12] and OpenCV [13]. Camera calibration is not the main focus of this thesis, and the reader may pursue calibration algorithms in the mentioned sources. However, the following information may be helpful for those wanting to calibrate a camera and understand the calibration parameters.

The method of camera calibration used in this thesis is based on the CameraCalibration2 function in OpenCV. Using a CCD camera, a chessboard of known dimension is viewed in various orientations. (See Figure 2.8.) In each position the Imperx VCE-Pro frame grabber, or similar hardware, is used to transfer the pixel data to host memory. Corner points are then detected using the cvFindChessboardCorners OpenCV function, and these object points are given to the CameraCalibration2 function, along with the number of corner points in the image. This causes the camera's intrinsic and extrinsic parameters to be calculated, and returned in matrix form. The results for three repetitions of this process are shown in Table 2.1, where twenty images were used in each run.



(a) Chessboard (b) CCD Camera

Figure 2.8: The chessboard and camera used in camera calibration.

Note that the x_0 and y_0 values are assumed to be the exact center of a 640 x 480 screen. The tangential distortion coefficients, p_1 and p_2, are also assumed to be zero. This is often done in camera calibration because it is convenient to measure radial distortion from the exact center of the screen, and tangential distortion is typically negligible. This can be seen in [11], where p_1 and p_2 are not even considered. Both of these assumptions are options of the CameraCalibrate2 function.

Table 2.1: Camera calibration results.

Camera Parameters	f_x	f_y	x_0	y_0	k_1	k_2	p_1	p_2
Calibration 1	501.161	498.554	319.5	239.5	-0.360	0.188	0.0	0.0
Calibration 2	495.686	492.112	319.5	239.5	-0.361	0.219	0.0	0.0
Calibration 3	514.109	509.664	319.5	239.5	-0.369	0.171	0.0	0.0
Calibration Average	503.652	500.110	319.5	239.5	-0.363	0.193	0.0	0.0

Some might call the camera lens a 60° field-of-view lens, but the above calibration parameters are much more descriptive. In [12] there is an informative description of the radial distortion coefficients. These follow the "Plumb Bob" model introduced by Brown in 1966 [12]. However, the lens used in this thesis does not represent extreme radial distortion, which means that the reduced camera model is warranted [12]. Therefore, k_1 and k_2 will be assumed to accurately represent radial distortion, and higher order terms will not be considered. The radial distortion equation then becomes

$$\breve{r}_m = k_2 {r_m}^5 + k_1 {r_m}^3 + r_m, \tag{2.16}$$

where

$$\breve{r}_m = \sqrt{x_{\breve{u}_c}^2 + y_{\breve{u}_c}^2}$$

and

$$r_m = \sqrt{x_{u_c}^2 + y_{u_c}^2}.$$

From Figure 2.9 it can be seen that radial distortion is a phenomenon that distorts an element's radial distance from the camera z-axis. This happens naturally when looking

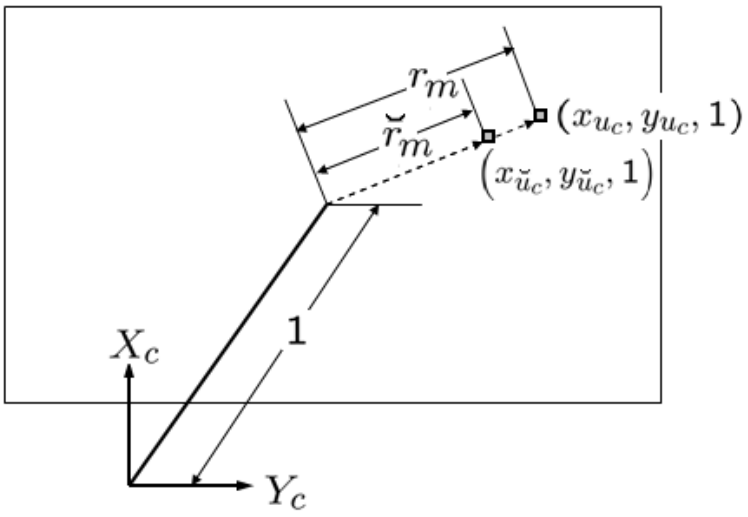


Figure 2.9: Radial distortion. Color information that would appear at radial distance r_m from the image center is distorted to distance $\breve{r}_m$ by the lens.

through a lens, and must be understood to accurately estimate a target's direction. The goal is to obtain the un-distorted radius, r_m, in terms of the distorted radius, $\breve{r}_m$, using the k_1 and k_2 parameters obtained in camera calibration.

Unfortunately, Eq.(2.16) provides no direct solution for r_m. However, the fifth-order polynomial may be approximated by a second-order polynomial within the region of interest. Thus, for a 640 x 480 image with $f_x = 503.652$, and $f_y = 500.110$, the furthest distorted radial distance in the image is 0.796 m (at an assumed focal length of 1 m). Within this range, points may be generated from Eq.(2.16) as shown in Figure 2.10, and a second-order least-squares approximation can be made of the form

$$\breve{r}_m = k_4 {r_m}^2 + k_3 r_m. \tag{2.17}$$

This new approximation has a k_4 of -0.217, and a k_3 of 1.026. A value for r_m in terms of $\breve{r}_m$ may now be readily obtained as

$$r_m = \frac{-k_3 + \sqrt{k_3^2 + 4k_4 \breve{r}_m}}{2k_4}. \tag{2.18}$$

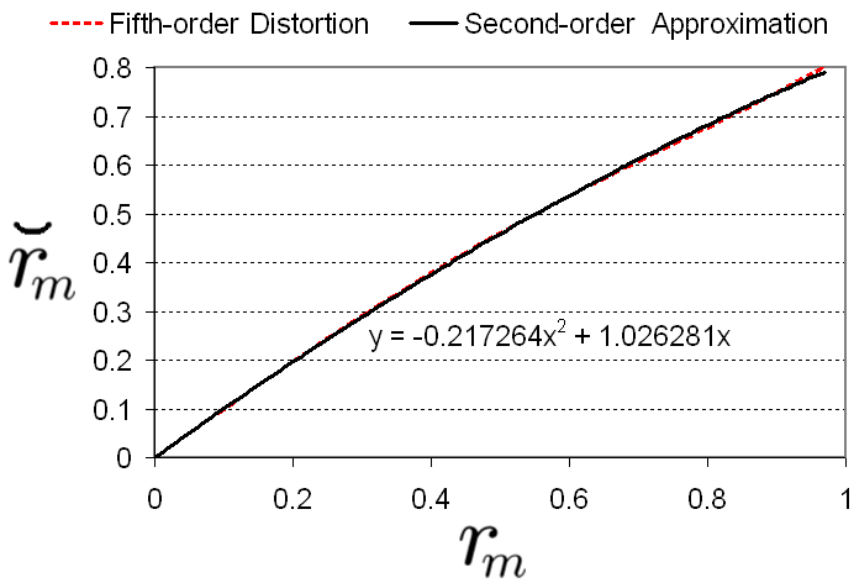


Figure 2.10: Radial distortion approximation. The fifth-order distortion model can be approximated using a second-order function.

Note that the region of interest is only for positive r_m, so only the positive root applies. Added insight may be gained by re-writing Eq.(2.17) as a scaling of r_m where

$$\breve{r}_m = r_m k_s \qquad \text{and} \qquad k_s = k_4 r_m + k_3. \tag{2.19}$$

Substituting the r_m from Eq.(2.18) into Eq.(2.19), the image skew constant becomes

$$k_s(\breve{r}_m) = \frac{k_3}{2} + \sqrt{\left(\frac{k_3}{2}\right)^2 + k_4 \breve{r}_m}. \tag{2.20}$$

Eq.(2.19) can also be separated into its x and y components such that

$$x_{u_c} = \frac{x_{\breve{u}_c}}{k_s} \quad \text{and} \quad y_{u_c} = \frac{y_{\breve{u}_c}}{k_s}. \tag{2.21}$$

It is important to note here that Eq.(2.21) represents a re-definition of Eq.(2.3). In other words, the original image frame target direction

$$\breve{\mathbf{u}}_\mathbf{c} = \begin{bmatrix} S_y\left(y_0 - y_{tp}\right) \\ S_x\left(x_{tp} - x_0\right) \\ 1 \end{bmatrix} = \begin{bmatrix} x_{\breve{u}_c} \\ y_{\breve{u}_c} \\ 1 \end{bmatrix} \tag{2.22}$$

should be de-skewed to obtain the true target direction

$$\mathbf{u_c} = \begin{bmatrix} \frac{x_{\breve{u}_c}}{k_s} \\ \frac{y_{\breve{u}_c}}{k_s} \\ 1 \end{bmatrix} \tag{2.23}$$

before it is normalized and rotated into the world frame.

2.3 Target Quality

Once the target center has been calculated as shown in Section 2.1, it is possible to get a measurement of the target quality. There are two attributes of a contact set that are desired when calculating the target center. Ideally, all target contacts would pass through a single point, and each contact would have a direction as different as possible from the other contacts to enhance triangulation accuracy. In reality, contacts in a set do not all pass through a single point, and there is limited variety in contact direction. However, measurements can be made that reveal to what degree a contact set exhibits these properties.

In the presence of false targets and multiple targets, measurement of these characteristics, along with the size of the contact set, will also be correlated with a confidence that the target exists, and that only one target exists.

Consider the contact set in Figure 2.11. The *target axis* is a ray that has origin $\mathbf{C}$, and direction $\hat{\mathbf{t}}$, where $\mathbf{C}$ is the target center as defined in Eq.(2.15), and $\hat{\mathbf{t}}$ is

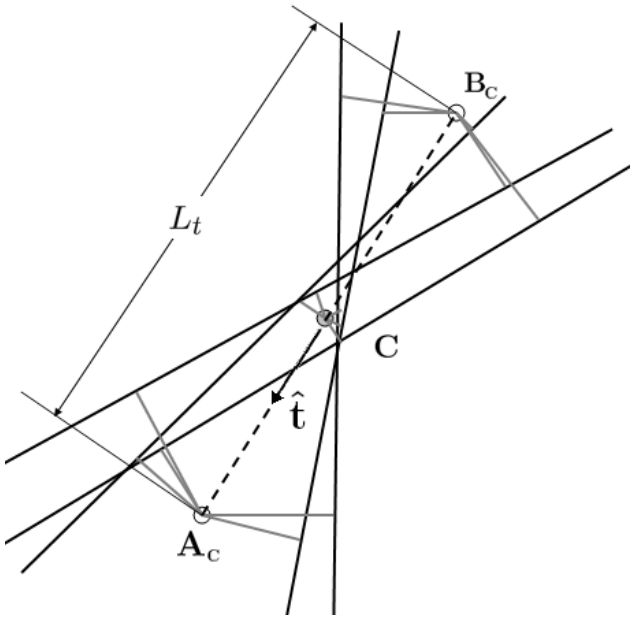


Figure 2.11: Target quality calculation. A target's quality is calculated by comparing the error sum at points $\mathbf{A_C}$ and $\mathbf{B_C}$ with the error sum at point $\mathbf{C}$. The target axis direction, $\hat{\mathbf{t}}$, is the average direction of all the contacts in the set. L_t is the test length.

$$\hat{\mathbf{t}} = \frac{\sum_{i=1}^{n} \hat{\mathbf{u}}_\mathbf{i}}{\left\| \sum_{i=1}^{n} \hat{\mathbf{u}}_\mathbf{i} \right\|}. \tag{2.24}$$

The axis direction, $\hat{\mathbf{t}}$, represents the average direction of all contacts in a given set. Along this axis, two new test points, $\mathbf{A_C}$ and $\mathbf{B_C}$, are defined as

$$\mathbf{A_C} = \mathbf{C} + \frac{L_t}{2}\hat{\mathbf{t}} \quad \text{and} \quad \mathbf{B_C} = \mathbf{C} - \frac{L_t}{2}\hat{\mathbf{t}}, \tag{2.25}$$

where L_t is some test length.

The target quality may now be defined as

$$Q = s_q \left(1 - \frac{2\mathbf{E}\left(\mathbf{C}, \mathbf{\Gamma}\right)}{\mathbf{E}\left(\mathbf{A_C}, \mathbf{\Gamma}\right) + \mathbf{E}\left(\mathbf{B_C}, \mathbf{\Gamma}\right)} \right), \tag{2.26}$$

where $\mathbf{E}(*, \mathbf{\Gamma})$ is the error sum of a test point as defined in Eq.(2.14), and s_q is a quantity scale factor that prevents the target quality from becoming artificially large before enough data exists to make an accurate quality assessment.

In this thesis the s_q factor is chosen to be

$$s_q = \begin{cases} \frac{n}{N_q} & ; \; n < N_q \\ 1 & ; \; n \geq N_q \end{cases} \tag{2.27}$$

where n is the number of contacts in a given contact set as defined in Section 2.1 and N_q is some quantity threshold. The s_q component of Eq.(2.26) serves to linearly increase the quality's maximum bound from zero to one until at least N_q contacts are present, after which s_q is one. This causes Q to be a measurement of not only the geometric qualities of the set, but also how many contacts are present. The quality measurement is designed to indicate how well target localization has occurred, and a sub-standard Q indicates that something has gone wrong. The s_q technique aligns with this objective by causing a noticeable reduction in Q as an indication that the small sample size may be a problem.

The reason for using s_q is that the experimental setup used in this thesis can have runs of three or four contacts that are generated from very similar flight positions and dynamic conditions. These contacts can be uniformly biased such that they generate an overly-optimistic target quality. This thesis uses a value of 10 for N_q because it is about twice the maximum run size, but other values may be warranted if the maximum expected run size changes. Another alternative is to eliminate the s_q factor and evaluate a target's raw quality along with the size of the contact set.

In addition to the s_q factor, Eq.(2.26) also contains a normalized measurement of the error accumulation at the target center compared to the the error accumulation at a specified distance away from the target center. A target's quality is not defined for a set with only one contact. This would cause $\mathbf{E}(\mathbf{A_C}, \mathbf{\Gamma}) = \mathbf{E}(\mathbf{B_C}, \mathbf{\Gamma}) = \mathbf{E}(\mathbf{C}, \mathbf{\Gamma}) = 0$, which means that the target axis would align perfectly with the only contact. Thus, the contact set is assigned a quality of zero in the single contact case. In the multiple contact case, the target quality compares a contact set's precision with its directional diversity. If the set does not cluster very tightly around the target center, then quality goes down. If all of the set's contacts

point in nearly the same direction, then quality again goes down. A low target quality causes the error to be sensitive in the $\hat{\mathbf{t}}$ direction.

Notice that points $\mathbf{A_C}$ and $\mathbf{B_C}$ are calculated using L_t, and that Q is calculated using the error sum at points $\mathbf{A_C}$ and $\mathbf{B_C}$. This relationship causes the target quality to increase as the test length increases. It is, therefore, beneficial to choose some characteristic length as a standard for comparison. Let L_t be the average distance between each contact origin and the target center:

$$L_t = \frac{\sum_{i=1}^{n} \|\mathbf{w_i} - \mathbf{C}\|}{n}. \tag{2.28}$$

This definition for L_t can be used in Eq.(2.25), and the resulting points $\mathbf{A_C}$ and $\mathbf{B_C}$ can then be used in Eq.(2.26) to calculate the target quality.

The concept of target quality offers a unique contribution to both target localization and search techniques. It is a number between zero and one that serves as a standard for comparison between various locating tasks. Accurate long-range sensors may produce a target quality in the .95 to .99 range. Mid-range locating may warrant a comparison in the .7 to .8 range, and low-accuracy sensors may operate in the .4 to .6 range[2]. Thus, target quality offers a standard of comparison between different search scenarios and equipment capabilities. Additionally, if a given setup typically generates target qualities in a certain range, then qualities lower than that range give rise to appropriate questions such as: "Was there more than one target?", "Was there a target there at all?", and "Were there too few target contacts?". In any case, a target quality outside of the expected range is justification for an additional look.

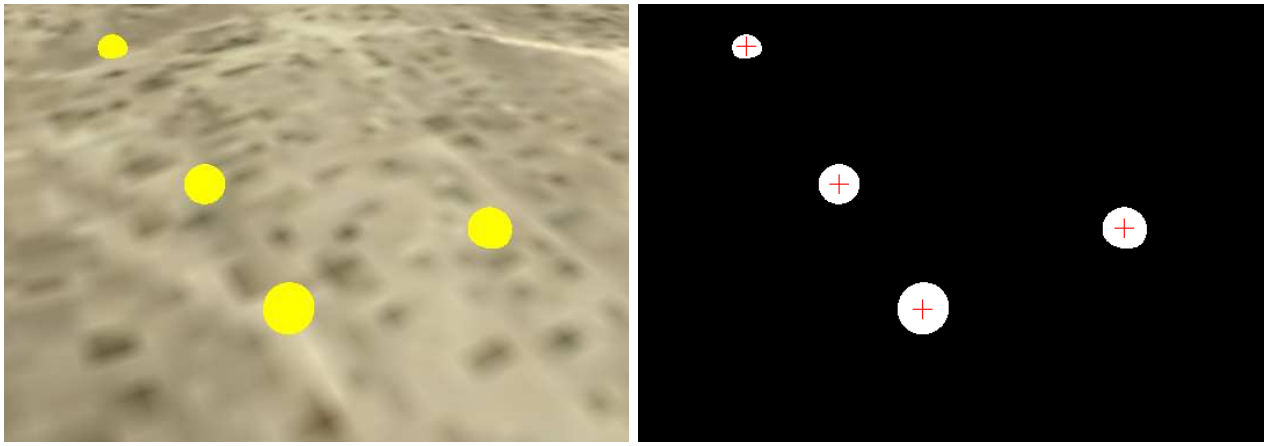
2.4 The Differentiation Algorithm

In addition to the original contributions of 3D target localization and target quality, this thesis also offers a target differentiation algorithm to enable UAVs to carry out a meaningful search. Target differentiation using computer vision is the process of distinguishing

[2]These numbers are only hypothetical, but calculated target qualities may be seen in Table 2.3. For applications where some contact error exists, an increase in contact accuracy causes the target quality to asymptotically approach unity. A target quality of $1.\overline{00}$ would indicate zero error, such that all contacts exactly intersect each other at a single point in space.

between multiple targets in an image, creating one contact for each detected target, and assigning each contact to a separate contact set.

To distinguish between targets, the method of color segmentation is used to identify pixels on the screen that are within predefined color thresholds. These thresholds are measured in hue, saturation and value (HSV). The pixels that are within the HSV thresholds are considered to be part of a target. As long as the in-threshold pixel groups do not overlap and the target pixels are interpreted correctly, a connected component algorithm can verify that a group is completely isolated from any other. (See Figure 2.12.)



(a) Simulation Image (b) Color Segmented Image

Figure 2.12: Target identification and separation using HSV color segmentation. In the color segmented image, the pixels that are within the HSV thresholds are shown in white, while others are shown in black. Each connected group has cross-hairs positioned at the average of its pixel locations.

Once pixels groups have been established, the average pixel location for each group may be calculated. This average pixel location is then used to create $\breve{\mathbf{u}}_\mathbf{c}$ as in Eq.(2.22), which then has its image skew removed to obtain $\mathbf{u_c}$. This vector must then be normalized and transformed into the world-frame, where a new contact is born according to Eq(2.9). This process should then be repeated for each pixel group in the image, and all of these contacts should be put into a contact set, $\mathbf{\Gamma_I}$, called the *image set*. Here it is also convenient

to define a *target set*, $\mathbf{\Gamma_j}$, as the contact set containing contacts that have been assigned to a single target, where $j = \{1, 2, 3, \ldots, m\}$, and m is the total number of target sets.

One key to target differentiation is to make sure that each γ in $\mathbf{\Gamma_I}$ is assigned to a different $\mathbf{\Gamma_j}$. Thus, additional information concerning the minimum number of targets can be brought from the image, and directly applied to contact assignment in the world frame. The algorithm assigns each image contact to the target set with the closest target center as long as no other image contact is closer to the target center. It does this only if the contact-to-target distance is less than some maximum assignment distance, $d_{\gamma m}$. (Contact assignment pseudo-code is included in Appendix A.2.) However, if all of the nearby targets have already been assigned a new γ from $\mathbf{\Gamma_I}$, and there are still more contacts in $\mathbf{\Gamma_I}$, then a new $\mathbf{\Gamma_j}$ is created.

This assignment algorithm can mis-assign contacts in some cases. For example, if there is enough error to cause a contact that should be assigned to $\mathbf{\Gamma_1}$ to be closest to $\mathbf{\Gamma_2}$, then that contact may be assigned to $\mathbf{\Gamma_2}$, or to a new target set. (Error in this case is the distance from the contact to the target center point as defined in Eq.(2.13).) This mis-assignment tends to happen when contact error is as large as the target spacing. Also, if a target moves off the screen, and a completely new target moves onto the screen, then a new contact could be assigned to the old target if it is still within $d_{\gamma m}$ from the old target. These situations tend to occur when there is large contact error compared to the target spacing.

For this thesis the specific contact mis-assignments are not recorded. They are simply allowed, and then a target center calculation is made. If mis-assignment occurs, then the misfit contacts may affect the solution somewhat. However, as described at the end of Section 2.1, they should not modify the center location significantly as long as a sizeable cluster of properly-allocated contacts already exists.

In summary, target differentiation has the following steps:

1. Identify separate targets in an image, and mark the target center pixels.
2. Calculate target directions, de-skew them and transform them to the world frame.
3. Tie target direction with UAV position to form contacts.
4. Put all contacts, γ, from an image into the same image set, $\mathbf{\Gamma_I}$.

5. Assign each γ in $\mathbf{\Gamma_I}$ to a different $\mathbf{\Gamma_j}$.

6. Calculate target centers for each $\mathbf{\Gamma_j}$ that received a contact.

7. Repeat steps 1 through 6 for each processed image.

2.5 Flight Testing

2.5.1 Test Setup

This section describes both the equipment and the procedure used in the target differentiation flight test. The UAV platform is a Unicorn flying wing design, as shown in Figure 2.13. It has a wingspan of 1.05 m and cruises at about 13 m/s for 45 minutes. It utilizes a complement of remote control (RC) aircraft batteries, servos, and other hardware that are typical for an aircraft of its size.

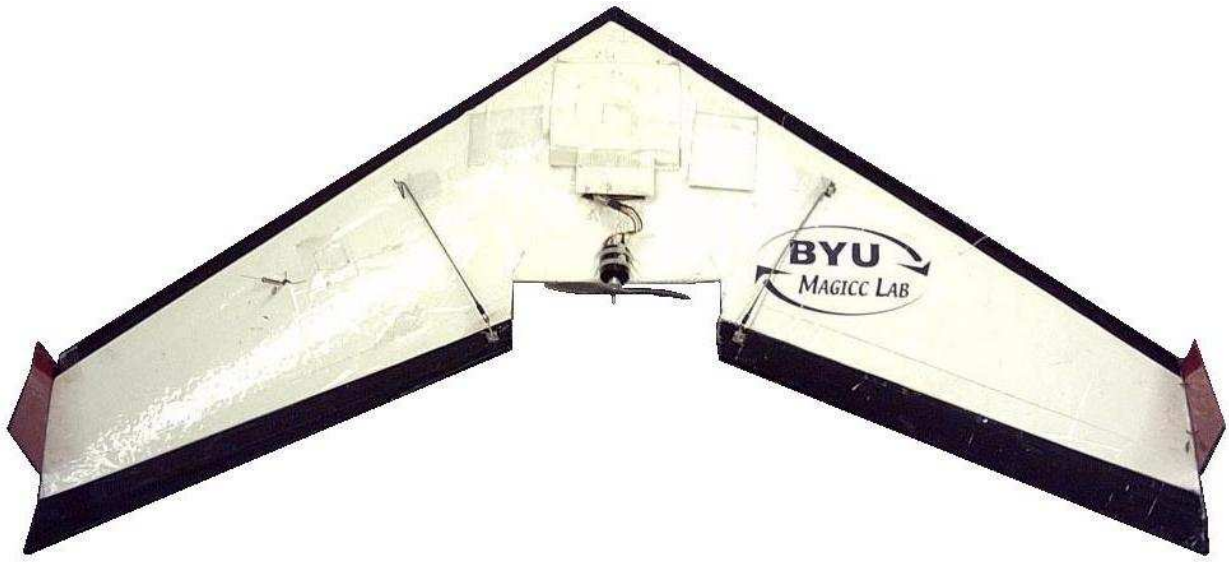


Figure 2.13: The SUAV used in the flight test.

The camera is secured inside a foam pod on the underside of the UAV, as shown in Figure 2.14. The exact orientation of the camera relative to the UAV body frame is not known precisely, but is estimated visually before the flight test. The flight test includes a camera orientation calibration to account for this unknown.

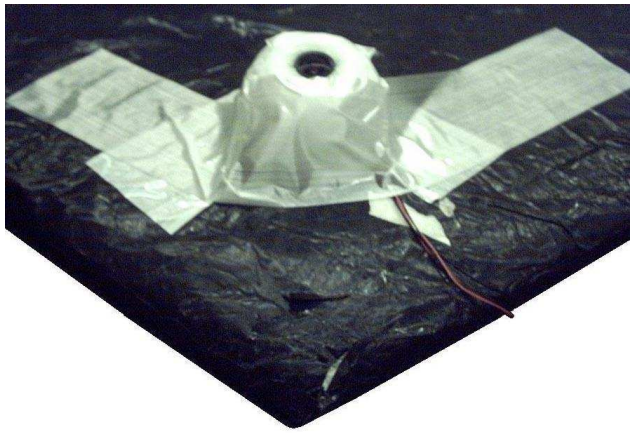


Figure 2.14: Camera mounted on the underside of the SUAV.

Video is sent from the UAV and received at the ground station using a video transmission system from Black Widow AV [9]. The video signal is then transferred through the ImperX VCE-Pro frame grabber to a laptop, where it can be accessed by Virtual Cockpit for video processing. (See Figure 2.15.)

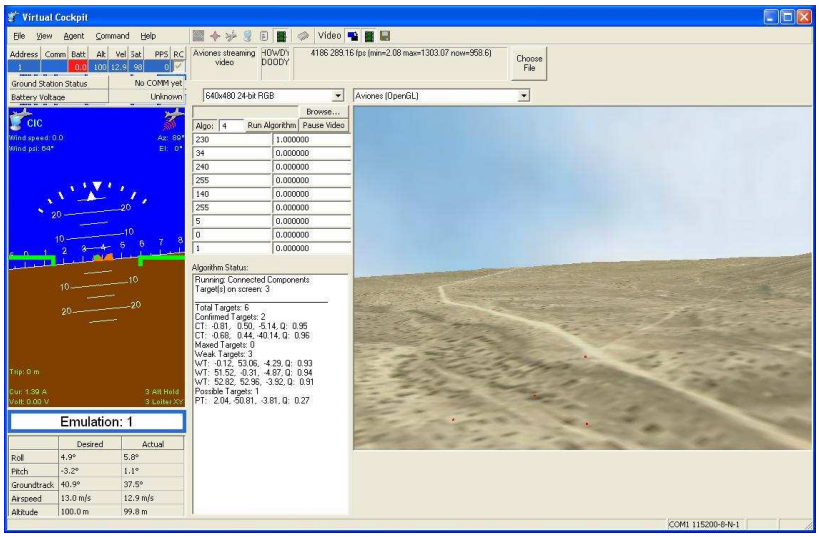
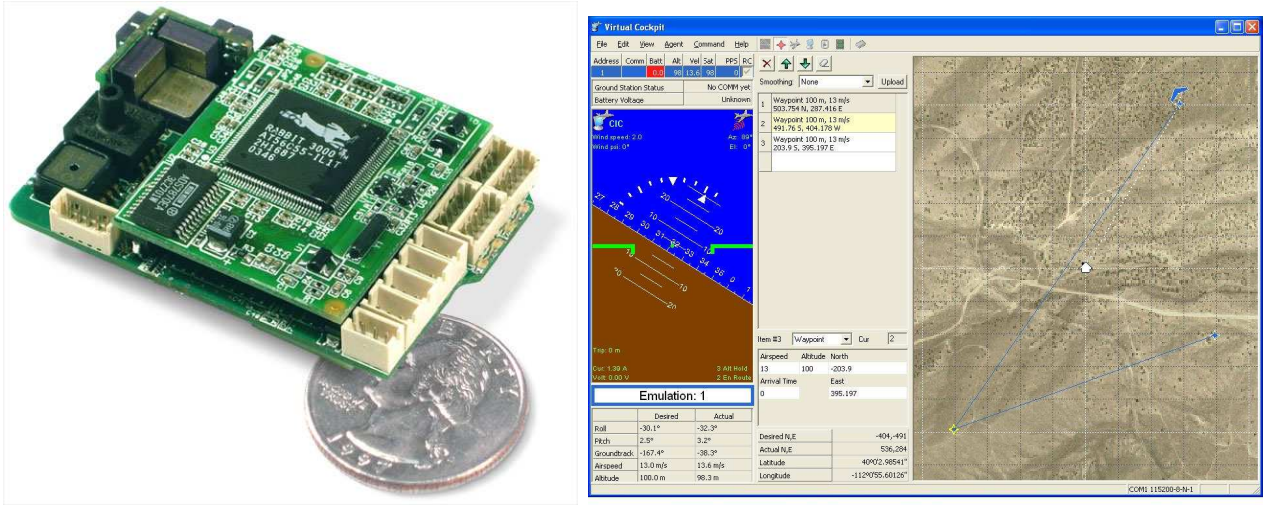


Figure 2.15: Simulated video in virtual cockpit.

In addition to the task of video processing, Virtual Cockpit also monitors the UAV and generates the UAV flight path. Once the flight path has been generated by Virtual

Cockpit, it is sent to the UAV autopilot via modem. The UAV uses a Foruno GH-80 GPS unit to locate itself in the world coordinate frame, along with the autopilot's air pressure reading for altitude measurement. The autopilot then controls the UAV along its path, and telemetry is transmitted back to the ground station by modem. After receiving this update, Virtual Cockpit matches the telemetry and video information to generate target contacts. Detailed information about the ground station and Kestrel autopilot may be found in [1] and [14]. (See Figure 2.16.)



(a) Kestrel Autopilot 2.X. (b) UAV control in Virtual Cockpit.

Figure 2.16: The Kestrel autopilot provides on-board UAV control, while Virtual Cockpit provides path information and receives UAV telemetry.

The procedure for this test begins with arranging multiple sheets of red cloth on the ground. These sheets have an area of approximately 1 m^2, and their red color makes them easy to identify through color segmentation. The location of each cloth is measured by placing the UAV on the target, and recording its position. These are the known targets.

The camera is focused visually by viewing the video output and adjusting the lens position. It is then calibrated using the method described in Section 2.2, and attached to the UAV. The camera azimuth and elevation angles are estimated visually relative to the body frame, and all of the camera parameters are recorded for use during the flight test.

After the camera mounting is complete, the UAV is launched and follows a simple flight plan consisting of one waypoint that is 200 m north of the known targets, and one waypoint 200 m south of the targets. The UAV flies from one waypoint to the other and back at an altitude of 60 m, passing over the targets each time.

The largest contact error is expected to be approximately 10 to 15 m, so a safe $d_{\gamma m}$ is 30 m. In other words, $d_{\gamma m}$ specifies that a contact will be allocated to the closest estimated target if it is not more than 30 m from it. To keep contact sets from becoming too large, a maximum of 30 contacts is allowed in each set. Beyond that, the contact that is furthest from the estimated target center will be deleted to make room for any incoming contacts.

On the first pass, target pixels in an image are analyzed to determine the appropriate hue, saturation and value limits for target detection. These limits are used in subsequent passes to allow the computer to identify the targets in an image. Contacts are then generated in Virtual Cockpit as targets appear in analyzed images, and center locations are calculated. The camera orientation is calibrated by finding the azimuth and elevation which minimize the total distance between each contact, and the nearest known target (see Section A.3 for calibration code). Each target's quality is monitored to see that reasonable data is obtained, and the data is stored in a file for post-test evaluation.

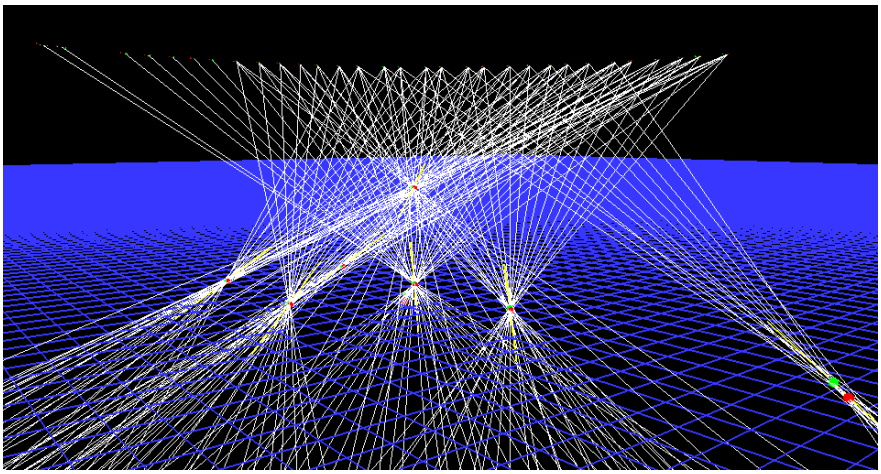


Figure 2.17: Visual 3D analysis of target differentiation performance.

As always, the last step in testing is the analysis. For target differentiation, the most effective analysis is a visual inspection in a 3D virtual world as shown in Figure 2.17. This is done through an application that renders one line per contact. Each line starts at the contact's origin and extends in the contact's direction. The application also renders several green and red spheres, which represent the estimated and known target positions respectively. It is important that the analyst be able to zoom, pan and rotate in the virtual world to gain full comprehension of the decisions made by the target differentiation algorithm. For this thesis, the described application is programmed using OpenGL visual libraries.

2.5.2 Results

Table 2.2 shows the measured positions of the known targets. All measurements are in meters, and the altitude is measured in height above ground. Figure 2.18 shows these targets as seen from the airborne UAV.

Table 2.2: Known target positions.

East	North	Altitude
1	-5	0
-11	3	0
-24	-13	0
-21	10	0
-10	12	0
-9	18	0

The UAV made a total of 15 passes over the known targets after the hue, saturation and value limits had been set. Images were processed once every half second, which caused target contacts to be created and center positions to be calculated.

The azimuth and elevation calibration was also performed, which decreased the average distance between each contact and its nearest known target from 6.11 to 5.89 m (see Figure 2.19 for the distance distributions). The optimum camera orientation, $(\alpha^*_{az}, \alpha^*_{el})$, was found to be $(.0041, 1.3878)$ radians, as compared to the estimated $(0, 1.35)$ radians. The

Figure 2.18: Targets observed from the airborne UAV during the flight test (red darkened for grayscale contrast).

pre-calibration and post-calibration target estimates are shown in Table 2.3, and the 2D position of each target with a non-zero quality is shown in Figure 2.20.

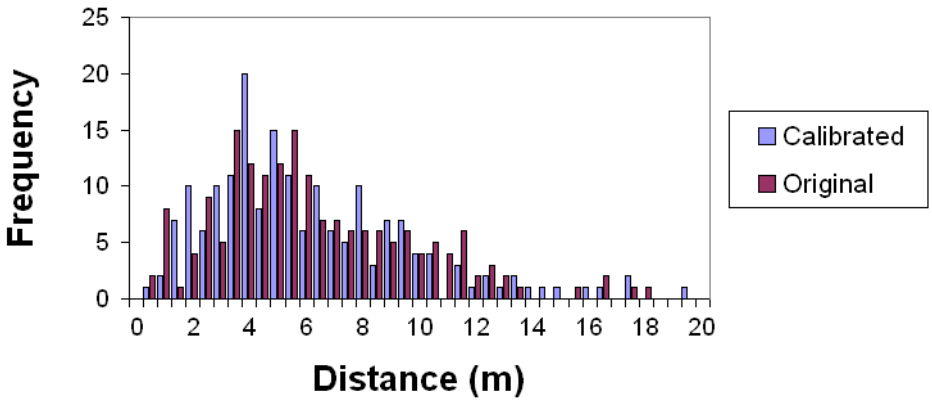


Figure 2.19: Contact distance from the nearest known target.

After the field test was over, the data was viewed in a 3D virtual world to determine the complexity of the data and the effectiveness of the target differentiation algorithm. A virtual view of the target area is shown in Figure 2.21. The qualitative results are difficult to show on paper, but one can see the complex web of contacts that the target differentiation

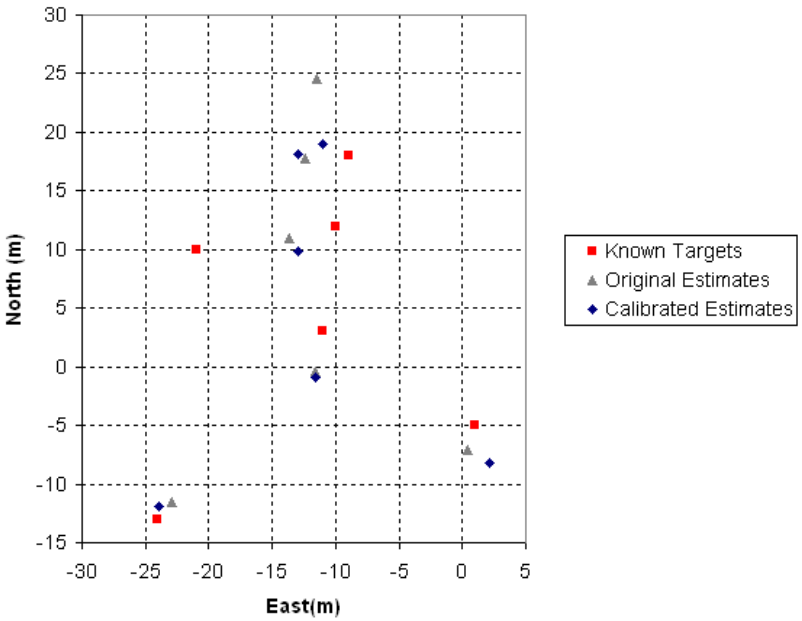


Figure 2.20: 2D comparison of the known and estimated targets.

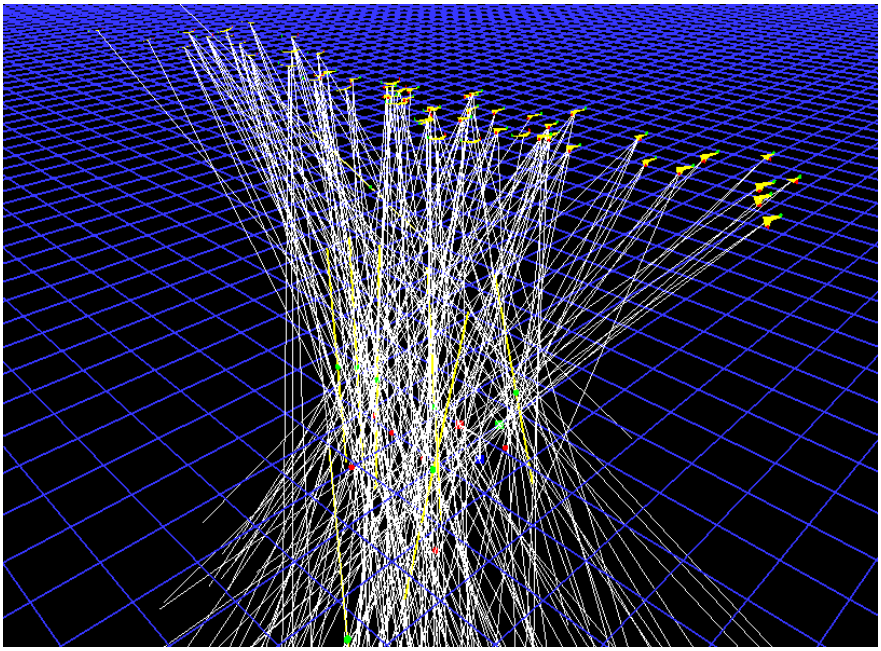


Figure 2.21: 3D visualization of the field test.

Table 2.3: Known vs. estimated target positions.

Known Targets			Original Estimates				Calibrated Estimates			
East	North	Alt.	East	North	Alt.	Q	East	North	Alt.	Q
-10	12	0	-13.63	11.00	13.69	0.729	-12.95	9.88	12.34	0.690
-11	3	0	-11.60	-0.40	13.07	0.673	-11.64	-0.85	9.74	0.647
-9	18	0	-12.35	17.79	12.32	0.660	-12.98	18.10	9.07	0.586
-24	-13	0	-22.88	-11.48	14.03	0.571	-23.89	-11.86	9.57	0.562
1	-5	0	0.46	-7.07	13.08	0.640	2.10	-8.20	8.96	0.538
-21	10	0	-11.46	24.56	8.78	0.450	-11.03	19.03	-11.45	0.393
			64.70	104.05	0.00	0.000	65.73	107.24	0.00	0.000
			37.30	-127.16	0.00	0.000	39.39	-128.33	0.00	0.000
			6.73	-120.87	0.00	0.000	10.04	-122.24	0.00	0.000
			-43.05	-15.68	0.00	0.000				

algorithm was dealing with. A total of 183 contacts were produced during the flight test, and upon visual inspection it is difficult even for a human to tell where the targets should be placed. However, the estimated target positions seem reasonable based on the contacts that were generated.

2.5.3 Discussion of Results

There are many positive outcomes from the field test. First, the test reveals the presence of false targets, and shows one way that false targets can occur in image processing. Analysis of Figure 2.22 reveals that static video contains primarily red and green clusters, with white lines across the image. False targets were primarily seen during turns far from the video receiver, and were probably a result of the vision algorithm interpreting these red clusters as targets. It is important to note that if any of these false targets had occurred near the known targets, and had increased the number of on-screen targets to more than six, then the algorithm would have generated more than six valid (non-zero quality) estimates.

The second positive outcome from the field test is that the target differentiation algorithm was able to identify the correct number of targets. The test involved six known targets, and in both the original and calibrated cases there were six valid estimated targets. This result seems to indicate that the target differentiation algorithm has a tendency to identify the correct number of targets, even when contact accuracy is poor. There is no

Figure 2.22: Static video from the field test.

guarantee that this is always the case, but if the maximum contact assignment distance, $d_{\gamma m}$, is within certain bounds, then this tendency is manifest.

For example, the choice of 30 m for $d_{\gamma m}$ was a good one for the field test. Some contacts had as much as 19.1 m of error (excluding false targets), so an assignment distance as high as 30 m did not separate any contacts from their parent target. However, $d_{\gamma m}$ was also small enough to prevent contacts from being assigned to targets outside the image. For this particular case, if $d_{\gamma m}$ had been more than about 70 m, then this could have occurred. As long as $d_{\gamma m}$ is within these reasonable bounds, then the target differentiation algorithm has the tendency to estimate the correct number of targets even with poor contact accuracy.

The third positive outcome from the field test is the surprising accuracy of the calibrated target estimates. Despite all of the challenges, the 2D difference between the calibrated and known targets is less than four meters on five out of the six valid estimated targets. (See Table 2.4.) In addition, the sub-standard target quality of the sixth estimate indicates that mis-assignment may have occurred. These results are surprisingly accurate when compared to the given contact error. It is important to note that the sensitive direction to error is nearly straight down, and the altitude consistency of the first five target estimates poses a question as to the accuracy of the UAV altitude measurement. However, even if the 3D difference is considered, the maximum distance between the estimated and actual targets is less than 18 m. Some of the known targets are just too close together,

which assumedly causes one inaccurately placed target estimate (altitude of -11.45 m) from the resulting mis-assignment.

Table 2.4: Distance between calibrated estimates and known targets (m).

Calibrated Target Estimates				Known Targets			Difference	
East	North	Alt.	Q	East	North	Alt.	2D Distance	3D Distance
-23.89	-11.86	9.57	0.562	-24	-13	0	1.15	9.64
2.1	-8.19	8.96	0.538	1	-5	0	3.38	9.58
-12.95	9.88	12.34	0.690	-10	12	0	3.64	12.87
-11.64	-0.85	9.74	0.647	-11	3	0	3.9	10.49
-12.98	18.1	9.07	0.586	-9	18	0	3.98	9.91
-11.03	19.03	-11.45	0.393	-21	10	0	13.5	17.67

After the test was finished, comparison between Figure 2.18 and Figure 2.20 brought up a question as to the accuracy of the known target measurements. Figure 2.23 shows the position error associated with placing the test UAV on the ground and obtaining stationary position readings. The large change in position near the beginning of the experiment is due to a drop in the number of GPS satellites in communication. However, even with a healthy number of satellites communicating, the GPS reading tends to vary. The type of variation present in Figure 2.23 is also expected to be present in the position measurements of the known targets. A better position estimate could be obtained in the future by using a more accurate differential GPS unit, or by measuring target position for a long time.

In addition to the UAV position uncertainty shown in Figure 2.23, there is probably significant error in attitude estimation that would be directly correlated with the contact error already shown. However, there is currently no method for measuring the true attitude of this SUAV in flight.

Understandably, the presented work does not serve as a proof of target differentiation's dependability. Several more flight tests would be preferable to further validate the robustness of the presented method, but this task is left to those pursuing future work on this subject.

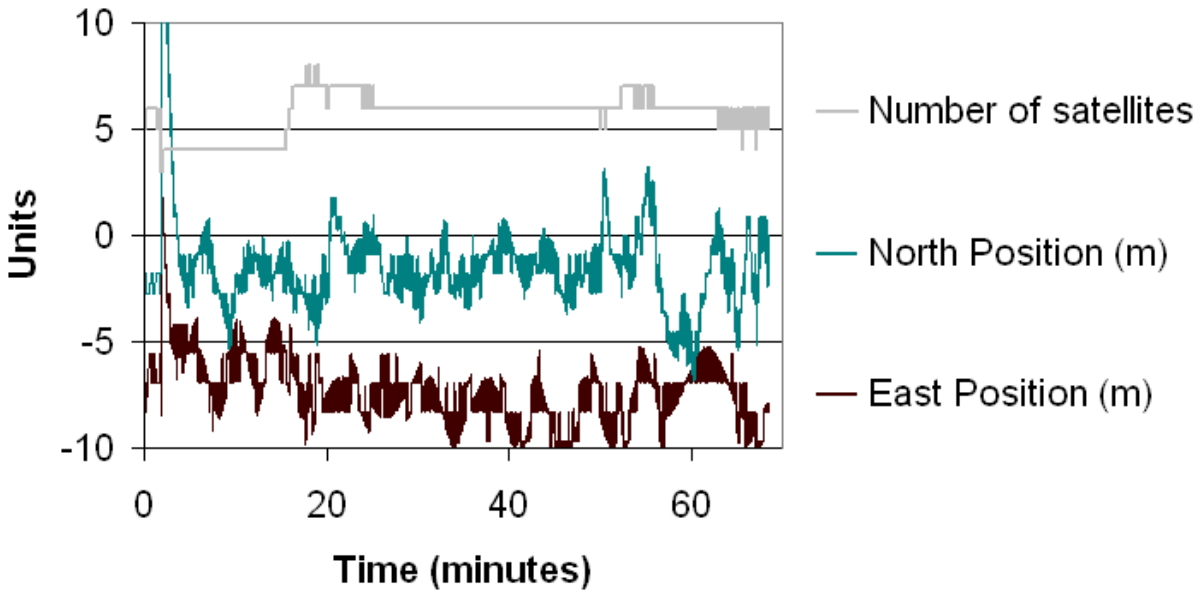


Figure 2.23: Position measurements for a stationary UAV.

2.5.4 Conclusion

These test results are valuable in understanding the behavior of the target differentiation algorithm when contact accuracy is poor. There are contacts with as much error as 19.1 m (excluding false targets), and yet the algorithm is able to count the correct number of targets when some are as close as 6.1 m apart. In addition, the 2D target estimates are less than four meters from the measured positions on five out of six targets, and the sub-standard target quality of the sixth estimate appropriately indicates a problem.

The field test shows that the original contributions of 3D target localization, target quality, and target differentiation can be applied to real-world tasks. In the future, target estimation accuracy should increase as UAV position and attitude estimation improve. For now, these results lay the groundwork of a meaningful search for multiple targets, where they can be differentiated and located.

Chapter 3

Optimal Search Theory

It is very helpful to review and understand some basic principles in optimal search theory before planning any search strategy. This chapter does not represent original contributions of this thesis, nor is it intended to be a full derivation of optimal search theory. However, it is intended to usefully depict some of the applicable contributions, as well as to direct interested readers to the appropriate published works. The material in this chapter is based on references [2] and [6], written by Bernard O. Koopman and Lawrence D. Stone. Although these men were contemporary researchers, much of Stone's work is based on the work of Koopman, and Koopman's work is derived from extensive collaboration and World War II experience. Much of the theory in this chapter is attributed to Koopman.

Some of the most applicable ideas from Koopman include the following:

- A probability model for visual detection
- The definition of a lateral range curve, and its associated search width
- An upper bound, lower bound and a mid-line case to describe target detection probability as a function of search effort in an area
- An optimal allocation of search effort between multiple areas
- Search strategy reversal with large search effort
- Guidelines for optimal scanning

Some of the most applicable ideas from Stone include the following:

- The dependence of detection probability on path placement accuracy
- An intuitive description of the changes to a probability distribution throughout an optimal search.

- Proofs regarding optimal search in the presence of false targets
- Guidelines in optimal nonadaptive, semiadaptive and adaptive searches

3.1 Two Detection Models

There are two detection models that are often used in optimal search theory: the definite range law, and the inverse cube law. Given that a target is present at range[1] r from the observer, the *definite range law* assumes that there is some definite range, R, beyond which there is zero probability of detecting the target, and within which the detection probability is unity. In other words,

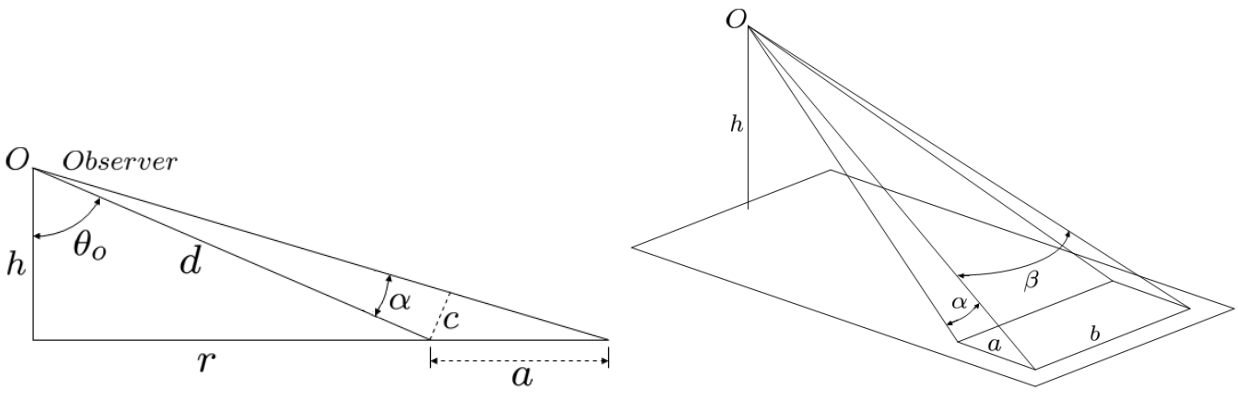
$$p\Big(D\Big|I(r), M_{DR}\Big) = \begin{cases} 1 & ; \ r \leq R \\ 0 & ; \ r > R \end{cases} . \tag{3.1}$$

This thesis uses D to indicate detection, and $I(*)$ to indicate an instantaneous glimpse of a target that is present at $*$, where $*$ can be expressed as range r or absolute distance d. The term M_{DR} is used to denote the definite range detection model. Thus, Eq.(3.1) represents the probability of detecting a target if given an instantaneous look at the target from range r using the definite range detection model. This quantity does not take into account the possibility of false targets, but only describes the probability of successfully detecting a target given that it is there.

The definite range law is not a very realistic model, but is frequently used as a standard for comparison and a quick rule of thumb. In contrast, the *inverse cube law* is a probability model for visual detection based on the solid angle of a target to its observer. Specifically, the model is developed for an airborne observer that is searching for the wake of an ocean vessel. In Figure 3.1 the approximate vertical and horizontal viewing angles are

$$\alpha \approx \frac{c}{d} \quad \text{and} \quad \beta \approx \frac{b}{d}.$$

[1]Range is distance along the ground.



(a) (b)

Figure 3.1: Solid angle diagrams. The airborne observer O is viewing the wake of an ocean vessel from height h. The wake has area $a \times b$, and is range r away from the observer.

By similar triangles, $\frac{c}{a} = \frac{h}{d}$ and $\alpha = \frac{ah}{d^2}$. The solid angle then becomes

$$Solid\ angle = \alpha\beta = \frac{abh}{d^3}.$$

Or, for an arbitrary wake with total area A,

$$Solid\ angle = \frac{Ah}{(h^2 + r^2)^{\frac{3}{2}}}. \tag{3.2}$$

The instantaneous detection probability is then assumed to be

$$p\Big(D\Big|I(r), M_{IC}\Big) = \frac{kh}{(h^2 + r^2)^{\frac{3}{2}}}, \tag{3.3}$$

where k is a constant that contains the area of the wake and provides the freedom to adjust for other factors such as lighting, the observer's ability, etc. The term M_{IC} is used to denote the inverse cube detection model. If $h \ll r$, then Eq.(3.3) takes the form

$$p\Big(D\Big|I(r), M_{IC}\Big) = \frac{kh}{r^3}. \tag{3.4}$$

The inverse cube law of sighting is widely recognized in this form, and gets its name from the r^3 term in the denominator. However, notice that without any further assumptions

Eq.(3.3) can also take the form

$$p\Big(D\Big|I(d), M_{IC}\Big) = \frac{k}{d^2}\frac{h}{d} = \frac{k}{d^2}\cos\theta_o. \tag{3.5}$$

In other words, Koopman's inverse cube law can be considered an inverse square law that includes a factor for the orientation of the target, and is written in terms of the absolute distance from the observer to the target.

3.2 The Lateral Range Curve and Search Width

Once a detection model is assumed, it can be used to generate a lateral range curve. Lateral range is depicted in Figure 3.2, and is the shortest distance from some point to the observer's path.

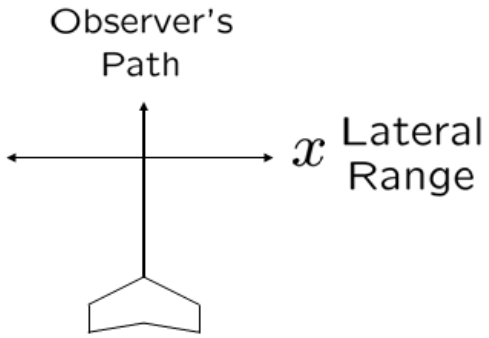


Figure 3.2: The lateral range of a point is the minimum distance between it and the observer's path. Consequently, measurements of lateral range are orthogonal to the path.

A lateral range curve indicates the probability of detecting a target given that the observer passes by it at lateral range x. (See Figure 3.3.) The term $P(x)$ is used to denote the target glimpses associated with a long, straight pass by the target at lateral range x. The term M is used to denote some detection model, which implies that M could be legitimately replaced by M_{DR}, M_{IC}, etc.

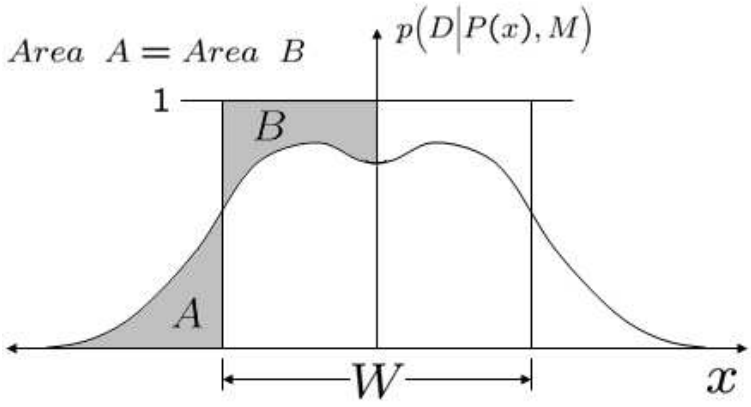


Figure 3.3: A lateral range curve and its search width. The lateral range curve indicates the probability of detecting a target given that the observer passes by it at lateral range x. W is the area under the curve, and can be used as the spacing between paths to allocate a fairly uniform search coverage.

The *effective search width* is defined as the area under the lateral range curve,

$$W = \int_{-\infty}^{+\infty} p\Big(D\Big|P(x), M\Big)dx. \tag{3.6}$$

It also happens to be a parameter that defines the range where area A is equal to area B, as shown in Figure 3.3. The importance of this condition is that it causes parallel passes placed W apart to yield fairly uniform coverage, and prevents wasted overlap or missed areas. The $A = B$ characteristic does not guarantee perfect search uniformity, but is a useful heuristic that creates a condition where the "missed" probability from one pass is somewhat covered by the next. Perhaps the most compelling reason to use the given definition for W is that it is a convenient way to predict the search rate, as shown in Section 3.3.

A lateral range curve can be obtained in closed form for the definite range model such that within range R the probability of detection is unity, and outside of that range it is zero as shown in Figure 3.4. The associated search width in this case is $2R$.

A lateral range curve may also be obtained in closed form for the inverse cube detection model. Let $S(t)$ represent some specified set of target glimpses occurring from time zero to time t and consider an observer who is stationary at range r from a target. In the case of continuous looking and a fixed instantaneous detection probability, the probability

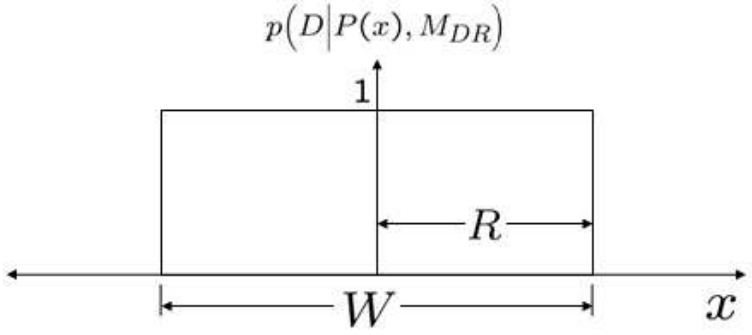


Figure 3.4: The lateral range curve of the definite range detection model.

of target detection by time t is

$$p\Big(D\Big|S(t), M_{IC}\Big) = 1 - \exp\left(-p\Big(D\Big|I(r), M_{IC}\Big) \times t\right). \tag{3.7}$$

When $p\big(D\big|I(r), M_{IC}\big)$ changes over time, the probability of target detection by time t is

$$p\Big(D\Big|S(t), M_{IC}\Big) = 1 - \exp\left(-\int_0^t p\Big(D\Big|I\big(r(t)\big), M_{IC}\Big)dt\right). \tag{3.8}$$

Now imagine an observer traveling in the $+y$ direction from $-\infty$ to $+\infty$ at height h and velocity v as shown in Figure 3.5. Also imagine a target placed at position x relative to the observer's path.

Assuming that the observer in Figure 3.5 begins a journey from $y = -\infty$ at time zero and continues on to $y = \infty$, the set $S(t)$ in Eq.(3.8) can be replaced by $P(x)$. In other words, the specified set of target glimpses is now declared to be the set of glimpses generated by passing a target at lateral range x. Eq.(3.8) then becomes

$$p\Big(D\Big|P(x), M_{IC}\Big) = 1 - \exp\left(-\int_{-\infty}^{+\infty} p\Big(D\Big|I\big(r(x,y), M_{IC}\big)\Big)\frac{dy}{v}\right). \tag{3.9}$$

Substituting the inverse cube law model from Eq.(3.4) in for $p\big(D\big|I(r(x,y)), M_{IC}\big)$ yields

$$p\Big(D\Big|P(x), M_{IC}\Big) = 1 - \exp\left(-\int_{-\infty}^{+\infty} \frac{kh}{\big(r(x,y)\big)^3}\frac{dy}{v}\right) = 1 - \exp\left(-\int_{-\infty}^{+\infty} \frac{kh}{(x^2+y^2)^{\frac{3}{2}}}\frac{dy}{v}\right).$$

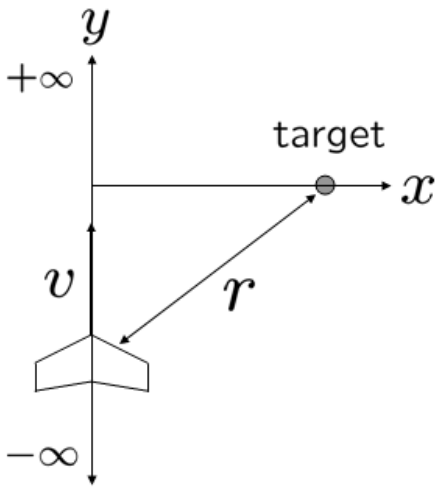


Figure 3.5: A long, straight pass by a target at lateral range x. The observer is traveling at speed v, and views the target from range r.

Trig. substitution of $dy = x\sec^2\theta d\theta$ and $\sqrt{x^2+y^2} = x\sec\theta$ may then be used to obtain

$$p\Big(D\Big|P(x), M_{IC}\Big) = 1 - \exp\left(-\frac{kh}{vx^2}\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}}\cos\theta d\theta\right),$$

which becomes

$$p\Big(D\Big|P(x), M_{IC}\Big) = 1 - \exp\left(-2\frac{kh}{vx^2}\right). \tag{3.10}$$

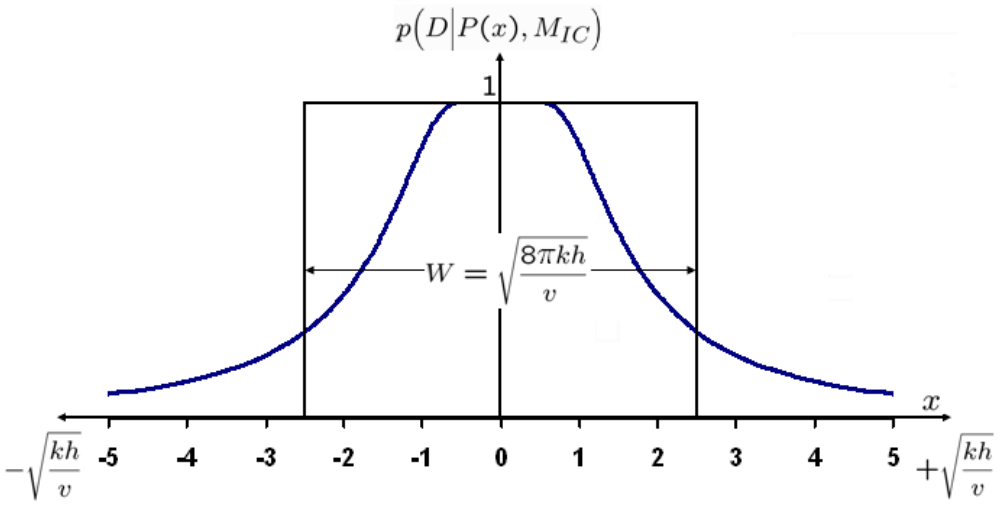


Figure 3.6: The lateral range curve of the inverse cube detection model.

The lateral range curve from Eq.(3.10) is shown in Figure 3.6, and the associated search width can be calculated by substituting the result from Eq.(3.10) into Eq.(3.6). Integration is easiest in the form

$$W = 2\int_0^\infty 1 - \exp\left(-2\frac{kh}{vx^2}\right) dx,$$

which is designed to use the known solution

$$\int_0^\infty \exp\left(-z^2\right) dz = \frac{\sqrt{\pi}}{2}.$$

The resulting search width is

$$W = \sqrt{\frac{8\pi kh}{v}}. \tag{3.11}$$

The same method[2] used to calculate the lateral range curve and search width for this detection model may be applied to any detection model. However, for complex detection models a numeric solution may be preferable.

3.3 Search in an Area

Consider the area shown in Figure 3.7, where the total length of search effort is divided into $n_{_L}$ pieces. For now the definite range law is assumed, which means that each piece of search effort, $\frac{L}{n_{_L}}$, indicates that an area $\frac{WL}{n_{_L}}$ has been completely searched. Given that a single target is within area A, and given that the observer follows a random path that is confined to A, the probability of detection for each piece is $\frac{WL}{n_{_L}A}$. This assumes that each piece is an independent trial in A.

Thus, the probability that detection will fail for each piece is $1 - \frac{WL}{n_{_L}A}$, and the probability that detection will succeed along the whole length is given by

$$p\Big(D\Big|R_A(L), C_A, M_{DR}\Big) = 1 - \left(1 - \frac{WL}{n_{_L}A}\right)^{n_L}, \tag{3.12}$$

[2]A lateral range curve is obtained by considering an observer following a long, straight path by a target at lateral range x, and considering the target's accumulated (time-integrated) probability of detection after the path has been followed. The search width is then found by integrating under the lateral range curve according to Eq.(3.6).

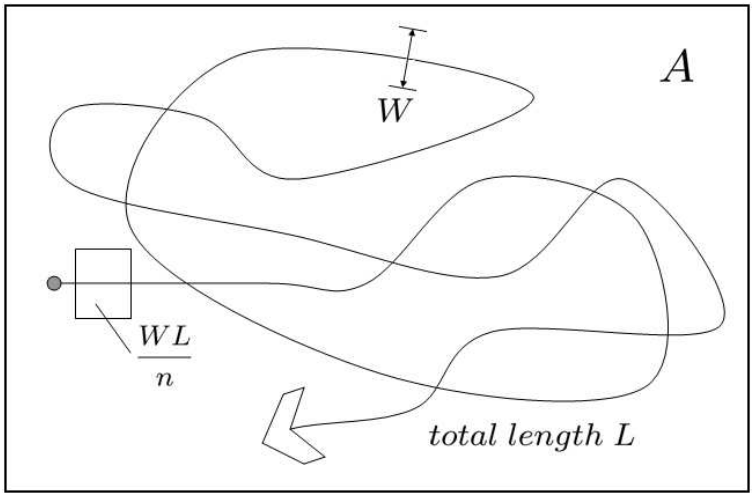


Figure 3.7: Random search in an area.

where $R_A(L)$ represents a random search path of length L in area A, and C_A denotes that the target is contained within area A.

Notice the fundamental change in notation here. The probability of detection still depends on the detection model (M_{DR} in this case), but the target location is no longer known precisely. This also means that the glimpse set is no longer known, and must be represented not only by the observation path, but also by information that describes how likely it is for the target to be at any given location. Without any further information, C_A implies a uniform probability distribution, where it is equally likely for the target to be at any location within A. In short, the new notation shown on the left side of Eq.(3.12) indicates that a target's detection probability is dependent on observer motion, the target's spatial probability distribution, and the detection model.

Now let the number of pieces ($n_{_L}$) approach infinity. Eq.(3.12) then becomes

$$p\Big(D\Big|R_A(L), C_A, M_{DR}\Big) = 1 - e^{\left(-\frac{WL}{A}\right)}. \tag{3.13}$$

This is a *detection function* known as the *random search law*. A detection function tells how the probability of detection increases as a function of search effort (L).

Now consider the scenario where area A is traversed by the same observer, but with a series of parallel lines that are spaced W apart. In this case $p\big(D\big)$ rises linearly with

increasing search effort until $L = \frac{A}{W}$ has been expended. In other words,

$$p\Big(D\Big|Z_A(L), C_A, M_{DR}\Big) = \min\left(\frac{WL}{A}, 1\right), \tag{3.14}$$

where $Z_A(L)$ represents a set of parallel paths with combined length L in area A. After $L = \frac{A}{W}$ is expended the probability of detection is unity and the target has been found. This detection function is called the *definite range law* as shown in Figure 3.8.

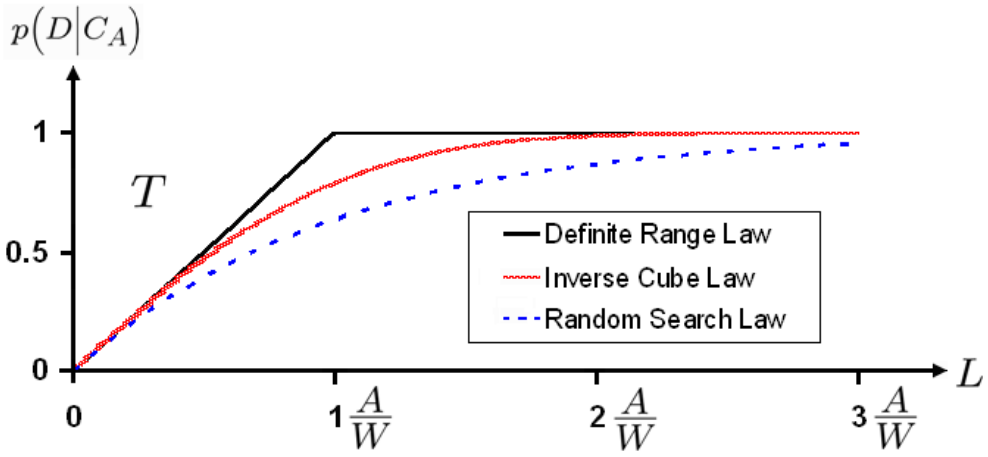


Figure 3.8: A comparison of the characteristic detection functions.

The *inverse cube law* detection function is not explicitly derived in this thesis, but is

$$p\Big(D\Big|Z_A(L), C_A, M_{IC}\Big) = \text{erf}\left(\frac{\sqrt{\pi}}{2}\frac{WL}{A}\right), \tag{3.15}$$

where

$$\text{erf}(u) = \frac{2}{\sqrt{\pi}}\int_0^u e^{-s^2} ds.$$

The derivation may be explored further in [2, p.75-78].

Sometimes there is confusion between the characteristic detection functions, which each correspond to a specific detection model and a specific search path. Hopefully table 3.1 clarifies the issue.

Table 3.1: Clarification of the characteristic detection functions.

Function Name	Search Path Used	Detection Model
Definite Range Law	Parallel Sweeps	Definite Range Law
Inverse Cube Law	Parallel Sweeps	Inverse Cube Law
Random Search Law	Random	Any

The confusion is understandable, because the detection function and the detection model have identical names in some cases. When a law is mentioned, it is important to use context clues to determine whether the subject is a detection model or a detection function. Note that although the random search law was derived using the definite range detection model, any model can be said to apply. This is because any detection model may be used without violating the assumption that search effort is applied randomly in A.

Figure 3.8 shows that the inverse cube law "is close to a middle case, a circumstance which indicates its frequent empirical use, even in cases where the special assumptions upon which its derivation was based are largely rejected." [2, p.79] One might question whether or not these detection functions deserve the title 'law', but this thesis denotes them as such in conformity with prior work. Koopman asserts that the definite range law and the random search law are upper and lower bounds for any search within area A using search width W.

It is helpful to recognize that an observer's search width characterizes its ability to search, and directly contributes to the rate of the search. According to Koopman, a "scanning method is optimum if it renders W a maximum" [2, p.163]. This is useful to keep in mind when the shape of observation is not fixed. However, Koopman's anti-submarine background also indicates that all searching should not necessarily be done laterally from the observer's path, because it is often imperative to detect the target early.

Stone relates an interesting result when searching an area. He shows that the detection function is dependent on one's ability to accurately place parallel paths at a distance W apart [6, p.26]. Assume that there is variation in path placement such that the mean distance between paths is W, with a standard deviation of σ. Then, even when search effort $L = \frac{A}{W}$ has been applied and the definite range detection model is used, the probability of detection may not reach unity. (See Figure 3.9.)

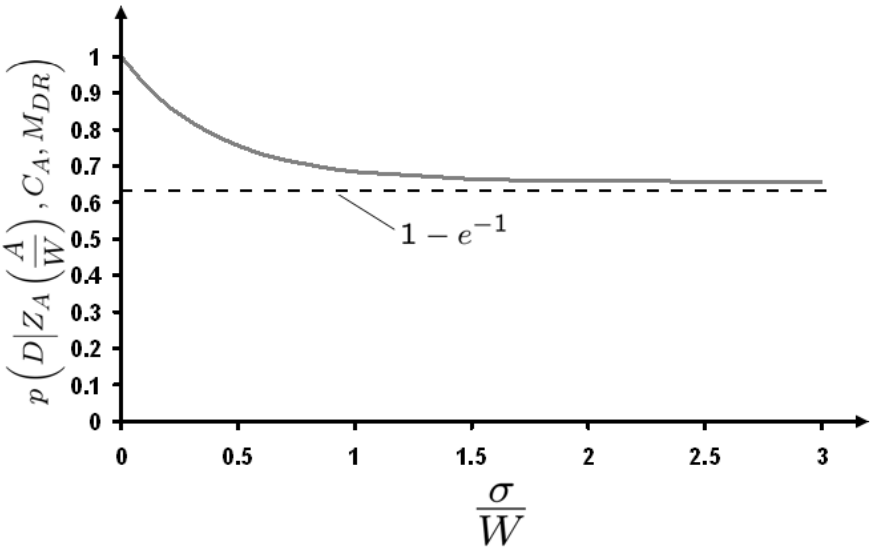


Figure 3.9: The dependence of detection probability on path placement accuracy.

The three characteristic detection functions are yardsticks by which the success of a search strategy may be measured, and understanding them is a key to valid search results. There are occasions where researchers claim to have better results than the definite range law. However, a closer look reveals that the yardstick was not held constant, such that the miraculous improvement is really a change in A or W. In addition, Koopman claims that in most cases, despite the diligent efforts of the searcher, the resulting detection function is actually closest to the random search law. This can be due to target motion, inaccurate maps, low-accuracy path planning and other factors. Thus, it is often best to plan careful paths that do not overlap, but to still assume the random search law when deciding the allocation of search effort. Optimal search theory is built upon this assumption.

The previous results in this section focus on understanding the probability of detection for some search behavior given that the target is contained in area A. Often, however, the target is only believed to be contained in area A according to some probability, $p(C_A)$. When this is the case, the vertical axis in Figure 3.8 changes from $p\big(D\big|C_A\big)$ to $p\big(D\big)$, and the maximum value on the axis becomes $p(C_A)$. In addition, it is useful for the search planner to know the probability that successful detection occurs for a given search behavior based on an estimate of the probability that the target is in the area. In other words, while

searching, how likely is it for the target to be detected in A *and* actually be in A? This can be estimated using

$$p\Big(D \cap C_A \Big| P_A(L), M\Big) = p(C_A) \times p\Big(D \Big| P_A(L), C_A, M\Big), \tag{3.16}$$

where $P_A(L)$ is some path in A of length L, M is some detection model, and $p(C_A)$ is an initial estimate of the probability that area A contains the target.

Figure 3.10 shows a Venn diagram, which is one way to gain insight about search in an area. One can see that the probability of detecting something, $p\big(D\big)$, can be larger than $p\big(D \cap C_A\big)$. This indicates that the possibility of false targets exists. Stone addresses this issue in [6], but this thesis focuses on the probability of successful detection, or $p\big(D \cap C_A\big)$. Thus, if a detection probability is stated, it is the probability of detecting a target and having it present, even though the probability of detecting something may be greater.

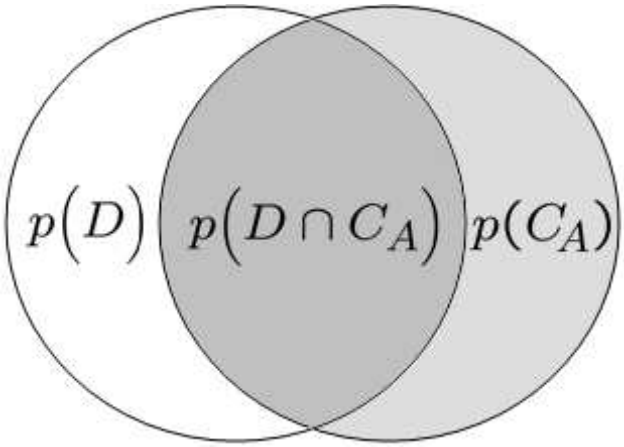


Figure 3.10: This Venn diagram illustrates the relationship between the probability of two events: target detection (D), and the target being contained in area A (C_A). For a search in area A, the detection probability $p(D)$ can be greater than $p(D \cap C_A)$. If detection occurs in A and the target is not within A, then it is considered to be detection of a *false target*. The probability $p(D \cap C_A) = p(C_A)p(D|C_A)$ represents the probability of successful detection.

3.4 Search Between Multiple Areas

Consider a target that may exist within one of two areas, A_1 or A_2. Each area has a different probability of containing the target, $p(C_1)$ and $p(C_2)$. The question is how to divide the total search effort, L, between the two areas in a way that maximizes the total detection probability. A decision of how much search effort to spend in A_1 will also determine how much search effort is spent in A_2, according to $L_2 = L - L_1$. The probability of detecting the target in one of the areas is

$$p\Big(D \cap C_@ \Big| P_{12}(L), M\Big) = p(C_1)p\Big(D\Big|P_1(L_1), C_1, M\Big) + p(C_2)p\Big(D\Big|P_2(L_2), C_2, M\Big),$$

where $C_@$ indicates that the target is contained within the area where detection occurs, and $P_{12}(L)$ indicates a path of length L in areas A_1 and A_2.

Koopman suggests that the detection function of even the most structured search effort is typically closest to the random search law. Using this assumption, the result from Eq.(3.13) is used to obtain

$$p\Big(D \cap C_@ \Big| P_{12}(L), M\Big) = p(C_1)\left(1 - e^{-\frac{WL_1}{A_1}}\right) + p(C_2)\left(1 - e^{-\frac{WL_2}{A_2}}\right). \tag{3.17}$$

Now let y be the exponential terms of Eq.(3.17) according to

$$y \;=\; y_1 + y_2 \;=\; p(C_1)e^{-\frac{WL_1}{A_1}} + p(C_2)e^{-\frac{W(L-L_1)}{A_2}}.$$

One approach to maximizing $p(D \cap C_@|P_{12}(L), M)$ is to minimize y, which can be accomplished by setting the derivative

$$\frac{dy}{dL_1} = -p(C_1)\frac{W}{A_1}\, e^{-\frac{WL_1}{A_1}} + p(C_2)\frac{W}{A_2}\, e^{-\frac{W(L-L_1)}{A_2}}$$

equal to zero and solving for L_1. The result is

$$L_1 = \frac{\frac{A_1 A_2}{W} \ln\left(\frac{p(C_1)A_2}{p(C_2)A_1}\right) + A_1 L}{A_1 + A_2}. \tag{3.18}$$

For a numerical example, consider the following conditions:

$$L = 5000\,m \quad W = 10\,m$$

$$A_1 = 10,000\,m^2 \quad A_2 = 20,000\,m^2$$

$$p(C_1) = 0.4 \quad p(C_2) = 0.1$$

Eq.(3.18) then yields a solution of $L_1 = 3053\,m$ as shown in Figure 3.11.

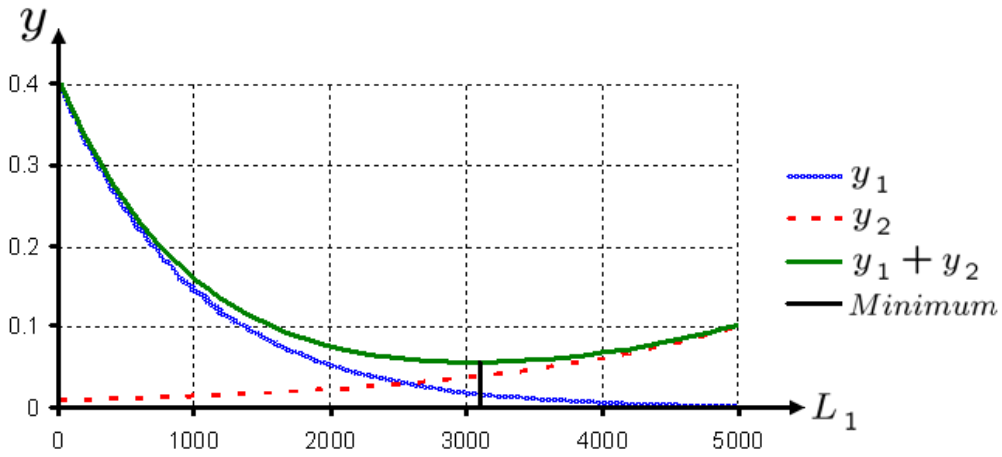


Figure 3.11: Minimizing the probability of search failure. Failure to detect the target in area A_1 is represented by y_1, and failure to detect the target in A_2 is represented by y_2. The target may only be found in one area or the other, so minimizing $y = y_1 + y_2$ minimizes the probability of not detecting the target.

Notice that the provided initial conditions put the solution for L_1 within the range $(0 < L_1 < L)$. Analyzing the derivation for Eq.(3.18) and the minimum point in Figure 3.11 reveals that the optimum allocation of search effort occurs when the slopes of y_1 and y_2 are equal and opposite. However, different conditions could lead to $L_1 < 0$ or $L_1 > L$. A solution

less than zero would indicate that all available search effort should be spent in A_2, while a solution greater than L would indicate that all available search effort should be spent in A_1.

To further explore the above example, consider the following quantities:

$$\begin{array}{cc} \rho_1 = \frac{p(C_1)}{A} & \rho_2 = \frac{p(C_2)}{A_2} \\ \phi_1 = \frac{WL_1}{A1} & \phi_2 = \frac{WL_2}{A_2} \\ A = A_1 + A_2 & \Phi = WL \end{array} \tag{3.19}$$

where

$$\rho_* = \text{the probability density in region *},$$

$$\phi_* = \text{the search effort density in region *},$$

$$A = \text{the total search area},$$

$$\Phi = \text{the total available search effort}.$$

Note that under the assumption of random search, ρ_n and ϕ_n are assumed to be uniform in region n. This does not necessarily mean that the actual path planning should take place in a random or distributed fashion. Observers should follow methodical paths in the region and attempt to avoid missed sections and overlap. However, for the purposes of search allocation *between* areas A_1 and A_2, uniformity in ρ_n and ϕ_n is assumed.

Search effort density ($\phi_n = WL_n/A_n$) is a term that compares the search effort in a region with the area of that region. It is analogous to mass density (kg/m^3) in that it is a ratio between two terms, one being a measurement of some quality and the other being a defined region in space. In this case search effort is defined in units of m^2 as WL and is compared to an area, so the search effort density is unit-less. However, a search effort density greater than unity does not mean that the region has been completely searched (except in the case of the definite range law). It is simply a way of quantifying how much searching has been done in comparison to the area.

Eq.(3.18) can now be re-written in terms of these quantities, as

$$\phi_1 = \ln \rho_1 - \frac{1}{A}(A_1 \ln \rho_1 + A_2 \ln \rho_2) + \frac{\Phi}{A} \tag{3.20}$$

and for ϕ_2 as

$$\phi_2 = \ln \rho_2 - \frac{1}{A}(A_1 \ln \rho_1 + A_2 \ln \rho_2) + \frac{\Phi}{A}. \tag{3.21}$$

If ϕ_1 is less than or equal to zero, then all search effort should be allocated to A_2. This corresponds to the condition

$$\rho_1 \leq \rho_2 e^{-\frac{\Phi}{A}}. \tag{3.22}$$

If ϕ_2 is less than or equal to zero, then all search effort should be allocated to A_1. This corresponds to the condition

$$\rho_2 \leq \rho_1 e^{-\frac{\Phi}{A}}. \tag{3.23}$$

So far the optimum search allocation has been discussed in terms of quantity only. However, Eqs.(3.22) and (3.23) also lead to an understanding of how the optimal search should proceed in time. As Φ begins from zero and progressively increases, the region with the highest probability density is searched first, and either Eq.(3.22) or (3.23) is valid. After a time, the search will reach a point where

$$\rho_1 > \rho_2 e^{-\frac{\Phi}{A}} \quad \text{and} \quad \rho_2 > \rho_1 e^{-\frac{\Phi}{A}}. \tag{3.24}$$

This marks a specific turning point where the higher probability density has been reduced by search effort such that it is equal to the lower probability density. Thereafter both areas are searched according to the ratio dictated by Eqs.(3.20) and (3.21). If search continues for a significant time following this point, then the largest region receives the greatest amount of search effort. The switch from searching the region with the largest ρ to searching the region with the largest A is known as *reversal for large* Φ. This property can also be verified by evaluating Eqs.(3.20) and (3.21) in the limit as Φ goes to infinity:

$$\lim_{\Phi \to \infty} \frac{\phi_1}{\phi_2} = 1. \tag{3.25}$$

Another interesting phenomenon can be seen by re-arranging Eqs.(3.20) and (3.21) according to

$$\phi_1 = \ln k\rho_1 \,, \quad \phi_2 = \ln k\rho_2 \tag{3.26}$$

where

$$k = \exp\left(-\frac{1}{A}(A_1 \ln \rho_1 + A_2 \ln \rho_2) + \frac{\Phi}{A}\right).$$

This means that a region's optimum search effort density is equal to the natural logarithm of a quantity that is proportional to the probability density of that region.

In order to present a simple and directly-applicable solution with many areas, the notation introduced in Eq.(3.19) will now be dropped. However, it is still useful to refer back to these definitions in the following example, just as it was in the two-area example.

J.W. Gibbs offers a helpful principle in extending the optimum solution to n areas [15] [2, p.152]:

> *If $f_1(x_1), \ldots, f_n(x_n)$ are n continuously differentiable functions of $x_1, \ldots, x_n$ for non-negative values, and if their sum $y = f_1(x_1) + \cdots + f_n(x_n)$, considered for all non-negative values of x_i adding up to a given constant c, has a maximum at $x_i = x_i^*$, then there exists a constant λ having the following "discriminating" property for the derivatives:*

$$\text{If } x_1^* > 0 \text{ then } f_i'(x_i^*) = \lambda\,; \quad \text{if } x_i^* = 0 \text{ then } f_i'(x_i^*) \leq \lambda \tag{3.27}$$

Many know the constant λ as the Lagrange multiplier, which is a function of the given constant c. Using $P_{1n}(L)$ to denote some path in regions 1 through n of total length L, Gibbs' theorem is applied to the problem of optimal search as follows:
If

$$p\Big(D \cap C_@ \Big| P_{1n}(L), M\Big) = p(C_1)\left(1 - e^{-\frac{WL_1}{A_1}}\right) + \cdots + p(C_n)\left(1 - e^{-\frac{WL_n}{A_n}}\right) \tag{3.28}$$

and

$$L_1 + L_2 + \cdots + L_n = L$$

then

$$L_i > 0 \quad \text{implies that} \quad p(C_i)\frac{W}{A_i}e^{-\frac{WL_i}{A_i}} = \lambda \quad \text{and} \quad \frac{p(C_i)W}{A_i} > \lambda \tag{3.29}$$

and

$$L_i = 0 \quad \text{implies that} \quad \frac{p(C_i)W}{A_i} \leq \lambda.$$

Now let all of the considered search regions be ordered such that

$$\frac{p(C_1)W}{A_1} \geq \frac{p(C_2)W}{A_2} \geq \cdots \geq \frac{p(C_n)W}{A_n}. \tag{3.30}$$

Also consider the set $A_L = \{A_1, \ldots, A_m\}$; $m \leq n$, containing the regions that are currently being searched ($L_i > 0$). The value of L_i for any member of this set may be calculated as a function of λ from Eq.(3.29) as

$$L_i \;=\; \frac{A_i}{W} \ln\left(\frac{p(C_i)W}{A_i \lambda}\right) \;=\; \frac{A_i}{W}\left(\ln \frac{p(C_i)W}{A_i} - \ln \lambda\right). \tag{3.31}$$

At the beginning of a search, the optimum allocation will first be found by considering some small $L > 0$. This starting search effort will be completely applied in $A_L = \{A_1\}$ such that $L = L_1$. Due to the fact that A_1 is the only participating region,

$$\lambda = p(C_1)\frac{W}{A_1} e^{-\frac{WL_1}{A_1}}. \tag{3.32}$$

This will continue with increasing L until

$$\lambda = p(C_1)\frac{W}{A_1} e^{-\frac{WL_1}{A_1}} = p(C_2)\frac{W}{A_2} e^{-\frac{W(0)}{A_2}}. \tag{3.33}$$

Thereafter the search effort will be spent in $A_L = \{A_1,\, A_2\}$ such that $L = L_1 + L_2$, and the probability densities of these areas will be reduced at a mutual rate until

$$\lambda = p(C_1)\frac{W}{A_1} e^{-\frac{WL_1}{A_1}} = p(C_2)\frac{W}{A_2} e^{-\frac{WL_2}{A_2}} = p(C_3)\frac{W}{A_3}. \tag{3.34}$$

And so on.

Thus, A_L is known for all L, because the $L = L(i)$ at which each A_i starts participating in the search can be calculated as

$$L(i) = \frac{A_1}{W}\left(\ln\frac{p(C_1)W}{A_1} - \ln\frac{p(C_i)W}{A_i}\right) + \cdots + \frac{A_{i-1}}{W}\left(\ln\frac{p(C_{i-1})W}{A_{i-1}} - \ln\frac{p(C_i)W}{A_i}\right). \quad (3.35)$$

Between these points, the L_i applied to each area can be found using Eq.(3.31) and $L = \sum_{i=1}^{m} L_i$ to obtain

$$\ln\lambda = \frac{A_1 \ln\frac{p(C_1)W}{A_1} + \cdots + A_m \ln\frac{p(C_m)W}{A_m} - WL}{A_1 + \cdots + A_m}, \quad (3.36)$$

where m is the number of regions currently being searched. The resulting value of λ can then be used to determine the optimal effort allocation at each point in the search as per Eqs.(3.32),(3.33) and (3.34).

Often when dabbling in writings about optimal search, the search effort, Φ, can be a representation of time, search length, covered area, etc. Figure 3.12 shows an optimal detection function based on any type of search effort. Koopman asserts that the expected effort spent up to target detection is the area of region E [2, p.179]. This means that if Φ is a measurement of time, then integration of region E reveals the mean time to detection (assuming $p(D|\Phi)$ eventually reaches unity).

It is useful to note that the multiple-area example also extends to the continuous case, where the probability of each dA is assigned according to its location by some continuous 2D function. However, evaluation of the discrete case invokes the needed familiarity with optimal search concepts for this thesis, and the reader may refer to the published works of Koopman [2] and Stone [6] for more detail.

Stone offers significant insight into the progression of an optimal search. He describes a situation where the target's probable location is a Gaussian distribution about the origin. For an optimal search in this region, the search effort will begin at the origin, and "the

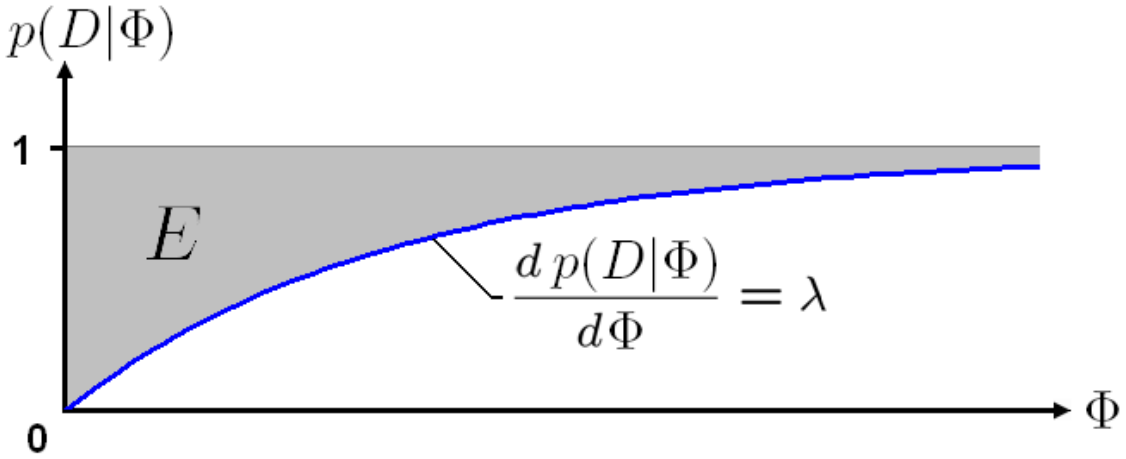


Figure 3.12: An optimal detection function as a function of search effort Φ. Integration of region E indicates the expected amount of search effort before detection occurs.

posterior density of the target distribution is constant inside [a circle] centered at the origin. ... As time increases and the target has not been detected, the disc over which the posterior density is constant expands and the value of the density in that disc decreases. The result is that the posterior target distribution is flattening and spreading as the target fails to be detected." [6, p.53]

Thus, for an optimal search of any shape, all regions having the maximum probability density will be searched uniformly until they have been reduced in density to that of the immediately-surrounding regions. Then the maximum probability density is redefined to be this new, lower density, which applies to a larger area. This continues in the same widening and flattening manner indefinitely. It is interesting to note that "the longer [an observer] searches in this situation and fails to find the target, the worse one's prospects become, in the sense that the mean time remaining to [target detection] is increasing." [6, p.54]

3.5 Search in the Presence of False Targets

Stone's major contribution to search theory is his work on optimal search in the presence of false targets. His proofs are not duplicated in this thesis, but some important definitions, principles and results are helpful in search planning.

A *false target* is any evidence that might indicate the presence of a target when a real target is not present. A *target contact* is any evidence that could indicate the presence of a target. Based on these terms, Stone indicates specific meanings for the words *detected*, *identified*, and *found*:

- *Detected* - A target contact appears.
- *Identified* - Contact investigation confirms a real target.
- *Found* - A target has been both detected and identified.

When working with false targets, search strategies are classified as nonadaptive, semiadaptive, or adaptive plans. A *nonadaptive* plan is one that proceeds according to the optimal search strategy as outlined in Section 3.4, and does not change based on the discovery of real or false targets. A *semiadaptive* plan is one that has a specified sequence of update times where the strategy may be updated. This plan approximates an *adaptive* plan, which continuously considers all information gained during the search to revise the target distribution and the false-target distribution in a Bayesian manner [6, p.167].

Stone proves that the mean time to target detection for the adaptive, semiadaptive and nonadaptive plans is ordered as follows:

$$\mu_a \leq \mu_s \leq \mu_n. \tag{3.37}$$

"Thus, if one has a choice, he would prefer to use the optimal adaptive plan for searches involving false targets." Unfortunately, "optimal adaptive plans have been found only for the situation of a search with two cells and one false target known to be in the second cell." [6, p.166] However, due to the order established in Eq.(3.37), it is desirable to approximate the adaptive plan if possible. For the reader interested in a cohesive introduction to semiadaptive plans, it may be best to start reading on p.165 of [6], and dissect the next 13 pages until the end of the chapter. Stone gives detailed theory, an instructive example, and concludes by stating the effectiveness of adaptation.

In the simple two-cell case for which an adaptive plan is known, the difference in mean detection time between the adaptive and nonadaptive plans is less than 1%. However, as complexity increases, a semiadaptive plan with continuous updating can produce as much as 30% smaller mean times to detection than the optimal nonadaptive plan. A semiadaptive plan seems to have the greatest advantage over a nonadaptive plan when the target distribution is uniform with a few false targets in each cell and a large contact investigation time [6, p.178]. Adaptation may also be significant in the presence of flawed assumptions, where Bayesian updates could help to correct the probability distribution.

One of the most important points to consider from Stone's work is the proof that *immediate and conclusive contact investigation is optimal in the presence of false targets for nonadaptive plans* [6, p.154]. In other words, if search effort can be applied anywhere and at any time, then once a target has been detected it is best to immediately take the time to positively or negatively identify the target. This result is based on the assumption that it takes a fixed time to identify target contacts. Stone suggests that this course of action will minimize the mean time to *find* the target, and claims that "this result agrees with the intuitive feeling that there is no point to broad searching in an area if contacts are not investigated." [6, p.151] Although the proof exclusively applies to nonadaptive plans, "only immediate and conclusive contact investigation [is] considered" in Stone's semiadaptive plans as well [6, p.166].

This principle has implications in many real path-planning tasks. Unlike optimal search theory, path planning does not have the luxury of assuming that search effort may be expended anywhere. Thus, if immediate and conclusive contact investigation is the best policy in optimal search theory, then it is even better in a real search due to the observer's temporary proximity to a new target contact. Of course, this assumes that target detection and identification are done by the same agent.

The search theory presented in this chapter may be used for comparison in path planning, and offers many guiding principles as to how to conduct an effective search. Much

of the material presented by Koopman and Stone is not included here, and could also apply to a particular search scenario. Thus, the reader is also encouraged to read [2] and [6].

Chapter 4

Search Planning

> "*In the relatively unusual cases in which the immobility of the target and the accuracy of navigation—as well as the exactness of the knowledge concerning the factors affecting the probability of its detection and the probabilities [$p(C_i)$]—warrant using a law other than the exponential one above, explicit solutions of the result of applying the Lagrange multiplier method cannot in general be obtained, and modern computers must usually be used.*" — B.O. Koopman [2, p.154]

With a basic understanding in optimal search theory, the task at hand is to create a useful search technique. The above quotation might cause one to think that assuming anything other than the random search law is walking on thin ice. However, the advent of a global positioning system and UAV technology may represent a valid objection to this philosophy. Then again, Koopman may still be largely right. Thus, the following methods are presented with the hope that if target mobility, navigation accuracy, detection probability and location probability are not sufficiently known at present, then technological advances will soon help them to become so.

Chapters 4 and 5 represent original contributions of the author using optimal search theory. The underlying principles are largely the same, but the contributions are specific to the detection method of a UAV with a CCD camera. The given technique that is used to measure search rate with a probability grid is a key to applying this theory.

4.1 Target Detection Probability

The first task in search planning is to identify a detection model. Many researchers simply assume that the definite range law is accurate enough for path planning. However, in the minds of some this does not provide a reasonable approximation for an authentic search.

4.1.1 CCD Camera Model

In this section a target's detection probability is based on the connected-component analysis described in Section 2.4. This uses HSV color segmentation to identify the pixels that are similar in color to the target. The goal is to obtain an estimate of the target detection probability based on its distance. However, an accurate estimate of this probability requires the following steps:

1. Derive a target's image size based on its distance and cross-sectional area.
2. Find representative color distributions for the target and its background.
3. Model the transition from target color to background color in an image.
4. Decide the minimum target size (in pixels) that the algorithm will acknowledge.
5. Using the color distributions from step 2 and the color transition from step 3, generate many images that display a target of known image size. Then attempt to identify the target in these images using a connected-component analysis and the minimum target size from step 4. The success ratio obtained from a large number of images should closely represent the detection probability for a target of that size.
6. Combine data from 1 and 5 to determine a target's detection probability with distance.

Consider an unobstructed spherical object that is observed through a charge-coupled device (CCD) camera as shown in Figure 4.1. By similar triangles $\frac{r_{objp}}{f_p} = \frac{r_{obj}}{d}$, and the

number of target pixels on-screen (occupied by an object of radius r at d units away) is

$$P_s = \pi r_{objp}^2 = \left(\frac{f_p}{d}\right)^2 \pi r_{obj}^2.$$

It can also be represented as

$$P_s = \left(\frac{f_p}{d}\right)^2 A_{obj} \tag{4.1}$$

where

$$f_p = \text{focal length in average pixels} \left(\frac{f_x + f_y}{2}\right)$$

$$f_x = \text{focal length in pixel widths (See Section 2.1.)}$$

$$f_y = \text{focal length in pixel heights (See Section 2.1.)}$$

$$d = \text{object distance}$$

$$A_{obj} = \text{cross-sectional area of the object.}$$

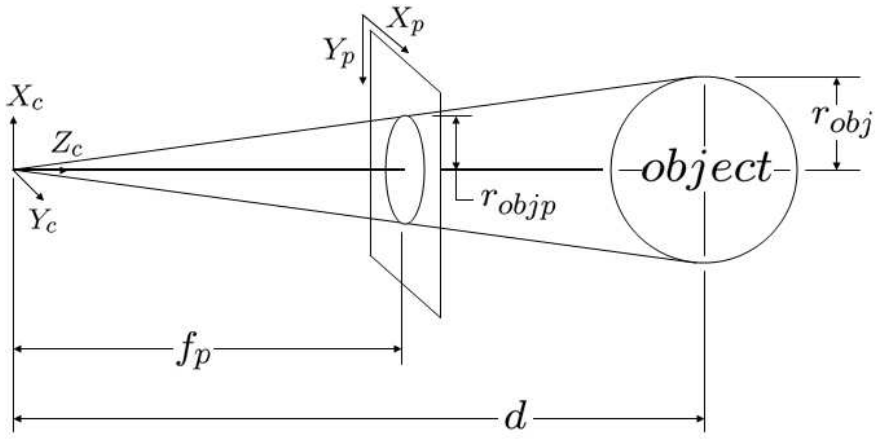


Figure 4.1: A spherical object viewed by a CCD camera.

Using Eq.(4.1), one can determine the expected number of target pixels in an image based on the target's size and distance. Notice that this formula is very similar to the solid

angle used in Koopman's inverse cube law, except that it does not depend on an assumed orientation, and the output is in pixels. (See Eq.(3.2) and Eq.(3.5).) Also note that P_s is not necessarily an integer value but represents the precise size of the target projected onto the image plane in units of square pixels.

Before discussing the color distributions it is important to understand that color is one of the most noticeable characteristics of a passive target. For example, BYU's recent search and rescue exercise (SAREX) was based on a lost victim who had been seen carrying an orange jacket and wearing a red or white t-shirt and blue jeans. This color information was the *only* information used when establishing initial target contact. Comments like "I thought I saw something white" or "There's an orange spot!" were typical observations.

Although the BYU SAREX was based on human detection, color is also a key factor in computer-aided search. Figure 4.2 is a zoom-in on one of the targets from Figure 2.18.

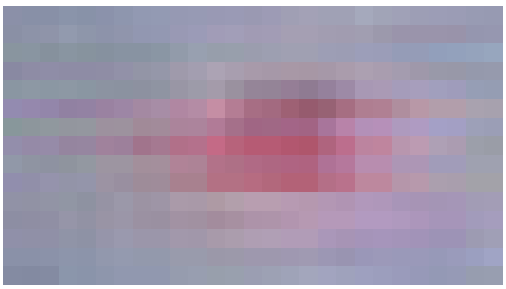


Figure 4.2: A close-up view of a red target as seen by a CCD camera (red darkened for grayscale contrast).

The color variation in this image can be surprising, because the target is just a flat piece of red cloth on a road. However, a CCD image can vary for several reasons, including:

1. Environmental lighting changes
2. Object position and orientation differences
3. Lens distortion

4. Shutter speed variation

5. The Bayer filter

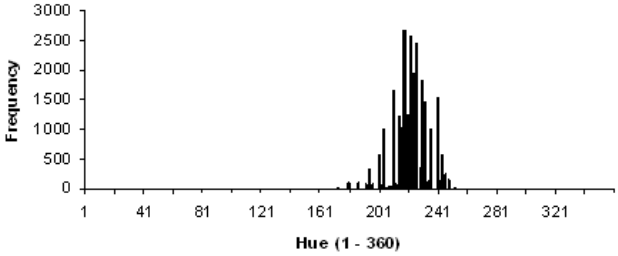
6. Interlacing

7. Signal noise.

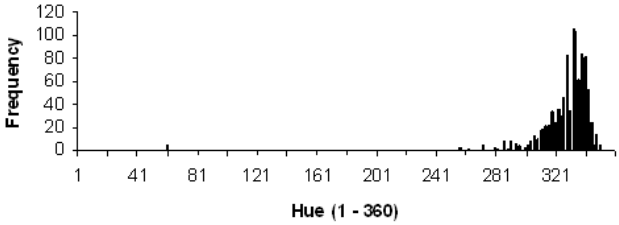
These sources of color variation generate a unique color distribution in each situation that can be difficult to predict. Figure 4.3 shows the measured HSV distributions of the gray road and red targets present during the flight test in Chapter 2. As shown in Figure 2.18, six cloths of identical color were laid on the ground, and were viewed by the UAV from around 63 meters in altitude. The lighting conditions were those generated by an outdoor flight on an overcast day, and there was no noticeable variation in ambient lighting during the experiment. The color data was obtained by selecting sections in many images from the Target Differentiation field test video, and accruing the HSV information in histogram form.

When compared to some other scenarios, searching for a red piece of cloth on a road may seem to be very simplistic. However, the color distributions of Figure 4.3 still embody many complex factors. For instance, the bi-modal nature of the road value distribution was seen during the field test as a series of dark diagonal lines that (according to testing in the MAGICC Lab) are likely to have been caused by signal noise from the electronic speed control. In addition, the left tail of the target hue distribution probably represents impure sampling, where some pixels already appear partially blended between the target and road colors. An unknown effect also caused the frequencies recorded in some bins to be zero while the frequencies in neighboring bins were very significant. The complexity of these and other factors often causes each experiment to have unique color distributions.

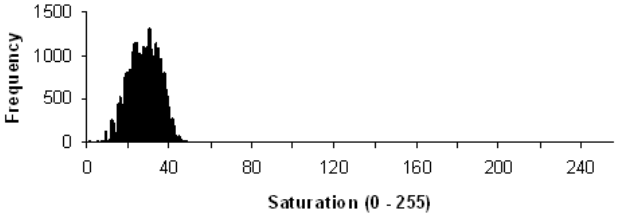
The goal is to use the color data from a given target and background to model the probability of detecting the target. An example of this is described later in this section, and the probability is found by generating many target images based on assumed color distributions. One approach to modeling these color distributions might be to pick randomly



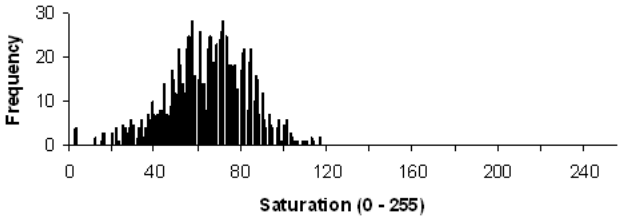
(a) Road hue.



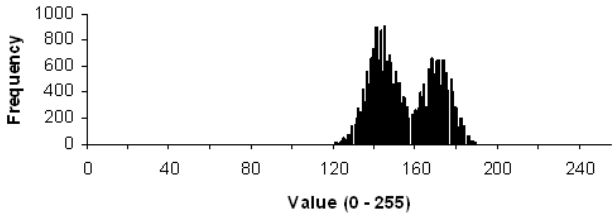
(b) Target hue.



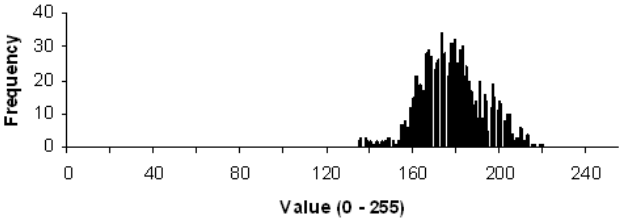
(c) Road saturation.



(d) Target saturation.



(e) Road value.



(f) Target value.

Figure 4.3: These are the HSV color distributions for the road and target. They are obtained by using an application that allows the user to repetitively select different areas of pixels, and put the pixel data into a collective histogram. Sections of road were sampled in several images to get an accurate distribution for road color, and small sections were sampled at the center of each target (many times, and in many images) to get an accurate distribution for target color. The bi-modal nature of the road value distribution was seen during the field test as a series of dark diagonal lines that are likely to have been caused by signal noise from the electronic speed control. (See Figure 2.18.)

from the exact set of colors that was generated by sampling target/background images. Another approach is to calculate the mean and standard deviation for each distribution (See Table 4.1.), and generate target images based on Gaussian distributions with these parameters.

Although the Gaussian assumption may not fit all of the data exactly (See Table 4.2.), some of the distributions are primarily Gaussian, and others could be Gaussian in the absence

Table 4.1: Mean and standard deviation for the HSV distributions.

	Road Color		Target Color	
	Mean	St.Dev	Mean	St.Dev.
Hue	222.37	11.83	327.09	21.86
Saturation	27.63	7.51	65.37	18.60
Value	154.80	15.34	178.19	14.62
Pixels Sampled	24,749		985	

of the un-intended experimental conditions described. One might expect the uniform target color and lighting conditions to generate a true color distribution characterized by a single hue, saturation, and value. Thus, the bell-shaped variation experienced in the actual measurements is assumed to be random error that follows a Gaussian distribution. Accordingly, the discrepancies in skew and kurtosis shown in Table 4.2 are rejected, and the Gaussian model is used.

Table 4.2: Skew and kurtosis for the HSV distributions.

	Skew	Kurtosis
Road Hue	-0.459	4.025
Road Saturation	-0.136	2.568
Road Value	0.180	1.786
Target Hue	-0.366	1.044
Target Saturation	-0.324	3.048
Target Value	0.051	3.018
Normal Distribution	0.000	3.000

Now that color distributions have been characterized for the target and its background, the region of blurred color at the target boundary must be understood. It is helpful to know that light undergoes a large number of effects as it enters the camera, including changes from defocusing, diffraction, photosensitive variation, etc. The imperfect pixel pattern that results from a viewed point of light can be modeled as a Gaussian distribution called the *point spread function* (PSF) [16].

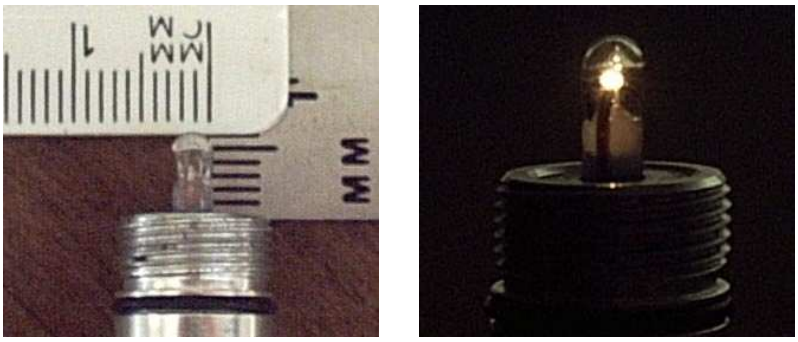
Due to distortion complexity, the PSF parameters are often estimated empirically. One way to do this is to view a distant point source of light against a black background [16]. However, when measuring the PSF for the test camera (The camera is shown in Figure 2.8(b).), there are two considerations that will help in obtaining reasonable results: First, a light source should be chosen and placed far away, such that its projected area is significantly less than one pixel. Second, the shutter speed should be similar to what it would be during an outdoor search, even though the PSF is measured in a dark room.

For comparison, the projected area of one pixel at a distance of one meter is

$$A_p = S_x S_y = \frac{1}{503.652}\ \frac{1}{500.110} = 3.97 \times 10^{-6}\ \text{m}^2.$$

In addition, the light source selected for this experiment is the diode inside a Mag-Lite® Solitaire® flashlight, as shown in Figure 4.4. When the diode is placed a known distance away (limited to 7.9 m by the room), Figure 4.4(a) can be used to calculate the diode's projected area at one meter away:

$$A_{diode} = \frac{.003}{7.9}\ \frac{.006}{7.9} = 2.88 \times 10^{-7}\ \text{m}^2.$$



(a) Size of the Mag-Lite® diode. (b) The Mag-Lite® diode lit.

Figure 4.4: The Mag-Lite® diode used to determine the camera's point spread function.

Thus, if the light source is .003 m × .006 m at a distance of 7.9 m, then the diode's projected area is .0726 square pixels. This is considered sufficiently small to give a reasonable PSF estimate, because (as will be seen) the point spread function of this camera has a standard deviation on the order of one pixel.

A feature of the test camera that must be dealt with is that it has automatic shutter speed control based on the ambient lighting. As a result, the test must begin with full lighting. The lights are then turned off and the PSF is measured in the next frame, before the shutter speed has time to adjust. A view of the diode using this method is shown in Figure 4.5.

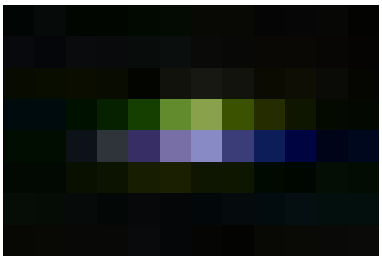


Figure 4.5: A view of the diode through the test camera (7.9 m away).

This image can be analyzed to determine the standard deviation of the PSF by using each pixel's color value (also called intensity) to indicate the degree of illumination. Figure 4.6 graphically shows the value of pixels in the vicinity of the light source.

One interesting characteristic of the test camera is that its PSF has a greater standard deviation in the X_p direction (2.052 pixels) than in the Y_p direction (.688 pixels), independent of the diode orientation. This reveals the true shape of the point spread function for this camera and lens. However, for simplicity an average standard deviation of $\sigma_{PSF} = 1.37$ pixels is used in the simulations that follow.

Notice that the point spread function describes the effect of light as it enters the camera, but can also be used to describe the sources of light that illuminate each pixel.

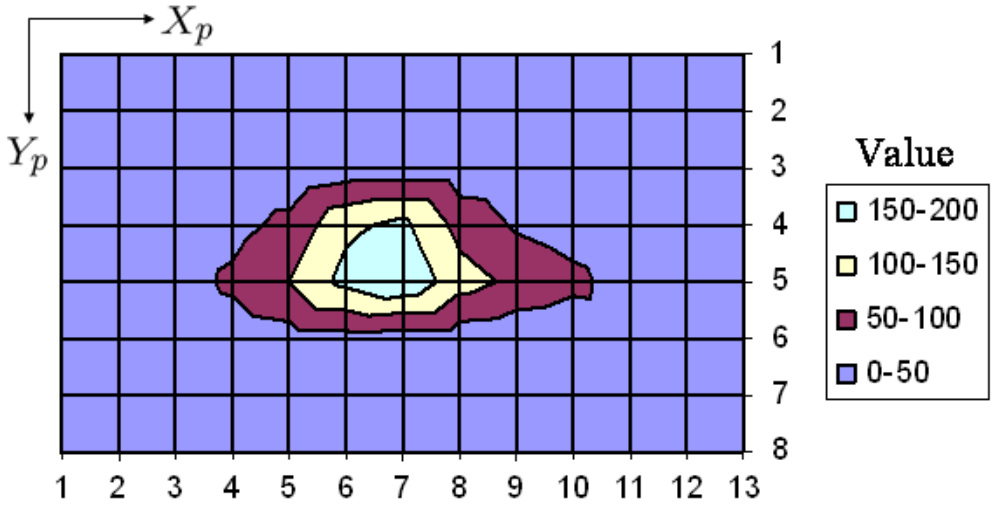


Figure 4.6: The camera's point spread distribution.

Thus, if a PSF is assumed to be centered on each pixel, then a pixel's color depends on what projected regions the PSF covers.

To illustrate this principle, consider the pixel shown in Figure 4.7. Let $\chi_R = [h_R, s_R, v_R]^T$ represent the HSV road color and $\chi_T = [h_T, s_T, v_T]^T$ represent the target color. These colors are independent random samples in H,S and V from the target and road distributions described in Figure 4.3 and Table 4.1. The color for any pixel in the image can now be defined as

$$\chi_P = \chi_R(1 - \alpha_T) + \chi_T\alpha_T \tag{4.2}$$

where α_T is the portion of the PSF that overlaps the target projection. Notice that although the PSF is a Gaussian distribution, it represents the color blending that occurs and is not a probability distribution. Also note that Eq.(4.2) can be used in this case because of the sequential nature of the road and target hue. However, for hue distributions that span 360° (say $\mu_1 = 10°$ and $\mu_2 = 350°$), a method must be used that appropriately transitions between these colors.

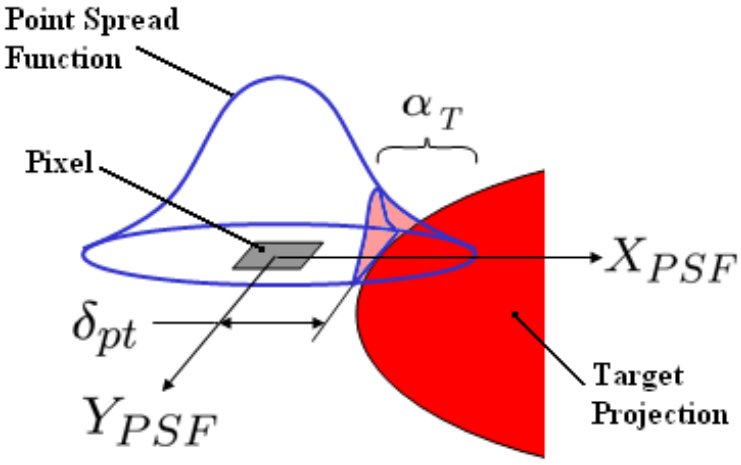


Figure 4.7: A pixel's color depends on how much of its point spread function overlaps the target projection.

For the pixel shown in Figure 4.7, an approximation for α_T can be obtained by integrating under the Gaussian curve:

$$\alpha_T = \int_{\delta_{pt}}^{\infty}\int_{-\infty}^{\infty} \frac{1}{2\pi\sigma_{PSF}^2} \exp\left(-\frac{x^2+y^2}{2\sigma_{PSF}^2}\right) dydx \quad = \quad \frac{1}{2}\left(1 - \text{erf}\left(\frac{\delta_{pt}}{\sigma_{PSF}\sqrt{2}}\right)\right). \tag{4.3}$$

Eq.(4.2) can now be used to estimate the color of any image pixel based on its distance from the target boundary. For a circular target projection with an area of P_s square pixels and a target radius of $r_t = \sqrt{P_s/\pi}$, δ_{pt} can be defined as $r_p - r_t$, where r_p is a given pixel's radial distance from the target center in pixels.

By defining each pixel in this way, one can generate simulated images of the target. However, to analyze the detection probability some pixel threshold, P_0, must be used to identify the target. Thus, a collection of qualified (connected and within color limits) pixels less than size P_0 is not considered a target, but is ignored. It is hoped that image noise and color distributions from other objects will rarely generate P_0 qualified pixels, and that an actual target will often generate P_0 qualified pixels. While searching for red targets using the given CCD camera and wireless transmission system, a practical value for P_0 is 20 pixels.

This is just large enough for the algorithm to distinguish between the intended targets and image noise or background objects in most cases.

Using the P_0 size threshold and the derived pixel colors, many images can be created and analyzed using connected-component color segmentation. Figure 4.8 shows the detection probability for various target sizes and color thresholds, and Figure 4.9 shows a simulated target image.

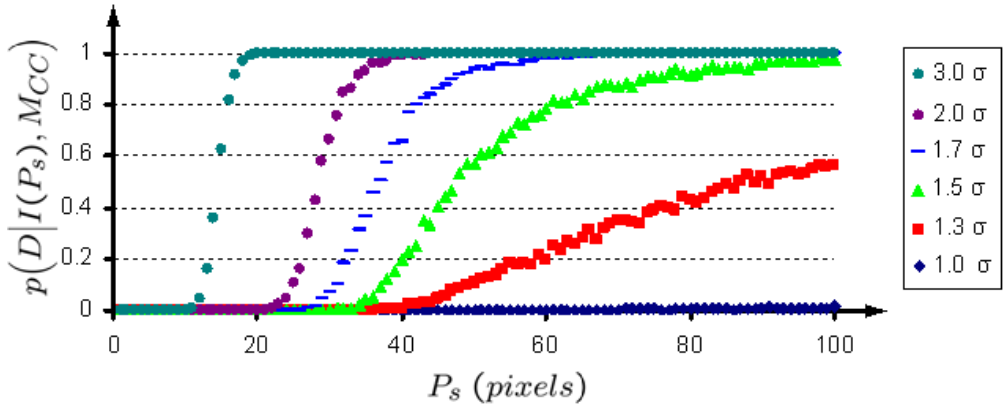


Figure 4.8: Target detection probability is shown as a function of target size (pixels). Each point represents the success ratio for 1000 trials. Images were analyzed using connected-component color segmentation on a target of size P_s. The color thresholds used in this analysis are centered on the mean, and spaced according to the series label. For example, using data from Table 4.1, the series labeled '1.0 σ' identifies target pixels within a hue of 327.09±21.86, a saturation of 65.37±18.60 and a value of 178.19±14.62.

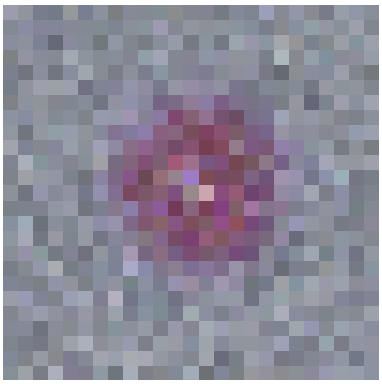


Figure 4.9: A simulated target image (red darkened for grayscale contrast).

The points in Figure 4.8 represent the *connected component detection model*, which can be closely approximated as

$$p\Big(D\Big|I(P_s), M_{CC}\Big) = \max\left(\frac{1-\exp\left(-\dfrac{P_s-s_1}{a_1}\right)}{1+\exp\left(-\dfrac{P_s-s_2}{a_2}\right)}, 0\right), \tag{4.4}$$

where s_1, s_2, a_1 and a_2 are the constants shown in Table 4.3, $I(P_s)$ denotes an instantaneous look (one processed image) at a target with an image projection size of P_s pixels, and M_{CC} denotes the connected component detection model. Notice that the detection probability is a direct multiplication between a sigmoid and an exponential curve. Figure 4.10 shows the simulation results and the approximations together.

Table 4.3: Target detection model parameters.

	1.3 σ	1.5 σ	1.7 σ	2.0 σ	3.0 σ
s_1	38	32	26	19.8	6.5
s_2	46	40	34	27.8	14.5
a_1	72.5	19	9	5	2.5
a_2	8	3.8	2.8	1.6	0.96

If the detection probability is dependent on the color thresholds, then what are the color thresholds likely to be? At present the appropriate thresholds are estimated visually after viewing a number of color histograms from the background and the target. These thresholds depend on how close the background color is to the target color, and how accurately these colors have been sampled or estimated. Often the color thresholds are only vague estimates based on a statement like "a bright orange jacket", and there is no way to know the true color of the target. However, threshold estimates can be significantly improved before the search by sampling the color of something similar to a target in the lighting that will be present during the search.

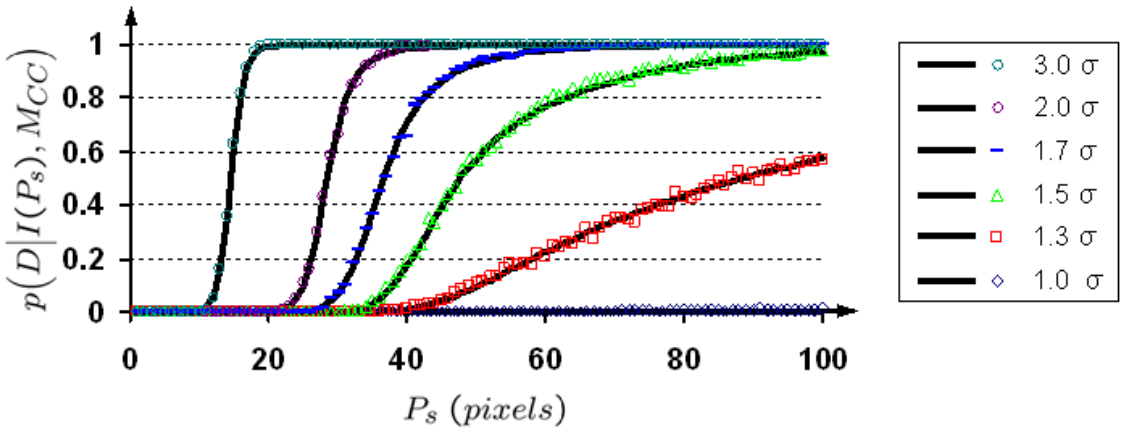


Figure 4.10: Detection probability results, along with their approximations. The data points here are identical to those in Figure 4.8. However, a black approximation curve is also shown for each case. The approximation curve is generated using values from Table 4.3 in Eq.(4.4).

Due to these uncertainties, color thresholds typically do not extend as far as 3 σ from the mean, nor are they precisely centered on the mean. Accurately approximating this human-based threshold would require an extensive experiment where many users would set the color thresholds in a variety of conditions. Such an experiment is outside the scope of this thesis. However, during a flight test it was observed that it becomes nearly impossible to detect these targets with this camera and algorithm if the UAV is further than 100 m away. As will be shown, this could indicate that a detection probability closer to the 1.7 σ curve is somewhat representative for visual threshold setting. Undoubtedly the accuracy of the detection model could be improved by more sophisticated methods, but this offers a starting point for now. Certainly this assumption is much less of a stretch than assuming the definite range law! It is based on real experience with a common sensor.

Based on the 1.7 σ assumption, the detection probability becomes

$$p\left(D\middle|I(P_s), M_{CC}\right) = \max\left(\frac{1-\exp\left(-\frac{P_s-26}{9}\right)}{1+\exp\left(-\frac{P_s-34}{2.8}\right)}, 0\right), \tag{4.5}$$

Substituting the result from Eq.(4.1) into Eq.(4.5) yields an expression for the instantaneous detection probability as a function of the observer's distance from the object:

$$p\Big(D\Big|I(d), M_{CC}\Big) = \max\left(\frac{1-\exp\left(-\frac{\left(\frac{f_p}{d}\right)^2 A_{obj}-26}{9}\right)}{1+\exp\left(-\frac{\left(\frac{f_p}{d}\right)^2 A_{obj}-34}{2.8}\right)}, 0\right). \tag{4.6}$$

In addition, if the object of interest has a typical cross-sectional area of $A_{obj} = 1$ m^2, and (f_x, f_y) are taken from Table 2.1, then the instantaneous detection probability is as shown in Figure 4.11.

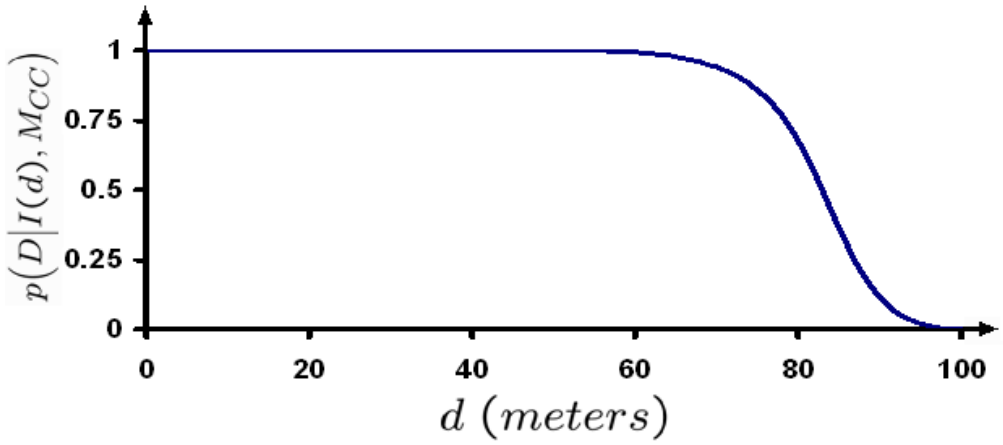


Figure 4.11: Instantaneous detection probability as a function of distance.

This predicts that if the UAV has a distance greater than 98 m, then no targets will be detected. Also, being much closer than about 60 meters to a target does not significantly increase the probability of target detection. This means that the extent to which a region is searched can now be measured and the effectiveness of path planning strategies can be tested for CCD camera agents. Of course, the accuracy of the model still depends on whether or not the conditions and assumptions used to derive it are valid in a given environment. The

target color must be different enough from the environment color such that thresholds can be set that achieve the performance modeled by the 1.7 σ curve.

One might ask how motion affects this detection probability. The answer is that the detection probability can be affected by motion if the detection algorithm is analyzing interlaced video. Otherwise the shutter speed is typically fast enough that any foreseeable UAV motion does not cause detectable blurring.

Interlacing is common in CCD cameras, and is a practice that takes every other pixel line from one shutter opening and combines that information with the rest of the pixel lines from the next shutter opening. This common process originated when data transmission was more costly, and interlacing was an attempt to alleviate some of that cost.

However, even detection using interlaced video is typically not disrupted unless unless there is significant motion. The test camera streams video at a rate of 30 frames per second, with 1/60 s between shutter openings. If image motion is enough to move the target more than one target width in 1/60 s, then target pixels in every other line will not be connected. One way to correct for this problem is to only run the algorithm on half of the image, and interpolate the color values of the pixels that are missing. If this is done, then the detection probability should remain as derived.

The detection probability developed in this thesis depends on well-lit conditions. In the dark, the shutter speed is slower and color blurring from motion is more prevalent. In addition, hue measurements are much less accurate without adequate light.

4.1.2 Image Skew Considerations

Unfortunately, typical image skew causes an object's image size to be less than it would be if no image skew were present. This means that if an object's expected size in an image is P_s, based on the object's distance and size alone, then its expected image size will be $P_{\check{s}}$ after image skew.

Consider some small rectangular region, p, whose projected area onto a plane one meter away is $A_p = S_x S_y$. (See Figure 4.12.) Due to image skew, the information that would be viewed through region p is actually viewed through the smaller region $\breve{p}$, with area $A_{\breve{p}} = S_{\breve{x}} S_{\breve{y}}$. The position shift for any point in the image is

$$x_{\breve{u}_c} = x_{u_c} k_s \qquad y_{\breve{u}_c} = y_{u_c} k_s, \tag{4.7}$$

where k_s is the image skew factor whose value depends on radial distance from the image center. (See Section 2.2.)

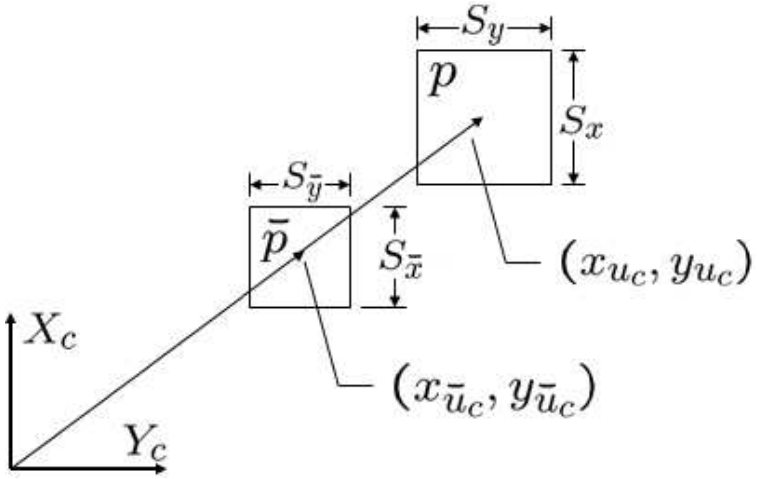


Figure 4.12: The shrinkage effect of radial distortion.

With the understanding gained from Section 2.2, one can compare the size of $\breve{p}$ with the size of p by equating their upper and lower boundaries through the factor k_s:

$$\left(x_{\breve{u}_c} + \frac{S_{\breve{x}}}{2}\right) - \left(x_{\breve{u}_c} - \frac{S_{\breve{x}}}{2}\right) = \left(x_{u_c} + \frac{S_x}{2}\right) k_s - \left(x_{u_c} - \frac{S_x}{2}\right) k_s. \tag{4.8}$$

Simplifying then yields

$$S_{\breve{x}} = S_x k_s, \quad \text{and similarly} \quad S_{\breve{y}} = S_y k_s. \tag{4.9}$$

This means that the area of $\breve{p}$ is related to the area of p through

$$A_{\breve{p}} = S_{\breve{x}} S_{\breve{y}} = (S_x k_s)(S_y k_s) = A_p {k_s}^2. \tag{4.10}$$

Thus, if image skew reduces a region's radial distance to 0.9 times its original value ($k_s = 0.9$), then its x and y dimensions are also reduced to 0.9 times their original value such that the area is reduced to $(0.9)^2$ times its original value. This is true as long as regions p and $\breve{p}$ are small enough that the same skew factor applies to the whole region with sufficient approximation.

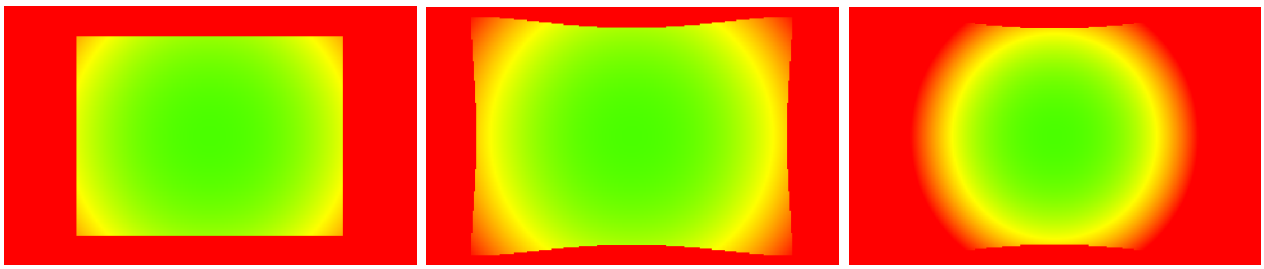
Now call region $\breve{p}$ an image pixel, and let the region $\breve{c}$ represent a collection of qualifying target pixels that is small enough to assume uniform k_s in $\breve{c}$. If the number of pixels within c is P_s, then the number of pixels within $\breve{c}$ is $P_{\breve{s}}$, where $P_{\breve{s}} = P_s {k_s}^2$. Thus, a more accurate version of Eq.(4.6) is

$$p(D|I(d,\breve{r}_m), M_{CC}) = \max \left(\frac{1 - \exp\left(-\frac{\left(\frac{k_s(\breve{r}_m) f_p}{d}\right)^2 A_{obj} - 26}{9} \right)}{1 + \exp\left(-\frac{\left(\frac{k_s(\breve{r}_m) f_p}{d}\right)^2 A_{obj} - 34}{2.8} \right)}, 0 \right), \tag{4.11}$$

where $I(d, \breve{r}_m)$ denotes an instantaneous look from distance d, with the target appearing at radius $\breve{r}_m$ from the image center. M_{CC} still denotes the connected component detection model, and the term $k_s(\breve{r}_m)$ shows that k_s depends on $\breve{r}_m$, as in Eq.(2.20). Figure 4.13 shows what happens to the sensor footprint when image skew is accounted for.

4.2 Discrete Search

Attention must now be given to a practical method of recording these detection probabilities as they accrue. This data can then be used to approximate an agent's search



(a) Detection probability without image skew taken into account.

(b) Detection probability with image skew directionality taken into account.

(c) Detection probability with image skew directionality and shrinkage taken into account.

Figure 4.13: The effect of image skew on detection probability and sensor footprint. A plane is viewed from 80 meters away using the camera calibrated in Table 2.1. Pure green indicates $p(D|I(d, \breve{r}_m), M_{CC}) = 1$, and pure red indicates $p(D|I(d, \breve{r}_m), M_{CC}) = 0$. Although image skew increases the viewing angle, it also decreases the peripheral detection probability.

width, as well as to measure UAV search performance in a given scenario. This data can also be accessed during a search to indicate location probabilities and direct UAV movement.

4.2.1 Probability Map Generation

In order to plan a search and measure its effectiveness using a computer, one must enter the realm of discrete search. Particularly, it is helpful to generate a *probability grid* containing N discrete nodes to approximate the probability distribution in the search area. The grid nodes are spaced some distance δ_g apart in the x and y directions over the region of interest. Each node, n_i, represents an area of δ_g^2, and records two probabilities:

1. $p(C_i)$, the probability that the target is contained in area A_i (given by the initial probability map).

2. $p(D(t)|C_i)$, the probability that the target has been detected by time t, given that it is contained in area A_i.

At any given point in the search, the overall probability of successful detection can be calculated according to

$$p\big(D(t) \cap C_{@}\big) = \sum_{i=1}^{N} p(C_i) p\big(D(t)\big|C_i\big), \tag{4.12}$$

where $C_{@}$ indicates that the target is contained within the area where detection occurs.

If an observer is allowed to view node n_i at time t, then an update can be made to describe the new detection probability for a target in A_i. The detection probability becomes

$$p\big(D(t)\big|C_i\big) = p\big(\tilde{D}(t)\big|C_i\big) + p\big(D^*(t)\big|C_i\big) - p\big(\tilde{D}(t)\big|C_i\big)\; p\big(D^*(t)\big|C_i\big) \tag{4.13}$$

where

$$D(t) = \text{target detection by time } t,$$

$$\tilde{D}(t) = \text{target detection before time } t, \text{ and}$$

$$D^*(t) = \text{target detection at time } t.$$

For example, let $I(d_{ij}, \breve{r}_m, t)$ be an instantaneous look at node n_i from UAV u_j at a distance of d_{ij}, where the target's image projection is $\breve{r}_m$ away from the image center, all at time t. Assume that one UAV is viewing node n_i such that $p(D|I(d_{ij}, \breve{r}_m, t), C_i, M_{CC}) = 0.6$. Also assume that previous observations have been made such that a target in A_i already has a probability of 0.9 of being detected. In this case, the probability of detecting a target in A_i by time t is

$$p\big(D(t)\big|C_i\big) = 0.9 + 0.6 - (0.9)(0.6) = 0.96.$$

Thus, a node is used to approximate the detection probability of a target, given that the target is in the area represented by the node.

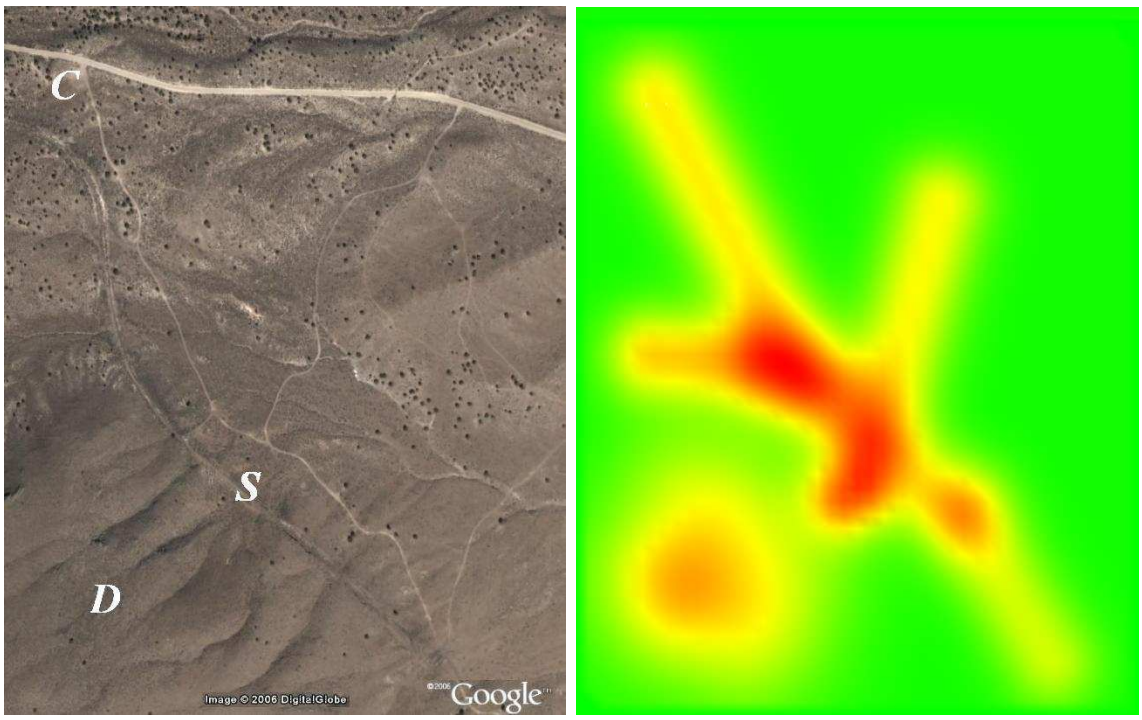
At the beginning of the search each node has zero probability of having a detected target in its area, and the probability that a target is in any given area is defined by some

initial *probability map*. This probability map is generated by considering factors such as the last seen position, probable target behavior, prominent landmarks and other evidence. The probability grid is the medium on which the probability map is stored and updated. It is a three-dimensional checklist that records whether or not all of the possible target locations have been searched, and how well. The underlying grid can be updated in simulation or during a real search, and can be helpful in planning effective paths. Grid updates require access to both the camera calibration parameters and UAV telemetry.

To better understand the concept of a probability map, it is often helpful to see one that has been generated based on a realistic search situation. This can illustrate the motivation behind using a probability map, and show some of the considerations that come into play. The following example is based on a recent BYU search and rescue exercise (SAREX) that occurred just south-west of Utah Lake. The purpose of the exercise was to test the usefulness of BYU's current SUAV technology in supporting ground team search and rescue operations. The test was also used to help identify the most important areas for improvement in BYU's SUAV systems.

Figure 4.14(a) shows the SAREX environment, where the search team encountered the following scenario:

> A group of Boy Scouts left their campground this morning at about 8:45 a.m., headed towards location D on four-wheelers. Among them was a boy named Jimmy Nielsen: 13 years old, 210 lb., 5'9", and diabetic. Along the trip Jimmy was not feeling well and decided that he would walk back to camp. He was last seen at point S, headed towards camp alone and without food or water. Jimmy may not have taken any insulin this morning. When Jimmy left camp he was carrying an orange jacket in his hands, and wearing blue jeans and a red or white t-shirt. At 9:45 a.m. the scout group returned and discovered that Jimmy had not made it back to camp. The time is now 10:30 and minimal searching has been accomplished, but an SUAV is available to aid in the search.



(a) A map of the area of interest. (b) The generated probability map.

Figure 4.14: A probability map is generated from the SAREX scenario where terrain data, landmarks, last seen location and probable target behavior are all considered. The area map contains the locations C = campground, D = destination, S = last seen location. Red indicates the areas where Jimmy is more likely to be, while green indicates the less likely areas.

Based on this search scenario, a probability map such as that shown in Figure 4.14(b) could be generated to describe Jimmy's probable location. After separating from the group, Jimmy would probably have headed back to camp along the main road. On the other hand, if he became lost he might try to find a lookout point or the group. The probability that Jimmy is within the area of interest at 10:30 a.m. is only 0.95, which is the sum of all probabilities shown in Figure 4.14(b). However, the distribution is significantly biased towards roads, the group destination, the last seen location and other topological features. There are many assumptions that can be made when defining a probability map, and these are primarily the responsibility of the search planner.

Probability map derivation is not the focus of this thesis, but hopefully the reader can see that generating an accurate probability map can be a significant step towards maximizing

search efficiency. If this probability map is searched in high probability regions first, followed by a thorough sweep of its remaining areas, then the detection probability can be maximized at each point in time. The first step in optimal search is generating this map.

It is also useful to note that the probability map and grid discussed in this section record the probabilities associated with the presence and detection of one target only. For the multiple target case another map and grid might be present to record the same information related to each additional target. Indeed, this could be necessary if two targets are not expected to behave the same. However, if the initial probability map and detection probabilities are assumed to be identical between more than one target, then only one map and grid may be necessary.

For example, say that at some point in time the information from map and grid 1 indicates a detection probability for target 1 of 0.5, and the information from map and grid 2 indicates a detection probability for target 2 of 0.6. To calculate the probability of detecting at least one target, one would find it to be $0.5 + 0.6 - (0.5)(0.6) = 0.8$. However, if the information from map and grid 1 was assumed to be representative of the detection probability for both target 1 and target 2, then the answer would be $0.5 + 0.5 - (0.5)(0.5) = 0.75$.

4.2.2 Search Width Characterization

In order to optimize UAV search performance, some underlying search parameters must first be chosen. Assume for a moment that there is a single Dell Inspiron 600m computer (1.8 GHz, 512MB RAM) at the site, and that this computer carries the task of running Virtual Cockpit. This includes running the connected-component video processing to identify targets, as well as maintaining constant communication with the UAV through the ground-station modem. (See Section 2.5.1.) With this setup the video processing can interfere with other essential functions if frames are analyzed much faster than twice per second. This dictates the rate and times at which the grid is updated.

Here it is necessary to introduce a quantity called the *area detection probability*, Λ. This is the average node detection probability in an area, multiplied by that area, or

$$\Lambda \;=\; \frac{\sum_{i=1}^{N} p(C_i)p\big(D(t)\big|C_i\big)}{N} A_N \;=\; \delta_g^2 \sum_{i=1}^{N} p(C_i)p\big(D(t)\big|C_i\big), \tag{4.14}$$

where A_N denotes the area represented by the N nodes being considered. The area detection probability is a measurement of not only how much detection probabilities have changed, but over what area.

If a UAV flies over a previously un-searched region, then the rate of the area detection probability change is directly proportional to the effective UAV search width according to

$$\dot{\Lambda} = WV_g \tag{4.15}$$

where

$$W = \text{the UAV effective search width,}$$

$$V_g = \text{the UAV groundspeed.}$$

Thus, a maximum $\dot{\Lambda}$ corresponds to a maximum W for a given groundspeed.

Due to the correlation between search rate and UAV velocity, one might push the UAV to its maximum velocity during a search. However, the maximum velocity can also cause high fuel consumption, high down time and few target glimpses. Assuming negligible wind such that the flight speed is the groundspeed, the average search rate for multiple flights can be described by

$$\overline{\dot{\Lambda}} \;=\; W\frac{d_f}{t_f + t_g} \;=\; WV_g\,\frac{t_f}{t_f + t_g}, \tag{4.16}$$

where $\overline{\dot{\Lambda}}$ is the average search rate, d_f is the distance covered by a single flight, t_f is the time of a single flight, and t_g is the time spent on the ground between flights. The time of a single flight is determined by the parameters discussed in [17], and can be approximated as

$$t_f = \frac{n_B C_B V_B}{P_r/\eta} \tag{4.17}$$

where

$$n_B = \text{the number of batteries (2 in this case)},$$

$$C_B = \text{battery storage in Coulombs} \left(1300\ mAh \times 3.6 \frac{C}{mAh} \right),$$

$$V_B = \text{battery voltage (12V in this case), and}$$

$$P_r/\eta = \text{the electric power required.}$$

($P_r = .1968V_g^2 - 4.4275V_g + 33.483$, and $\eta \approx .3$ are approximations from [17].)

If the search width is assumed to be constant in Eq.(4.16), then as t_g becomes small the $\overline{\dot{\Lambda}}$-maximizing velocity is the UAV maximum velocity. However, as t_g becomes large, the $\overline{\dot{\Lambda}}$-maximizing velocity is the UAV maximum range velocity. For this thesis a large t_g is assumed such that the optimum flight velocity is the maximum range velocity, or approximately 13 m/s for the small Unicorn UAV platform shown in Figure 2.13 [17, p.26].

The next parameters to be selected are UAV height above ground and camera angle. To select these, a series of simulations were run ($V_g = 13$ m/s, Frame rate = 2 frames/s) in which the camera passed over a probability grid at various altitudes and camera angles. The maximizing set (h, θ_o), illustrated in Figure 4.15, was found by running the UAV over the grid using a constant height and camera angle for each run. Altitude was evaluated from 0 to 100 m in increments of 1 m, and camera forward angle was evaluated from 0° to 90° in increments of 1° Figure 4.16 shows the $\dot{\Lambda}$ maximum and the $\dot{\Lambda}$-maximizing camera angle for each altitude. The figure presents results both with and without image skew.

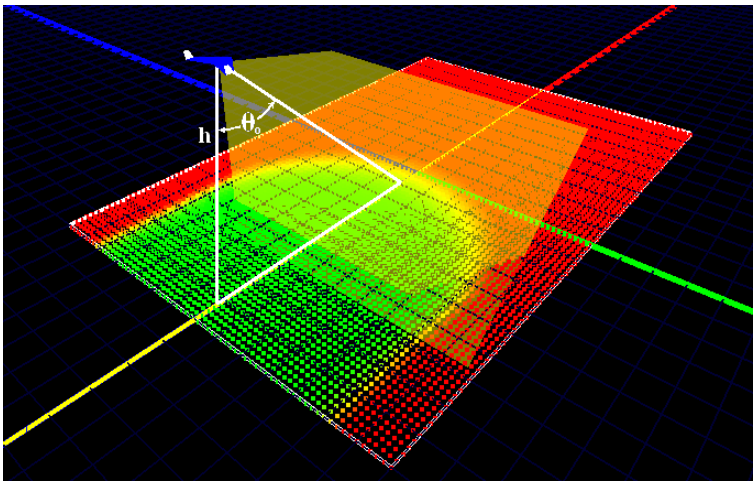
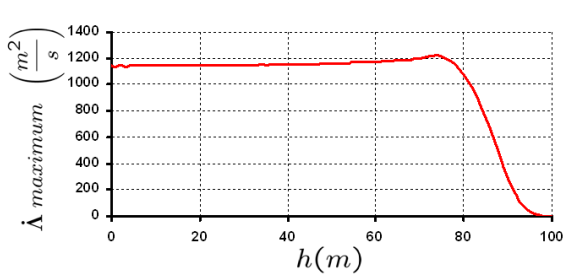
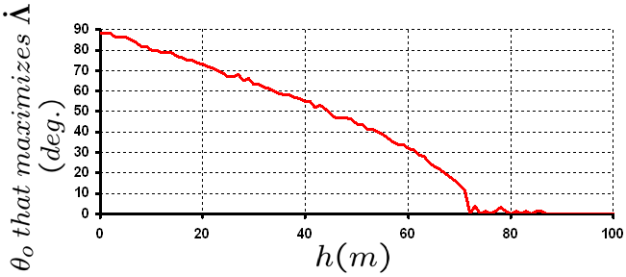


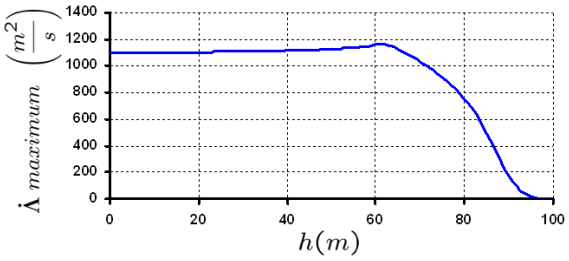
Figure 4.15: Selecting the $\dot{\Lambda}$-maximizing value for h and θ_o. The UAV is traveling over a probability grid, and the probability of detection at each node is being recorded. The solid green nodes have a detection probability near unity,and the solid red nodes have a detection probability of zero.



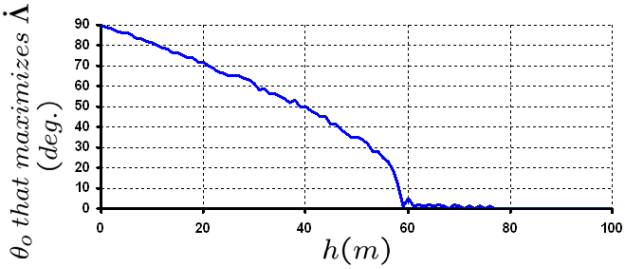
(a) Maximum $\dot{\Lambda}$ without image skew.



(b) Maximizing θ without image skew.



(c) Maximum $\dot{\Lambda}$ with image skew.



(d) Maximizing θ with image skew.

Figure 4.16: Subfigures (a) and (c) show the maximum $\dot{\Lambda}$ possible at each altitude. Subfigures (b) and (d) show the maximizing θ_o at that altitude. The top two subfigures represent data where no image skew is accounted for, and the bottom subfigures represent data where image skew is accounted for. Nodes are updated every 0.5 seconds according to Eq.(4.13).

It is interesting to note the difference between Figures 4.16(a) and 4.16(c). The image skew seems to reduce the camera's search capability between the altitudes of about 60 and 85 meters. This is the range where the corners of the image would be particularly helpful at increasing a target's detection probability through time integration. However, when the corner detection range is reduced through image skew, Figure 4.16(c) shows that it is advantageous to stay lower to the ground. The maximum value in this figure occurs at $(h = 62\text{m}, \theta_o = 2^\circ)$, which is selected as the optimal search configuration when using this CCD camera detection model. Remember that the data in Figure 4.16 depends on the camera calibration parameters, the target size ($A_{obj} = 1$ m^2), the UAV velocity (13 m/s), and a grid update once every half second.

The data obtained to calculate $\dot{\Lambda}$ can also be used to find the effective search width if the $p(D|C_i)$ for each node is recorded and the search path is known. As the simulated SUAV passes over an area, the resulting grid updates will reveal the time-integrated detection probability at each node. This detection probability may then be compared with the node's distance from the UAV path to form the lateral range curve. (See Figure 4.17.) The search width may then be calculated by integrating under the lateral range curve.

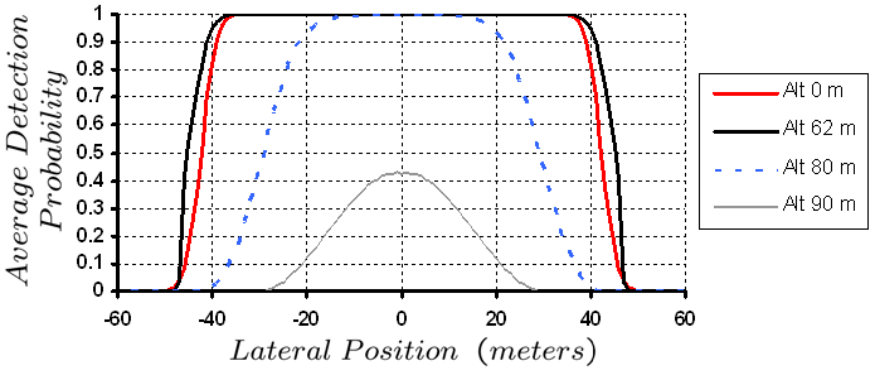


Figure 4.17: The maximum lateral range curve at various altitudes. This plot uses the altitudes shown in the key, along with the corresponding camera angles from Figure 4.16(d).

Figure 4.18 shows the maximum search width possible at each altitude. This search width corresponds directly to the rate of change in the area detection probability. Notice that the data in Figure 4.18 is an exact scaling of the data in Figure 4.16(c). The two are related through Eq.(4.15), where $V_g = 13$ m/s. However, the data in Figure 4.16(c) was measured by recording $\dot{\Lambda}$ during each UAV pass, and the data in Figure 4.18 was measured by integrating under the lateral range curve.

For an altitude of 62 meters and a θ_o of 2°, the effective search width is 89.3 meters. This information can now be used to space search paths in a real search.

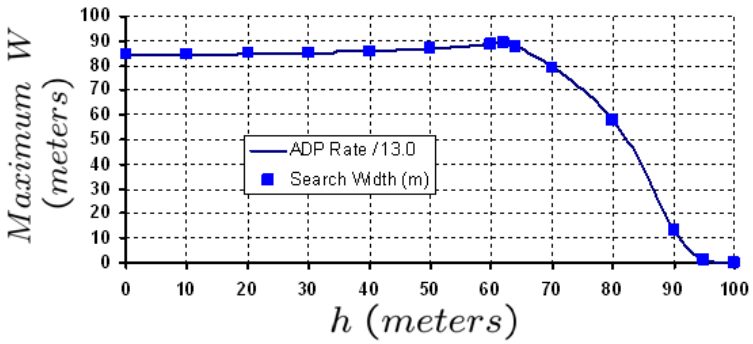
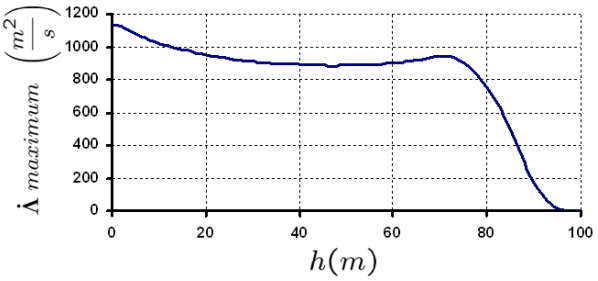


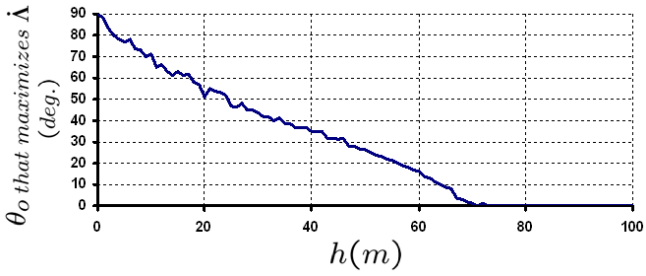
Figure 4.18: Maximum search width as a function of altitude. This plot uses the altitudes shown along the horizontal axis, along with the corresponding camera angles from Figure 4.16(d).

A similar test can be run to evaluate the effectiveness of a side-looking camera. To create Figure 4.19, $\dot{\Lambda}$ is calculated over the entire range $0 \leq \theta_o \leq 90$, then the maximum $\dot{\Lambda}$ is selected for each altitude. The difference is that θ_o is measured laterally from the flight path and the down direction.

Although there is a region where Figure 4.19 shows a greater $\dot{\Lambda}$ than in Figure 4.19, it is only for altitudes of 3 m or less, with the camera looking out nearly parallel to the ground. However, this region is undesirable in real flight due to the difficulty of avoiding obstacles at this altitude. Thus, with the intent of applying these principles to real SUAVs, a forward-looking camera is used. It is also interesting to note that if the algorithm is given the freedom



(a) Maximum $\dot{\Lambda}$ for a side-looking camera.



(b) Maximizing θ_o for a side-looking camera.

Figure 4.19: Subfigure (a) shows the maximum $\dot{\Lambda}$ possible at each altitude for a side-looking camera. Subfigure (b) shows the maximizing θ_o at that altitude. The azimuth of the camera is $\pm 90°$, or lateral to the path, and θ_o is the angle between the camera direction and straight down. Image skew is included in the detection model used to generate the data, and nodes are updated every 0.5 seconds according to Eq.(4.13).

to point in the $\dot{\Lambda}$-maximizing direction at each time step, then it ends up scanning back and forth (changing θ_o for the side-looking camera) to maximize search width. Understandably, this would be the optimum behavior for computer detection using a gimbaled camera, but it is very sickening for a human to watch.

4.2.3 Turning

Traveling in a straight line is fabulous if the search area is only one search width wide. However, most searches require the UAV to turn, which can leave considerable doubt as to the search effectiveness in those regions. So, there are several questions to answer for path-planning purposes: Can a turn be used as valid coverage of an area? How much of a turn is reasonable? What is the detection rate in a turn?

The simulations that answer these questions are very similar to those used to characterize a search width. However, the turning simulations all take place at an altitude of 62 m, a forward camera angle of $\theta_o = 2°$ and a velocity of 13 m/s. The only independent variable now is the flight radius. Figure 4.20 shows this type of simulation, where the UAV is put into a turn and allowed to view the probability grid. The lateral range curve is again calculated based on each node's distance from the UAV path. These curves are shown in Figure 4.21.

Notice that the lateral range curve generated by a 300 m radius turn is nearly symmetric, similar to that of a straight path. This is because there is only a small bank angle required to hold a 300 m radius turn, and the camera can point almost directly at the flight path. On the other hand, tighter turns result in a detection probability bias towards the inside of the turn. The sharp inner slope seems to be caused by the image boundary, while the more gradual outer slope is caused by nodes that are too far away to be fully detected.

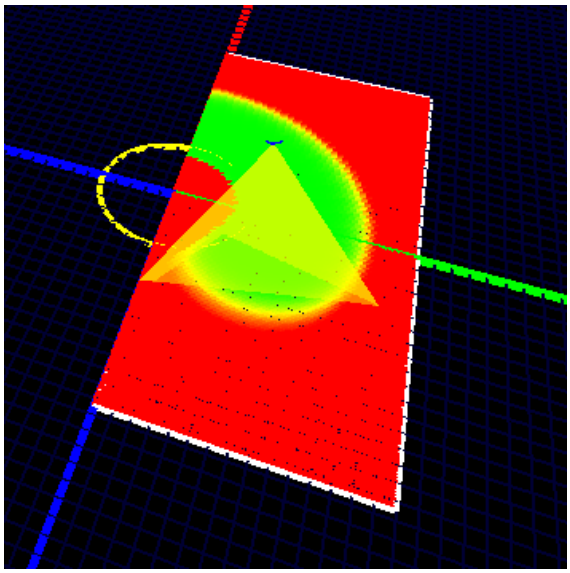


Figure 4.20: Characterization of search performance in a turn.

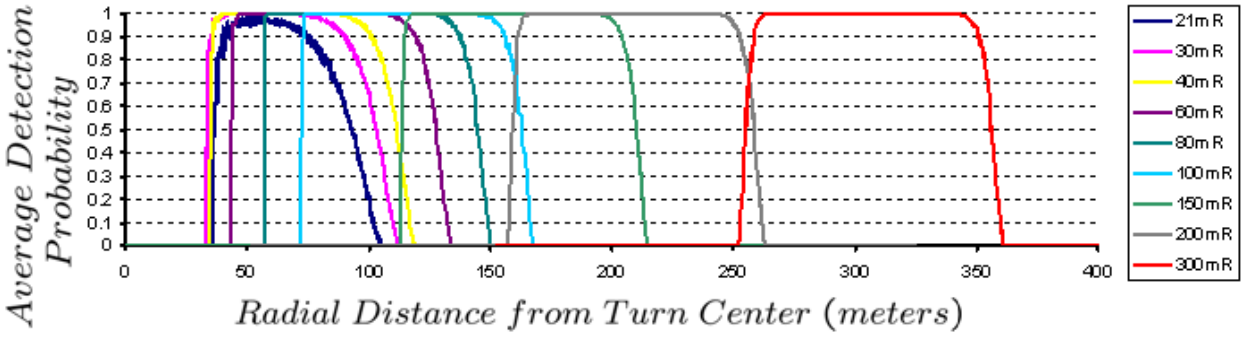


Figure 4.21: Lateral range curves during various turns at 60 m altitude.

Comparing Figure 4.21 with Figure 4.22 reveals that even though the search width is decreasing for tighter turns, $\dot{\Lambda}$ is increasing. The UAV velocity is still 13 m/s, but the camera is pointing outward from the turn, and is able to search at a greater rate than normal. Thus, with decreasing flight radius, the camera is duplicating less of the searching effort by looking at new area in greater quantity. The increasing $\dot{\Lambda}$ effect continues until it is overcome by the camera leaving spaces between consecutive frames (frame rate of 2 frames/s) and/or the viewed region being too distant for detection. The function printed in Figure 4.22 is an approximation of this increasing $\dot{\Lambda}$ behavior, where the data point at 21 meters is suffering from some of the ill effects described.

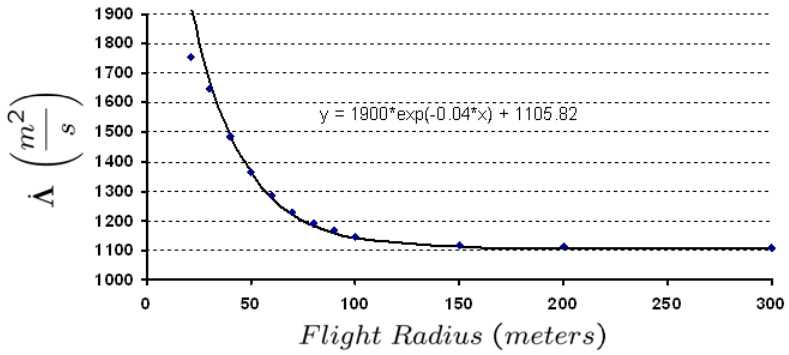


Figure 4.22: $\dot{\Lambda}$ as a function of turning radius.

Figure 4.23 is a surprisingly linear set of data comparing the search width to the bank angle. The point at $(\frac{\pi}{2}, 0)$ is a theoretical limit where the UAV is executing an instantaneous turn, and going over everything so fast that the detection probability for a single pass is zero. If this point is removed, then the linear fit remains largely un-changed. This indicates that in some cases the characterization of search width in a turn may be largely predicted by a line between the non-turning search width and this theoretical limit.

This analysis shows that significant searching can be done in turns, even with a fixed camera. However, although the rate of search can actually increase in a turn, the search

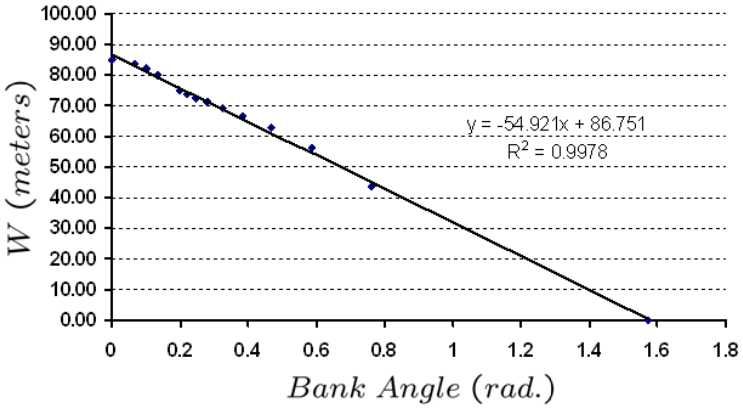


Figure 4.23: W as a function of bank angle.

width seems to diminish. These are important properties to understand when planning UAV search paths.

Although the results from this section are not explicitly used in Chapter 5, they are offered for those whose path planning can incorporate these principles to receive greater precision and fewer missed areas while turning. In general, during a turn:

1. The search rate increases.

2. Paths should be placed closer together.

This means that turns should not necessarily be avoided, but should be planned with a purpose. They can expedite the search rate for a fixed-camera agent if they are used properly.

4.2.4 Optimal Discrete Search

When dealing with a non-uniform discrete probability map it can be useful generate an optimum detection function. This serves as a standard for comparison when judging the effectiveness of path planning approaches. To generate the optimum detection function, one must know the *node search rate*

$$\dot{N} = \sum_{j=1}^{N_{UAV}} \dot{N}_j, \tag{4.18}$$

where $\dot{N}_j = \dot{\Lambda}_j/\delta_g^2 \ \frac{nodes}{second}$ for each agent, and $\dot{\Lambda}_j$ is obtained from Eq.(4.15) for each agent. $\dot{N}$ defines how fast the UAVs can collectively search nodes during straight and level flight.

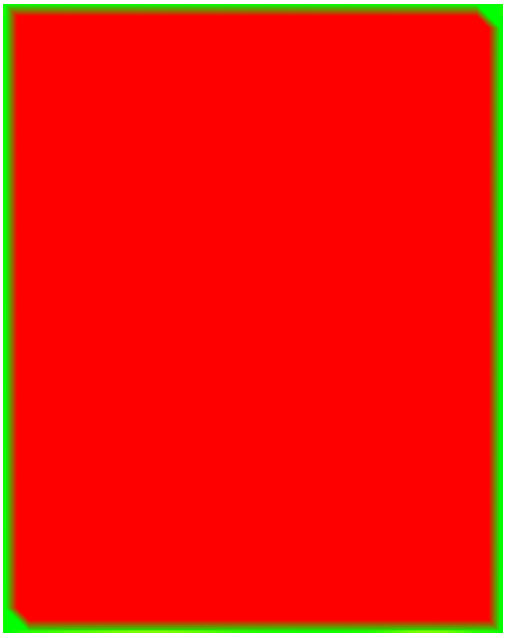
Using this definition, an optimum detection function may be created as follows:

- Estimate or measure the search width and groundspeed for each agent.
- Calculate $\dot{\Lambda}$ for each agent using Eq.(4.15).
- Order the nodes in descending probability from $i = 1 \ldots N$.
- Assume that the nodes are searched at a rate of $\dot{N}$.
- Step through the nodes from first to last, recording a new time and probability after each node. Thus, evaluation of node i yields $\Delta t = 1/\dot{N}$ and $\Delta p(D \cap C_@) = p(C_i)$.

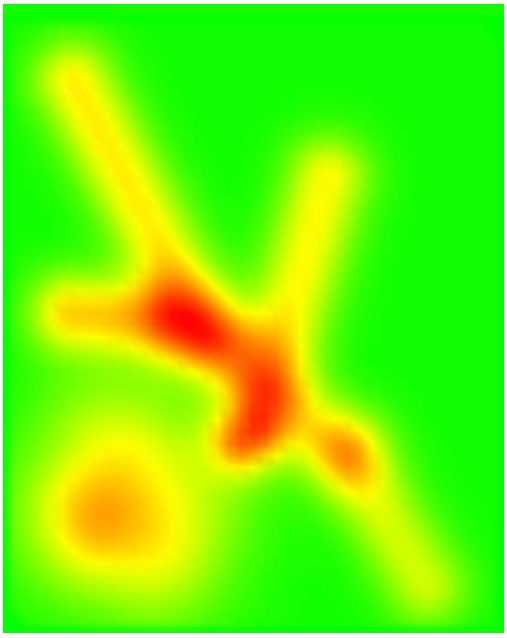
This will generate a detection function that is the discrete equivalent to Koopman's definite range law for evaluating a uniform area, as shown in Figures 4.24(a) and 4.24(c). However, this method also provides a useful analysis for more complex probability maps as shown in Figure 4.24(b) and 4.24(d).

This is a fairly simple way to evaluate how well the search would progress if the agents could be at the most beneficial location on the map at all times. Real agents are typically unable to do this, but significant differences between the optimum detection function and an actual detection function tend to point out where improvements in the search strategy can be made.

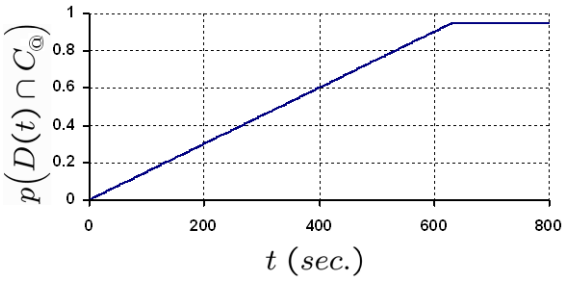
Several original contributions have been made in this chapter. These include a new CCD detection model, a discrete method for measuring search width and search rate using a probability grid, and a method for calculating an optimal detection function for any probability map.



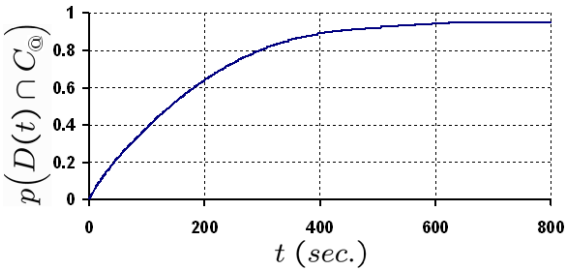
(a) A uniform discrete probability map.



(b) The SAREX discrete probability map from Section 4.2.1.



(c) The optimum detection function for the uniform map.



(d) The optimum detection function for the SAREX map.

Figure 4.24: In these two discrete probability maps, the total contained target probability is 0.95, and one UAV is searching. In the SAREX case it is possible for the detection probability to increase more rapidly at the beginning due to a search of the highest probability nodes first. However, notice that the optimum search of both maps becomes complete at the same time.

Chapter 5

Path Planning Strategy

SUAVs are approaching the point where they can contribute significantly to many missions. For instance, the MAGICC Lab has developed the capability of hand-launching an SUAV into a mission that is completely controlled by the autopilot all the way from the auto-takeoff to the auto-land cycle [18]. SUAV search paths can also be followed through mountain canyons or in support of a spiral viewing pattern [18]. As these capabilities continue to develop, it is becoming very important to measure the effectiveness of the applied search method. This chapter shows how search theory can be applied to the path planning task.

In defining a path planning strategy there are many options. Often an area is searched using some kind of offset path, as is done in a Zamboni or contour search[7, 19]. However, if an optimal search strategy is to be followed, then perhaps the agents should go straight to the high-probability regions. Then again, maybe a combination would be better.

The offset path approach typically ignores the fact that some regions are more likely to contain the target than others. Thus, it can be slower at finding the target in some situations. However, the redeeming feature of the offset path approach is that upon arriving at the end of the last path, the probability of having detected the target is greater than if the same search effort had been applied in some other way[1].

In this chapter the metrics of search efficiency and search completeness are offered as original contributions. In addition, two plans are developed and combined into a composite strategy. The first is designed to be efficient. The agents seek out high probability regions

[1]This is true if (a)the target is stationary, (b)the other search is incomplete or has redundancy, and (c)the search environment consists of a planar surface such that altitude changes are not required.

using a locally greedy search, coordinated by a global assignment. The second plan seeks to be complete. It generates offset paths from a local boundary according to search width, and assigns each agent to follow the nearest available path.

These plans are carried out in the Search Simulator, where UAVs are assigned approximate dynamic and aerodynamic constants according to the size, shape and weight of MAGICC Lab UAVs. As such, some of the numbers used for UAV control in these simulations are presented for demonstration purposes, but will need to be altered for use with other airframes.

5.1 Efficiency vs. Completeness

So which is better, the greedy or offset path method?

The answer depends on which is more important to the success of the search: efficiency or completeness. These two qualities apply particularly to a finite search where limited search effort is available. The search efficiency, $\xi(t^*)$, and search completeness, $\zeta(t^*)$, of a given detection function are shown in Figure 5.1. They are defined as

$$\xi(t^*) = \frac{\int_0^{t^*} p\big(D(t) \cap C_{@}\big)dt}{t^*} \quad \text{and} \quad \zeta(t^*) = p\big(D(t^*) \cap C_{@}\big), \tag{5.1}$$

where t^* marks the time at the end of the search.

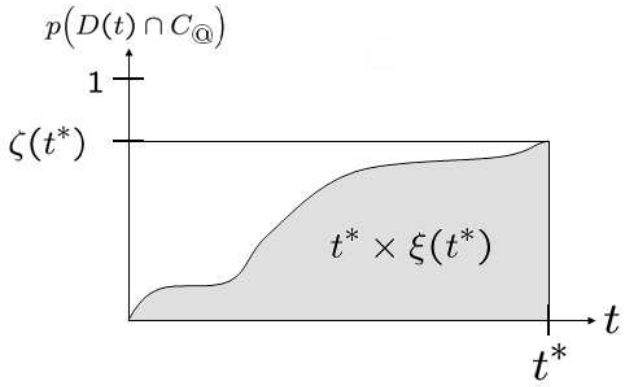


Figure 5.1: Efficiency and completeness for a given detection function.

Notice that the efficiency and completeness of two different detection functions are only comparable for a given ending time. As the search begins, a strategy that is more efficient is also more complete. However, as time progresses a less efficient strategy may attain higher completeness than its competitor. Thus, a highly complete strategy also has a certain level of efficiency, and a highly efficient strategy also has a certain level of completeness. A re-evaluation of Figure 3.12 reveals that an increase in efficiency leads to a decrease in mean detection time. Completeness, on the other hand, is a measure of the target detection probability when all of the available search effort has been expended. In certain situations it is beneficial to switch off between an efficient plan and a complete plan the updated probability map changes.

5.2 The Greedy Search

Consider the agent in Figure 5.2. Assume that this agent has position $\mathbf{w}$ and heading ψ in the world coordinate frame such that the UAV X_s stability axis points in the ψ direction. One way to execute a greedy search is to evaluate local probabilities at a distance R_p and angle ψ_p relative to the UAV, and go towards the highest probability.

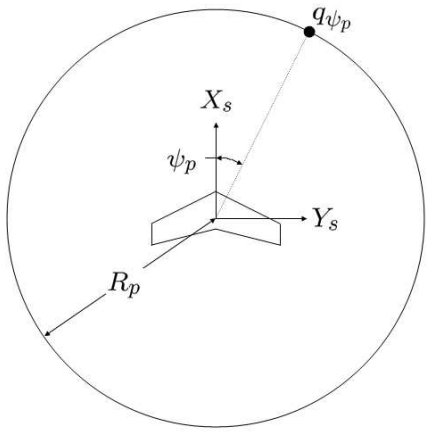


Figure 5.2: Testing a local probability.

Each node in the probability grid is already keeping track of the probability a target is within the node's area, and the probability that it has been detected given that it is in the node's area. However, the probability of interest to a UAV is the probability that any node's area contains an un-detected target:

$$p\big(U(t) \cap C_i\big) = p\big(U(t)\big|C_i\big)p(C_i) \tag{5.2}$$

or

$$p\big(U(t) \cap C_i\big) = \big(1 - p\big(D(t)\big|C_i\big)\big)p(C_i). \tag{5.3}$$

To examine the most profitable direction, a point q_{ψ_p} can be located in the world frame according to some assumed direction, ψ_p, and some assumed range, R_p. This point is

$$q_{\psi_p} = \begin{bmatrix} w_x + R_p \cos(\psi + \psi_p) \\ w_y + R_p \sin(\psi + \psi_p) \\ 0 \end{bmatrix}.$$

To examine the benefit of viewing q_{ψ_p}, some measurement must be made to determine the average $\bar{p}_{\psi_p} = \sum_{i=1}^{N_{\psi_p}} p(U \cap C_i)/N_{\psi_p}$ in its vicinity, where N_{ψ_p} is some number of selected nodes around q_{ψ_p}. This average measures the benefit of traveling in the ψ_p direction and viewing the surrounding area of q_{ψ_p}. Due to the fact that the traveling UAV will sweep out a path whose width is less than or equal to the maximized search width, an estimate for $\bar{p}_{\psi_p}$ may be obtained by averaging $p(U \cap C_i)$ for all nodes within distance $W/2$ from q_{ψ_p}.

The next task is to select the R_p and ψ_p values. If R_p is too small, then the UAV will end up getting $\bar{p}_{\psi_p}$ estimates that include some recently-viewed nodes, which can cause fidgety behavior. On the other hand, if R_p is too large, then the UAV may change course before having an opportunity to pass over the previously selected q_{ψ_p}. Fortunately, in the primarily downward-looking case, the UAV does not affect the current probability of nodes at a range more than $W/2$ away. This distance, along with the $W/2$ sampling radius described

above lead to an R_p total value of W. At this range, the angle at which q_{ψ_p} is considered must be chosen to allows the UAV enough turning time to view them. However, a maximum ψ_p range is desirable to allow the UAV greater choice among local maximums. This thesis uses $\psi_p = \{-2.0, -1.8, \ldots, 1.8, 2.0\}$ radians, which spans $\pm 115^\circ$ because simulations have shown that the modeled UAV can turn quickly enough to view q_{ψ_p} points that are 89.32 m away at these angles.

An algorithm could certainly look further ahead than R_p if desired, and some researchers have investigated more sophisticated algorithms where the UAV flies along some locally-optimal path [8]. However, the given procedure is one that fulfills the requirement of viewing nearby locations in a greedy fashion. Due to the fact that this algorithm is intended for use with one or more UAVs, each UAV u_j is discouraged from searching near other UAVs by modifying $\bar{p}_{\psi_p}$ according to

$$\bar{p}'_{\psi_p} = \bar{p}_{\psi_p} \prod_{n=1}^{N_{UAV}} k_{un}, \quad n \neq j, \tag{5.4}$$

where N_{UAV} is the number of UAVs, and UAV u_j is the one making the directional decision. This uses

$$k_{un} = \begin{cases} 0 & \text{if } d_{un} < R_p \\ \frac{d_{un}}{R_p} - 1 & \text{if } R_p \leq d_{un} < 2R_p \\ 1 & \text{if } d_{un} \geq 2R_p \end{cases}, \tag{5.5}$$

where d_{un} is the distance from some other UAV, u_n, to the point q_{ψ_p}. In other words, k_{un} is a scale factor that modifies the average probability associated with q_{ψ_p} such that $\bar{p}'_{\psi_p}$ is zero within R_p of another UAV and rises linearly until $\bar{p}'_{\psi_p} = \bar{p}_{\psi_p}$ at $d_{un} \geq 2R_p$. The range R_p is used in k_{un} because it is the same radius that is used in directional decision-making. Thus, the given logic always makes heading towards a proximate UAV look un-attractive in the greedy search.

Each UAV is commanded towards the maximum $\bar{p}'_{\psi_p}$ value at each time step according to

$$\psi_g = \frac{\tau\psi_{g,last} + \Delta t\psi_{p,max}}{\tau + \Delta t}, \tag{5.6}$$

where $\psi_{p,max}$ is the ψ_p corresponding to the maximum $\bar{p}'_{\psi_p}$ value, τ is a time constant, and Δt is the time step size. Eq.(5.6) is a low-pass filter on the commanded heading, where τ has been tuned to produce the desired dynamics with the assumed UAV airframe. The time constant is chosen to allow the UAV to turn relatively quickly, but the low-pass filter prevents an over-aggressive response as the UAV travels over a changing probability terrain. For the simulated UAV, a value of $\tau = 1.2$ is used.

One of the problems in executing a purely greedy search is that agents can wander around for a long time chasing a local maximum that is not nearly as profitable as in other regions of the map. The greedy search[2] is part of an efficient plan, but lacks a needed holistic view. This can be added by directing agents to more profitable areas according to a global mandate, but still allowing them to execute a locally greedy search.

In order to steer agents towards the most profitable regions on a probability map, these regions must first be identified. (See Figure 5.3.) This is done using a method that is similar to that of a connected-component video processing algorithm. The nodes that are higher than a given probability threshold are identified and collected into groups. These groups are characterized according to their contained probability

$$p(U \cap C_k) = \sum_{i=1}^{N_k} p(U \cap C_i) \tag{5.7}$$

and their total area

$$A_k = \delta_g^2 N_k, \tag{5.8}$$

where N_k is the number of nodes in group G_k, δ_g is the node spacing, and $p(U \cap C_i)$ is defined in Eq.(5.3).

[2]Refers to being locally greedy, or greedy in the spatial domain.

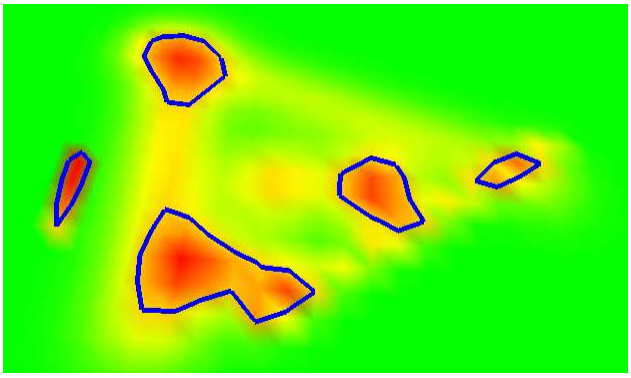


Figure 5.3: On this probability density surface, red indicates a high $p(U \cap C_i)$ compared to other regions, and green indicates a low $p(U \cap C_i)$ compared to other regions. Groups of nodes with a probability above a certain threshold have been identified and bounded with a blue line.

When searching with more than one UAV, any given group, G_k, most likely has a different probability rate to UAV u_j than it does to any other UAV. This rate can be defined as

$$\dot{p}_{jk} = \frac{p(U \cap C_k)V_g}{d_{jk} + A_k/W}, \tag{5.9}$$

where d_{jk} is the closest distance from UAV u_j to group G_k. Notice that the denominator is an approximation of the travel distance required to get to group G_k and search it, and the numerator contains a sum of the node probabilities within G_k. Multiplying this probability length density by V_g yields an estimate of the probability rate associated with searching group G_k. Due to the fact that $p(U \cap C_k)$ and A_k are constant for a given node group, the maximum $\dot{p}_{jk}$ for a given node group also indicates the closest UAV[3]. However, Eq.(5.9) is only an estimate of the probability rate, and does not take into account the benefit of viewing regions on the way to G_k.

This thesis executes search assignment as follows:

1. Each UAV-group pair is assigned a probability rate according to Eq.(5.9.

2. The list of probability rates is ordered from highest to lowest.

[3]This assumes that an identical velocity and search width common to all agents.

3. The UAV and node group corresponding to the maximum probability rate are assigned to each other.

4. The list is sequentially evaluated, and UAV/node group assignments are made when the probability rate corresponds to a UAV and node group that have been previously un-assigned.

This means that every UAV will be assigned a group if there are more groups than UAVs, but that some UAVs will go un-assigned if there are more UAVs than groups. The un-assigned UAVs continue to search in a locally greedy fashion. Once a UAV has been assigned to a group, it is commanded to fly towards it if it is further than d_G from it. Once it comes within distance d_G from the group it begins a local greedy search. This thesis uses a value of $d_G = R_p$ to ensure that the UAV comes close enough to detect the desired node group.

Although it is desirable to direct UAVs to more profitable areas, meddling with their search by giving them a new assignment can pull them away from some smaller probabilities that may need to be re-searched later. Thus, it is beneficial to establish an update rate that will not allow the UAVs to search in un-profitable regions for too long, but which offers only periodic guidance. Accordingly, the groups and assignments are calculated once every $t_A = 20$ seconds, with node probability updates still occurring once every half second[4]. This greedy approach ensures that profitable nodes are accounted for in the search plan whether they are nearby or not.

As mentioned previously, this global assignment depends on some number of identified nodes that have been placed into node groups, where the appropriate number of nodes is experimentally determined. There should be enough nodes to give the UAVs guidance, but not so many nodes that the UAVs are directed to huge, un-helpful node groups. This thesis

[4]As described in Section 4.2.2, a probability update occurs once every half second because that is the fastest that the given ground station is able to process a video frame while coordinating UAV flight.

groups $t_G = 7$ seconds-worth of nodes once every $t_A = 20$ seconds. Based on t_G, the actual number of nodes that is grouped is $N_G = \dot{N} t_G$, where $\dot{N}$ is defined in Eq.(4.18).

5.3 The Contour Search

The greedy search can be an effective way to search a complex probability distribution, but there often arrives a time when the probability is uniform within a defined region. It is more efficient in this case for a UAV to systematically follow offset paths that are spaced W apart, rather than to seek out probability differences.

Generation of offset paths has been extensively studied in the field of pocket milling. Hatna claims that for regular shapes a zigzag pattern is more efficient, but for complex shapes a contour pattern is more efficient [20]. (The word 'efficient' here refers to the shortest path length.) Yao and Gupta elaborate on this statement by developing a branch and bound method that generates a composite cutter path using a combination of zigzag and offset contour paths. They claim that this new path is guaranteed to be shorter than traditional zigzag or contour paths alone [21].

After a greedy search has exploited the most profitable regions, the remaining probability map often has a shape that is quite complex. Thus, path generation could be done with the branch and bound method, but for simplicity this thesis implements a pure contour pattern from the VRONI algorithm of Martin Held. This algorithm generates contour paths by trimming offset contour segments to the boundaries of an extended Voronoi diagram.

Path generation is not the focus of this thesis, and is not discussed in more detail here. However, readers interested in this subject are invited to to explore the works of Martin Held in [22] and [23]. For now, it is sufficient to understand that some method has been used to generate offset paths at specified intervals (search widths) from the boundary.

Once these offset paths have been created, it is important to note that great care has been taken to ensure that they do not overlap, and that they are spaced W apart. Thus, a contour search requires that they be followed in some organized and non-overlapping fashion.

This could be done by simply allowing the UAV to follow the contours using the vector-field approach presented in [24]. However, this method can cause a drastic and un-predictable viewing pattern when the UAV encounters sharp direction changes. Another way to follow the contour lines is to use the sliding segment technique shown in Figure 5.4. This technique can be used to predict turns in the upcoming contour path, and keep the camera largely pointed at the path.

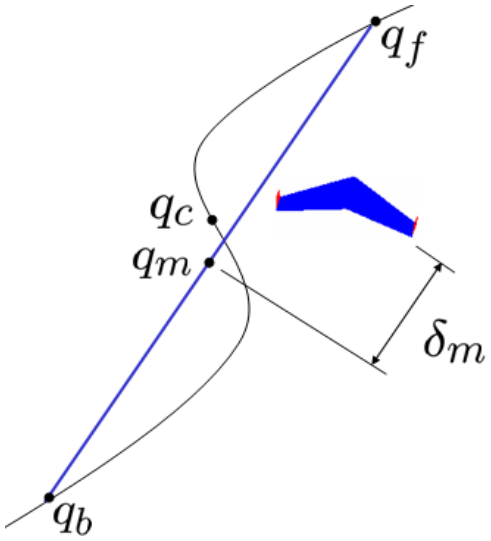


Figure 5.4: The sliding segment method.

This technique is accomplished by keeping track of q_c, a point representing the forward progress of the UAV *along the contour path*. Using a segment length of $L_s = 90$ m, q_f is created $L_s/2$ in front of q_c along the contour path, and q_b is created $L_s/2$ behind q_c along the path. The UAV is then commanded to fly along segment $\overline{q_b q_f}$ using a vector-field approach, and the segment midpoint, q_m, is used as a benchmark for UAV forward progress *along the segment*. At each time step the UAV moves forward some distance δ_m along segment $\overline{q_b q_f}$, and point q_c is advanced the same δ_m *along the contour path*. Then q_f, q_b, q_m and δ_m are calculated over again.

The sliding segment method is one predictable way to control the UAV system. The UAV follows a 'sliding segment' that changes direction in anticipation of upcoming curves, keeping the camera largely pointed at the path. Together, offset contours and the sliding segment method represent one way to search an area in a systematic manner with a fixed camera.

Un-like the greedy search, the contour search does not have time or node thresholds. All of the non-zero nodes are simply grouped and a boundary is drawn around them. Then the offset contours are generated and followed by the nearest available UAV. This method of assignment illustrates one of the offset contour search assumptions: probability uniformity. UAVs are assigned to follow the nearest available contour because all contours are assumed to have equal benefit other than their distance from UAVs.

5.4 The Composite Search

A composite search can be executed by switching off between the greedy search and the offset contour search at different times. In general, the greedy search is used unless a certain degree of uniformity is present, and then the contour search is used. The composite search method is not un-like the greedy search. It simply uses a new time threshold, t_C to determine how many seconds-worth of nodes must be within a single group before a contour search is warranted. The rate at which UAVs are capable of searching ($\dot{N}$) then determines the precise node count according to $N_C = \dot{N} t_C$. Once contours have been drawn they must all be followed before the UAVs switch back to greedy searching.

Rather than add more detail about how to carry out such an algorithm, it is helpful to see a small example of when the greedy and contour searches might be appropriate. Assume that there are two UAVs searching a probability map, each traveling at a velocity of 12 m/s and having a search width of 8 m. Also assume that the underlying probability grid has a spacing of $\delta_g = 20$ m. Under these conditions, and assuming no overlap, nodes are searched at a rate of

$$\dot{N} = \frac{N_{UAV} V_g W}{\delta_g^2} = .48 \ \frac{\text{nodes}}{\text{s}}, \tag{5.10}$$

where $\dot{N}$ is the node search rate defined in Eq.(4.18), N_{UAV} is the number of UAVs, V_g is groundspeed, and W is agent search width.

The greedy search and the offset contour search are both dependent on groups of nodes that have been identified and contained in a boundary. These nodes are identified because they are each above some probability threshold, P_T. For the greedy search, assume that at any one time it is desirable to have at least $t_G = 6$ s worth of bounded nodes, where t_G is the time threshold for the greedy search. This means that it is desirable to have at least $N_G = \dot{N} t_G = 2.88$ nodes bounded at any one time, where N_G is the node threshold for the greedy search.

Now consider the probability map and node list shown in Figure 5.5. If it is desirable to have at least $N_G = 2.88$ nodes bounded at any time, then by stepping down the node list to the first node with $i \geq 2.88$, the probability threshold P_T is found. This threshold can then be used in the spatial domain to identify and establish boundaries around all nodes with probabilities greater than or equal to P_T. The given method for selecting P_T causes the collective number of nodes in all boundaries, N_B, to always be greater than N_G.

When P_T is selected in this way, each resulting group, G_k, contains some number of nodes, N_k. However, all nodes with $p(U \cap C_i) \geq P_T$ are included, which makes it possible for the N_k of some groups to become large. When this happens, some time threshold, t_C, can be used to establish how many seconds-worth of nodes must be contained in any one group to justify starting a contour search. If $t_C = 23$ s, then the number of nodes that must be contained is $N_C = \dot{N} t_C = 11$ nodes, where N_C is the node threshold for a contour search. If more than 11 nodes are contained within any one boundary, then a contour search begins.

To illustrate this logic, assume that at some point the UAVs were able to search the probability map in Figure 5.5 such that the three nodes in the upper-right hand corner all have values of $p(U \cap C_i) = .05$. The new situation is shown in Figure 5.6.

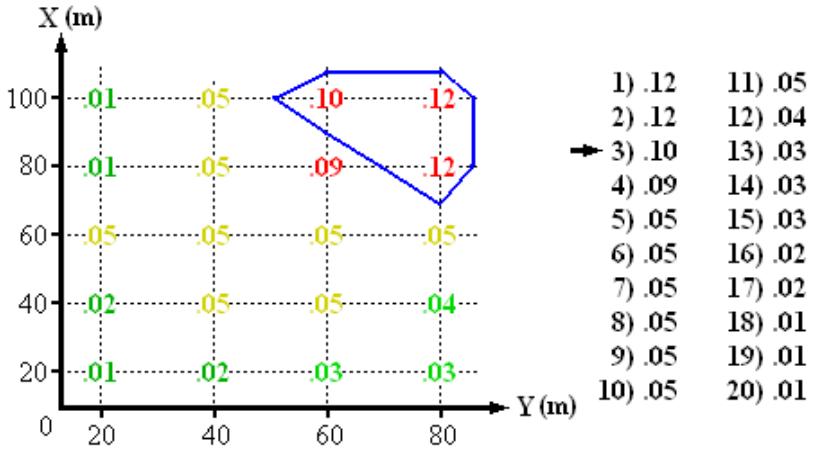


Figure 5.5: A small example probability map and the list of nodes, ordered highest to lowest. The numbers shown in the grid represent the probability $p(U \cap C_i)$ for each node n_i. The bounded nodes are above some probability threshold, P_T, which is chosen by stepping down the ordered list to the first node with $i \geq N_G$.

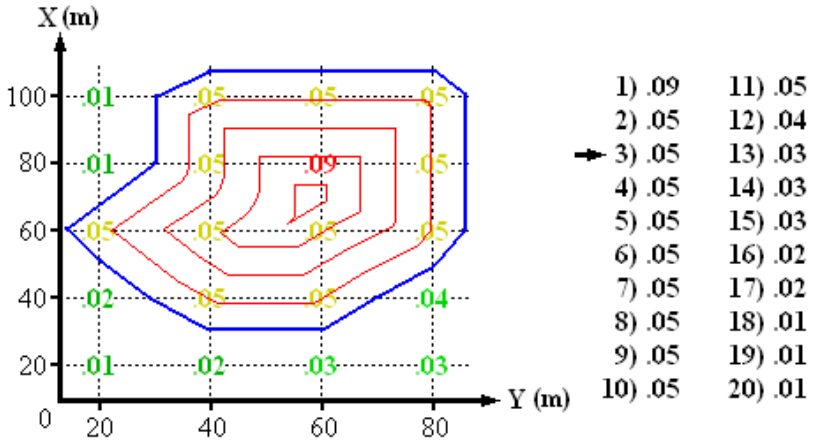


Figure 5.6: Offset contours on the example probability map. If the probability threshold is $P_T = .05$, then $N_B = 12$ nodes are bounded as shown. If the contour threshold is $N_C = 11$ nodes, and this is less than N_k, then offset contours are drawn and a contour search begins.

Stepping down the list to the node with $i \geq N_G$ selects a probability threshold of $P_T = .05$, which causes a boundary to be drawn around 12 nodes. This also causes N_k to be greater than or equal to N_C, or $12 \geq 11$, which causes a contour search to begin.

Hopefully these examples have been helpful in understanding the method of composite search for this thesis. In contrast to the above examples, this thesis uses the values of $t_G = 7$s and $t_C = 100$s as the time thresholds for the greedy and contour search respectively. In the

greedy search UAVs are directed towards profitable areas that are identified using t_C. This parameter must be high enough that the UAVs typically have bounded areas to work on, but low enough that they are not directed to large areas that offer little guidance. On the other hand, t_G is chosen to be high enough that a contour search will begin when a sizeable search area exists, but low enough that the search areas do not have to be excessively large for this to happen. Both t_G and t_C are chosen experimentally.

The presented logic is one way to identify when a sufficient degree of uniformity exists to begin a contour search. In a more general sense, it also indicates when to perform a search that focuses on completeness vs. one that focuses on efficiency. Notice that this strategy parallels the principle of reversal for large Φ discussed in Section 3.4 in that the regions of high probability are searched until a fairly uniform distribution remains. At this point it becomes profitable to not only change the region of search, but also the method of search.

5.5 Simulation Results and Discussion

There are trade-offs in choosing to search using the greedy algorithm vs. the offset contour algorithm, and a composite search can produce better results in some cases. In this section three probability maps are searched using the greedy algorithm, the offset contour algorithm, and the composite algorithm. Each map is searched using a different number of UAVs, and is designed to show specific cases where each algorithm might be most effective. However, the "effectiveness" of each algorithm is subject to interpretation, and depends on the relative importance of efficiency vs. completeness to the success of the search. This section ends with a comparison between efficiency and completeness for each algorithm, and compares them against the uniformity of the probability map.

The first simulation shows when it might be best to use the greedy search algorithm described. The probability map uses a Gaussian distribution to define $p(U \cap C_i)$ for each node, as shown in Figure 5.7. Red represents a high $p(U \cap C_i)$ relative to other nodes, and

green represents a low $p(U \cap C_i)$ relative to other nodes. This region is searched with one UAV.

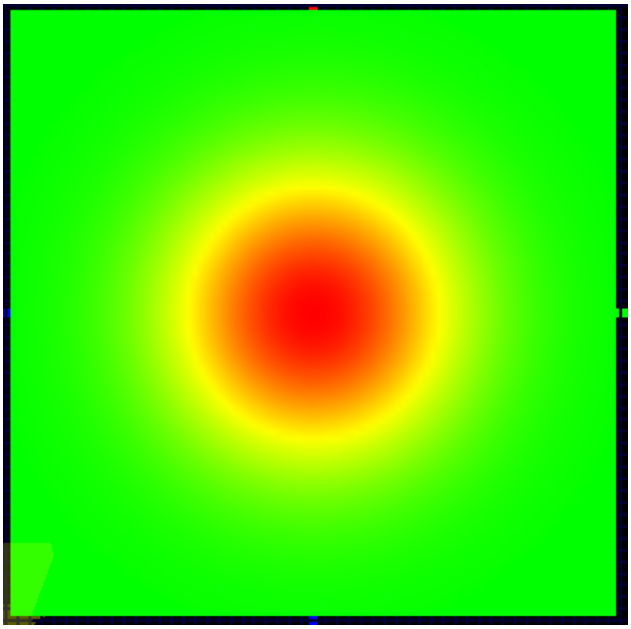


Figure 5.7: A Gaussian probability map.

Figure 5.8 shows the detection functions for a search of this region using both greedy and offset contour methods. The optimum detection function is also shown for comparison, and the actual paths are shown in Figure 5.9.

The blue curve in Figure 5.8 is labeled 'Composite / Greedy' because a composite search is used to obtain this curve, but it is always appropriate to use a greedy search for the given situation and time frame. Notice that in this case the greedy search follows the optimum much better than the contour search. This is because the agent goes straight to the center of the Gaussian distribution, then follows the local probability in a manner that approximates a spiral search. The contour search, on the other hand, follows a methodical set of paths to cover the region, which causes the detection probability to rise more sharply at times when the more profitable regions are being viewed.

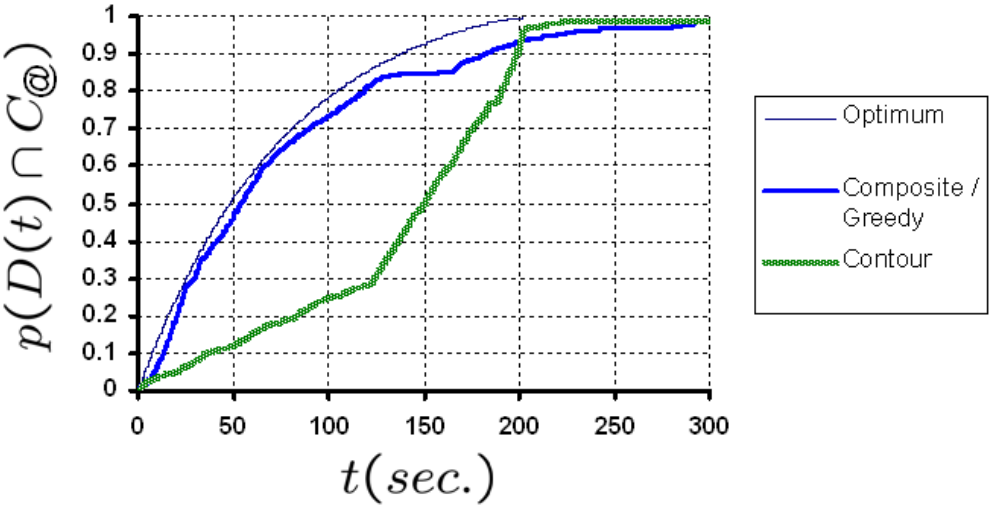
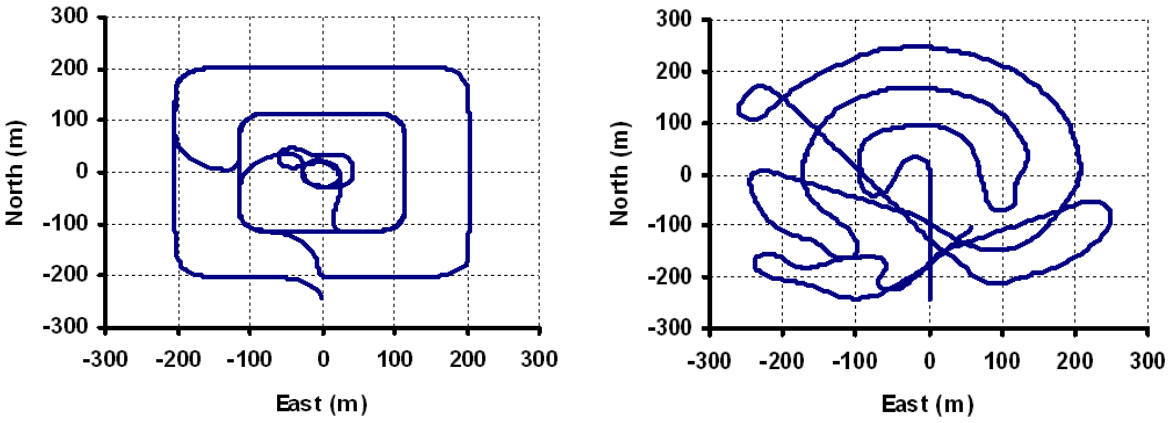


Figure 5.8: Detection functions for a search of the Gaussian probability map.



(a) The offset contour pattern.

(b) The greedy search pattern.

Figure 5.9: The search paths used on the Gaussian probability map. These paths are applied to the map in Figure 5.7 to generate the detection functions shown in Figure 5.8. The contour pattern is one type of offset contour method, and generates a typical systematic detection function. The greedy path is less organized and less complete, but searches high probability regions early.

The completenesses and efficiencies for each of the different search methods are shown in Table 5.1. Notice that the greedy search yields an efficiency that is significantly higher than the contour search, and arrives at a completeness that is comparable to that of the contour search. However, the contour search systematically proceeds until it is more complete than the greedy search at 200 seconds. Thus, the early efficiency of this type of greedy search sacrifices some degree of future completeness.

Table 5.1: Completeness and efficiency for searches of the Gaussian map.

	$\zeta(200)$	$\xi(200)$
Optimum	0.996	0.694
Greedy	0.933	0.642
Offset Contour	0.926	0.325
Composite	0.933	0.642

Notice that offset path plans like the Zamboni or contour search are not always the best. In fact, this case is an example where an alternate plan yields a 50% increase in efficiency as compared to an offset path plan. In addition, the greedy approach would also be the more complete plan if the search ended before 198 seconds. Thus, the decision of which plan to use greatly depends on the shape of the probability distribution, the benefit of efficiency vs. completeness, and the amount of search effort available.

The next example is a two-agent search of a uniform probability map, and is an example where the offset contour search might be best. (See Figure 5.10.) Each node within the red region has an identical $p(U \cap C_i)$, and nodes outside the red region have a $p(U \cap C_i)$ of zero.

Figure 5.11 shows the detection functions for a search of this region. The 'Composite / Contour' curve is labeled as such because it is a composite search, but always exhibits offset contour behavior. The UAVs have finished following all of the available contours at about 570 s, and a new set of contours is generated. Figure 5.11 shows that even at the end

Figure 5.10: A uniform probability map.

of the first contour search there are still missed areas that must be re-visited. Figure 5.12 shows the actual paths that are used.

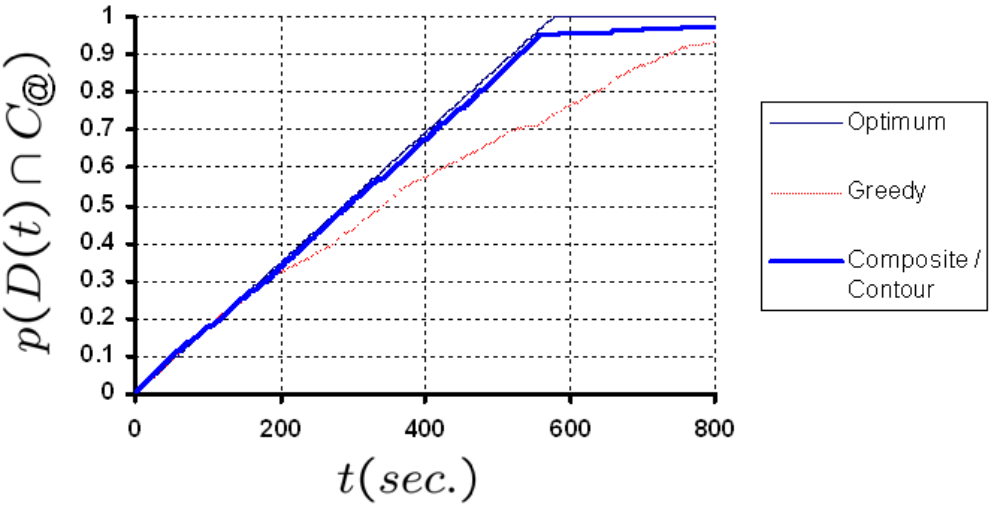
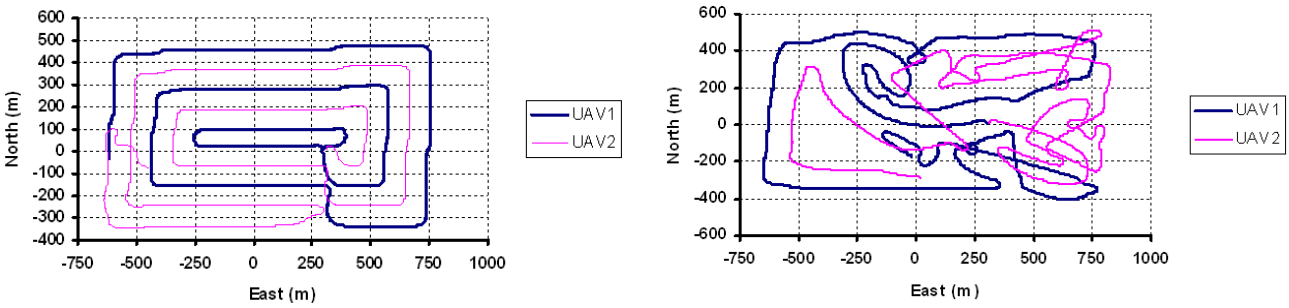


Figure 5.11: Detection functions for a search of the uniform probability map.

Table 5.2 shows that the contour search is both more efficient and more complete than the greedy search after 577 seconds. However, the greedy search is not far behind.

The drawback of the greedy search is that there are times when the agents will cross over previously searched regions to get to some maximum. As this happens more and more, the 'Swiss cheese' effect shown in Figure 5.13 becomes prevalent. The cross-overs cause more



(a) The composite/offset contour pattern.

(b) The greedy search pattern.

Figure 5.12: The search paths used on the uniform probability map. These paths are applied to the map in Figure 5.10 to generate the detection functions shown in Figure 5.11. The composite path begins by following offset contours, then switches over to a greedy search after a time.

Table 5.2: Completeness and efficiency for searches of the uniform map.

	$\zeta(650)$	$\xi(650)$
Optimum	0.999	0.500
Greedy	0.742	0.419
Offset Contour	0.955	0.490
Composite	0.955	0.490

holes, and the holes cause more cross-overs in a pattern of diminishing returns. Perhaps this is the reason that searches presented in much of the literature ([2],[7],[19]) favor some type of methodical contour method. Using a greedy strategy, and without the aid of a computer, it can become very difficult to locate and remember everywhere that an agent has been. Even with a computer, Figure 5.13 seems to indicate that the coverage of a parallel-pass method might have the benefit of increased uniformity.

For the third example, Figure 5.14 shows a uniform probability distribution that has been augmented with several other Gaussian shapes. This example illustrates when a composite search might be best, and uses three agents are used to search the map.

The detection functions shown in Figure 5.15 are all separate runs, although the greedy search curve hides behind the composite curve most of the time. This is because the composite search uses the greedy method until t =220 s. After this point the composite

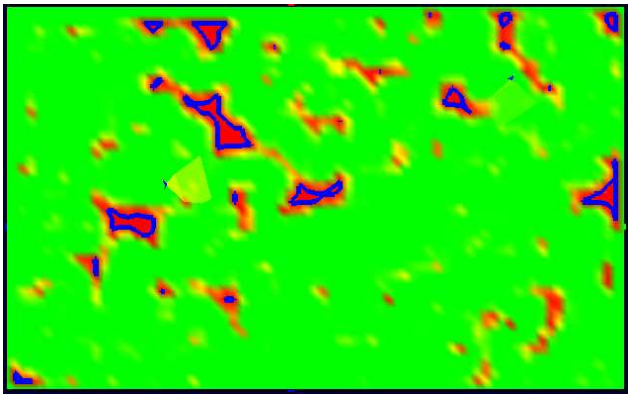


Figure 5.13: The 'Swiss cheese' effect of a greedy search on a uniform probability map.

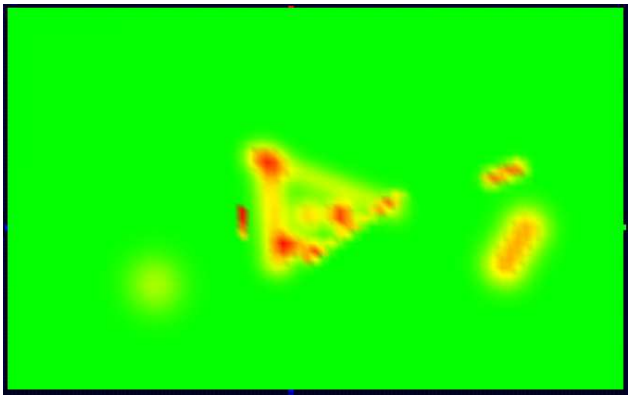


Figure 5.14: A complex probability map.

search switches over to a contour search to best cover the uniform distribution that is left. This slightly increases the slope of the detection function such that the composite search comes out better than greedy search in both efficiency and completeness. (See Table 5.3.) The actual paths are shown in Figure 5.16.

Table 5.3: Completeness and efficiency for searches of the complex map.

	$\zeta(400)$	$\xi(400)$
Optimum	1.000	0.913
Greedy	0.969	0.815
Offset Contour	0.981	0.376
Composite	0.974	0.816

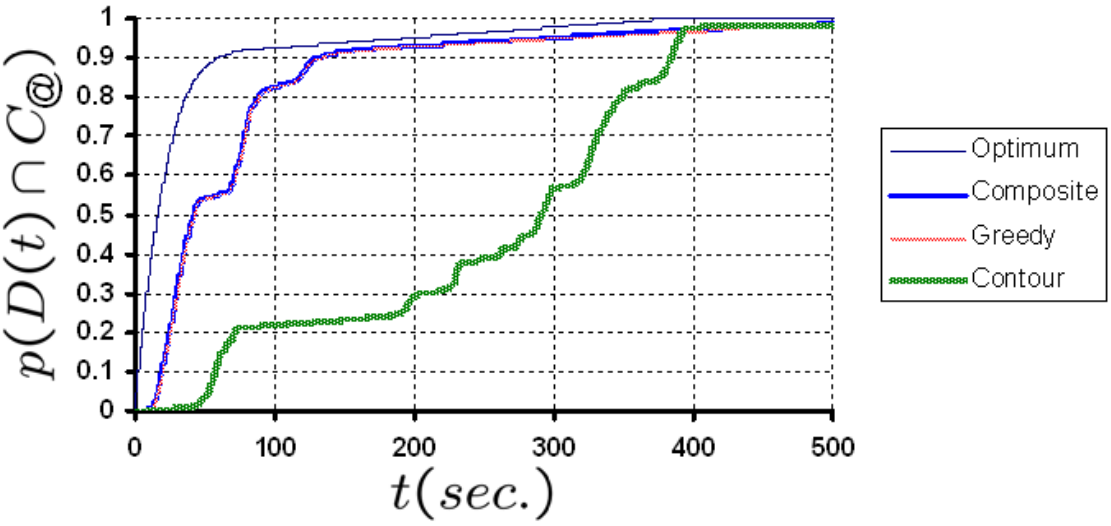
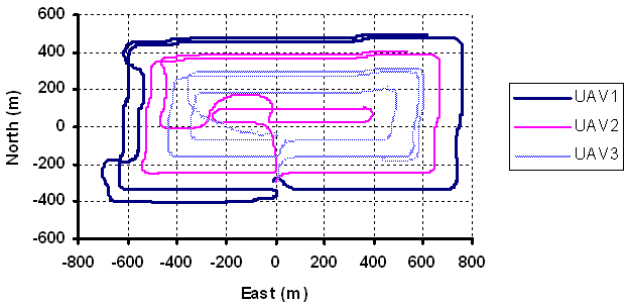
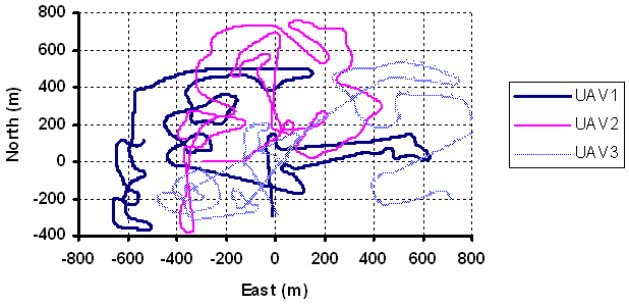


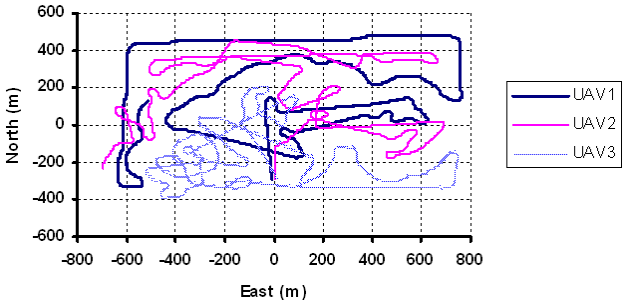
Figure 5.15: Detection functions for a search of the complex probability map.



(a) The offset contour pattern.



(b) The greedy search pattern.



(c) The composite pattern.

Figure 5.16: The search paths used on the complex probability map. These paths are applied to the map in Figure 5.14 to generate the detection functions shown in Figure 5.15. A dot indicates the starting point for the agents.

One might argue that the benefit of switching over to the contour search does not provide much improvement, and is not worth the effort. That sounds fair enough when looking at the detection functions. However, after searching for 220 s and not finding the target, there is still a probability of 1.0 that must be divided between all of the remaining possible locations. When considering the future benefit in this way, the contour search looks just as advantageous as it does in Figure 5.11.

The previous three examples have shown when it might be beneficial to use the greedy, contour, or composite search strategies. However, selection of an appropriate strategy is based on the relative importance of efficiency and completeness, along with the available amount of search effort. The final example is designed to give an overview that helps the reader develop a rule of thumb for when to use each strategy. In order to do this, a constant area is searched for an amount of time that is consistent between experimental runs. Un-like the preceding examples, however, the probability distribution changes such that the tradeoff between efficiency and completeness can be seen.

Consider the variable probability map presented in Figure 5.17. This map is un-like the previous maps in that it is defined by a variable region of non-uniform probability, A_{NU}, and a variable region of uniform probability, A_U. Together these two regions make up the total area, $A_T = A_{NU} + A_U$, and are related through the ratio $R_{NT} = A_{NU}/A_T$. The variable map is generated based on this ratio as it is varied from zero to one, and the completeness and efficiency of each search algorithm is recorded.

The total region is a square with edge length $L_T = 750$ m, and the non-uniform region is a square with edge length $L_{NU} = L_T\sqrt{R_{NT}}$. The regions A_{NU} and A_T share the same lower left-hand corner.

When calculating the node probabilities in the variable map, the first step is to allocate a probability of $1 - R_{NT}$ to all of the nodes within A_T. Then, an additional probability of R_{NT} is allocated to the nodes that are within A_{NU}. This second allocation is done in a non-uniform fashion, based on a 5×5 matrix of Gaussian distributions. Each distribution

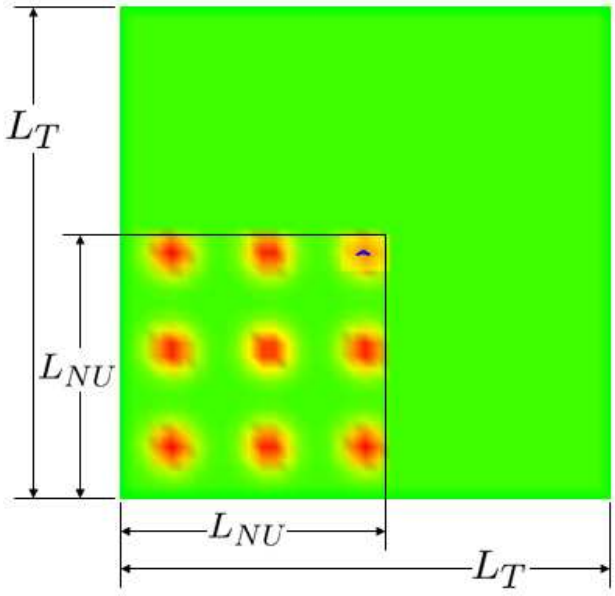


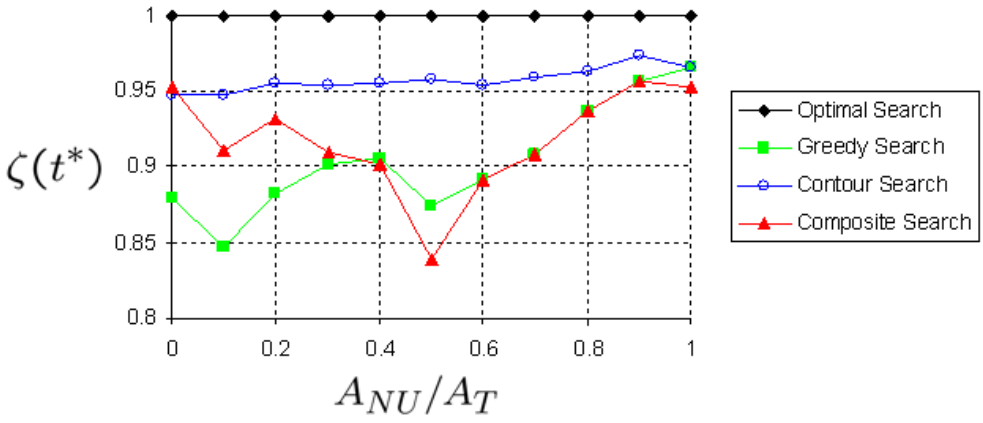
Figure 5.17: A variable probability map. The map contains a non-uniform probability region of area L_{NU}^2, and a uniform probability region of area $L_T^2 - L_{NU}^2$.

has a standard deviation of 25 m and is placed at some x and y-increment of 150 m throughout A_T, all centered within A_T. However, these Gaussian distributions are only allowed to affect the nodes in A_{NU}. As R_{NT} grows from zero to one, more of the map is affected by the Gaussian distributions, and becomes non-uniform.

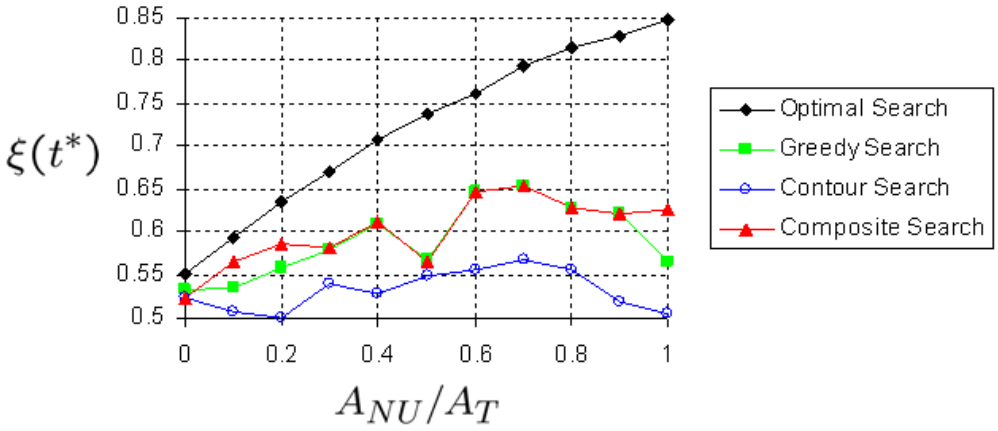
A single UAV searches each generated map by starting in the center of A_T, and each experimental run stops at time $t^* = A_T/V_g W = 484.43\ s$. Theoretically, this is the time necessary for the UAV to make one complete sweep of the total area. The results for these simulations are shown in Figure 5.18.

It is important to note in Figure 5.18 that the results are dependent upon the specific situation presented in each scenario. In some cases the map configuration may cause a UAV to end up finishing a contour right next to a region that needs to be searched, or being pulled away from something just before finishing its search. So, the key is to look for general trends rather than at specific data points.

Despite this variation, Figure 5.18 is very helpful in illustrating the trade-off between completeness and efficiency. Note that in almost every case the contour search is the method



(a) A comparison of completenesses for algorithms run on the variable probability map.



(b) A comparison of efficiencies for algorithms run on the variable probability map.

Figure 5.18: Efficiency and completeness comparison on a variable probability map.

highest in completeness and lowest in efficiency. Also note that the composite algorithm is the most advantageous within the range $R_{NT} = 0$ to 0.3. This is the range where a significant amount of uniform probability remains after the most profitable regions have been searched. This range depends on the relationship between t_C and A_T. In other words, the range could extend higher if the total number of uniform nodes remaining were greater than $\dot{N}t_C$ after the more profitable (variable) regions had been searched. Within the range the composite search can achieve efficiencies at least equal to those of the greedy search, but can augment the overall completeness to be above that of the greedy search.

It is hoped that this example provides the needed intuition for the reader. For a map with certain uniformity, the composite search strategy can be beneficial in augmenting the completeness of an efficient search strategy. Remember that the importance of efficiency is affected by the available search effort, the search rate, and the time-dependent consequences of not finding the target. On the other hand, the offset path search strategy should be used if completeness is highly important relative to efficiency. These are the rules of thumb sought for in doing the variable-map experiments.

It is important to note that a search simulation like those presented above may be executed in tandem with a real-world search. Telemetry can be fed into the simulation regarding where UAVs have been, and the resulting probability distribution can then be used to guide the evolving search effort.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The intent of this thesis is to bridge the gap between the principles of optimal search theory and the principles of path planning. However, in order to perform a meaningful search it is often necessary to detect and locate more than one target. Thus, a method of target differentiation is given that includes the following contributions:

- A practical numerical method for locating targets in three dimensions.
- A way to count and separate multiple targets as seen from multiple images.
- The concept of target quality.

Results from a field test are discussed in which multiple targets as close as 6.1 m apart are located and counted correctly. The estimated 2D locations for these targets exhibit less than four meters of error in five out of six target estimates. In addition, the concept of target quality is used to identify when faulty data is likely to be present.

Following the chapter on target differentiation, chapter three is entirely devoted to a review of optimal search theory. This chapter collects concepts from the work of two other researchers, and can be considered an in-depth literature review of the prevailing search theory.

This thesis seeks to show that applying optimal search theory to a coordinated UAV search can improve search performance. Chapters four and five present original work where optimal search theory is applied to UAV searching by way of a discrete probability grid. The original contributions include:

- A target detection probability model based on CCD camera sensing.
- Discrete methods for measuring search rate using a probability grid.
- A method for determining search width using a probability grid.
- An optimal detection function for any given probability map.
- The metrics of search efficiency and search completeness.
- A search strategy that can be applied to any given probability map.

The CCD detection model is developed using the color data obtained in the field test. This data is used to reconstruct many target images, and Monte Carlo simulations are run to determine a target's detection probability based on its size in the image. Functions are chosen to represent a target's detection probability based on its distance from the camera, and image skew is also considered to obtain a more accurate model.

The concept of a discrete probability grid and a probability map are presented as tools to measuring search progress. Using these tools and the CCD model for target detection, the search width and turn performance for the given SUAV are also developed.

In the last chapter the metrics of search efficiency and search completeness are offered. Three search situations are presented, and each of the greedy, contour, and composite strategies is applied. This does not constitute a proof of the conditions under which an efficient vs. a complete strategy should be used, because that would depend on the perceived importance of efficiency vs. completeness in a given search scenario. However, the concluding simulation illustrates some strategy trade-offs that should be considered in path planning. This result shows that considerations prompted by optimal search theory can be used to select the most effective search strategy based on the importance of efficiency and completeness.

6.2 Future Work

There are four candidate areas for immediate work. These are:

1. Detection logic
2. Ground team coordination
3. Probability map extensions
4. UAV search algorithms

As stated in Section 4.1.1, color is one of the most important considerations when planning a search. Further work should be done to quickly and easily understand the color distributions of a target and its background before beginning a search. If these distributions are understood, then algorithms for effective and/or dynamic color thresholding could drastically increase a target's detection probability through a CCD camera. The detection probability could also be significantly improved by not only identifying a known target color, but also by filtering out colors that are different from that of the background.

In order to make these principles useful, they must be integrated into the larger picture. The SUAVs described here are often hand-launched by a ground team that has a specific purpose. As such, further studies in ground team operations should be done to develop a platform and user-interface that cause the SUAV to be a valuable part of the team effort.

Additionally, the usefulness of the probability map could be extended by tailoring the update method to fit a particular purpose. For example, a learning algorithm could use Bayesian updates to change the probability map based on targets or other clues that are identified. Another possibility is the moving-target case, where the probability map could be updated based on probable target motion.

Of course, the algorithms presented in this thesis are not perfect. Particularly, the global assignment of the greedy search mandates that new regions be searched without

considering whether or not the new assignment is better than what the UAV is currently searching. In addition, the global benefit of each option does not take into account the proximity of other SUAVs. In the contour search, tight corners still cause some missed areas despite the benefit of the sliding segment method. Further research in perfecting these and other details could improve the efficiency and completeness of these methods.

On a final note, remember that the principles of optimal search theory originated from an effort to save lives during World War II. These principles were tested by pilots and observers who were searching for submarine invaders, in a situation where every failure might have caused a loss of life. The knowledge that we have today did not come without a price, and it is hoped that the reader will honor the legacy of those researchers and participants who fought for freedom. Please honor those who have given so much.

Bibliography

[1] June 2006, http://www.procerusuav.com. 1, 28

[2] B. O. Koopman, *Search and Screening, General Principles with Historical Applications.* New York, USA: Pergamon Press Inc., 1980. 1, 3, 37, 46, 47, 54, 56, 60, 61, 113

[3] Z. Tang, "Information-theoretic management of mobile sensor agents," M.S. thesis, Ohio State University, Columbus, Ohio, 43210, 2005, http://www.ohiolink.edu/etd/send-pdf.cgi?osu1126882086. 2, 4

[4] J. D. Redding, "Vision-based target localization from a small, fixed-wing unmanned air vehicle," M.S. thesis, Brigham Young University, Aug. 2005, http://contentdm.lib.byu.edu/ETD/image/etd895.pdf. 2, 12

[5] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge, UK, CB2 2RU: Cambridge University Press, 2003, ch. 12. 2, 15

[6] L. D. Stone, *Theory of Optimal Search*, ser. Mathematics in Science and Engineering. New York, USA: Academic Press, 1975, vol. 118. 3, 11, 37, 47, 49, 56, 57, 58, 59, 60

[7] H. Wollan, "Incorporating heuristically generated search patterns in search and rescue," M.S. thesis, University of Edinburgh, Edinburgh, Scotland, EH8 9YL, 2004, http://www.aiai.ed.ac.uk/project/ix/project/wollan/2004-msc-wollan-sar-patterns.pdf. 3, 95, 113

[8] M. Flint, M. Polycarpou, and E. Fernandez-Gaucherand, "Cooperative control for multiple autonomous UAV's searching for targets," in *Proc. IEEE Conference on Decision and Control*, June 2002. 3, 99

[9] June 2006, http://www.blackwidowav.com. 4, 27

[10] T. K. Moon and W. C. Stirling, *Mathematical Methods and Algorithms for Signal Processing.* Upper Saddle River, NJ 07458: Prentice Hall, 2000. 12

[11] Z. Zhang, "A flexible new technique for camera calibration," Microsoft Corporation, Redmond, WA 98052, Tech. Rep., 1998. 16, 17

[12] June 2006, http://www.vision.caltech.edu/bouguetj/calib_doc/. 16, 17

[13] June 2006, http://www710.univ-lyon1.fr/ ameyer/devel/opencv/docs/ref/opencvref_cv.htm. 16

[14] R. S. Christiansen, "Design of an autopilot for small unmanned aerial vehicles," M.S. thesis, Brigham Young University, Aug. 2004, http://contentdm.lib.byu.edu/ETD/image/etd445.pdf. 28

[15] J. W. Gibbs, *On the Equilibrium of Heterogeneous Substances*, ser. Collected Works. New York: Longman, Green & Co., 1928, vol. 1, p. 66. 54

[16] S. Baker and T. Kanade, "Limits on super-resolution and how to break them," in *IEEE Computer Society Conference on Computer Vison and Pattern Recognition*, 2000. 67, 68

[17] J. N. Ostler, "Flight testing small, electric powered unmanned aerial vehicles," M.S. thesis, Brigham Young University, Apr. 2006, http://contentdm.lib.byu.edu/ETD/image/etd1223.pdf. 85

[18] M. Quigley, B. Barber, S. Griffiths, and M. Goodrich, "Towards real-world searching with fixed-wing mini-uavs," 2005. 95

[19] D. Enns, D. Bugajski, and S. Pratt, "Guidance and control for cooperative search," in *Proc. IEEE American Control Conference*. Honeywell Technology Center, May 2002. 95, 113

[20] A. Hatna, R. Grieve, and P. Broomhead, "Automatic CNC milling of pockets: Geometric and technological issues," in *Computer Integrated Manufacturing Systems*, 1998, vol. 11, no. 4, pp. 309–330. 103

[21] Z. Yao and S. K. Gupta, "Cutter path generation for 2.5D milling by combining multiple different cutter path patterns," in *International Journal of Production Research*. Taylor & Francis Ltd, June 2004, vol. 42, no. 11. 103

[22] M. Held, "Vroni: An engineering approach to the reliable and efficient computation of voronoi diagrams of points and line segments," in *Computational Geometry: Theory and Application*, vol. 18, no. 2, pp. 95–123. 103

[23] June 2006, http://www.cosy.sbg.ac.at/~held/projects/vroni/vroni.html. 103

[24] D. R. Nelson, D. B. Barber, T. W. McLain, and R. W. Beard, "Vector field path following for miniature air vehicles," in *IEEE Transactions on Robotics and Automation*, Dec. 2005. 104

Appendix A

Target Differentiation Code

A.1 Target Center Calculation

This code calculates the target center estimate, **C**, using the contacts in a contact set. If there are less than four contacts in the set, then a zero-altitude estimate is calculated. If there are at least four contacts in the set, then the error minimzation approach is used to find the 3D target center. The function *compute_target_center()* calls the function *calcError()* to find the total error between all the contacts in the target set, and a given test point. This function is not shown, but is the same calculation as shown in Eq.(2.14). The test point is then moved to the position with least error. The iteration starts off with a large step size, and then reduces its step size near the local minimum for accuracy.

A.2 Contact Assignment

The code for contact assignment is only pseudo-code due to its length. It assigns all of the contacts in an image set to the closest target as long as no other contact is closer to that target. If no nearby target estimates exist, then a new one is created.

A.3 Azimuth and Elevation Calibration

This code calculates the camera azimuth and elevation angles that minimize the total distance between all of the contacts in all target sets, and a set of previously-known targets called "loaded_targets". The function *contact_distance()*, called from *calc_avg_distance()*, is the same calculation shown in Eq.(2.13).

```
void target::compute_target_center()
{
  if(m_num_contacts < 4) //If there are less than 4 contacts, then calcluate the new target center
                         //using the zero altitude estimate.
  {
    m_target_location = Point3D(0,0,0);
    for(vector<contact>::iterator itr=contacts.begin(); itr!=contacts.end() ; itr++)
    {
      Point3D origin = itr->ray_origin();
      Point3D direction = itr->ray_direction();
      m_target_location.x+= origin.x - (origin.z/direction.z)*direction.x;
      m_target_location.y+= origin.y - (origin.z/direction.z)*direction.y;
    }
    m_target_location = m_target_location / m_num_contacts;
  }
  else //If there are 4 or more contacts, then calculate a 3D target estimate.
  {
    bool minimizing_error = true;
    double inc = 10.0;
    Point3D one_old_target_loc = Point3D(0,0,0);
    Point3D two_old_target_loc = Point3D(0,0,0);

    while(minimizing_error) //Iterates to find the 3D location that corresponds to the least error.
    {
      double x_increment, y_increment, z_increment;
      double current_error = calcError(Point3D(m_target_location.x,m_target_location.y,m_target_location.z));
      double x_pos_error = calcError(Point3D(m_target_location.x+inc,m_target_location.y,m_target_location.z));
      double x_neg_error = calcError(Point3D(m_target_location.x-inc,m_target_location.y,m_target_location.z));
      double y_pos_error = calcError(Point3D(m_target_location.x,m_target_location.y+inc,m_target_location.z));
      double y_neg_error = calcError(Point3D(m_target_location.x,m_target_location.y-inc,m_target_location.z));
      double z_pos_error = calcError(Point3D(m_target_location.x,m_target_location.y,m_target_location.z+inc));
      double z_neg_error = calcError(Point3D(m_target_location.x,m_target_location.y,m_target_location.z-inc));

      if(x_pos_error >= current_error && x_neg_error >= current_error)
        x_increment = 0;
      else if(x_pos_error >= current_error)
        x_increment = -inc;
      else
        x_increment = inc;

      if(y_pos_error >= current_error && y_neg_error >= current_error)
        y_increment = 0;
      else if(y_pos_error >= current_error)
        y_increment = -inc;
      else
        y_increment = inc;

      if(z_pos_error >= current_error && z_neg_error >= current_error)
        z_increment = 0;
      else if(z_pos_error >= current_error)
        z_increment = -inc;
      else
        z_increment = inc;

      m_target_location = Point3D(m_target_location.x+x_increment,
                                  m_target_location.y+y_increment,
                                  m_target_location.z+z_increment);

      //Breaks the loop if the algorithm settles on a circular reference.
      if(m_target_location == two_old_target_loc)
      {
        x_increment = 0;
        y_increment = 0;
        z_increment = 0;
      }

      if(x_increment == 0 && y_increment == 0 && z_increment == 0 && inc == 10.0)
        inc = 1.0;  //Initiate iteration with a smaller increment.
      if(x_increment == 0 && y_increment == 0 && z_increment == 0 && inc == 1.0)
        inc = .1;  //If found target location to within 1.0 meters, then iterate with a smaller increment.
      else if(x_increment == 0 && y_increment == 0 && z_increment == 0 && inc == .1)
        inc = .01;
      else if(x_increment == 0 && y_increment == 0 && z_increment == 0 && inc == .01)
        minimizing_error = false;  //If found target location to within .01 meters, then end.

      two_old_target_loc = one_old_target_loc;
      one_old_target_loc = m_target_location;
    }
  }
}
```

```
for(i=1 ; i<=num_contacts ; i++)
{
  assign_contact_to_target(contact[i],0, 30);
}

assign_contact_to_target(current_contact, threshold_dist, d_max)
{
  Find the closest target to the current_contact where
    (distance(closest_target,current_contact) > threshold_dist).
  Name that target "selected_target".

  if(a target was selected)
  {
    if(distance(selected_target,current_contact) < d_max)
    {
      if(selected_target is not already assigned)
      {
        Add the current_contact to selected_target.
        Mark the selected_target as assigned.
      }
      else //if that target is already assigned
      {
        if(the current_contact is closer to the selected_target center than the last contact that was added)
        {
          Remove the last contact that was added from selected_target, and name it "outplaced_contact".
          Add the current_contact to the selected_target.
          assign_contact_to_target (outplaced_contact, distance(selected_target,current_contact, d_max))
        }
        else //if the current_contact is further from it.
        {
          assign_contact_to_target (current_contact, distance(selected_target,current_contact, d_max))
        }
      }
    }
    else //if the contact is not close enough to the target.
    {
      assign_contact_to_target (current_contact, distance(selected_target,current_contact, d_max))
    }
  }
  else //if a target was not selected
  {
    Create a new target, and add the current contact.
  }
}
```

```
void calibrator::calc_azel()
{
  bool minimizing_error = true;
  float inc = (float)0.1;
  float one_old_AZ = 0;
  float two_old_AZ = 0;
  float one_old_EL = 0;
  float two_old_EL = 0;
  float avg_dist = 0;

  while(minimizing_error) //Iterates to find the azimuth and elevation that minimize the average
  {                       //distance between each contact and nearest known target.

    float AZ_inc, EL_inc;
    avg_dist = calc_avg_distance(azi,ele,f_x,f_y,k3,k4);
    float distAZpos = calc_avg_distance(azi + inc, ele,f_x,f_y,k3,k4);
    float distAZneg = calc_avg_distance(azi - inc, ele,f_x,f_y,k3,k4);
    float distELpos = calc_avg_distance(azi, ele + inc,f_x,f_y,k3,k4);
    float distELneg = calc_avg_distance(azi, ele - inc,f_x,f_y,k3,k4);

    if(distAZpos >= avg_dist && distAZneg >= avg_dist) AZ_inc = 0;
    else if(distAZpos >= avg_dist) AZ_inc = -inc;
    else AZ_inc = inc;

    if(distELpos >= avg_dist && distELneg >= avg_dist) EL_inc = 0;
    else if(distELpos >= avg_dist) EL_inc = -inc;
    else EL_inc = inc;

    azi += AZ_inc;
    ele += EL_inc;

    //Breaks the loop if algorithm settles on circular reference.
    if((two_old_AZ == azi && two_old_EL == ele) ||
       (one_old_AZ == azi && one_old_EL == ele))
    {
      AZ_inc = 0;
      EL_inc = 0;
    }

    //If the minimum average distance is found to within .01 radians, then use a smaller increment.
    if(AZ_inc == 0 && EL_inc == 0 && inc == (float).1)
      inc = (float).01;
    else if(AZ_inc == 0 && EL_inc == 0 && inc == (float).01)
      inc = (float).001;
    else if(AZ_inc == 0 && EL_inc == 0 && inc == (float).001)
      inc = (float).0001;
    else if(AZ_inc == 0 && EL_inc == 0 && inc == (float).0001)
      minimizing_error = false;
    //If the minimum average distance is found to within .0001 radians, then end.

    two_old_AZ = one_old_AZ;
    two_old_EL = one_old_EL;
    one_old_AZ = azi;
    one_old_EL = ele;
  }
}
```

```
float calibrator::calc_avg_distance(float az, float el, float fx, float fy, float k_3, float k_4 )
{
  //Re-calculate all contacts using the new azimuth and elevation.
  for(vector<contact>::iterator itr=all_contacts.begin() ; itr!=all_contacts.end() ; itr++)
  {
    Point3D dir1 = itr->ray_direction();

    Point3D dir = calculated_targets.get_contact_direction(itr->plane_attitude(), az, el, fx, fy,
                                                           k_3, k_4, itr->pixel_x(), itr->pixel_y());
    itr->set_contact_data(itr->ray_origin(), dir, itr->plane_attitude(), itr->pixel_x(), itr->pixel_y());
  }

  double summed_distances = 0;
  //Sum the distances between each contact and the closest known target.
  for(vector<contact>::iterator itr1=all_contacts.begin() ; itr1!=all_contacts.end() ; itr1++)
  {
    double closest_distance = 99999.999;
    for(vector<Point3D>::iterator itr2=loaded_targets.begin() ; itr2!=loaded_targets.end() ; itr2++)
    {
      double current_distance = itr1->contact_distance(*itr2);
      if(current_distance < closest_distance) closest_distance = current_distance;
    }

    summed_distances += closest_distance;
  }

  return (float)summed_distances/all_contacts.size();
}
```