Dynamic Sonar Perception

by

Richard J. Rikoski

S.M., Ocean Engineering
Massachusetts Institute of Technology, 2001
B.S., Mechanical Engineering and Economics
Carnegie Mellon University, 1998

Submitted to the Department of Ocean Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Marine Robotics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2003

Author ....
Department of Ocean Engineering
June 1, 2003

Certified by ....
John J. Leonard
Associate Professor of Ocean Engineering
Thesis Supervisor

Accepted by ....
Professor Michael S. Triantafyllou
Professor of Ocean Engineering
Chairman, Departmental Committee on Graduate Students

Dynamic Sonar Perception

by

Richard J. Rikoski

Submitted to the Department of Ocean Engineering
on June 1, 2003, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Marine Robotics

Abstract

Reliable sonar perception is a prerequisite of marine robot feature-based navigation. The robot must be able to track, model, map, and recognize aspects of the underwater landscape without *a priori* knowledge. This thesis explores the tracking and mapping problems from the standpoint of observability. The first part of the thesis addresses observability in mapping and navigation. Features are often only partially observable from a single vantage point; consequently, they must be mapped from multiple vantage points. Measurement/feature correspondences may only be observable after a lag, and feature updates must occur after a delay. A framework is developed to incorporate temporally separated measurements such that the relevant quantities are observable. The second part of the thesis addresses observability in tracking. Although there may be insufficient information from a single measurement to estimate the state of a target, there may be enough information to observe correspondences. The minimum information necessary for a dynamic observer to track locally curved targets is derived, and the computational complexity is determined as a function of sonar design, robot dynamics, and sonar configuration. Experimental results demonstrating concurrent mapping and localization (CML) using this approach to early sonar perception are presented, including results from an ocean autonomous underwater vehicle (AUV) using a synthetic aperture sonar at the GOATS 2002 experiment in Italy.

Thesis Supervisor: John J. Leonard
Title: Associate Professor of Ocean Engineering

Acknowledgments

In appreciation of their efforts over my graduate career, I would like to suggest that the non-dimensional numbers in my thesis be named as follows:

N_1 = the Leonard number
N_3 = the Henrik number (Schmidt was already taken)
N_4 = the Kuc number
N_5 = the Choset number
N_6 = the Xanthos number
N_7 = the Kimball number
N_8 = the Damus number

N_2 will be left to the reader to name.

Of course, none of this would be possible without the other members of the marine robotics laboratory, including Jacob Feder, Tom Fulton, Chris Cassidy, Sheri Chang, John Fenwick, Mike Slavik, Jay Dryer, and Paul Newman. And of course, thanks to my family.

This research has been funded in part by NSF Career Award BES-9733040, the MIT Sea Grant College Program under grant NA86RG0074, and the Office of Naval Research under grants N00014-97-0202 and N00014-02-C-0210.

Contents

List of Figures

List of Tables

Nomenclature

A = amplitude, ocean wave

$\mathbf{A}$ = feature covariance at initialization

$\mathbf{B}$ = feature/map covariance at initialization

c = speed of sound, approximately 1500 $\frac{m}{s}$ in water

D = spacing between elements for a binaural sonar

f = frequency, of sound

f_s = sampling rate, i.e. the rate at which the sonar transmits

f_d = Doppler shift

$\mathbf{f}(\cdot)$ = vehicle dynamic model

$\mathbf{F_x}$ = vehicle dynamic model Jacobian

g = gravitation acceleration, 9.8 $\frac{m}{s^2}$

$\mathbf{g}(\cdot)$ = feature initialization model

$\mathbf{G_x}$ = feature initialization Jacobian

h = water depth

$\mathbf{h}(\cdot)$ = measurement model

$\mathbf{H_x}$ = measurement Jacobian

k = time index

k = wave number (acoustic wave)

K = wave number (ocean wave)

L = array length

m = measurement degrees of freedom

n = feature degrees of freedom

N_1 = non-dimensional number describing contribution of velocity to range rate

N_2 = contribution of velocity to range second derivative

N_3 = contribution of yaw rate to range second derivative

N_4 = contribution of acceleration to range second derivative

N_5 = contribution of robot velocity to target bearing rate

N_6 = contribution of robot yaw to target bearing rate

N_7 = contribution of deep water waves to range rate

N_8 = contribution of shallow water waves to range rate

p = roll rate

p_a = apparent roll rate

$\mathbf{P}$ = state covariance

$\mathbf{P_{ff}}$ = covariance of all features

$\mathbf{P_{rf}}$ = robot/feature covariance

$\mathbf{P_{rr}}$ = covariance for all robot states

$\mathbf{P_{r_1r_1}}$ = covariance for all of robot 1's states

$\mathbf{P_{r_1r_1}}(k|k)$ = robot 1's covariance at time k given information through time k

q = pitch rate

q_a = apparent pitch rate

$\mathbf{Q}$ = process noise covariance

r = yaw rate

r_a = apparent yaw rate

r_m = measured range to a target

r_c = range to the center of curvature of a target

$\mathbf{R}$ = measurement noise covariance

T_l = TOF, measured by the left transducer of a binaural sonar

T_r = TOF, measured by the right transducer of a binaural sonar

TOF = time of flight

u = surge, or forward velocity

u_a = apparent surge velocity

v = sway, or side velocity

v_a = apparent sway velocity

$\mathbf{v_a}$ = apparent velocities

$\dot{\mathbf{v}}_\mathbf{a}$ = apparent accelerations

V = vehicle speed

$\dot{V}$ = vehicle acceleration

w = heave, or vertical velocity

w_a = apparent heave velocity

x = globally referenced x position

$\dot{x}$ = globally referenced x velocity

$\ddot{x}$ = globally referenced x acceleration

x_c = x component of center of curvature of target in body coordinates

$\mathbf{x_c}$ = center of curvature of target in body coordinates

$\dot{\mathbf{x}}_\mathbf{c}$ = velocity of center of curvature of target in body coordinates

$\ddot{\mathbf{x}}_\mathbf{c}$ = acceleration of center of curvature of target in body coordinates

$\mathbf{x}$ = true state

$\mathbf{x}[k]$ = true state at time k

$\hat{\mathbf{x}}$ = estimated state

$\hat{\mathbf{x}}[k|k]$ = estimated state at time k given information through time k

$\hat{\mathbf{x}}[k+1|k]$ = estimate at time $k+1$ given information through time k (prediction)

$\hat{\mathbf{x}}[k-1|k]$ = estimate at time $k-1$ given information through time k (smoothing)

$\mathbf{x}_r$ = robot's state

$\mathbf{x_r}$ = set of robot states

$\mathbf{x_{r_1}}$ = for multiple robots, robot 1's trajectory

$\mathbf{x}_{r_1}(k)$ = robot 1's state at time k

$\mathbf{x_f}$ = set of all feature states

$\mathbf{x_{f_1}}$ = feature 1's trajectory, for a dynamic feature

$\mathbf{x}_{f_1}(k)$ = feature 1's state at time k

y = globally referenced y position

$\dot{y}$ = globally referenced y velocity

$\ddot{y}$ = globally referenced y acceleration

y_c = y component of center of curvature of target in body coordinates

z_c = z component of center of curvature of target in body coordinates

$\mathbf{Z}^k$ = set of all measurements made through time k

$\mathbf{z}[k]$ = set of measurements made at time k

$\mathbf{z_a}[k]$ = measurements from time k which have been associated with features

$\mathbf{z_{\neg a}}[k]$ = measurements from time k which have not been associated with features

$\mathbf{z}_{f_i}[k]$ = measurement at time k of feature i

$\mathbf{z}_m^{r_n}[k]$ = measurement of feature m by robot n and time k

λ = wavelength (acoustic wave)

Λ = wavelength (ocean wave)

η = ocean wave elevation

ν = innovation

ϕ = vertical angle of target

ρ = radius of curvature

σ_{xx} = variance of x

σ_{xy} = covariance of x and y

θ_b = sonar beam half angle

θ = target bearing

θ_r = vehicle heading

$\dot{\theta}_r$ = robot yaw rate, two dimensional

ω_a = apparent angular rate of robot

ω = angular frequency (acoustic wave)

Ω = angular frequency (ocean wave)

Υ = maximum angular rate of robot (for non-dimensional analysis)

Chapter 1

Introduction

> It looks like a torpedo, but this device doesn't explode. It searches for mines and other objects that do. The Woods Hole scientists who built it over the course of ten years dubbed their handiwork REMUS, for Remote Environmental Monitoring UnitS. In their first-ever use in hostile waters, the undersea drones were used as part of a team that helped to clear mines from the Iraqi port of Umm Qasr, according to the US Office of Naval Research. Their success allowed 232 tons of badly needed food, water, blankets, and other supplies to reach Iraqi civilians over the weekend. ... Before REMUS is put to work, two sound-emitting transponders are placed in nearby waters and their positions set by portable global navigation devices.
>
> "Undersea drones pull duty in Iraq hunting mines", Jack Coleman, *Cape Cod Times*, 2 April 2003.

1.1 Why Marine Robots?

Few environments justify mobile robots as well as the ocean. Ocean exploration is hazardous and expensive — robots costing hundreds of thousands of dollars can pay for themselves in weeks. This thesis investigates feature-based navigation in which the goal is to enable an autonomous underwater vehicle (AUV) to build a map of an unknown environment while concurrently using this map for navigation. This capability is anticipated to significantly increase the autonomy and reliability of AUV operations for a wide range of ocean applications.

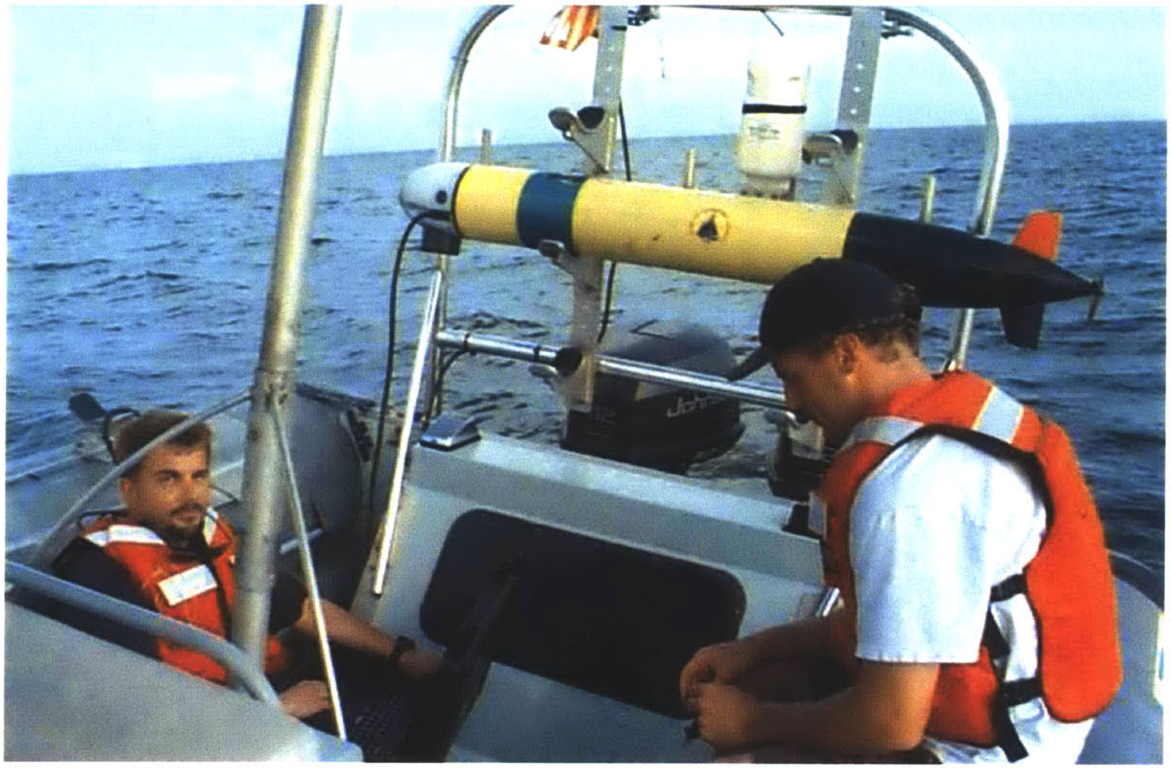


Figure 1-1: The REMUS AUV, developed by the Ocean Systems Laboratory of the Woods Hole Oceanographic Institution (Courtesy of Timothy Prestero).

As illustrated by the successful deployment of REMUS in the Second Gulf War, the field of AUV research has advanced dramatically over the last decade. Other successful recent AUVs include the Odyssey class of vehicles (Figure 1-2), developed by MIT Sea Grant, and the Autonomous Benthic Explorer (ABE), developed at the Deep Submergence Laboratory of the Woods Hole Oceanographic Institution (Figure 1-3). Ten years ago, these vehicles were being placed in the water for the first time [11, 95, 84]. Today, these AUVs are highly capable and reliable platforms, and many new applications are coming to fruition. Many difficult research issues, however, still remain.

Deep sea exploration is expensive and dangerous. Ocean vessels typically cost $25,000 per day to operate. Any device that can shorten the time needed to accomplish a mission will save money. Ocean exploration is dangerous. Manned submersibles are subjected to environmental extremes unseen in other applications and require expensive life support systems. Underwater construction using compression

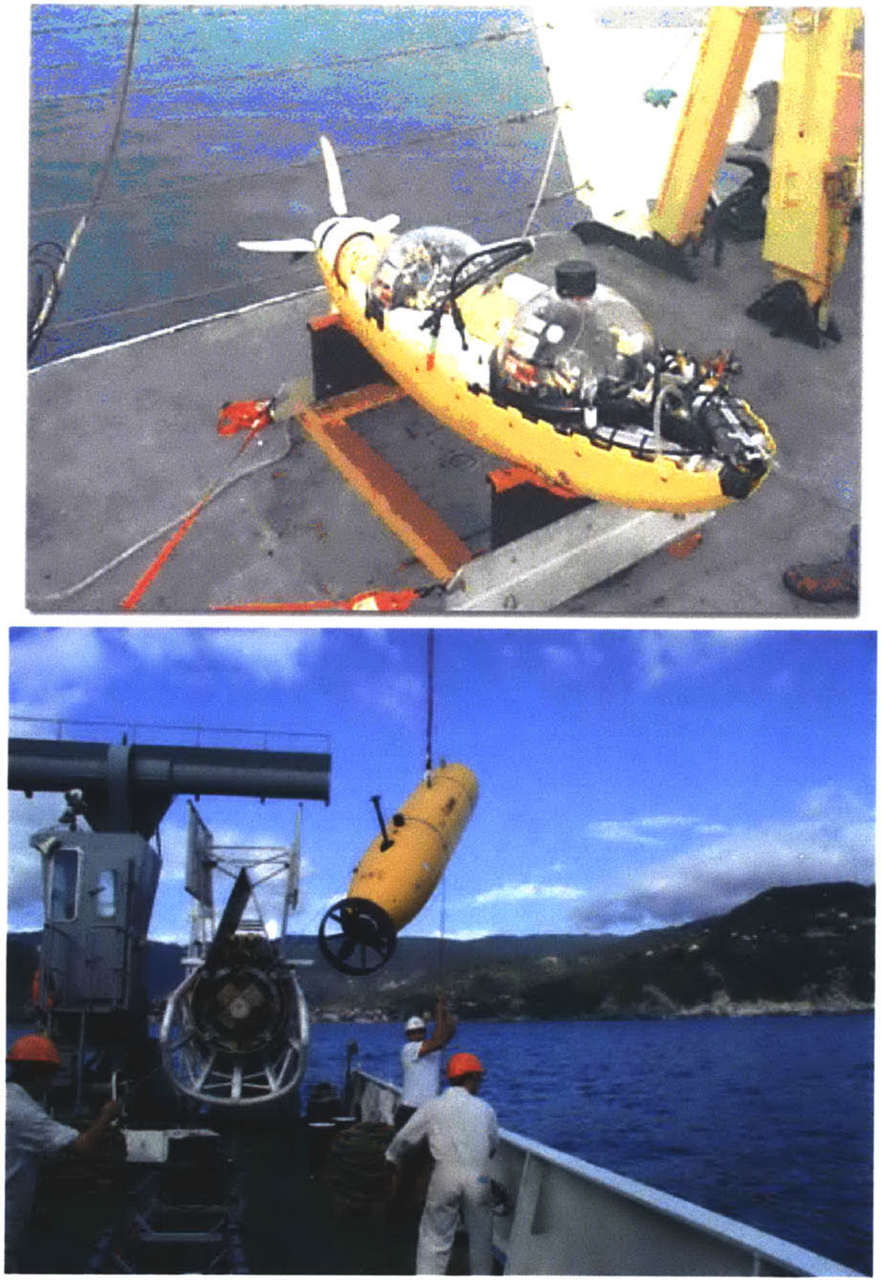


Figure 1-2: Top: MIT Odyssey IIb autonomous underwater vehicle with outer, free-flooding hull opened, exposing payload pressure spheres. Bottom: MIT Odyssey III autonomous underwater vehicle shown during the GOATS-2002 experiment (NATO SACLANT Research Centre).

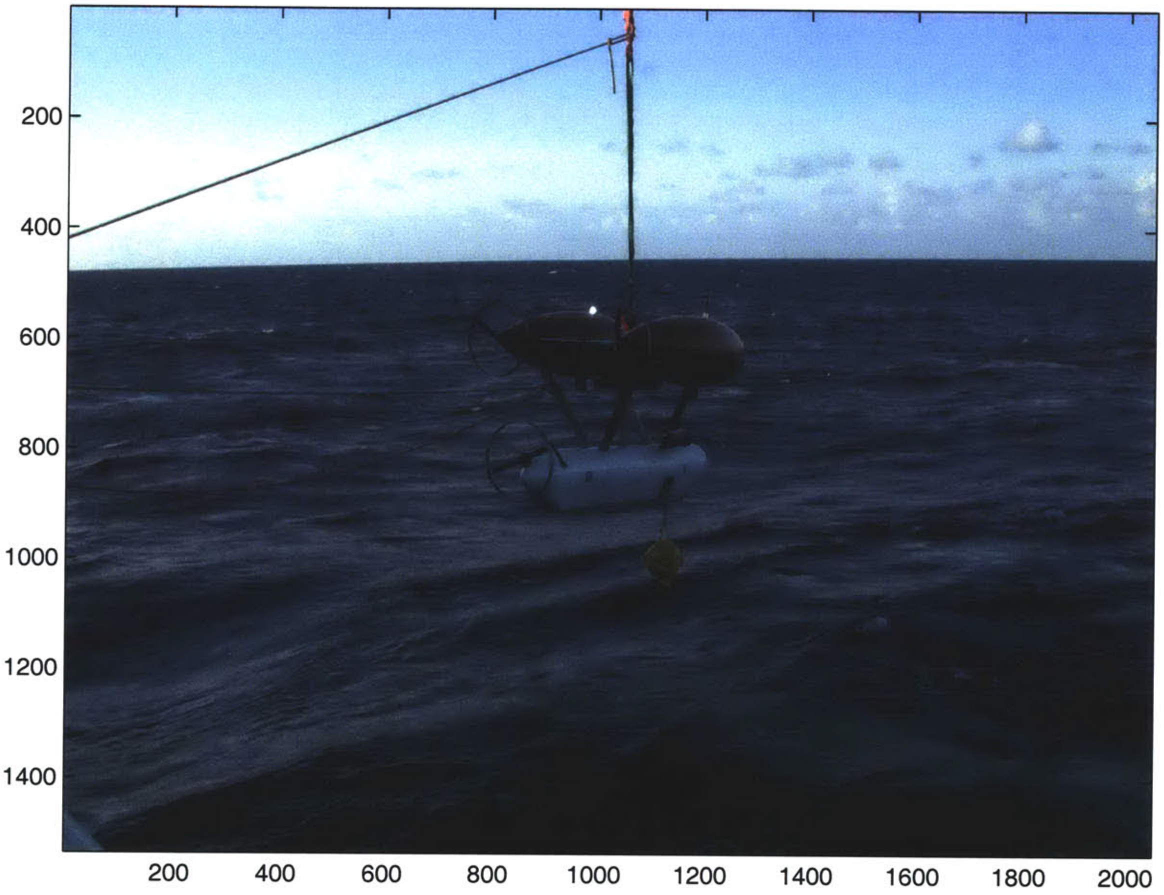


Figure 1-3: The Autonomous Benthic Explorer (Courtesy of Albert Bradley).

diving and exotic gas mixtures is risky. The use of robots potentially allows us to avoid these risks and expenses.

For instance, towed sonar sleds are often used for deep sea surveying [45]. To tow a sled at a depth of several kilometers, a significantly larger amount of cable needs to be used since drag dominates the tension. While the drag of the sled is insignificant, the drag of the cable dominates. A ship dragging a ten kilometer cable can take up to twelve hours to turn around, at a cost of roughly $10,000 per turn. An untethered survey robot can turn around in a fraction of a minute, and many robots can be deployed by a single ship.

Oceanographers are interested in the temperature and salinity of ocean water as a function of depth. Seawater density drives ocean flows. Assuming that the density of seawater will vary inversely with temperature is often inadequate. As it warms, seawater expands, reducing its density. However, at the surface evaporation takes place, increasing the salinity and density. These two conflicting mechanisms make ocean flows unstable [97]. To estimate ocean flows, oceanographers gather conductivity, temperature, and depth (CTD) data. (The salinity is determined by the conductivity.) A typical ocean survey uses a system of cannisters lowered on a cable. At each relevant depth, a cannister opens and gathers water. This sort of survey typically takes four hours.

The *Flying Fish* [16] AUV was developed to expedite such surveys. A streamlined vehicle without a propeller, the *Flying Fish* is driven entirely by buoyancy. Initially it has a chunk of lead in the nose and dives at $9\frac{m}{s}$, sampling water on the way down. Once the vehicle reaches its bottom depth, the lead is released, and it surfaces at $9\frac{m}{s}$. Using a unique recovering mechanism (it is snagged by a second robot), this AUV's entire mission can be completed in forty minutes.

Manned exploration of the deep sea is expensive in part due to the necessity of life support systems. First, the pressure makes the environment extremely challenging. In space, the difference between the inside and outside pressure is one atmosphere, or roughly 100kPa. Moreover, the pressure vessels are typically in tension. Underwater, a "full ocean depth" vehicle has to withstand a compressive load of 600 atmospheres, or roughly 60MPa. By convention, 6000m is considered full ocean depth, and a vehicle designed for 6000m will be able to access the vast majority of the ocean. However, when the *Trieste* bottomed out in the Marianas trench, it was closer to 11,000m, or a pressure of 110MPa. Since these are compressive loads, pressures that a vessel might withstand in tension can cause a buckling failure. To avoid buckling, spherical pressure vessels are often used. The manned submersible *Alvin* has a 2m diameter, four inch thick titanium sphere which has a limited fatigue life.

Near hydrothermal vents, water temperatures are high enough that, in the absence of light, the rocks glow red hot. The water exiting the vents is hot enough to melt the windows of *Alvin*.

A submersible can become entangled in seaweed, trapped under an overhang, disabled due to a battery failure, or lose its life support systems. Once, *Alvin* came under attack from a swordfish. The safety record of manned ocean exploration is a testament to the engineers who design and maintain the craft. However, this comes at a price, and clearly there is a strong rationale for unmanned exploration with robots

Even at a cost of hundreds of thousands of dollars, a robot that can save the user $10,000 every time it turns around can be economically justified. A robot that saves thousands of dollars per CTD survey and dive multiple times per day can be economically justified. A robot which spares the user the costs associated with saving or losing human lives can also be justified. A tremendous amount of money is spent on ocean exploration, and what would seem an outrageous sum for a land robot is a sound investment in the marine community.

1.2 Enabling Technologies

There are a multitude of problems that must be solved for marine robots to become pervasive. Five key marine robotics research areas are power, locomotion, navigation, perception, and cognition.

Power consumption and energy source design are critical to any deployment. Robots should be designed to minimize the hotel load (the non-propulsive power draw) and travel at a velocity that optimizes range [15]. Batteries or alternative energy sources such as fuel cells need to be developed to provide the greatest neutrally buoyant energy density [79, 1].

Locomotion is another serious design issue. The dynamics of a robot depend substantially on the design goals. An AUV such as REMUS [89] that is designed to perform sidescan sonar surveys could have a simple, streamlined design. An ROV like JASON [7] that manipulates objects, such as amphora, might be better designed as a hovering vehicle.

Navigation is a chronic underwater problem. GPS is unavailable because water quickly attenuates electromagnetic signals. Inertial navigation systems drift over time. Dead reckoning leads to unbounded error. The best solution is to use a calibrated network of underwater beacons. This is how REMUS and most other AUVs navigate today. The use of beacons increases the cost and reduces the flexibility of marine robot deployments. The ultimate goal for our research is for robots to navigate relative to distinct aspects of terrain.

Navigating relative to terrain will require substantial advances in the fourth key marine robotics research area: perception. Perception is the process of transforming sensory data into higher level representations, or constructs. Simple examples of perception include triggering on signals that exceed thresholds and rejecting spurious measurements. High-level perception typically concerns object recognition and may involve comparing high-level representations to other high-level representations. This thesis will examine methods for processing sensory data towards the ultimate goal of terrain based navigation.

Cognition is the final, and perhaps least explored, marine robotics problem. Given that a robot has the perceptive capacity to develop an understanding of its surroundings, it would be desirable for the robot to autonomously achieve some broader goal. Most marine robots are either directly controlled by people (such as ROVs) or are preprogrammed to follow specific paths (survey vehicles such as the Odysseys, Ocean Explorer, ABE, and REMUS). With the development of cognition, it will become feasible to consider a richer variety of robot-feature interactions. We could consider having robots perform minor tasks such as picking things up or repairing or building simple structures.

The generic oceanographic array technology sonar (GOATS) concept, illustrated in Figure 1-4 [83], provides a compelling vision for AUV technology development. The overall goal is to enable networks of many AUVs to rapidly survey large areas and to detect and classify objects in real time.

1.3 Thesis Goals

This thesis concerns feature-based navigation. At the most primitive level, the feature-based navigation problem can be broken down into: 1) finding features, 2) building a map of features, and 3) recognizing features. Finding and recognizing features using sonar has received little attention in the robotics literature compared to the equivalent problem in vision. The first part of the thesis develops a framework for mapping partially observable features. The second part of the thesis develops methods for early tracking of features. Object recognition is left for future work. For the purposes of this thesis, grouping measurements based only on measurements, without knowing what the target is, over extremely short time scales, will be considered early tracking or correspondence. Recognizing targets after prolonged periods without observations will be considered object recognition and will be left as future work. This thesis will be restricted to very short time scales on the order of ten seconds.

We want robots to navigate. We want to send them places, have them get there, and have them return successfully. We would like them to explore environments, find noteworthy things, come back, and either tell us how to get to those things or return to them themselves. On smaller scales, we would like for robots to develop an understanding of an environment so they can maneuver around obstacles, manipulate objects, and perform other high-level robot/object tasks.

Some robots possess some of these capabilities. Using GPS or inertial navigation, robots have been able to visit predefined areas. Using homing beacons or predefined targets, some robots have returned to areas and docked. Obstacle avoidance tech-

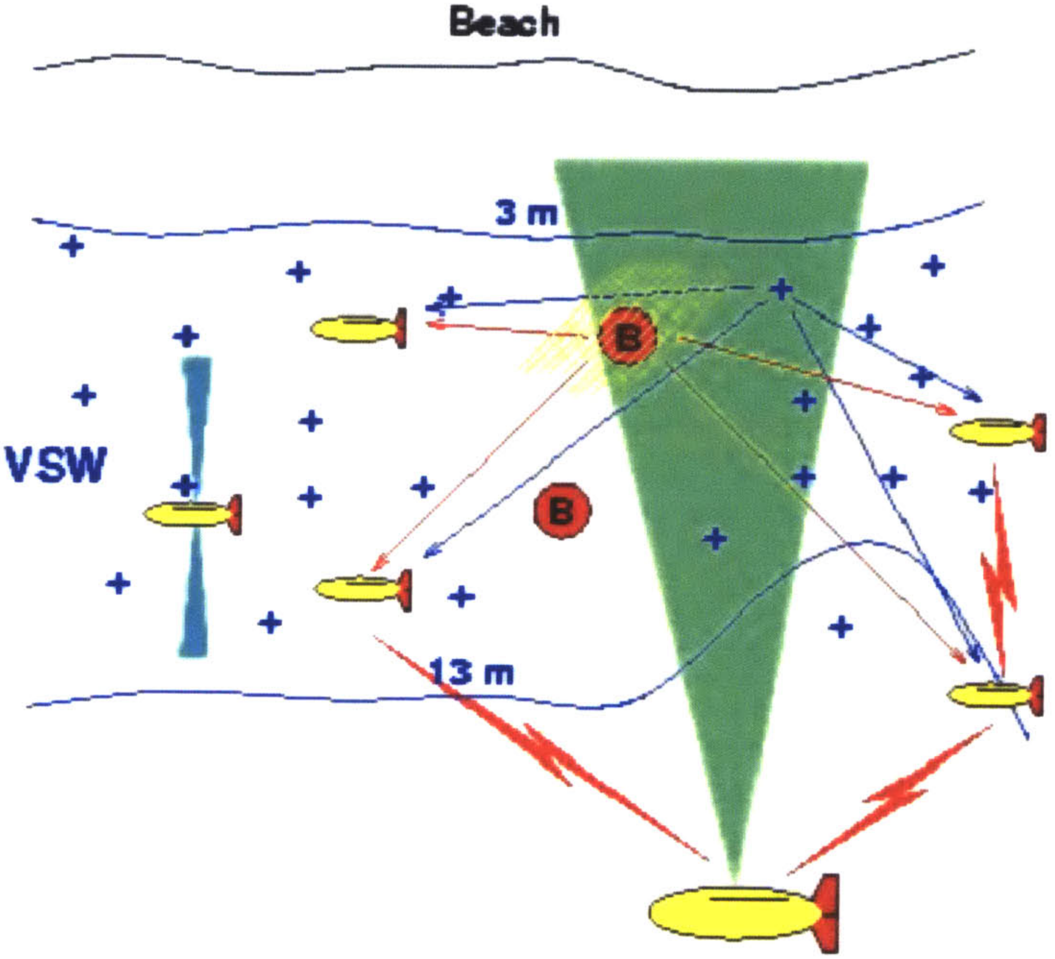


Figure 1-4: The generic oceanographic array technology sonar (GOATS) concept. The vision of GOATS is to enable a network of multiple AUVs to perform rapid detection and classification of proud and buried targets over large areas using multistatic acoustic sensing.

niques have been developed for collision avoidance. Industrial robots have performed manipulation tasks in highly controlled environments, but what we want goes further. We want robots to enter environments and use their sensors to develop true situational awareness. We want them to do this without external beacons or homunculi. We want feature-based navigation.

The goal of feature based navigation just leads to more questions. What does it mean for a robot to be situationally aware? How should it use its sensors to develop an understanding of its environment? How does this depend on the environment? Are these goals achievable?

For the purposes of this thesis, we will restrict situational awareness to simply understanding environmental geometry. In other words, the robot has to understand where things are well enough to get around. This is still very open ended, as no specific environmental representation has yet been chosen.

The perception problem, or how the robot should use its sensors to build an understanding of the world, is arguably the toughest problem in robotics. The world is ambiguous, partially observable, cluttered, complicated, and dynamic. What is required of perception, and hence the form of the instantiation, depends significantly on the selected world representation.

Conventional undersea sonar data interpretation is based primarily on an imaging paradigm. The goal is to utilize as narrow beams as possible to attempt to create sharp pictures that a human operator can readily interpret. For example, Figure 1-5 shows a 500 kHz sidescan sonar image for a set of undersea targets observed during the GOATS 2002 experiment. Figure 1-6 shows data from a 675 kHz Imagenex mechanically scanned sonar taken in a tank. Each of these sensors has a beamwidth on the order of 1 degree. While these images display some acoustic artifacts, such as multipath in the tank sonar image, a well-trained human operator can clearly make some inferences about the structure of the scene being imaged.

Part of the rationale for the design of imaging sonars is to get as few pings as possible on a given target. Then, using many pings obtained systematically over a given area, one can create a picture. However, if we look at the design of biosonar systems,

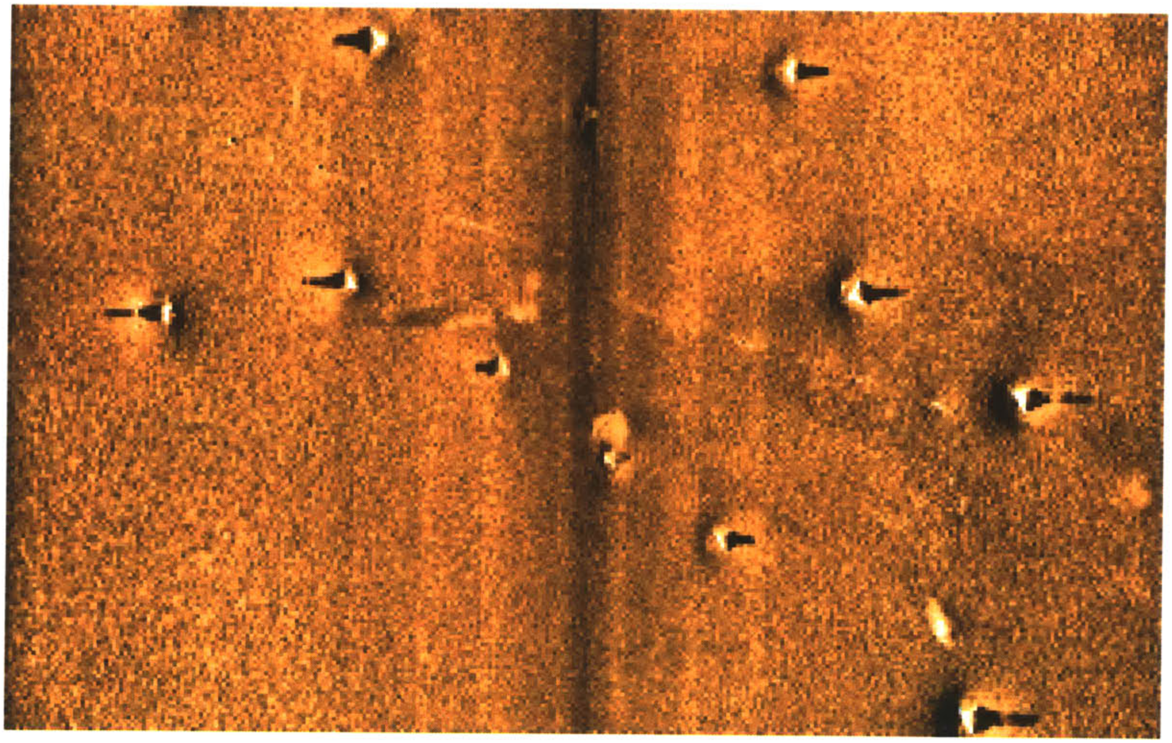
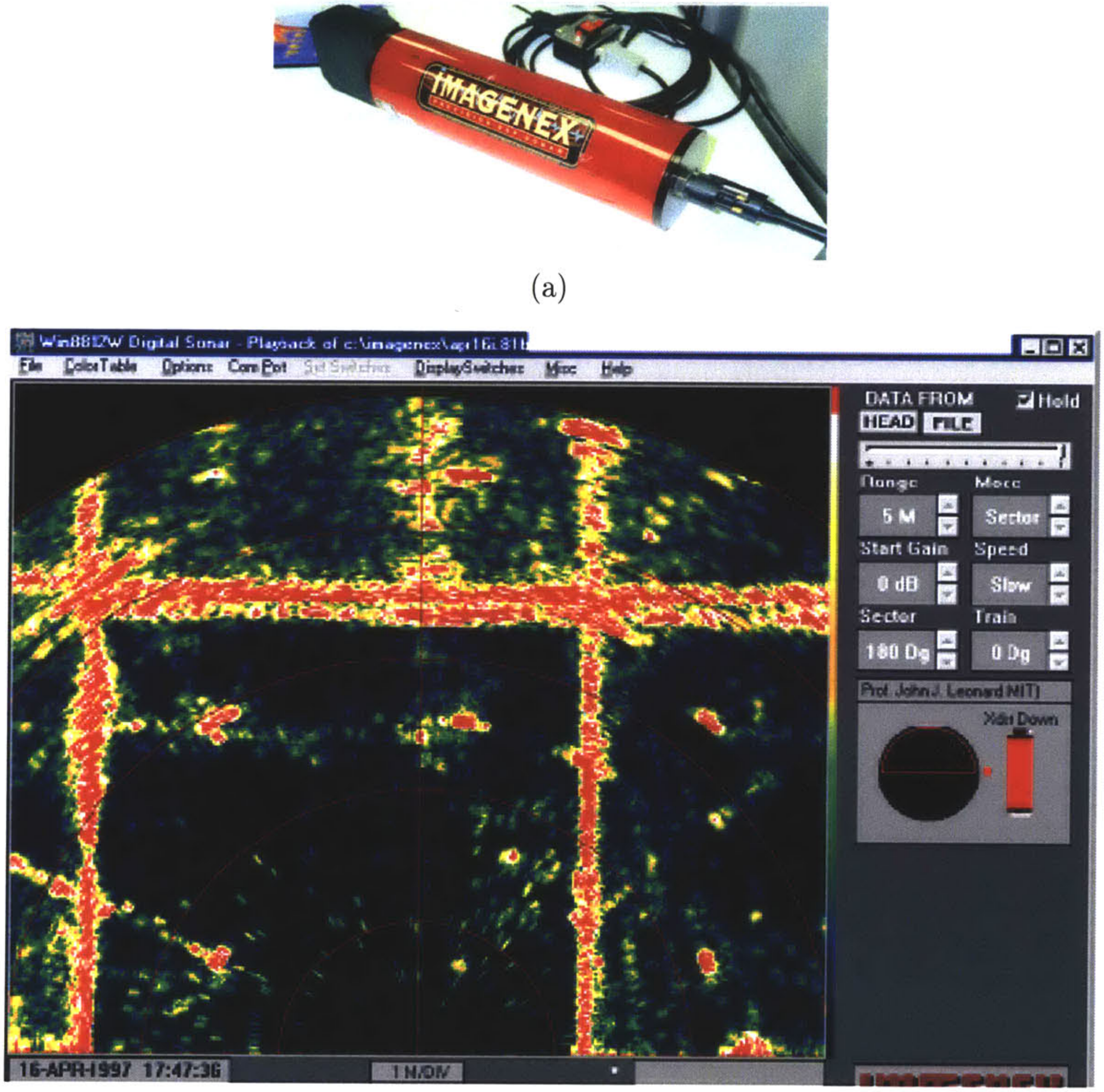


Figure 1-5: A sidescan image of targets observed during GOATS 2002 using a Klein DS5000 500 kHz sonar system. The narrow beams of the Klein sonar provide images in which the shadows cast by objects are clearly visible. (Image courtesy of GESMA.)



(a)

Win881ZW Digital Sonar - Playback of c:\imagenex\apr16L81I

☑ Hold

Range: 5 M

Mode: Sector

Start Gain: 0 dB

Speed: Slow

Sector: 180 Dg

Train: 0 Dg

Prof. John J. Leonard MIT)

Xdcr Down

16-APR-1997 17:47:36

(b)

Figure 1-6: (a) Imagenex 675 kHz mechanically scanned sonar. (b) Sonar data acquired with this sensor in a testing tank. The tank is roughly three by three meters in dimensions with four cylindrical posts protruding upwards at several locations in the tank.

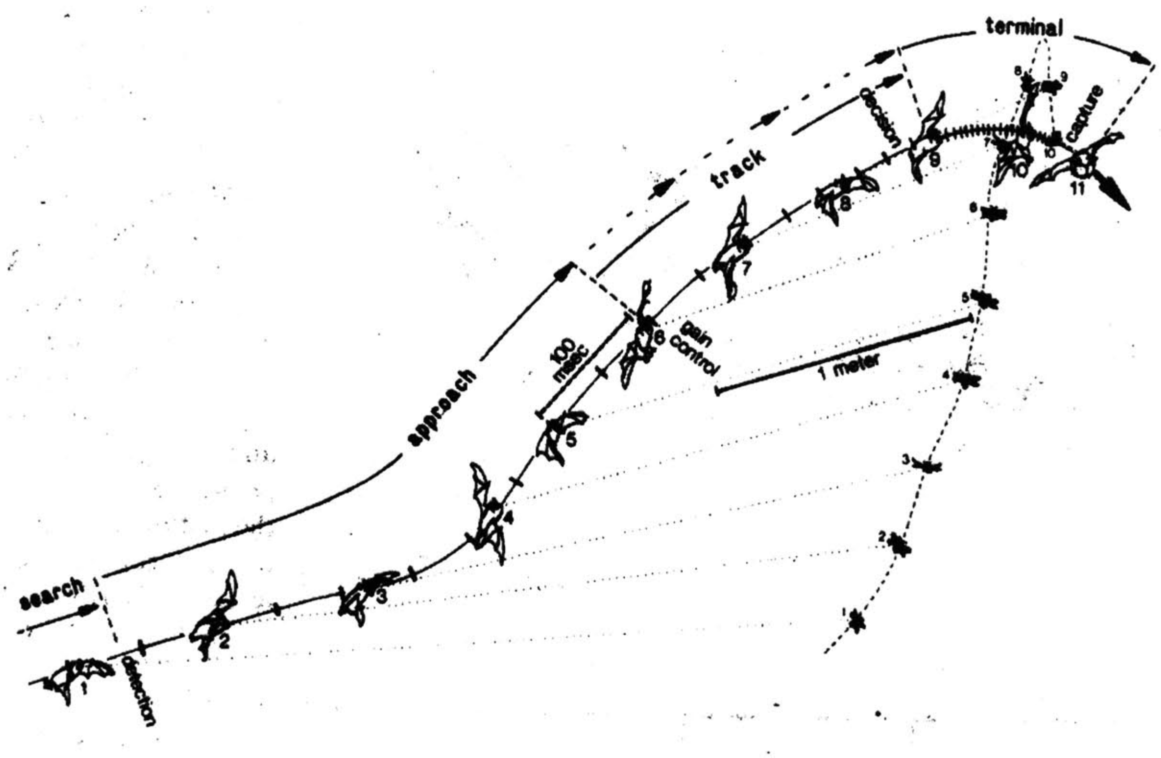


Figure 1-7: A bat chasing a moth (from [4]).

we see a divergence from the way that man-made sonar systems are developed.

A motivation for the approach that we follow in this thesis comes from the remarkable sonar capabilities of dolphins and bats [4, 2]. In contrast to man-made imaging sonars, bats and dolphins employ wide beams. They also utilize dynamic motion strategies when investigating and tracking targets. For example, Figure 1-7 shows a bat attempting to capture a moth. The motion of the bat clearly differs from the way in which a man-made sonar sensor is typically scanned through the environment. The question of how to exploit dynamic motion control for echolocation with a wide beam sonar has not received much attention in the the undersea robotics community. This is one of the key questions that we consider in this thesis. Can we use a wide beamwidth to advantage? Is it possible to maintain contact with features so that we can continually use them as navigational references?

1.4 Thesis Outline

This thesis has two parts. The first develops a framework for mapping and navigation, creating an explicit infrastructure for delayed perception and mapping of partially observable features. The second part investigates how to perform early sonar perception. By early sonar perception, we mean the initial tracking of features, prior to modeling and object recognition.

Chapter 2 reviews prior work in mobile robot navigation and mobile robot sonar usage. Chapter 3 develops stochastic mapping with working memory, a new framework for mapping and navigation in situations with partial observability. Chapter 4 develops an early correspondence technique based on the minimal information necessary to track locally curved objects. Chapter 5 presents results from the GOATS 2002 AUV experiment. Finally, Chapter 6 provides a summary of our contributions and makes suggestions for future research.

Chapter 2

Problem Formulation

This chapter formulates our approach to the problem of feature-based navigation using wide beam sonar. We begin by discussing the task of choosing a representation for navigation and mapping. We proceed to review the feature-based formulation of the concurrent mapping and localization (CML) problem, the approach that we follow due originally to Smith, Self, and Cheesemman [87]. After discussing some of the open issues in CML research, we proceed to formulate our approach to the sonar perception problem. We discuss relevant work in computer vision, such as optical flow, that inform our investigation of widebeam sonar perception for feature-based navigation. The principle of natural modes [81] serves as a guiding principle for developing percerptual models.

2.1 Representations for Navigation

It is necessary to identify and define the parameters of successful robot navigation. For instance, suppose we want to send a robot to an arbitrary location. How should it proceed, and how will it know that is has gotten there?

If the distance to be travelled is short, the robot could use dead reckoning [12]. By estimating its heading and velocity, the robot can estimate its position, maneuvering until it believes it is at the final location. Typically, when dead reckoning, the robot uses a model of its dynamics, its control inputs, and measurements such as heading a compass or gyro and velocity from a Doppler Velocity Log (DVL). Unfortunately, when the robot dead reckons, errors grow with distance travelled and time. An inertial navigation unit could be used to reduce the error growth rate [56], but inertial navigation also results in unbounded error growth.

For long transits, in which dead reckoning and inertial navigation are insufficient, the robot will need to use external information. We need to define what external information is required, and how the robot should use that information. This leads directly to the heart of the feature-based navigation problem; the choice of a representation. There are two major forms of feature based navigation: procedural navigation and globally referenced navigation.

If procedural navigation is used, the robot transits from place to place without knowing where it is globally (ie having its position defined in a coordinate system). Instead, it uses an ordered set of instructions. Navigation is decomposed into behaviors and cues. For instance, the following are directions, taken from an (uncorrected) e-mail for how to get to an MIT laboratory.

> NW13-220 You can get in either by showing your ID at the front door of the reactor and heading up the stairs then down the hall make a right, through the doors and my lab is right there, my office is inside or by going in NW14 up the stairs down the hall a bit first left through the red doors, all the way down the hall through another red door, then another, right down the hall first left and my office is right before you get to the grey doors.

Such instructions presuppose high-level object recognition such that one can recognize “the red doors” in order to turn left through them.

The advantage of this approach is that by decomposing navigation into paths and cues, if one can recognize the cues and stay on the paths, then navigation is fairly assured. Similar approaches have been used nautically. River navigators cannot stray from their path, ignoring obstacle avoidance issues (trees and sandbars can be particularly hazardous) a navigator only needs to recognize when they approach the appropriate city [26]. Navigation along coastlines proceeds similarly. Polynesian navigators [64] travelled between islands using "star paths". By steering at the location where a sequence of stars appeared on the horizon, they could maintain a constant heading and steer towards a distant island. For landfall, birds were observed at dawn and dusk. At dawn, birds would fly out from shore to feed, at dusk they would fly home. The heading of the birds defined a path to the island.

Consider a second example:

> **Hugh Durrant-Whyte:** Where's the bathroom?
> **John Leonard:** It's left, then another left, and on your left. Do you want Rick to show you?
> **Hugh Durrant-Whyte:** That's ok, I'll find it, it's left, left, and then left?
> (10 minutes pass)
> **Hugh Durrant-Whyte:** Now how do I get to the airport?

Clearly, Durrant-Whyte's trip to the bathroom takes advantage of substantial prior knowledge, but it is a very procedural trip. Leonard does not provide a globally referenced bathroom position; rather he provides a set of instructions. Given sufficient perceptive and cognitive capabilities, including some prior knowledge of hallways and bathrooms, one can parse the instructions and arrive at the desired destination. One might wonder about the phrasing of the questions. In the first, he asks "where", in the second, he asks "how". Perhaps for things that are very close, a vectorized approach is occasionally sufficient, but for distant objects, a procedural representation is necessary. Or perhaps we have over analyzed his diction.

Examples of procedural navigation in robotics include the behavior based navigation of Mataric and Brooks [18], the cognitive mapping of Endo and Arkin [31], the semantic spatial hierarchy of Kuipers [55], and the usage of generalized Voronoi graphs by Choset [24] .

Mataric and Brooks [18] developed a robot that navigated based on its reactions to the environment. The robot had behaviors such as obstacle avoidance and wall following. By establishing sequences of behaviors that resulted in the robot being in specific locations, a sort of robotic portolan [26] was established.

Similarly, Endo and Arkin [31] created a cognitive map, which includes both spatial and behavioral information. The goal was to create a representation that adequately represented the information needed to move between locations.

Choset [24] used a Generalized Voronoi Graph (GVG) for navigation. The GVG is essentially a set of points that are equidistant from the closest objects. In the planar case, a sort of lattice with curves is generated. The nodes of the GVG are of special importance, for they define locations where numerous objects are equidistant. Once a GVG is defined, localization can occur simply by knowing which nodes the robot is between.

Globally referenced navigation differs from procedural navigation by representing the robot position in terms of globally referenced coordinates. Such coordinates could be latitude and longitude or a locally referenced coordinate system. The robot maps terrain, and by reobserving terrain estimates its globally referenced position. Some of the key questions include how terrain should be modeled, how the terrain should be mapped, and when terrain is reobserved, how that information should be used to improve the robot's estimate of it's locatoin?

Hybrid metric/topological representations, consisting of a network of places or submaps, have been successfully exploited for mobile robot navigation and mapping by a variety of researchers including Kuipers [55, 54], Gutmann and Konolige [40], and Thrun [91, 94].

Arguably, one of the most successful feature based navigation approaches is the globally referenced grid based approach of Thrun [91]. Using a combination of occupancy grids and particle filters, Thrun estimates the true probability distribution of measurements, which is in turn used for navigation. However, discrete features are not represented, so it is unclear how this scales to higher level operations such as manipulation.

An alternative approach, termed "stochastic mapping" after a seminal article by Smith, Self, and Cheeseman [87], breaks the world down into discrete modeled features. By combining reobservations with an extended Kalman filter, measurements of objects are used for navigation. This is the approach that we follow in this thesis. In the next section, we review the stochastic mapping approach and illustrate it with a simple two-dimensional example. In the following section, we discuss some of the active research topics in the field of CML today, such as the map scaling problem and the data association problem.

2.2 Stochastic Mapping

Stochastic mapping is a feature-based, Kalman filter based approach to mapping and navigation relative to a global reference that was first published by Smith, Self, and Cheeseman [87] and Moutarlier and Chatila [70].

The stochastic map consists of a state vector $\mathbf{x}$ and the covariance $\mathbf{P}$. The state vector $\mathbf{x}[k|k]$ contains the robot state $\mathbf{x}_r[k|k]$ and the feature states $\mathbf{x_f}[k|k]$. We use the variable k to represent discrete timesteps.

$$\mathbf{x}[k|k] = \begin{bmatrix} \mathbf{x}_r[k|k] \\ \mathbf{x_f}[k|k] \end{bmatrix} \tag{2.1}$$

For a two dimensional example, a robot's state could be its x and y coordinates, its heading θ, and its velocity u

$$\mathbf{x}_r[k|k] = \begin{bmatrix} x_r \\ y_r \\ \theta_r \\ u \end{bmatrix}. \tag{2.2}$$

Two point features, $\mathbf{x}_{f_1}$ and $\mathbf{x}_{f_2}$, would be described by their respective coordinates

$$\mathbf{x_f}[k|k] = \begin{bmatrix} \mathbf{x}_{f_1}[k|k] \\ \mathbf{x}_{f_2}[k|k] \end{bmatrix} = \begin{bmatrix} x_{f_1} \\ y_{f_1} \\ x_{f_2} \\ y_{f_2} \end{bmatrix}. \tag{2.3}$$

In this case, $\mathbf{x_f}[k|k]$ is the vector of features, and $\mathbf{x}_{f_1}[k|k]$ and $\mathbf{x}_{f_2}[k|k]$ are the individual feature vectors.

The state vector describing the map would be

$$\mathbf{x}[k|k] = \begin{bmatrix} \mathbf{x}_r[k|k] \\ \mathbf{x}_\mathbf{f}[k|k] \end{bmatrix} = \begin{bmatrix} \mathbf{x}_r[k|k] \\ \mathbf{x}_{f_1}[k|k] \\ \mathbf{x}_{f_2}[k|k] \end{bmatrix} = \begin{bmatrix} x_r \\ y_r \\ \theta_r \\ u \\ x_{f_1} \\ y_{f_1} \\ x_{f_2} \\ y_{f_2} \end{bmatrix} \tag{2.4}$$

The covariance $\mathbf{P}$, which describes the uncertainty in the state vector, would be

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{rr} & \mathbf{P}_{r\mathbf{f}} \\ \mathbf{P}_{\mathbf{f}r} & \mathbf{P}_{\mathbf{ff}} \end{bmatrix}. \tag{2.5}$$

For the described two dimensional world, the covariance would be

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{rr} & \mathbf{P}_{r\mathbf{f}} \\ \mathbf{P}_{\mathbf{f}r} & \mathbf{P}_{\mathbf{ff}} \end{bmatrix} = \begin{bmatrix} \mathbf{P}_{rr} & \mathbf{P}_{rf_1} & \mathbf{P}_{rf_2} \\ \mathbf{P}_{f_1r} & \mathbf{P}_{f_1f_1} & \mathbf{P}_{f_1f_2} \\ \mathbf{P}_{f_2r} & \mathbf{P}_{f_2f_1} & \mathbf{P}_{f_2f_2} \end{bmatrix} \tag{2.6}$$

Decomposing the robot covariance would yield

$$\mathbf{P}_{rr} = \begin{bmatrix} \sigma^2_{x_r} & \sigma_{x_ry_r} & \sigma_{x_r\theta_r} & \sigma_{x_ru} \\ \sigma_{y_rx_r} & \sigma^2_{y_r} & \sigma_{y_r\theta_r} & \sigma_{y_ru} \\ \sigma_{\theta_rx_r} & \sigma_{\theta_ry_r} & \sigma^2_{\theta_r} & \sigma_{\theta_ru} \\ \sigma_{ux_r} & \sigma_{uy_r} & \sigma_{u\theta_r} & \sigma^2_u \end{bmatrix}. \tag{2.7}$$

Similarly, the decomposed state covariance would be

$$\mathbf{P}_{rr} = \begin{bmatrix} \sigma^2_{x_r} & \sigma_{x_r y_r} & \sigma_{x_r \theta_r} & \sigma_{x_r u} & \sigma_{x_r x_{f1}} & \sigma_{x_r y_{f1}} & \sigma_{x_r x_{f2}} & \sigma_{x_r y_{f2}} \\ \sigma_{y_r x_r} & \sigma^2_{y_r} & \sigma_{y_r \theta_r} & \sigma_{y_r u} & \sigma_{y_r x_{f1}} & \sigma_{y_r y_{f1}} & \sigma_{y_r x_{f2}} & \sigma_{y_r y_{f2}} \\ \sigma_{\theta_r x_r} & \sigma_{\theta_r y_r} & \sigma^2_{\theta_r} & \sigma_{\theta_r u} & \sigma_{\theta_r x_{f1}} & \sigma_{\theta_r y_{f1}} & \sigma_{\theta_r x_{f2}} & \sigma_{\theta_r y_{f2}} \\ \sigma_{u x_r} & \sigma_{u y_r} & \sigma_{u \theta_r} & \sigma^2_{u} & \sigma_{u x_{f1}} & \sigma_{u y_{f1}} & \sigma_{u x_{f2}} & \sigma_{u y_{f2}} \\ \sigma_{x_{f1} x_r} & \sigma_{x_{f1} y_r} & \sigma_{x_{f1} \theta_r} & \sigma_{x_{f1} u} & \sigma^2_{x_{f1}} & \sigma_{x_{f1} y_{f1}} & \sigma_{x_{f1} x_{f2}} & \sigma_{x_{f1} y_{f2}} \\ \sigma_{y_{f1} x_r} & \sigma_{y_{f1} y_r} & \sigma_{y_{f1} \theta_r} & \sigma_{y_{f1} u} & \sigma_{y_{f1} x_{f1}} & \sigma^2_{y_{f1}} & \sigma_{y_{f1} x_{f2}} & \sigma_{y_{f1} x_{f2}} \\ \sigma_{x_{f2} x_r} & \sigma_{x_{f2} y_r} & \sigma_{x_{f2} \theta_r} & \sigma_{x_{f2} u} & \sigma_{x_{f2} x_{f1}} & \sigma_{x_{f2} y_{f1}} & \sigma^2_{x_{f2}} & \sigma_{x_{f2} x_{f2}} \\ \sigma_{y_{f2} x_r} & \sigma_{y_{f2} y_r} & \sigma_{y_{f2} \theta_r} & \sigma_{y_{f2} u} & \sigma_{y_{f2} x_{f1}} & \sigma_{y_{f2} y_{f1}} & \sigma_{y_{f2} x_{f2}} & \sigma^2_{y_{f2}} \end{bmatrix} . \tag{2.8}$$

2.2.1 State Projection

As the robot moves, its positional uncertainty increases. This is due to the uncertainty in velocity and heading, and process noise. A nonlinear function $\mathbf{f}(\cdot)$ is used for state projection. An initial state

$$\mathbf{x}[k|k] = \begin{bmatrix} \mathbf{x}_r[k|k] \\ \mathbf{x_f}[k|k] \end{bmatrix} \tag{2.9}$$

would become

$$\mathbf{x}[k+1|k] = \begin{bmatrix} \mathbf{x}_r[k+1|k] \\ \mathbf{x_f}[k+1|k] \end{bmatrix} = \begin{bmatrix} \mathbf{f}(\mathbf{x}_r[k|k], \mathbf{u}[k]) \\ \mathbf{x_f}[k|k] \end{bmatrix} \tag{2.10}$$

with $\mathbf{u}[k]$) being the control input. For a two dimensional model with control input $\mathbf{u}[k] = [\delta\theta_r \ \delta u]^T$, the robot state projection would be

$$\mathbf{x}_r[k+1|k] = \mathbf{f}(\mathbf{x}_r[k|k], \mathbf{u}[k]) = \begin{bmatrix} x_r + u\cos(\theta_r)\delta t \\ y_r + u\sin(\theta_r)\delta t \\ \theta_r + \delta\theta_r \\ u + \delta u \end{bmatrix} \tag{2.11}$$

The projected covariance matrix is

$$\mathbf{P} = \mathbf{F_x}\mathbf{P}\mathbf{F_x}^T + \mathbf{Q} \tag{2.12}$$

where $\mathbf{F_x}$ is the Jacobian of the projection function $\mathbf{f}(\cdot)$ and $\mathbf{Q}$ is the process noise. In the two dimensional map with two features, the Jacobian would be

$$\mathbf{F_x} = \begin{bmatrix} \mathbf{F}_{\mathbf{x}_r} & 0 \\ 0 & I \end{bmatrix}, \tag{2.13}$$

with the vehicle Jacobian $\mathbf{F}_{\mathbf{x}_r}$ being

$$\mathbf{F}_{\mathbf{x}_r} = I + \begin{bmatrix} 0 & 0 & -u\sin(\theta_r)\delta t & \cos(\theta_r)\delta t \\ 0 & 0 & u\cos(\theta_r)\delta t & \sin(\theta_r)\delta t \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{2.14}$$

2.2.2 Feature Mapping

Features are mapped by augmenting the feature vector $\mathbf{x_f}$ of the state vector $\mathbf{x}$ using the nonlinear function $\mathbf{g}(\cdot)$. For a map with i features, a new feature $i+1$ would be mapped at time k using measurement $\mathbf{z}_{f_{i+1}}[k]$ through the following state augmentation:

$$\mathbf{x}[k|k] = \begin{bmatrix} \mathbf{x}_r[k|k] \\ \mathbf{x_f}[k|k] \end{bmatrix} = \begin{bmatrix} \mathbf{x}_r[k|k] \\ \mathbf{x_f}[k|k] \\ \mathbf{g}(\mathbf{x}_r[k|k], \mathbf{z}_{f_{i+1}}[k]) \end{bmatrix}. \tag{2.15}$$

Applying the two dimensional robot model, and measuring the range and bearing to the target, $\mathbf{z}_{f_{i+1}}[k] = [r\ \theta]^T$, the initialization function would be

$$\mathbf{g}(\mathbf{x}_r[k|k], \mathbf{z}_{f_{i+1}}[k]) = \begin{bmatrix} x_{f_{i+1}} \\ y_{f_{i+1}} \end{bmatrix} = \begin{bmatrix} x_r + r\cos(\theta_r + \theta) \\ y_r + r\sin(\theta_r + \theta) \end{bmatrix} \tag{2.16}$$

Next, the covariance matrix is augmented using submatrices $\mathbf{A}$ and $\mathbf{B}$

$$\mathbf{P} = \begin{bmatrix} \mathbf{P} & \mathbf{B}^T \\ \mathbf{B} & \mathbf{A} \end{bmatrix} \tag{2.17}$$

where **A** and **B** are

$$\mathbf{A} = \mathbf{G_x P G_x}^T + \mathbf{G_z R G_z}^T \tag{2.18}$$

$$\mathbf{B} = \mathbf{G_x P}. \tag{2.19}$$

The state and measurement Jacobians $\mathbf{G_x}$ and $\mathbf{G_z}$ in this case would be

$$\mathbf{G_x} = \begin{bmatrix} 1 & 0 & -r\sin(\theta_r + \theta) & 0 & \dots & 0 \\ 0 & 1 & r\cos(\theta_r + \theta) & 0 & \dots & 0 \end{bmatrix} \tag{2.20}$$

$$\mathbf{G_z} = \begin{bmatrix} \cos(\theta_r + \theta) & -r\sin(\theta_r + \theta) \\ \sin(\theta_r + \theta) & r\cos(\theta_r + \theta) \end{bmatrix} \tag{2.21}$$

.

If the range and bearing measurements were uncorrelated, the measurement covariance would be the diagonal matrix

$$\mathbf{R} = \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\theta^2 \end{bmatrix}. \tag{2.22}$$

If the range and bearing observations were correlated, this would be reflected by nonzero off diagonal terms.

2.2.3 Measurement Update

When measurements of features are made, the state is updated using an extended Kalman filter (EKF) [47, 9]. First, the measurement is predicted using the nonlinear prediction function $\mathbf{h}(\cdot)$. Then, the innovation ν is calculated, which is the difference between the actual and predicted measurement:

$$\nu = \mathbf{z} - \mathbf{h}(\cdot). \tag{2.23}$$

The associated innovation covariance $\mathbf{S}$ is calculated using the measurement Jacobian $\mathbf{H_x}$, the state covariance $\mathbf{P}$, and the measurement covariance $\mathbf{R}$

$$\mathbf{S} = \mathbf{H_x}\mathbf{P}\mathbf{H_x}^T + \mathbf{R}. \tag{2.24}$$

Next, the Kalman gain is calculated

$$\mathbf{K} = \mathbf{P}\mathbf{H}^T\mathbf{S}^{-1} \tag{2.25}$$

Using the Kalman gain, the updated state and covariance can be calculated

$$\mathbf{x}[k+1|k+1] = \mathbf{x}[k+1|k] + \mathbf{K}\nu \tag{2.26}$$

$$\mathbf{P} = \mathbf{P} - \mathbf{K}\mathbf{S}\mathbf{K}^T \tag{2.27}$$

For instance, for a robot in a map with a single point feature

$$\mathbf{x}[k|k] = \begin{bmatrix} \mathbf{x}_r[k|k] \\ \mathbf{x}_f[k|k] \end{bmatrix} = \begin{bmatrix} x_r \\ y_r \\ \theta_r \\ u \\ x_{f_1} \\ y_{f_1} \end{bmatrix} \tag{2.28}$$

making a measurement $\mathbf{z} = [r\ \theta]^T$, the predicted measurement would be

$$\mathbf{h}(\mathbf{x}(k+1|k)) = \begin{bmatrix} \sqrt{(x_{f_1} - x_r)^2 + (y_{f_1} - y_r)^2} \\ \arctan(\frac{y_{f_1} - y_r}{x_{f_1}) - x_r} - \theta_r \end{bmatrix} \tag{2.29}$$

and would have a measurement Jacobian

$$\mathbf{H_x} = \begin{bmatrix} \frac{\partial r}{\partial x_r} & \frac{\partial r}{\partial y_r} & \frac{\partial r}{\partial \theta} & \frac{\partial r}{\partial u} & \frac{\partial r}{\partial x_{f_1}} & \frac{\partial r}{\partial y_{f_1}} \\ \frac{\partial \phi}{\partial x_r} & \frac{\partial \phi}{\partial y_r} & \frac{\partial \phi}{\partial \theta} & \frac{\partial \phi}{\partial u} & \frac{\partial \phi}{\partial x_{f_1}} & \frac{\partial \phi}{\partial y_{f_1}} \end{bmatrix}, \tag{2.30}$$

or

$$\mathbf{H_x} = \begin{bmatrix} \frac{x_r - x_{f_1}}{r} & \frac{y_r - y_{f_1}}{r} & 0 & 0 & \frac{x_{f_1} - x_r}{r} & \frac{y_{f_1} - y_r}{r} \\ \frac{y_r - y_{f_1}}{r} & \frac{x_{f_1} - x_r}{r^2} & -1 & 0 & \frac{y_{f_1} - y_r}{r^2} & \frac{x_r - x_{f_1}}{r^2} \end{bmatrix}. \tag{2.31}$$

2.3 Research Issues

The Smith, Self, and Cheeseman approach to the CML problem has had a profound impact on the field of mobile robotics. Recent events such as the "SLAM Summer School" [25] document the high level of recent interest and progress in the problem of robot navigation and mapping. Topics of recent research in CML include the problems of map scaling, data association, and operation in dynamic environments.

The map scaling problem has been a key issue in CML research. The ideal solution to the map scaling problem would simultaneously satisfy the "Three C's" of consistency, convergence, and computational efficiency. The basic Kalman formulation of CML by Smith, Self, and Cheeseman suffers an $\mathcal{O}(n^2)$ growth in computational complexity in which n is the number of environmental features. Some methods can reduce the computation by a constant factor but still incur the $\mathcal{O}(n^2)$ growth. These include postponement (Davison [27] and Knight [50]), the compressed filter (Guivant and Nebot [39]), sequential map joining (Tardós *et al.*[90]), and the constrained local submap filter (Williams [101]). Approximation methods that achieve $\mathcal{O}(1)$ growth of complexity include decoupled stochastic maping [61] and sparse extended information filters [94, 93]. Recently, Leonard and Newman have developed a submap approach that achieves asymptotic convergence for repeated traversals of the environment while maintaining consistency and $\mathcal{O}(1)$ growth of complexity [57].

One criticism of the Kalman approach to CML is the assumption that Gaussian probability distributions can effectively represent uncertainty and can cope with nonlinearities in models for vehicle dynamics and sensor measurements. Several approaches have been published for a fully nonlinear approach to CML, including the factored solution to simultaneous localization and mapping (FastSLAM) [68], and the use of a sum of Gaussians models for representation of errors [66].

The Atlas framework of Bosse *et al.* [13] uses a hybrid metric/topological representation for CML to achieve real-time CML in large, cyclic environments. The approach combines and extends elements of Chong and Kleeman [22] and Gutmann and Konolige [40].

Another assumption of most recent in CML is that the environment consists of static features. Research that lifts this assumption, to integrate tracking of dynamic objects with mapping, has recently been performed by Wang *et al.* [96] and Hähnel *et al.*[41].

The data association problem concerns determining the correspondence of measurements to environmental features. Many SLAM researchers have used nearest-neighbor gating [8] for determing correspondence. Joint Compatibility Branch and Bound (JCBB), an improved method that simultaneoulsy considers associations to multiple features, has been developed by Neira and Tardós [75].

Nearly all of the work in CML listed above assumes that sensors provide complete observations of environmental features from a single vehicle position. This assumption, however, is violated in many important scenarios, notably when sensing with wide beam sonar data. In effect, CML assumes perception has been solved. However, as argued in Chapter 1, we feel that perception is arguably the toughest problem in robotics.

2.4 Perception

Perception is the process of transforming raw sensory data into a useful representation so a robot can interact with its environment. Raw sensory data is the output of a sensor. For a simple acoustic sensor, the output could be a waveform or a time of flight (TOF) for ranging. A camera would output an image, typically a bitmap. A laser scanner might output a set of ranges and angles. Sensors typically do not output high-level constructs such as, “my thesis advisor is looking skeptically at this sentence.” Perception is what transforms raw data into high-level representations.

Sonar research typically differs from vision research in its goals. Quite often, sonar researchers are trying to create an image that a human can process, rather than a representation for automated processing. (For example, see Figure 1-5 in Chapter 1.) When humans process visual images, they have the benefit of a highly evolved visual cortex, and substantial prior knowledge and reasoning. It would be desirable to develop sonar processing without having to replicate the human visual apparatus.

Most robotic perception work using sonar has concerned either obstacle avoidance or feature detection for navigation. Nearly all commercially available land mobile robots come equipped with a ring of Polaroid sonar sensors. Many researchers who have attempted to use sonar data in land robotics have been disappointed due to a fundamental misunderstanding of the nature of sonar data. The Polaroid sonar is a wide beam time of flight sensor (TOF) that typically triggers on specular echoes and which may trigger on multipath reflections. Multipath is when the sound does not travel directly between the transducer and the target but strikes an intermediate object. When multipath occurs, the range to the target is no longer simply the path length divided by two.

The SICK laser scanner became widely available in robotics in the mid-1990s. Reseachers have been much more successful with laser data. Figure 2-2 shows a comparison of SICK laser and Polaroid sonar data taken in a corridor at MIT. The layout of the corridor is shown in Figure 2-1. Both data sets are smeared by dead reckoning error; however, it is readily evident that the laser data provides a much better match to a visual map of the environment.

A smaller number of researchers (such as Kuc, Kleeman, Wijk, Choset, Brooks, Leonard, and Durrant-Whyte) have investigated how to use sonar in robotics, developing approaches that take advantage of its strengths while accommodating the weaknesses of specific sonar units.

What form the high-level representation takes depends heavily on the cognitive algorithm. For many feature-based navigation approaches, such as Leonard and

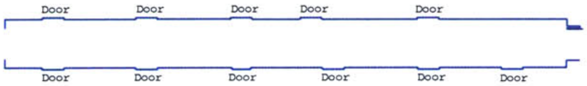


Figure 2-1: Hand-measured model of a corridor (total length approximately 25 meters).

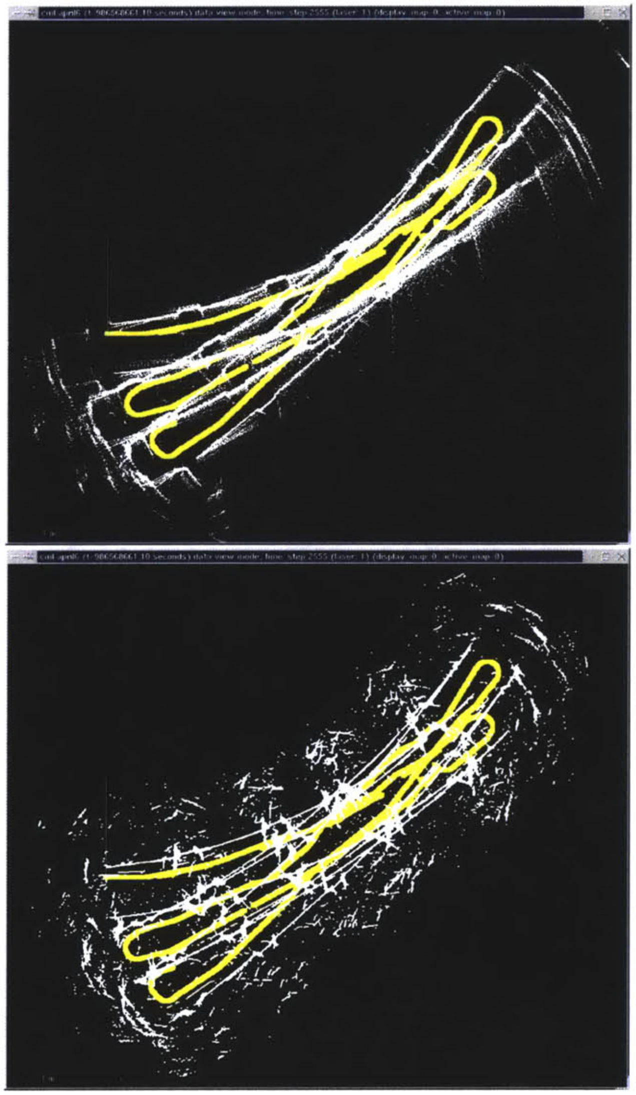

corl.april6 (1.986568661.10 seconds) data view mode: hmm step 2555 (laser: 1) (display map 0, active map 0)
corl.april6 (1.986568661.10 seconds) data view mode: hmm step 2555 (laser: 1) (display map 0, active map 0)

Figure 2-2: Laser (top) and sonar (bottom) data taken with a B21 mobile robot in the corridor shown in Figure 2-1, referenced to the dead-reckoning position estimate. The vehicle traveled back and forth three times, following roughly the same path. Each sonar and laser return is shown referenced to odometry. The laser data is slightly smeared due to a latency between the odometry and the laser data.

Feder [61, 34, 58], Moutarlier and Chatila [71], Smith, Self, and Cheeseman [87], Tardós *et al.*[90], Newman [77, 29], and Wijk and Christensen [98, 100, 99], the output must be geometric observations of individual, modeled features. So perception determines which measurements go with which features (the correspondence problem) and establishes correct models for the features (point, plane, cylinder, sphere, amphora, etc.).

Other approaches require less of perception. For instance, by devising a sufficiently complex estimator, Thrun [92] was able to abstract away all aspects of physical reality. Rather than decompose the world into features or objects, the world is modeled as a probability distribution. Measurements simply change the world distribution. Alternatively, Mataric and Brooks [18] bypassed traditional cognition, connecting perception directly to control. Rather than use high-level features, very simple perception, such as feature tracking, was used to provide an appropriate input for control behaviors.

2.4.1 Analyzing the Perception Problem

Early perception researchers had to address two problems simultaneously. In addition to solving the problems of their field, they had to define a rigorous methodology for approaching them.

Marr [67] proposed an "information processing" approach, by which perceivers do "what is possible and proceed from there toward what is desirable." Rather than try to transform raw data into a desired representation in a single step, a sequence of transformations is performed. Each transformation is specifically designed with respect to the others and each transformation is rigorously grounded in physics. The information processing transformations were described at three levels.

The first level, the Computational Theory, describes the inputs, outputs, and transformation of a processing stage. An input should be justified based on what can reasonably be expected of precursor processing stages. Any stage that requires an impossible input will likely fail in practice. Similarly, the output of a processing stage should be justified in terms of the broader goal of the processing. Transforming data from one representation to another provides no utility unless the new representation is useful. The output should be justified as an input to an appropriate next stage of processing. Finally, the process by which the input is transformed into the output must be explained. The physical basis for the transformation should be provided to demonstrate why the desired output will be uniquely computed from the input.

At the second level, Representation and Algorithm, the specific representations of the input and output are determined as is the algorithm for the transformation. Unlike the first level, which addresses questions of whether processing is possible and appropriate, this stage addresses the practicalities of implementation. Some ideas, such as exhaustive multiple hypothesis testing (MHT) [85], are optimal at the level of Computational Theory but intractable at the algorithmic level.

The final level, Hardware Implementation, addresses selecting the most appropriate architecture for the representations and algorithms. While many perceptual approaches are justified by the performance of humans and animals, digital computers are very different from animal brains. The form of the hardware will impact the selection of the representation and algorithm and may make certain desirable approaches impractical.

Richards [81] approached the perception problem in a manner similar to Marr's approach. He described four stages for analyzing a perceptual process. First, goals and givens must be identified. Second, a Theory of Competence is developed to show how a reliable and accurate representation can be determined from the input. Third, the representation and algorithm behind a processing stage is determined. Finally, a visual system is tested to establish the veracity of the suggested algorithm.

Richards also developed the Principle of Natural Modes [51, 51]:

> Structure in the world is not arbitrary and object properties are clustered about modes along dimensions important to the interaction between objects and environments.

Essentially, sensor data will be structured because the world is structured. Physics explains the structure. To develop algorithms that reliably interpret the regularities in sensor measurements, the underlying physics must be understood.

2.4.2 Visual Perception

Considering that the visual perception problem has received far more attention than sonar perception, it is worth studying for general information about how to proceed. Marr [67] strongly believed that the problem had to be broken into well defined subproblems, and that single step algorithms were unlikely to succeed.

> Finally, one has to come to terms with cold reality. Desirable as it may be to have vision deliver a completely invariant shape description from an image (whatever that may mean in detail), it is almost certainly impossible in only one step. We can only do what is possible and proceed from there towards what is desirable. Thus we arrive at the idea of a sequence of representations, starting with descriptions that could be obtained straight from an image but that are carefully designed to facilitate the subsequent recovery of gradually more objective, physical properties about an object shape [67].

In line with this reasoning, the vision community has developed several competences for use in visual processing. Three of these competences, edge detection, shape from shading, and optical flow, will be discussed to demonstrate the style of the approaches taken by the vision community. Subsequently, we will discuss these concepts in the context of the sonar perception problem.

Edge Detection

The simplest approximation of camera physics, perspective projection, uses a pinhole camera description [44]. The image is formed on an image plane, which is defined to be at position $z = -f'$. At the origin is the "pinhole", which all rays are defined to pass through. Relevant points in the world exist in positions (x, y, z), with the restriction that $z > 0$. A new coordinate system (x', y') is defined in the image plane, describing the inverted, or real, image.

$$\frac{x'}{f'} = \frac{x}{z} \tag{2.32}$$

$$\frac{y'}{f'} = \frac{y}{z} \tag{2.33}$$

A camera outputs a bitmapped image or pixels with intensity values. The image intensity, or irradiance, is the power per unit area hitting the image plane. The irradiance of an image is defined as $E(x, y)$.

Edges are lines or curves in an image along which there is a strong discontinuity in image intensity [67]. Edges occur for a variety of reasons. If the reflected light from two objects differs, and one object occludes another, then there will be an edge along the occlusion boundary. If the incident light falling onto an object has a sharp boundary, perhaps because of a shadow, then the reflected light will have a sharp boundary, leading to an edge. Since the reflectance of a surface may be angle dependent, a sharp change in the slope of an object may cause an edge in an image. Different materials often have different reflectance properties; as materials in a surface change, edges are often observed [102].

To find the sharp discontinuities in image intensity or image intensity gradient that are edges, the derivatives of the image intensity $E(x, y)$ are examined. Typically, the Laplacian of the image, $\nabla^2 E$, is used. The Laplacian is desirable because it is rotationally invariant and because it preserves the sign of the brightness difference across the edge [44].

Images are noisy, so the derivatives of images are noisy. Also, structure occurs on a variety of scales. To overcome this, an image is often processed using multiple edge detectors, each tuned to a different image scale.

Unfortunately, calculating the Laplacian can be difficult. Camera pixels are often arranged in a grid. If an edge truly has a sharp gradient, on the order of a pixel, then only the immediately adjacent pixels have information [44].

Shape from Shading

The reflectance of a surface often depends on its orientation with respect to the incident light. For constant illumination and material properties, the brightness of a surface will change as the slope changes. Unfortunately, when the orientation of the surface with respect to the source and receiver can be determined, there is one additional rotational ambiguity. If boundary conditions can be applied and the surface assumed to be continuous, shape can be reconstructed.

The reflectance map, $R(p, q)$, is a function which describes reflectance as a function of the surface normal angle. At a point (x, y) with surface gradient (p, q), the image intensity is written $E(x, y)$, while the reflectance map predicts a radiance $R(p, q)$ [44]. Unfortunately, the local slope has two components, but the intensity of a point only provides one constraint. The orientation of the surface cannot be determined without additional information.

If it is assumed that the depth of the point (x, y) is known, and the slope $(p, q) = (\frac{dz}{dx}, \frac{dz}{dy})$ is also known, it is possible to start to reconstruct the shape of the surface.

If a subsequent point $(x + \delta x, y + \delta y)$ is chosen, the depth will be perturbed by an amount $z = p\delta x + q\delta y$. The surface orientation is perturbed by

$$\delta p = \frac{\partial p}{\partial x}\delta x + \frac{\partial p}{\partial y}\delta y = \frac{\partial^2 z}{\partial x^2}\delta x + \frac{\partial^2 z}{\partial x \partial y}\delta y \tag{2.34}$$

$$\delta q = \frac{\partial q}{\partial x}\delta x + \frac{\partial q}{\partial y}\delta y = \frac{\partial^2 z}{\partial y \partial x}\delta x + \frac{\partial^2 z}{\partial y^2}\delta y. \tag{2.35}$$

Perturbing by some distance s yields

$$\frac{\partial p}{\partial s} = \frac{\partial p}{\partial x}\frac{\partial x}{\partial s} + \frac{\partial p}{\partial y}\frac{\partial y}{\partial s} = \frac{\partial^2 z}{\partial x^2}\frac{\partial x}{\partial s} + \frac{\partial^2 z}{\partial x \partial y}\frac{\partial y}{\partial s} \tag{2.36}$$

$$\dot{q} = \frac{\partial q}{\partial x}\frac{\partial x}{\partial s} + \frac{\partial q}{\partial y}\frac{\partial y}{\partial s} = \frac{\partial^2 z}{\partial y \partial x}\frac{\partial x}{\partial s} + \frac{\partial^2 z}{\partial y^2}\frac{\partial y}{\partial s}. \tag{2.37}$$

Differentiating the image intensity $E(x, y) = R(p, q)$ with respect to x and y, and using a chain rule expansion, one can get

$$\frac{\partial E}{\partial x} = \frac{\partial R}{\partial p}\frac{\partial p}{\partial x} + \frac{\partial R}{\partial q}\frac{\partial q}{\partial x} = \frac{\partial R}{\partial p}\frac{\partial^2 z}{\partial x^2} + \frac{\partial R}{\partial q}\frac{\partial^2 z}{\partial y \partial x} \tag{2.38}$$

$$\frac{\partial E}{\partial y} = \frac{\partial R}{\partial p}\frac{\partial p}{\partial y} + \frac{\partial R}{\partial q}\frac{\partial q}{\partial y} = \frac{\partial R}{\partial p}\frac{\partial^2 z}{\partial x \partial y} + \frac{\partial R}{\partial q}\frac{\partial^2 z}{\partial y^2} \tag{2.39}$$

If the change in the image plane position is in the direction of the reflectance gradient

$$\frac{\partial x}{\partial s} = R_p \tag{2.40}$$

$$\frac{\partial y}{\partial s} = R_q \tag{2.41}$$

then the slope in depth is

$$\frac{\partial z}{\partial s} = pR_p + qR_q \tag{2.42}$$

and the the image gradient is used to determine the change in surface slope

$$\frac{\partial p}{\partial s} = E_x \tag{2.43}$$

$$\frac{\partial q}{\partial s} = E_y. \tag{2.44}$$

Subject to very specific boundary conditions, the shape from shading equations can be solved to reconstruct a body. The general case, without constraints, is unsolvable [44].

Optical Flow

Optical flow is a technique for tracking objects in an image [44]. If at a point (x, y) in the image at time t, the image intensity is $E(x, y, t)$, and if the image velocity of that point is $u(x, y)$ and $v(x, y)$, then at time $t + \delta t$ the point would have moved to $(x + u\delta t, y + v\delta t$. Those two points would have the same intensity

$$E(x + u\delta t, y + v\delta t, t + \delta t) = E(x, y, t). \tag{2.45}$$

Taking a Taylor series expansion of the left side yields

$$E(x,y,t) + \delta x \frac{\partial E}{\partial x} + \delta y \frac{\partial E}{\partial y} + \delta t \frac{\partial E}{\partial t} = E(x,y,t). \qquad (2.46)$$

Removing terms that cancel, and dividing by ∂t, the equation becomes

$$\frac{\partial E}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial E}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial E}{\partial t} = 0. \qquad (2.47)$$

2.5 Sonar Perception

Sonar sensors can provide strong geometric target constraints. An active sonar can provide very precise ranging information. Sonars provide less accurate bearing information. The bearing resolution of a sonar depends on the aperture. A transmitter with a large aperture can create a very narrow beam, greatly constraining a target's location. However, a narrow beam will rarely be on any given target, making it difficult to continuously observe a target for navigation. A wide beam sonar can more easily continuously observe a target, but, with no other information, cannot provide a strong angular constraint. The key is having more information. By using more than one receiver and measuring the delay between the arrival at the various receivers, wide beam sonars with the angular resolution of narrow beam sonars can be created. The angular resolution of a receiving array is a function of the aperture, or size. The larger the array, the more precise. However, robots can only carry arrays of limited size.

Leonard and Durrant-Whyte [60] developed a clustering technique for classifying features to build geometric maps of features from polaroid sonar data. Applying the Freedman [37] model of echo formation to a scanning wide beam sonar, measurement artifacts known as RCDs (regions of constant depth) were extracted.

The Freedman model of echo formation predicts the echoes from a faceted surface. For a flat, relatively smooth surface, it predicts strong specular reflections. For a monostatic sonar (the same transducer acts as transmitter and receiver), these are normal reflections. So the Freedman model predicts normal reflections from surfaces when using a monostatic sonar. For bistatic or multistatic sonars, in which there are a transmitter and one or more receivers, the angle of incidence would equal the angle of reflection, leading to more complex echo paths. The Freedman model also predicts echoes in which there are discontinuities in a surface. For a faceted surface, the edges would scatter sound in all directions. Combining these two methods of echo formation, the Freedman model predicts than the return from a target will be a series of impulses.

Using the Freedman model, Leonard and Durrant-Whyte [60] recognized that the echo formation model depended only on position, not transducer orientation, and that range is rotation invariant. However, transducers are not omnidirectional. Based on their shape or aperture they have a beam pattern. This beam pattern describes the intensity of the outgoing sound and the sensitivity to incoming sound. The beam pattern of a sonar is based on the aperture and transmission frequency and typically has multiple intensity lobes. The main lobe has the highest intensity; the side lobes lower. However, the Polaroid sonar does not really have side lobes. Because it transmits a broadband pulse, it has a white spectrum. The overall beam pattern for the sonar is a combination of the beam patterns of the respective frequencies, leading to a single main lobe [52]. Approximating the beam pattern as a sector, the Freedman model was applied to targets in that sector. It was assumed that targets outside that sector likely received little energy. As was the convention at the time, sonar returns were plotted as if the object were directly in front of the sonar. If the sonar was rotated, or scanned, the object would be seen over a range of angles. When this was plotted, an arc covering some sector was observed. For a solitary target with polar coordinates (r, θ) with a beam half angle of θ_b, scanned measurements would form an arc from $(r, \theta - \theta_b)$ to $(r, \theta + \theta_b)$. Since this arc has a constant range, this was called a region of constant depth.

Classification was performed by comparing observations from multiple vantage points. The Freedman model predicts that edges will yield reflections, so a clustering technique was developed to find edges. Since all reflections from edges originate from the same point, the clustering technique looks for RCDs that intersect at a common point, or nearly intersect at a common point.

The Freedman model for planes predicts normal reflections for a monostatic source. The RCDs from a planar surface should have a common tangency.

Using these two clustering techniques, mapping was done with accurate navigation, and localization was done with accurate maps.

The Arc Transversal Median (ATM) method [74] was developed to provide a robust means for constructing a Generalized Voronoi Graph (GVG), a topological representation. The GVG is a set of points that are equidistant from the closest objects. If sonar is to be used, resolution is an immediate issue. If the sonar cannot resolve two targets, but instead fuses them into one, then the GVG is constructed with respect to a third more distant target. For land robots, insufficient resolution may preclude the robot from finding doorways.

The ATM approach modeled sonars as RCDs [60]. Since door edges were of interest, RCD intersections were explored. Noting that many intersections are ill-conditioned, only transverse intersections were used. Transverse intersections were defined as intersections at angles exceeding 30^o. Clusters were created, using arc intersections. To find the location of the edge, the median was used as a robust estimator.

Using this approach, the edges of the entrance to a corridor were found, which the robot successfully explored.

Triangulation Based Fusion (TBF) [99, 100, 98] was a method developed for mapping vertical edges from sonar data. A sliding window is used to store recent sonar measurements. The most recent set of measurements are compared to prior measurements in the window to find triangulation points. By assuming perfect navigation, ranges from two positions are intersected to estimate the potential location of an edge. If a measurement from the most recent timestep, when intersected with past measurements, consistently yields the same intersection, an edge is found.

The Hough transform has been applied to processing Polaroid sonar data for feature-based navigation [63]. For each feature type, a grid based feature space is created. Measurements are projected into the feature space and used to increase the occupancy of the relevant grid elements. For instance, to find points using a Polaroid sonar, a cartesian space is used. When a measurement is added, the pixels which the RCD passes through are incremented by one. Feature space locales of high occupancy indicate features.

Mataric and Brooks [18] were interested in sonar-based behaviors. A land robot, Toto, was constructed to negotiate an office environment. It was given a simple feature tracking behavior. Essentially, the robot maintained a fixed distance from a feature while driving forward. Decisions were made based on the robot's path. If, while staying a fixed distance from the feature, the robot drove in a straight line, the feature was judged to be a wall. If, while tracking a wall, the robot started to turn, this was a cue that the wall had ended. The robot established paths based on sequences of behaviors and cues. Although often overlooked in the history of robotic sonar perception, this work is noteworthy because features are tracked without explicit models and because cues are used to indicate transitions in the nature of tracked features (ie wall to corner).

Specialized sensors have been developed to disambiguate sonar measurements. Since it is often useful for a land robot to be able to differentiate between walls and corners, specialized sensors have been developed for their classification.

Barshan and Kuc [10] developed a sensor which differentiated between convex corners and walls using amplitude measurements from two transducers. Both transducers were able to transmit and receive. Because each could receive the transmission of the other, there were four observable acoustic paths. Using the method of images, each path's amplitude for the two hypotheses (corner and wall) was predicted. By comparing the measured amplitudes of the paths, corners and walls could be classified. Using the path lengths, the position of the corner or wall was then determined.

Kleeman and Kuc [49] used the method of images to distinguish corners and walls, using a sonar with two transmitters and two receivers. Each time a reflection occurs, the virtual image of the transmitter flips. For a single reflection, it flips once, for two reflections, it flips twice (which is the same as not flipping at all). If, from the received signal, the virtual image of the transmitter can be resolved, then the corners and walls can be classified by observing whether or not the virtual image is flipped. It was determined that, "Two transmitters and two receivers are necessary and sufficient for discriminating planes, corners and edges in two dimensions" [49].

Similarly, Peremanns *et al.*[78] developed a tri-aural sonar for estimating the range and bearing to targets. Using path lengths, planes and edges were classified. However, sensor noise made curvature estimation difficult.

Using a binaural sonar, Kuc [53] was able to classify targets based on their waveforms. The system would adaptively move with respect to the object, searching for "unique aspects". Using the waveform for recognition, the sonar could differentiate between heads and tails of a coin and identify targets such as o-rings.

Similarly, to differentiate between a diver and a remotely operated vehicle (ROV), Ruiz [82] developed a classifier for a sector scanning sonar. The classifier performed supervised classification using discriminant functions. Sonar images of training set objects were processed using typical vision techniques and used to generate a set of statistics describing the respective feature classes. Using what was learned from the training objects, the classifier was able to differentiate between a diver and an ROV.

One way of avoiding explicit perception algorithms is Multiple Hypothesis Testing (MHT). In the extreme case, all possible combinations of sensor data [85] are exhaustively evaluated. By representing all possible explanations, it is possible to guarantee a represention of the correct solution. In practice, there are far too many possibilities. For instance, in [85], after 5 timesteps there were 2×10^{249} hypotheses. For MHT to be successfully implemented, extremely aggressive pruning strategies must be used.

Towards this end, clustering, nearest neighbor gating, and delayed track initiation were used by Leonard and Feder [58, 33] for mapping and navigation. Leonard *et al.* [58] post-processed forward looking sonar data from a navy vehicle. Features were mapped without prior knowledge of the environment. Measurements were clustered, large clusters triggered mapping, and new features were added using the most recent observations. Feature reobservations were established using nearest neighbor gating. All features were treated as points. This limited the implementation. The robot did not have the capacity to process a prominent ridge, and the authors concluded that more advanced sonar perception was needed. Likewise, Feder [33] used clustering, nearest neighbor gating, and delayed track initiation to map and reobserve point targets in an MIT testing tank experiment.

2.6 Non-accidental Features

Many have argued that the perception problem is solvable because of world structure. By understanding this structure, measurements can be reliably interpreted.

Consider the Kanizsa triangles in Figure 2-3, taken from Donald Hoffman's excellent book *Visual Intelligence* [43]. In both cases, a viewer constructs a large white triangle in the middle. The reason for this is that each image contains a large number of redundant dimensions, or codimensions. Consider the bottom edge of the triangle in the image on the left. It is defined by the centers of the bottom two circles. Now, the bottom two circles have wedges cut out of them, giving them a "Pacman" shape. In each circle, one of those edges is colinear with the triangle bottom, meaning they share a common dimension. This reduces the total dimensionality of the figure by two. Notice the the lowest "V". The upper termination of the "V" occurs at the bottom edge of the white triangle. Since this occurs for the two lines, this further reduces the dimensionality by two. Without looking at the rest of the triangle, we have already found four redundant dimensions, or codimensions. It is these redundant dimensions that cause the triangle to jump out at us. The triangle on the right has even more codimensions.

Next, consider the white squares in Figure 2-4, again borrowed from Hoffman [43]. For the left and middle squares, because the line terminations are colinear, there is a substantial dimensional redundancy, causing a viewer to construct a square. (Although well outside the scope of this thesis, the right square is weak because the lines intersect at their termination. In the first two, the colinear terminations implied occlusion. However, it is unlikely that an occlusion would take place at colinear intersections, so we less vividly construct an occluding square in the final case.)

As a final example from Hoffman, consider the glowing blue square in Figure 2-

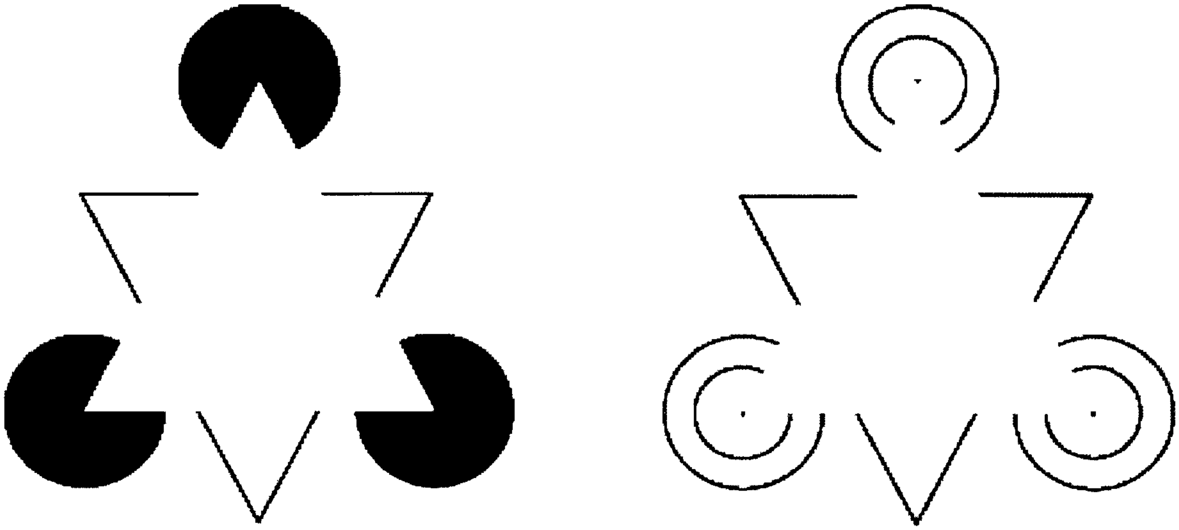


Figure 2-3: The Kanizsa triangle [43].

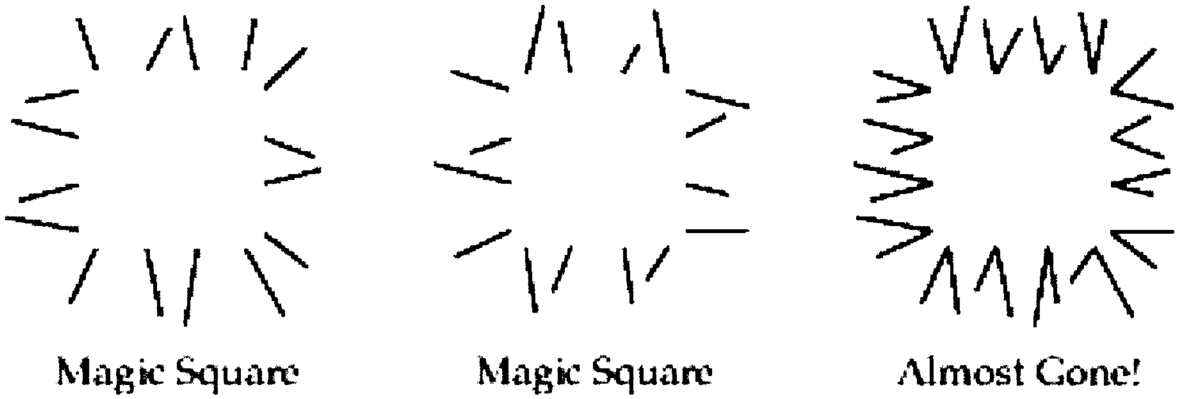


Figure 2-4: An imaginary square [43].

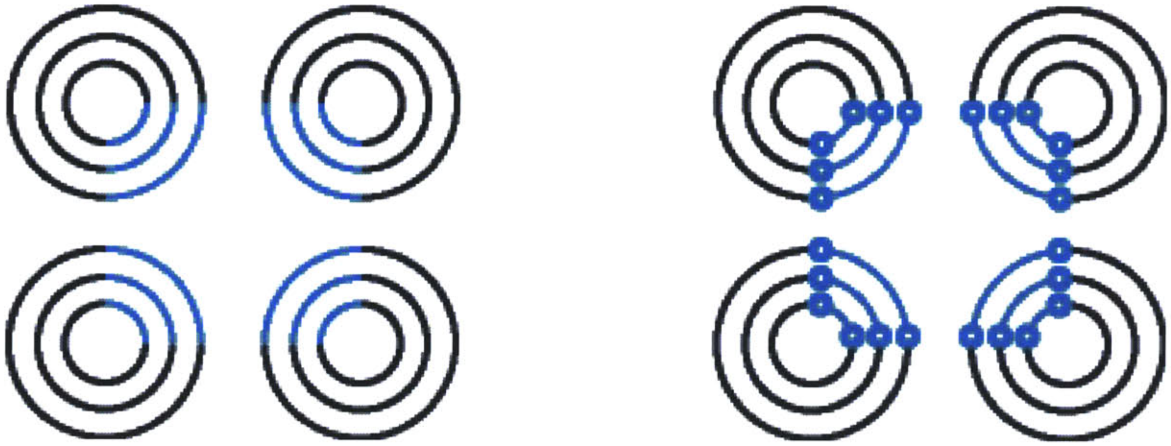


Figure 2-5: A neon blue square [43].

5. In the left square, a viewer sees glowing blue where the paper is white. The codimensions cause the viewer to believe it is being occluded by something blue; hence, that is what is constructed. The square on the left is less vivid because of the termination points; they imply there is some other explanation for the transition from black to blue.

Now, the reader may wonder what this has to do with sonar. This is important to sonar processing because it demonstrates how to approach the problem. We need to understand the underlying physics of the problem, to understand what sorts of regularities will exist in sensory data. Once we understand the dimensionality of the sensing process, we can determine when codimensions will exist in sensor data. Consider the data from an ocean experiment in Figure 2-6. On the right, we see a the path of the robot as it circled the target field(blue). The dots correspond to globally projected sonar measurements(red). The +'s correspond to measurements that originated from a single feature, and the solid dots along the robot trajectory correspond to the positions from which the robot made those measurements. This is the typical representation used in robotics. Now consider the plot on the left. The x axis is range; the y axis is time. The measurements that are the +'s in the left figure are plotted in the right figure. Notice the high degree of codimensionality of the figure. There are roughly 80 measurements, yet the entire sequence could be adequately described by a third or fourth order polynomial. The massive dimensional redundancy

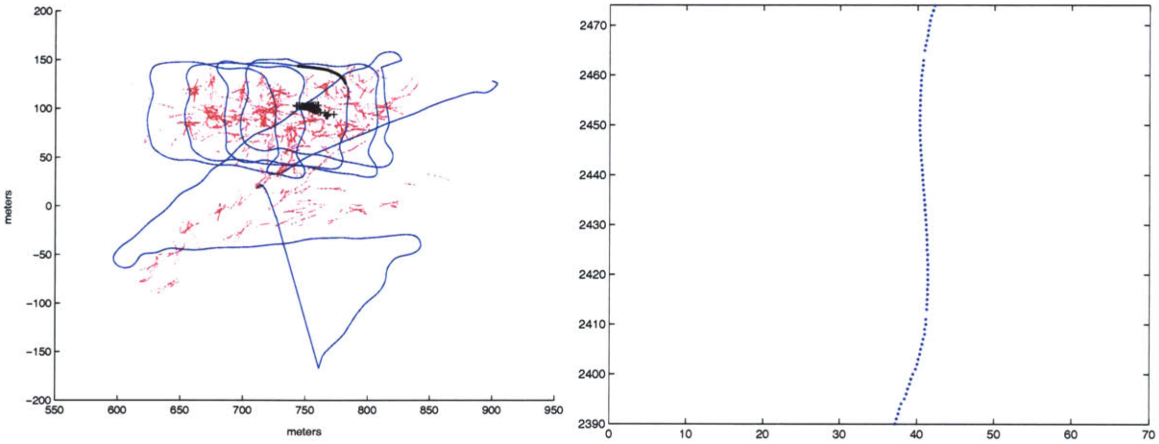


Figure 2-6: On the right, a global projection sonar data. The robot is circling a target field. Its sonar measurements are shown as dots. The +'s correspond to measurements of a specific target; the solid dots along the robot's trajectory are positions from which it measured the target. On the left, the same measurements are plotted in a different coordinate system. The x axis is range, the y axis is time. Clearly, the representation on the left has a high degree of codimensionality.

allows us to infer that the measurements had a common origin. It is just this sort of redundancy that we will exploit in the sonar perception portion of this thesis. We will determine the minimum information needed to establish codimensionality in the tracking problem, so we can efficiently group measurements of common origins.

2.7 Summary

In this chapter we have posed the problem under study in this thesis: CML for AUVs with wide beam sonar data. We have discussed alternative approaches to sonar data interpretation, selecting the biosonar characteristics of bats and dolphins (dynamic, wide-beam) for inspiration, in comparison to more traditional narrow-beam sonar imaging systems currently in use. We have reviewed previous work in CML, focusing on recent CML approaches with sonar, and we have highlighted the difficulty of coping with partial observability in prior CML approaches. Chapter 3 will describe a new CML framework, based on an approach called Working Memory, which handles these issues. Finally, we have motivated a novel approach to achieving correspondence for wide-beam sonar data acquired by a moving observer, called Trajectory Perception, which is described in Chapter 4.

Chapter 3

Stochastic Mapping with Working Memory

Stochastic Mapping works if perception occurs instantaneously and features are fully observable from a single measurement. This is unreasonable to expect in the general case; perception is very difficult and features may be of arbitrary complexity. In this chapter, the stochastic map is enhanced to include a short history of robot positions. This short history of positions, also known as working memory or short term memory, is used to accommodate partial observability and delays. Two experiments using this approach are presented at the end of the chapter.

3.1 Motivation

Stochastic mapping assumes instantaneous perception. When a robot initially observes a feature, it must model and map the feature from the first observation, or information is lost. For instance, Feder [33] used delayed track initiation to determine whether measurements were spurious. Under this scheme, only the last observation could be used in the feature initialization process. Stochastic mapping also assumes that features are fully observable from a single observation. This is not always valid. For instance, querying a Long Baseline (LBL) navigation beacon provides a range measurement. Range defines a sphere around the robot, not a point. With measurements from three positions, three spheres could be intersected to find possible beacon positions (with an ambiguity due to the quadratic). More than one observation is needed for mapping.

When features are reobserved, it is assumed that the robot will recognize them immediately. If the robot cannot immediately recognize a feature, the measurement cannot be used, and information is lost.

In this chapter, the stochastic map will be modified to include a history of robot positions to accommodate these sorts of operations.

We present two different types of experimental results with the method. In Section 3.6, we present a series of simplified examples that use manual data association to demonstrate the processes of multi-vantage point initialization and batch measurement processing. The results also demonstrate mapping of composite features and the initialization of a new robot position into a stochastic map. In Section 3.7, we describe the use of the method within a complete, real-time implementation of CML that uses the Hough transform [90] to find features. The results are for a B21 mobile robot navigating in typical indoor environments, such as a corridor, using odometry and Polaroid sonar data. Finally, in Section 3.9, we provide a further discussion of related research and describe a number of interesting topics for future research.

3.2 Problem statement

3.2.1 General formulation of the problem

CML is somewhat unconventional as a state estimation problem for two reasons: (1) data association uncertainty, and (2) variable dimensionality. Initially, the number of features in the environment is unknown and there are no initial location estimates for any features. The initial state vector is restricted to contain only the initial state of the robot. As the robot moves through its environment, it uses new sensor measurements to perform two basic operations: (1) adding new features to its state vector, and (2) updating concurrently its estimate of its own state and the locations of previously observed features in the environment. The robot also has to maintain its map, which can incorporate the fusing of two features that are hypothesized to be the same object [5, 22] and the deletion of features that are hypothesized to no longer be present [59]. In this manner, the number of elements in the stochastic map (and hence the size of the state space) varies through time.

Let us assume that there are n features in the environment, and that they are static. The true state at time k is designated by $\mathbf{x}[k] = [\mathbf{x}_r[k]^T \; \mathbf{x}_\mathbf{f}[k]^T]^T$, where $\mathbf{x}_r[k]$ represents the location of the robot, and $\mathbf{x}_\mathbf{f}[k]^T = [\mathbf{x}_{f_1}[k]^T \ldots \; \mathbf{x}_{f_n}[k]^T]^T$ represents the locations of the environmental features. We assume that the robot moves from time k to time $k+1$ in response to a known control input, $\mathbf{u}[k]$, that is corrupted by noise. Let U^k designate the set of all control inputs from time 0 through time k.

The sensors on the robot produce m_k measurements at each step k of discrete time. The set of sensor measurements at time k is designated by $Z[k]$, which is the set $\{\mathbf{z}_j[k] | j = 1 \ldots m_k\}$. Let Z^k designate the set of all measurements obtained from time 0 through time k. We assume that each measurement originates from a single feature, or it is spurious. For each measurements $\mathbf{z}_j[k] \in Z(k)$, there is a corresponding assignment index $\mathbf{a}_j$. The value of $\mathbf{a}_j$ is i if measurement $\mathbf{z}_j[k]$ originates from feature i, and it is zero if $\mathbf{z}_j[k]$ is a spurious measurement. Let A^k designate the set of all assignment indices from time 0 through time k. The cardinality of the sets Z^k and A^k are the same. Let n_k designate the number of features that have been measured up through time k (the number of features that have at least one measurement in A^k).

The objective for CML is to compute recursively the probability distribution for the location of the robot and the features and the assignments given the measurements and the control inputs:

$$p(\mathbf{x}[k], A^k | Z^k, U^{k-1}) = p(\mathbf{x}_r[k], \mathbf{x}_{f_1}[k], \ldots, \mathbf{x}_{f_{n_k}}[k], A^k | Z^k, U^{k-1}). \tag{3.1}$$

Before considering strategies for computing Equation 3.1, consider first the more restrictive problem of localization and mapping with prior knowledge of all the features and with no data association uncertainty. With perfect knowledge of $A[k]$, one could discard the outliers and combine the remaining measurements of $Z[k]$ into a composite measurement vector $\mathbf{z}[k]$. With prior knowledge of the number of features, and prior state estimates for all features, we are left with a "conventional", fixed-dimension state estimation problem. The general recursive solution applicable for fully non-linear and non-Gaussian systems is well-known [19, 88] and given by the following two equations:

$$p(\mathbf{x}[k]|Z^{k-1}, U^{k-1}) = \int p(\mathbf{x}[k]|\mathbf{x}[k-1], \mathbf{u}[k-1])p(\mathbf{x}[k-1]|Z^{k-1}, U^{k-2})d\mathbf{x}[k-1] \quad (3.2)$$

and

$$p(\mathbf{x}[k]|Z^{k}, U^{k-1}) = c_k p(\mathbf{z}[k]|\mathbf{x}[k])p(\mathbf{x}[k]|Z^{k-1}, U^{k-1}), \; k = 1, 2, \ldots \quad (3.3)$$

where $\frac{1}{c_k} = \int p(\mathbf{z}[k]|\mathbf{x}[k])p(\mathbf{x}[k]|Z^{k-1}, U^{k-1})d\mathbf{x}[k]$. Equation 3.2 is the Chapman-Komolgorov equation, and represents the use of the dynamic model $p(\mathbf{x}[k]|\mathbf{x}[k-1], \mathbf{u}[k-1])$ for state projection. Equation 3.3 is Bayes theorem, where $p(\mathbf{z}[k]|\mathbf{x}[k])$ is the measurement model. The direct application of Equations 3.2 and 3.3 entails a computational burden that grows exponentially with the number of features, rendering such application computationally intractable for typical feature-based CML applications in environments with hundreds or more features. Recent work in sequential Monte Carlo methods [30] has achieved successful performance for many challenging nonlinear, non-Gaussian state estimation problems; difficulties are encountered, however, in the application of sequential Monte Carlo methods in high-dimensional state spaces [65].

Equations 3.2 and 3.3 assume that the correspondence problem is known. When data association uncertainty (the correspondence problem) is added to the formulation, one is left with a hybrid (mixed continuous/discrete) estimation problem. Mori *et al.* [69] published a general recursive non-linear, non-Gaussian algorithm for state estimation with assignment ambiguity. Their solution generalized an earlier linear-Gaussian method by Reid [80], known as multiple hypothesis tracking (MHT). The solution builds an exponentially growing tree of hypotheses, with each leaf of the tree implementing a different solution to Equations 3.2 and 3.3, based on different hypothesized assignments. Probabilities are assigned recursively to each discrete hypothesis, and pruning is used to restrict the number of hypotheses. While the Mori *et al.* [69] solution can accommodate general non-linear, non-Gaussian models, to our knowledge it has never been implemented without simplifying assumptions. Even with the linear-Gaussian assumptions made by Reid's algorithm, the method is exponentially complex due to the combinatorics of discrete decision making. The problem bears some resemblance to object recognition in computer vision [38].

It is unclear how to incorporate variable-dimensionality (initialization of new features based on state estimates for the robot and other features in the map) into the Mori *et al.* [69] algorithm. Hence, it is unclear if one can consider the Mori *et al.* [69] as the general solution to Equation 3.1 for the CML problem. Our current opinion is that, because of the interactions between uncertainty and computational complexity, from a general theoretical perspective CML is an "unsolved" problem.

3.2.2 Linear-Gaussian Approximate Algorithms for CML

The method published in Smith, Self, and Cheeseman [86] is a linear-Gaussian approximation to the general solution of Equations 3.2 and 3.3. Nonlinear functions are linearized via a Taylor series expansion, and all probability distributions are approximated by Gaussian distributions. State updates are performed with the EKF. With these approximations, and assuming that data association is known, the computational complexity is reduced to $\mathcal{O}(n^2)$ [71].

The method recursively computes a state estimate $\hat{\mathbf{x}}[k|k] = [\hat{\mathbf{x}}_r[k|k]^T \; \hat{\mathbf{x}}_\mathbf{f}[k]^T]^T$ at each discrete time step k, where $\hat{\mathbf{x}}_r[k|k]^T$ and $\hat{\mathbf{x}}_\mathbf{f}[k]^T = [\hat{\mathbf{x}}_{f_1}[k]^T \ldots \; \hat{\mathbf{x}}_{f_n}[k]^T]^T$ are, respectively, the robot and feature state estimates. Based on assumptions about linearization and data association, this estimate is the approximate conditional mean of $p(\mathbf{x}[k]|Z^k, U^{k-1})$:

$$\hat{\mathbf{x}}[k|k] \approx E(\mathbf{x}[k]|Z^k, U^{k-1}). \tag{3.4}$$

Associated with this state vector is an estimated error covariance, $\mathbf{P}[k|k]$, which represents the errors in the robot and feature locations, and the cross-correlations between these states:

$$\mathbf{P}[k|k] = \begin{bmatrix} \mathbf{P}_{rr}[k|k] & \mathbf{P}_{r\mathbf{f}}[k|k] \\ \mathbf{P}_{\mathbf{f}r}[k|k] & \mathbf{P}_{\mathbf{ff}}[k|k] \end{bmatrix} = \begin{bmatrix} \mathbf{P}_{rr}[k|k] & \mathbf{P}_{rf_1}[k|k] & \cdots & \mathbf{P}_{rf_n}[k|k] \\ \mathbf{P}_{f_1r}[k|k] & \mathbf{P}_{f_1f_1}[k|k] & \cdots & \mathbf{P}_{f_1n}[k|k] \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P}_{f_nr}[k|k] & \mathbf{P}_{f_nf_1}[k|k] & \cdots & \mathbf{P}_{f_nf_n}[k|k] \end{bmatrix}. \tag{3.5}$$

The method uses three models, a plant model $\mathbf{f}(\cdot)$, a feature initialization model $\mathbf{g}(\cdot)$, and a measurement model $\mathbf{h}(\cdot)$. This chapter focuses on $\mathbf{g}(\cdot)$ and $\mathbf{h}(\cdot)$, presenting a generalized model for feature initializations and measurement updates from multiple uncertain vantage points. The plant model $\mathbf{f}(\cdot)$ is used to make predictions of future vehicle positions based on a control input. For a more general discussion of these models, see Feder and Leonard [34] or one of the other references on feature-based CML listed in Chapter 2. Before considering the problem of feature initialization in more detail, we provide a discussion of the data association problem for CML.

3.2.3 Data Association

To use the models $\mathbf{h}(\cdot)$ and $\mathbf{g}(\cdot)$ properly, stochastic mapping algorithms must make decisions about the origins of measurements. Spurious measurements must be ignored; however, it is often unclear which measurements are spurious. Measurements that are determined to originate from previously mapped features are used via $\mathbf{h}(\cdot)$ to perform a state estimated update. Measurements that are determined to originate from a new feature are used with $\mathbf{g}(\cdot)$ to add the feature to the map.

While there is no mention of the data association problem in Smith, Self, and Cheeseman [86], it is a crucial aspect of the CML problem. The options for data association are rather limited. Powerful tools exist, such as MHT [80] or probabilistic data association filter (PDAF) [8], but the computational burden of these approaches is very high when these techniques are applied to CML. The usual alternative is to employ "nearest-neighbor" gating techniques. For each feature in the state vector, predicted range and angle measurements are generated and are compared against the actual measurements using a weighted statistical distance in measurement space. For all measurements $\mathbf{z}_j[k]$ that can potentially be associated with feature $\hat{\mathbf{x}}_{f_i}[k]$, the innovation $\boldsymbol{\nu}_{ij}[k]$ and the innovation covariance $\mathbf{S}_{ij}[k]$ are constructed, and the closest measurement within the "gate" defined by the Mahalanobis distance

$$\boldsymbol{\nu}_{ij}[k]^T \mathbf{S}_{ij}[k]^{-1} \boldsymbol{\nu}_{ij}[k] \leq \gamma, \tag{3.6}$$

is considered the most likely measurement of that feature [8]. Such an approach will fail if the features in the environment are too close to one another.

In addition, simply testing the proximity of observations to predicted measurements for previously mapped features provides no indication of when a measurement comes from a new feature. Feature initialization is typically based on looking for several consecutive unexplained measurements that are close to one another, and far from any previously matched features. This policy is referred to as delayed track initiation [60, 34, 29]. In general, there is a tradeoff between being more likely to assign a measurement to an old feature versus using it to initialize a new feature. If one is is able to perform feature fusion [22], then it is probably better to err on the side of new feature creation. This is the strategy employed in the experiments in Section 3.7.

A variety of methods for attacking the correspondence problem have been developed in vision, such as RANSAC [35]. The general idea is to use techniques from robust statistics to find sets of measurements that collectively reinforce one another and yield a single, consistent interpretation. Recently, Neira *et al.*[75] have presented a joint compatibility testing method for data association that exploits correlation information when considering potential assignments for groups of measurements. The method succeeds in ambiguous situations when standard nearest neighbor gating fails. The general policy of looking for consensus among multiple measurements to resolve ambiguity is similar in spirit to RANSAC. Other data association strategies specific to sonar have been proposed. For example, Wijk and Christensen [99] have recently developed a technique called Triangulation-Based Fusion (TBF) that provides excellent performance for detection of point features from ring sonar data. The TBF method looks for sets of sonar returns obtained from adjacent positions that could all have originated from the same point object by efficiently computing circle intersection points and applying angle constraints. The method runs in real-time and has been successfully used for occupancy grid mapping, model-based localization, and relocation [99]. CML has also been implemented using the TBF for points features [103].

In this chapter, we use manual data association in Section 3.6 to illustrate various new types of feature initialization, and we use a Hough transform voting technique (fully documented in Tardós *et al.* [90]) to perform initialization of new point and line features when performing real-time CML in Section 3.7.

3.3 Solution to Stochastic Mapping Shortcomings

The limitations in the stochastic map are temporal in nature. The desired operations involve using temporally separated measurements. The stochastic map needs to be enhanced to allow measurements to be used asynchronously or simultaneously. This provides the robot with a "working memory", or the capacity to remember a few of its past states. The robot will be able to adaptively "remember" the information needed to use measurements and then forget what is no longer needed. Using this infrastructure, the robot will be able to escape the mechanical tedium of the Kalman filter cycle and process measurements more flexibly at its leisure.

3.4 Stochastic Mapping with Working Memory

The stochastic map consists of the state vector $\mathbf{x}$ and the covariance $\mathbf{P}$. The state vector $\mathbf{x}[k|k]$ contains the robot state $\mathbf{x}_r[k|k]$ and the feature states $\mathbf{x}_f[k|k]$.

$$\mathbf{x}[k|k] = \begin{bmatrix} \mathbf{x}_r[k|k] \\ \mathbf{x}_f[k|k] \end{bmatrix} \tag{3.7}$$

The first k in the $[k|k]$ notation is the time index of the state. For the robot state $\mathbf{x}_r[k|k]$, it indicates the robot's location at timestep k. The second k defines the information that is being used to estimate the state. So $\mathbf{x}_r(k|k)$ means that measurements through timestep k are used to estimate the state of the robot.

The prediction of the robot's next state would be $\mathbf{x}_r[k+1|k]$. The $k+1$, the time index, implies it is the next state. The k means that this is a prediction, and only measurements through timestep k have been used. Once measurements from timestep $k+1$ are used to update the state, the robot state would become $\mathbf{x}_r[k+1|k+1]$.

The representation needs to be expanded to include a "short term memory" of robot positions, to allow the robot to briefly remember where it has been. For instance, suppose the robot needs to remember where it was five timesteps ago. The state vector would be augmented to include the old state

$$\mathbf{x}[k|k] = \begin{bmatrix} \mathbf{x}_r[k|k] \\ \mathbf{x_f}[k|k] \\ \mathbf{x}_r[k-5|k] \end{bmatrix} . \tag{3.8}$$

The additional robot state, $\mathbf{x}_r[k-5|k]$, is a *smoothed* state. The information index k indicates that measurements from after the time index have been used to improve the estimate. Because more recent information can be used, smoothed estimates typically are better than real time estimates. If no measurements are available, and no updates took place after the real time state estimate, the smoothed state will be the same as the real time estimate.

A new vector, $\mathbf{x_r}[k|k]$, will be defined. Like $\mathbf{x_f}[k|k]$, the vector of feature states, $\mathbf{x_r}[k|k]$ will be a vector of robot states. Similarly, for occasions when it is desirable to maintain a temporal history of features (i.e. for dynamic features), the vector of states of the ith feature would be $\mathbf{x_{f_i}}$.

For a map containing m dynamic features and a history of n states, the expanded view of the state vector would be

$$\mathbf{x}[k|k] = \begin{bmatrix} \mathbf{x}_{\mathbf{r}}[k|k] \\ \mathbf{x}_{\mathbf{f}}[k|k] \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{\mathbf{r}}[k|k] \\ \mathbf{x}_{\mathbf{f_1}}[k|k] \\ \mathbf{x}_{\mathbf{f_2}}[k|k] \\ \vdots \\ \mathbf{x}_{\mathbf{f_m}}[k|k] \end{bmatrix} = \begin{bmatrix} \mathbf{x}_r[k|k] \\ \mathbf{x}_r[k-1|k] \\ \vdots \\ \mathbf{x}_r[k-n|k] \\ \mathbf{x}_{f_1}[k|k] \\ \mathbf{x}_{f_1}[k-1|k] \\ \vdots \\ \mathbf{x}_{f_1}[k-n|k] \\ \vdots \\ \mathbf{x}_{f_m}[k-n|k] \end{bmatrix}. \tag{3.9}$$

To keep a history of robot states, the prediction process must be modified. Rather than replace the old robot state with the predicted robot state, the state vector is augmented with the predicted state

$$\mathbf{x}[k|k] = \begin{bmatrix} \mathbf{x}_{\mathbf{r}}[k|k] \\ \mathbf{x}_{\mathbf{f}}[k|k] \end{bmatrix} = \begin{bmatrix} \mathbf{f}(\mathbf{x}_r[k|k], \mathbf{u}(k)) \\ \mathbf{x}_{\mathbf{r}}[k|k] \\ \mathbf{x}_{\mathbf{f}}[k|k] \end{bmatrix}. \tag{3.10}$$

Similarly, the covariance matrix is augmented using submatrices $\mathbf{A}$ and $\mathbf{B}$

$$\mathbf{P} = \begin{bmatrix} \mathbf{A} & \mathbf{B}^T \\ \mathbf{B} & \mathbf{P} \end{bmatrix}, \tag{3.11}$$

where $\mathbf{A}$ and $\mathbf{B}$ are

$$\mathbf{A} = \mathbf{F_x} \mathbf{P} \mathbf{F_x}^T + \mathbf{Q} \tag{3.12}$$

$$\mathbf{B} = \mathbf{F_x} \mathbf{P}. \tag{3.13}$$

When the state of a dynamic feature is projected, the corresponding vector of feature states is augmented. For instance, if the $k+1$th state of feature i was being predicted, the i feature vector would be augmented

$$\mathbf{x_{f_i}}[k+1|k] = \begin{bmatrix} \mathbf{f}(\mathbf{x}_{f_i}[k+1|k]) \\ \mathbf{x_{f_i}}[k|k] \end{bmatrix}. \tag{3.14}$$

$$\mathbf{x}[k+1|k] = \begin{bmatrix} \mathbf{x_r}[k|k] \\ \mathbf{x_f}[k+1|k] \end{bmatrix} = \begin{bmatrix} \mathbf{x_r}[k|k] \\ \mathbf{x_{f_1}}[k|k] \\ \mathbf{x_{f_2}}[k|k] \\ \vdots \\ \mathbf{x_{f_i}}[k+1|k] \\ \vdots \\ \mathbf{x_{f_m}}[k|k] \end{bmatrix} = \begin{bmatrix} \mathbf{x_r}[k|k] \\ \mathbf{x_{f_1}}[k|k] \\ \mathbf{x_{f_2}}[k|k] \\ \vdots \\ \mathbf{f}(\mathbf{x}_{f_i}[k+1|k]) \\ \mathbf{x_{f_i}}[k|k] \\ \vdots \\ \mathbf{x_{f_m}}[k|k] \end{bmatrix}. \tag{3.15}$$

Because the state vector is augmented in the middle, the covariance matrix augmentation is slightly more complicated. First, the covariance matrix is subdivided into submatrices

$$\mathbf{P} = \begin{bmatrix} \mathbf{P_1} & \mathbf{P_2}^T \\ \mathbf{P_2} & \mathbf{P_3} \end{bmatrix} \tag{3.16}$$

with submatrices

$$\mathbf{P_1} = \begin{bmatrix} \mathbf{P_{rr}} & \mathbf{P_{rf_1}} & \cdots & \mathbf{P_{rf_{i-1}}} \\ \mathbf{P_{f_1r}} & \mathbf{P_{f_1f_1}} & \cdots & \mathbf{P_{f_1f_{i-1}}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P_{f_{i-1}r}} & \mathbf{P_{f_{i-1}f_1}} & \cdots & \mathbf{P_{f_{i-1}f_{i-1}}} \end{bmatrix} \tag{3.17}$$

$$\mathbf{P_2} = \begin{bmatrix} \mathbf{P_{f_{i+1}r}} & \mathbf{P_{f_{i+1}f_1}} & \cdots & \mathbf{P_{f_{i+1}f_{i-1}}} \\ \mathbf{P_{f_{i+2}r}} & \mathbf{P_{f_{i+2}f_1}} & \cdots & \mathbf{P_{f_{i+2}f_{i-1}}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P_{f_mr}} & \mathbf{P_{f_mf_1}} & \cdots & \mathbf{P_{f_mf_{i-1}}} \end{bmatrix} \tag{3.18}$$

$$\mathbf{P_3} = \begin{bmatrix} \mathbf{P_{f_{i+1}f_{i+1}}} & \mathbf{P_{f_{i+1}f_{i+2}}} & \cdots & \mathbf{P_{f_{i+1}f_m}} \\ \mathbf{P_{f_{i+2}f_{i+1}}} & \mathbf{P_{f_{i+2}f_{i+2}}} & \cdots & \mathbf{P_{f_{i+2}f_m}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P_{f_mf_{i+1}}} & \mathbf{P_{f_mf_{i+2}}} & \cdots & \mathbf{P_{f_mf_m}} \end{bmatrix} . \tag{3.19}$$

The augmented covariance then has the form

$$\mathbf{P} = \begin{bmatrix} \mathbf{P_1} & \mathbf{B_1}^T & \mathbf{P_2}^T \\ \mathbf{B_1} & \mathbf{A} & \mathbf{B_2}^T \\ \mathbf{P_2} & \mathbf{B_2} & \mathbf{P_3} \end{bmatrix} \tag{3.20}$$

with submatrices

$$\mathbf{A} = \mathbf{F_x}\mathbf{P}\mathbf{F_x}^T + \mathbf{Q} \tag{3.21}$$

$$\mathbf{B_1} = \mathbf{F_x} \begin{bmatrix} \mathbf{P_1} \\ \mathbf{P_2} \end{bmatrix} \tag{3.22}$$

$$\mathbf{B_2} = \mathbf{F_x} \begin{bmatrix} \mathbf{P_2}^T \\ \mathbf{P_3} \end{bmatrix} , \tag{3.23}$$

where $\mathbf{Q}$ is the feature's process noise.

If it is desirable to have more than one robot in the map, additional robots can be used to augment the robot vector $\mathbf{x_r}$. If the robot state vector is expanded to include multiple robots, the state vector becomes

$$\mathbf{x}[k|k] = \begin{bmatrix}\mathbf{x}_{\mathbf{r}}[k|k]\\ \mathbf{x}_{\mathbf{f}}[k|k]\end{bmatrix} = \begin{bmatrix}\mathbf{x}_{\mathbf{r_1}}[k|k]\\ \mathbf{x}_{\mathbf{r_2}}[k|k]\\ \vdots \\ \mathbf{x}_{\mathbf{f_1}}[k|k]\\ \mathbf{x}_{\mathbf{f_2}}[k|k]\\ \vdots\\ \mathbf{x}_{\mathbf{f_m}}[k|k]\end{bmatrix} = \begin{bmatrix} \mathbf{x}_{r_1}[k|k]\\ \mathbf{x}_{r_1}[k-1|k]\\ \vdots\\ \mathbf{x}_{r_2}[k|k]\\ \mathbf{x}_{r_2}[k-1|k]\\ \vdots\\ \mathbf{x}_{r}[k-n|k]\\ \mathbf{x}_{f_1}[k|k]\\ \mathbf{x}_{f_1}[k-1|k]\\ \vdots\\ \mathbf{x}_{f_1}[k-n|k]\\ \vdots\\ \mathbf{x}_{f_m}[k-n|k]\end{bmatrix}. \tag{3.24}$$

To project the states of secondary robots, the state vector is augmented in the middle, as is done for dynamic features.

3.4.1 Delayed Decision Making

Once a robot has a short term history of states, or working memory, it no longer has to instantaneously process measurements. It can delay making decisions until sufficient information is available. Once a measurement is understood, it can be used for a state update.

In the delayed update case, the predicted measurement $\mathbf{h}(\cdot)$ of feature j is a function of a past robot state $\mathbf{x}_r[k-i|k]$. The innovation is

$$\nu = \mathbf{z}_{f_j}[k-i] - \mathbf{h}(\mathbf{x}[k|k]) = \mathbf{z}_{f_j}[k-i] - \mathbf{h}(\mathbf{x}_r[k-i|k], \mathbf{x}_{f_j}[k|k]) \tag{3.25}$$

The measurement Jacobian's nonzero terms correspond to the robot state from timestep $k-i$ and to feature j

$$\mathbf{H}_\mathbf{x} = \begin{bmatrix} 0 & \mathbf{H}_{r[k-i|k]} & 0 & \mathbf{H}_{r[k-i|k]} & 0 \end{bmatrix}. \tag{3.26}$$

The state vector is updated using the standard Kalman update equations. If, after using the measurement, the corresponding robot state is no longer needed, it can be discarded. If the unneeded state $\mathbf{x}_r[k-i|k]$ has four elements and takes up rows $[\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4]^T$, then to remove the element, simply remove those elements from the state vector, and remove rows and columns $[\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4]^T$ from the state vector.

3.4.2 Batch Updates

If the robot can update measurements after a lag, it can also update an entire sequence of measurements in one step. Very often, the reason a delay allows the robot to make a decision is because additional information can remove ambiguities. When this is true, often a single additional measurement will explain an entire sequence of measurements. In this case it is desirable to update all the measurements at once.

To update m measurements, a stacked innovation vector is created, which is a vector of innovations

$$\nu = \begin{bmatrix} \nu_1 \\ \nu_2 \\ \vdots \\ \nu_m \end{bmatrix} = \begin{bmatrix} \mathbf{z}_1 - \mathbf{h}_1(\cdot) \\ \mathbf{z}_2 - \mathbf{h}_2(\cdot) \\ \vdots \\ \mathbf{z}_m - \mathbf{h}_m(\cdot) \end{bmatrix} \tag{3.27}$$

with the subscripts $[1\ 2\ \ldots\ m]^T$ corresponding to the different measurements. The batch update Jacobian is a vector of the Jacobians corresponding to the respective measurements

$$\mathbf{H}_\mathbf{x} = \begin{bmatrix} \mathbf{H}_\mathbf{x1} \\ \mathbf{H}_\mathbf{x2} \\ \vdots \\ \mathbf{H}_\mathbf{xm} \end{bmatrix}. \tag{3.28}$$

Using the stacked innovation ν and Jacobian $\mathbf{H_x}$, the state update is calculated using the Kalman update equations.

3.4.3 Delayed Initializations

Often, when the robot initially observes a feature, it cannot tell for certain whether or not the measurement was spurious. It is undesirable to map noise. Consequently, the robot should wait until there is enough information to support the existence of a feature. Once the robot decides to map the feature, it may be desirable to map the feature using a past measurement. In this case, the mapping function $\mathbf{g}(\cdot)$ is a function of a past robot position and measurement

$$\mathbf{g}(\mathbf{x}[k|k], \mathbf{z}_{f_{i+1}}[k]) = \mathbf{g}(\mathbf{x_r}[k|k], \mathbf{Z}^k) = \mathbf{g}(\mathbf{x}_r[k-j|k], \mathbf{z}_{f_{i+1}}[k-j]). \tag{3.29}$$

The nonzero terms in the initialization Jacobian $\mathbf{G_x}$ now correspond to the $k-j$ robot state

$$\mathbf{G_x} = \begin{bmatrix} 0 & \mathbf{G}_{\mathbf{r}[k-j|k]} & 0 \end{bmatrix} \tag{3.30}$$

Using the measurement initialization function $\mathbf{f}(\cdot)$ and the initialization Jacobian $\mathbf{G_x}$, the feature is mapped by augmenting the state as in standard stochastic mapping. Additional measurements can be added using a batch update.

3.4.4 Mapping from Multiple Vantage Points

Features are not always fully observable from a single measurement or position. For instance, mapping a Long Baseline (LBL) beacon requires a minimum of three measurements. If three spheres are intersected (This assumes they do intersect. When poorly conditioned intersections are combined with measurement noise there exists the possibility that the spheres will not intersect. For instance if the robot stands still and makes three readings, noise may give three different readings, yielding three concentric spheres with no intersections.) there are two resulting points. Often a fourth measurement is needed to disambiguate the intersections. For convenience, assume there is no ambiguity because only one of the intersections corresponds to an underwater point, and since the air/water interface is a pressure release surface a reflection would cause a 180^o phase shift which the robot did not detect, and because the water is deep enough that the bottom is far away, removing the possibility of more than one surface bounce. Also, assume a known turn around time (TAT). The turnaround time is the delay between when the beacon receives the querying signal and when it transmits the reply. The TAT is often on the order of $.25s$; ignoring a $.25s$ TAT will cause an error of roughly 187.5 meters. If the TAT were unknown, a fourth measurement would be needed to estimate the beacon's relevant properties; however, we will assume the TAT is known. Subject to those assumptions, it is valid to assume that an LBL beacon can be mapped from three ranges. If observations were made at times $k - j_1$, $k - j_2$, and $k - j_3$, feature $m + 1$ would be mapped using an equation of the form

$$\mathbf{g}(\mathbf{x}[k|k], \mathbf{Z}^k) = \mathbf{g}(\mathbf{x}_r[k-j_1|k], \mathbf{z}_{f_{m+1}}[k-j_1], \mathbf{x}_r[k-j_2|k], \mathbf{z}_{f_{m+1}}[k-j_2], \mathbf{x}_r[k-j_3|k], \mathbf{z}_{f_{m+1}}[k-j_3]). \tag{3.31}$$

The feature initialization Jacobian $\mathbf{G_x}$ would have nonzero terms corresponding to the respective robot states

$$\mathbf{G_x} = \begin{bmatrix} 0 & \mathbf{G}_{\mathbf{r}[k-j_1]} & 0 & \mathbf{G}_{\mathbf{r}[k-j_2]} & 0 & \mathbf{G}_{\mathbf{r}[k-j_3]} & 0 \end{bmatrix}. \tag{3.32}$$

Again, once the initialization function and measurement Jacobian have been defined, the state augmentation occurs as in traditional stochastic mapping.

3.4.5 Using Partially Observable Measurements (Long Baseline Navigation)

Sometimes measurements cannot be fully observed from a single vantage point. For instance, if the robot is moving and makes an LBL time of flight (TOF) measurement, it will transmit and receive from different locations. The distance of flight is the distance from the transmit location to the beacon plus the distance from the beacon to the reception position. When the measurement is a function of two robot positions, from timesteps $k - j_1$ and $k - j_2$, the prediction is of the form

$$\mathbf{h}(\mathbf{x}[k|k]) = \mathbf{h}(\mathbf{x}_r[k - j_1|k], \mathbf{x}_r[k - j_2|k]). \tag{3.33}$$

Functions of even more vantage points are possible. For instance, when the robot observes it own reflection off the surface, it transmits the signal from one position, the signal reflects off the surface, reflects off the robot at a second position, reflects off the surface again, and finally is received by the robot at a third position. As long as all the relevant robot positions are represented in the state vector, the measurement can be used.

3.4.6 Mosaicking and Relative Measurements

Features do not necessarily need to be mapped for their measurements to be used. Since estimates of features are simply transformations of observations, the observations can be used directly for updates. However, some caution is necessary. Measurement noise is assumed to be uncorrelated with the state covariance; however, once an update is performed, the measurement is implicitly correlated. If a measurement is used to update more than one other measurement, all the updates must be done at once. If they are done sequentially, the correlation must be explicitly represented.

Using the two dimensional robot model, assume two range and bearing measurements are made of a point feature at timesteps $k-j_1$ and $k-j_2$. The first measurement could be used to initialize the feature, but instead it will be used to create a temporary feature state that will not be used directly.

$$\begin{bmatrix} x_p[k-j_1] \\ y_p[k-j_2] \end{bmatrix} = \begin{bmatrix} x_r[k-j_1] + r_{k-j_1}\cos(\theta_{k-j_1} + \theta_{r,k-j_1}) \\ x_y[k-j_1] + r_{k-j_1}\sin(\theta_{k-j_1} + \theta_{r,k-j_1}) \end{bmatrix} \tag{3.34}$$

A second measurement could be predicted as

$$\begin{bmatrix} r_{k-j_2} \\ \theta_{k-j_2} \end{bmatrix} = \begin{bmatrix} \sqrt{(x_p[k-j_2] - x_r[k-j_2])^2 + (y_p[k-j_2] - y_r[k-j_2])^2} \\ \arctan(\frac{y_p[k-j_2]-y_r[k-j_2]}{x_p[k-j_2]-x_r[k-j_2]}) - \theta_{r,k-j_2} \end{bmatrix} \tag{3.35}$$

Substituting for the temporary variables yields a predicted measurement that is a function of the prior measurement and the two robot states. Defining two temporary variables,

$$\Delta x = x_r[k-j_1] - x_r[k-j_2] \tag{3.36}$$

$$\Delta y = y_r[k-j_1] - y_r[k-j_2] \tag{3.37}$$

$$\mathbf{h}(\mathbf{x}, \mathbf{Z}^k) = \begin{bmatrix} r_{k-j_2} \\ \theta_{k-j_2} \end{bmatrix} = \begin{bmatrix} \sqrt{(\Delta x)^2 + (\Delta y)^2 + r_{k-j_1}^2 + 2r_{k-j_1}(\cos(\theta_{k-j_1} + \theta_{r,k-j_1})\Delta x + \sin(\theta_{k-j_1} + \theta_{r,k-j_1})\Delta y)} \\ \arctan(\frac{\Delta y + r_{k-j_1}\sin(\theta_{k-j_1} + \theta_{r,k-j_1})}{\Delta x + r_{k-j_1}\cos(\theta_{k-j_1} + \theta_{r,k-j_1})}) - \theta_{r,k-j_2} \end{bmatrix}. \tag{3.38}$$

A Kalman update is now possible using only robot states and measurements without intervening features. This is a useful technique for the mosaicking problem, in which images are correlated to produce an aggregate image as well as a reconstructed robot trajectory. Using this approach for mosaicking was the subject of Fleischer's Ph.D. thesis [36]. The working memory framework was developed simultaneously for delayed decision making and initializing partially observable features. It was not immediately recognized that working memory could be applied to the mosaicking problem. Likewise, Fleischer did not apply smoothed vehicle states to non-mosaicking applications.

3.4.7 Spatiotemporal Mahalanobis Testing

The Mahalanobis distance is a statistic which is commonly used for data association. Essentially, it determines which isoprobabilistic surface of an n dimensional ellipsoid a given innovation lies upon. Given the measurement function $\mathbf{h}(\cdot)$ and the measurement Jacobian $\mathbf{H}_{\mathbf{x}}$, the expanded and contracted forms of the Mahalanobis distance are

$$\gamma^2 = (\mathbf{z} - \mathbf{h}(\hat{\mathbf{x}}))^T(\mathbf{H}_{\mathbf{x}}\mathbf{P}\mathbf{H}_{\mathbf{x}}^T + \mathbf{R})^{-1}(\mathbf{z} - \mathbf{h}(\hat{\mathbf{x}})) \tag{3.39}$$

$$\gamma^2 = \nu^T\mathbf{S}^{-1}\nu. \tag{3.40}$$

The Mahalanobis distance is used most frequently in stochastic mapping to perform nearest neighbor gating [33, 75]. Observations of individual features are compared against predicted observations of individual features to try to determine correspondence.

Delayed gating allows future information to smooth the robot estimate for a given timestep, resulting in a more precise gate.

More generally, numerous observations taken through time of individual features can be tested to see whether or not they fit broader hypotheses. Using this form, measurements can provide context for other measurements. In this case, an expanded ν of the form utilized in the batch update would be used.

Most generally, observations taken throughout time of more than one feature can be tested to establish a metric for the broadest form of hypotheses.

3.5 Cooperative Stochastic Mapping with Working Memory

If the robot is to use information from other robots for mapping and navigation, it will need to represent their states in its state vector. It will also need some method of communication. Underwater communications are slow, and acoustic transmission rates are low compared to electromagnetic rates in air. Modems can also be jammed by other sound sources so that transmissions may have to be scheduled for when other acoustic instruments are silent. This will cause delays. These delays will have to be accommodated by explicitly representing a history of the secondary robots states.

3.5.1 Cooperation with Perfect Communication

The simplest case would be if instantaneous communication were available. Obviously, this is not realistic, but examining this case is useful.

If the robot wants to use robot n's observation of feature m, and robot n's state is in the state vector and feature m is mapped, the robot simply performs an update using an update equation of the form

$$\mathbf{h}(\mathbf{x}) = \mathbf{h}(\mathbf{x}_{r_n}, \mathbf{x}_{f_m}). \tag{3.41}$$

If the robot wants to use the nth robot's observation of the $m+1$th feature to map it, a mapping function of the form

$$\mathbf{g}(\mathbf{x}, \mathbf{z}[k]) = \mathbf{g}(\mathbf{x}_{r_n}, \mathbf{z}^{r_n}_{f_{m+1}}[k]) \tag{3.42}$$

is used.

3.5.2 Cooperation with Delayed Communication

In reality, delays occur. Robots will need to maintain estimates of where the other cooperating robots have been. If the robot wants to use the nth robot's measurement of the mth feature at timestep $k-j$, it must use a measurement prediction equation of the form

$$\mathbf{h}(\mathbf{x}) = \mathbf{h}(\mathbf{x}_{r_n}[k-j|k]). \tag{3.43}$$

If the robot wants to map the feature $[m+1]$ using the nth robot's observation at timestep $k - j$, the mapping equation is of the form

$$\mathbf{g}(\mathbf{x}) = \mathbf{g}(\mathbf{x}_{r_n}[k - j|k], \mathbf{z}_m^{r_n}[k - j]). \tag{3.44}$$

3.5.3 Cooperation with Communication Dropouts or Robots of Opportunity

Sometimes it is impractical to recreate where a cooperating robot has gone. For instance, suppose a cooperating robot has not been seen for days. Should our robot demand to know everything that the collaborator has done over that interval? If the primary robot knew every control input and every observation of the collaborator during the communications dropout, it could most accurately estimate that robot's present state. However, that is impractical. It requires an immense amount of communication. Moreover, it is a violation of the collaborator's privacy. Perhaps this is information that should not be shared with other robots, or perhaps it is undesirable to broadcast to the world where specific robots have been. In such cases, it will be necessary to remap the second robot.

Similarly, if a robot of opportunity shows up and offers to be useful, the robot should be able to incorporate the collaborator into its map and use its information.

The immediate idea one has is to merge the two robot's maps. This is a bad idea. If the two maps were entirely independent, then they could be merged flawlessly, but if information had been exchanged, they would be correlated. If they were correlated, merging the two maps, while assuming them to be uncorrelated, would result in information being reused. If the two robots had previously merged their maps, they would be correlated. If the first robot had merged its map with a third robot, which in turn merged its map with the second robot, then the first and second robots' maps would be correlated. Merging correlated maps and reusing information causes the robots to become overconfident, which is bad. If care is not taken, repeated map mergings can implicitly cause the same measurement to be used over and over again, essentially turning a single observation into several.

The easiest way to remap a cooperating robot, or map a robot of opportunity, is to use its observations of known features. The easiest case is if the second robot has sensors for measuring its orientation and velocity, in which case those components of its state can be directly initialized. To estimate position, the initialization calculates where the robot must have been to have made the observations it made. For instance, in a two dimensional environment, assuming the *nth* robot's heading θ could be measured using a compass, and assuming its observation of *mth* feature, a point object at $[x_p \; y_p]^t$, at timestep $k-j$ was $[r \; \phi]^T$, the initialization function would be of the form

$$\mathbf{g}(\mathbf{x}, \mathbf{Z}^k) = \mathbf{g}(\mathbf{x}_{f_m}(k|k), \mathbf{z}_{f_m}^{r_n}[k-j]) = \begin{bmatrix} x_{r_n}[k-j] \\ y_{r_n}[k-j] \\ \theta_{r_n}[k-j] \\ u_{r_n}[k-j] \end{bmatrix} = \begin{bmatrix} x_p - r\cos(\theta+\phi) \\ y_p - r\sin(\theta+\phi) \\ \theta_{r_n}[k-j] \\ u_{r_n}[k-j] \end{bmatrix}. \quad (3.45)$$

If, instead, the robot heading and velocity are unknown, the initialization is more problematic. In an experiment in Chapter 6, two sequential positions of a second robot are initialized. Those two positions are used to infer the second robot's heading and velocity.

First, by combining robot n's range measurements of two known features $\mathbf{x}_{f_{m_1}}$ and $\mathbf{x}_{f_{m_2}}$ at sequential timesteps $[k-j]$ and $[k-j+1]$, the two positions can be initialized as

$$\mathbf{g}_1(\mathbf{x}, \mathbf{Z}^k) = \mathbf{g}_1(\mathbf{x}_{f_{m_1}}(k|k), \mathbf{z}_{f_{m_1}}^{r_n}[k-j], \mathbf{z}_{f_{m_2}}^{r_n}[k-j]) = \begin{bmatrix} x_{r_n}[k-j] \\ y_{r_n}[k-j] \end{bmatrix} \quad (3.46)$$

and

$$\mathbf{g}_2(\mathbf{x}, \mathbf{Z}^k) = \mathbf{g}_2(\mathbf{x}_{f_{m_2}}(k|k), \mathbf{z}_{f_{m_2}}^{r_n}[k-j+1], \mathbf{z}_{f_{m_2}}^{r_n}[k-j+1]) = \begin{bmatrix} x_{r_n}[k-j+1] \\ y_{r_n}[k-j+1] \end{bmatrix}. \quad (3.47)$$

Next, a second initialization function $\mathbf{g}_3$ is used to estimate the initial robot heading and velocity. This initialization function does not directly use measurements, only the new state variables

$$\mathbf{g}_3(\mathbf{x}) = \mathbf{g}_3(\mathbf{g}_1(\mathbf{x}), \mathbf{g}_2(\mathbf{x})) = \begin{bmatrix} \theta_{r_n}[k-j] \\ u_{r_n}[k-j] \end{bmatrix} = \begin{bmatrix} \arctan(\frac{y_{rn}[k-j+1]-y_{rn}[k-j]}{x_{rn}[k-j+1]-x_{rn}[k-j]} \\ \frac{\sqrt{(x_{rn}[k-j+1]-x_{rn}[k-j])^2+(y_{rn}[k-j+1]-y_{rn}[k-j])^2}}{t_{k-j+1}-t_{k-j}} \end{bmatrix}. \tag{3.48}$$

In this function, $t_{k-j+1} - t_{k-j}$ is the time difference between states $k-j+1$ and $k-j$. Since the numerator of the velocity initialization is distance traveled, the denominator must be the time difference.

Finally, if a control input exists, it can be used in the initialization of the second heading and velocity

$$\mathbf{g}_4(\mathbf{x}) = \mathbf{g}_4(\mathbf{x}_r[k-j|k], \mathbf{u}[k-j]) = \begin{bmatrix} \theta_{r_n}[k-j+1] \\ u_{r_n}[k-j+1] \end{bmatrix} = \begin{bmatrix} \theta_{r_n}[k-j] + \delta\theta \\ u_{r_n}[k-j] + \delta u \end{bmatrix}. \tag{3.49}$$

3.5.4 Cooperative Mapping of Partially Observable Features

One problem with partially observable features is conditioning. If the state is not fully observable from a single vantage point, moving a small amount may make the feature barely observable but ill-conditioned. For instance, in a two dimensional environment using range only measurements, a robot may want to find a point by intersecting two circles. If the radii of the circles are large compared to the distance between their centers, the intersection of the circles will be ill-conditioned. A small change in either radius (due to measurement noise) will substantially alter the intersection point, as will small changes in either circle center (due to navigation error).

Additional robots can help address the conditioning problem. Appropriately positioned, multiple robots can make observations that yield a very well conditioned initialization. In this case, the initialization function $\mathbf{g}(\cdot)$ is a function of the different robot states and their respective measurements. For instance, to use measurements from two different robots r_{n_1} and r_{n_2} taken at timesteps $k - j_1$ and $k - j_2$ to initialize a feature f_{m+1}, the initialization function would take the form

$$\mathbf{g}(\mathbf{x}, \mathbf{Z}^k) = \mathbf{g}(\mathbf{x}_{r_{n_1}}[k - j_1|k], \mathbf{z}^{r_{n1}}_{m+1}[k - j_1], \mathbf{x}_{r_{n_2}}[k - j_2|k], \mathbf{z}^{r_{n2}}_{m+1}[k - j_2]) \tag{3.50}$$

3.5.5 Cooperative Spatiotemporal Mahalanobis Testing

Not only can multiple robots be used for mapping and navigation, but they can help each other understand their measurements. Single robots can use spatiotemporal Mahalanobis testing to test multiple measurements simultaneously, with each measurement providing context for other measurements. When information is available from multiple robots, all of their measurements can be tested jointly to provide the broadest possible context for measurement interpretation.

While the approach is generally applicable using any state estimation framework, in this chapter we describe the implementation of the approach in the context of stochastic mapping, yielding a method we refer to as "delayed stochastic mapping". The new method is summarized in Figure 3-1. The new components of the framework include trajectory state management, perceptual grouping, multiple vantage point initialization, and batch updating.

The growth of the state vector in this manner increases the computational burden as $\mathcal{O}(n^2)$, so caution must be taken. The new problem of trajectory state management is introduced. Old vehicle trajectory states and associated terms in the covariance need to be deleted once all the measurements from a given time step have been either processed or discarded. In practice, we have seen excellent performance by keeping the number of trajectory states restricted to a fixed size of 40. With a fixed window size, the process of adding past states is similar to a fixed-lag Kalman smoother [3].

3.5.6 Perception

The next step in the framework is to apply a perception algorithm to examine collectively the entire set of data that came from the current and past vehicle positions currently in the map. Instead of being forced to make an instantaneous decision about

```
while active mission do
    $\hat{\mathbf{x}}(k|k-1) = \mathbf{f}(\hat{\mathbf{x}}(k-1|k-1), \mathbf{u}(k))$ {state projection}
    $\mathbf{P} = \mathbf{F_x P F_x^{\mathit{T}}} + \mathbf{Q}$ {covariance projection}
    $\hat{\mathbf{z}}(k) = \mathbf{h}(\hat{\mathbf{x}}(k|k-1))$ {sensor prediction}
    $(\mathbf{a}, \neg\mathbf{a}) \leftarrow (\mathbf{z}(k), \hat{\mathbf{z}}(k))$ {data association}
    $\mathbf{S} = \mathbf{H_x P H_x^{\mathit{T}}} + \mathbf{R}$ {innovation covariance}
    $\mathbf{K} = \mathbf{P H_x}^{T} \mathbf{S}^{-1}$ {Kalman gain}
    $\hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + \mathbf{K}(\mathbf{z_a} - \hat{\mathbf{z}}_\mathbf{a})$ {Kalman state update}
    $\mathbf{P} = \mathbf{P} - \mathbf{K S K}^{T}$ {Kalman covariance update}
    $\hat{\mathbf{x}}(k|k) \leftarrow \begin{bmatrix} \hat{\mathbf{x}}(k|k) \\ \mathbf{g}(\hat{\mathbf{x}}(k|k), \mathbf{z}_{\neg\mathbf{a}}) \end{bmatrix}$ {mapping state}
    $\mathbf{P} \leftarrow \begin{bmatrix} \mathbf{P} & \mathbf{P G_x^{\mathit{T}}} \\ \mathbf{G_x P} & \mathbf{G_x P G_x^{\mathit{T}}} + \mathbf{G_z R G_z^{\mathit{T}}} \end{bmatrix}$ {mapping covariance}
    $k = k + 1$
end while
```

```
while active mission do
    $\hat{\mathbf{x}}(k|k-1) \leftarrow \begin{bmatrix} \mathbf{f}(\hat{\mathbf{x}}_r(k-1|k-1), \mathbf{u}(k)) \\ \hat{\mathbf{x}}(k-1|k-1) \end{bmatrix}$ {state augmentation with projection}
    $\mathbf{P} \leftarrow \begin{bmatrix} \mathbf{F_x P F_x^{\mathit{T}}} + \mathbf{Q} & \mathbf{F_x P} \\ \mathbf{P F_x}^{T} & \mathbf{P} \end{bmatrix}$ {covariance augmentation with projection}
    $\hat{\mathbf{z}}(k) = \mathbf{h}(\hat{\mathbf{x}}(k|k-1))$ {sensor prediction}
    $(\mathbf{a}, \neg\mathbf{a}) \leftarrow (\mathbf{z}(k), \hat{\mathbf{z}}(k))$ {data association}
    $\mathbf{S} = \mathbf{H_x P H_x^{\mathit{T}}} + \mathbf{R}$ {innovation covariance}
    $\mathbf{K} = \mathbf{P H_x}^{T} \mathbf{S}^{-1}$ {Kalman gain}
    $\hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + \mathbf{K}(\mathbf{z_a} - \hat{\mathbf{z}}_\mathbf{a})$ {Kalman state update}
    $\mathbf{P} = \mathbf{P} - \mathbf{K S K}^{T}$ {Kalman covariance update}
    $\hat{\mathbf{x}}(k|k) = \begin{bmatrix} \hat{\mathbf{x}}(k|k) \\ \mathbf{g}(\hat{\mathbf{x}}(k|k), \mathbf{z}_{\neg\mathbf{a}}) \end{bmatrix}$ {mapping state}
    $\mathbf{P} \leftarrow \begin{bmatrix} \mathbf{P} & \mathbf{P G_x^{\mathit{T}}} \\ \mathbf{G_x P} & \mathbf{G_x P G_x^{\mathit{T}}} + \mathbf{G_z R G_z^{\mathit{T}}} \end{bmatrix}$ {mapping covariance}
    Contract the state $\mathbf{x}$ and covariance $\mathbf{P}$ to remove unnecessary trajectory states and
    associated terms in the covariance matrix
    $k = k + 1$
end while
```

Figure 3-1: Comparison of conventional stochastic mapping (top) and delayed stochastic mapping (bottom). The notation is summarized as follows: $\mathbf{x}$ is the state vector, $\mathbf{P}$ is the covariance, $\mathbf{F}, \mathbf{G}$, and $\mathbf{H}$ are the Jacobians of their respective nonlinear functions, $\mathbf{Q}$ and $\mathbf{R}$ are the propagation and measurement covariance matrices, $\mathbf{u}$ is the control input, $\mathbf{z}$ are the observations, $\mathbf{a}$ labels associated observations, $\neg\mathbf{a}$ labels unmatched observations.

the origins of current measurements, delayed decision making is now possible. In general, a wide variety of perception algorithms are possible, such as the RANSAC [35] and Least Median [32] methods that have been successfully employed in vision. The subject of delayed decision making is very broad in scope, and in this chapter, we focus on the state estimation aspects of the problem only. For now, we assume that some perception has been employed to classify and associate measurements. Further discussion of perception is contained below in Section 3.7.

3.5.7 Composite Initialization

In general, the feature initialization function can use *any* of the information in the state estimation $\hat{\mathbf{x}}$ for the stochastic map, including the locations of other previously mapped features as well as as previous vehicle states. The new feature location can be a function of one or more previously mapped features, one or more measurements, and the robot state estimate corresponding to these measurements. For example, one can initialize a line that passes through a point feature $\hat{\mathbf{x}}_{f_i}$ and is tangent to one sonar return $\mathbf{z}_j(k)$. In this case, the feature initialize function is of the form:

$$\hat{\mathbf{x}}_{f_{n+1}} = \mathbf{g}(\hat{\mathbf{x}}_{f_i}, \hat{\mathbf{x}}_r(k), \mathbf{z}_j(k)). \tag{3.51}$$

Alternatively, one can initialize a point that lies at the intersection of a line currently in the map and a new sonar return. The equations for these two initialization scenarios are described in Appendix A. One can also initialize a new feature without any measurements, for example, hypothesizing the constraint that a new point feature exists at the intersection of two line segments currently in the map. Examples with real data for several of these scenarios are given below in Section 3.6.

3.6 Examples Using Manual Association

We now present several illustrations of the concepts presented above using real sonar data sets with manual data association. The first experiment uses 500 kHz underwater sonar data acquired in a testing tank, and the second uses data from a ring of 24 Polaroid sonar sensors. Both experiments employ manually-guided data association strategies that exploit *a priori* knowledge of the environment. While both environments are highly simplified, they are useful in illustrating the state estimation process for mapping from multiple uncertain vantage points. Fully automatic data association is used in the experiments in Section 3.7 as part of an integrated system that can perform CML in real-time using odometry and wide-beam sonar measurements.

An experiment was conducted using a robotic gantry to emulate the motion and sensing of an underwater vehicle. A 500 KHz binaural sonar was used [52, 4]. To show mapping of partially observable features, bearing information was discarded. Two objects were placed in the tank, a metal triangle and a point like object (a fishing bobber). The gantry was moved through two trajectories, one to the left and one to the right of the objects, emulating cooperative CML by two vehicles. All processing was post processing. Data association was done by hand since it is not the focus of this chapter. The manually-associated returns used for feature initialization are labeled in Figures 3-4 and 3-7 and are listed in Table 3.1. The initialization strategies used for each feature and for the position of the second robot are listed in Table 3.2. We consider the set of measurements from the right side of the objects to originate from "robot 1" and the measurements from the left side to be from "robot 2".

The gantry operates in a tank that is 10 meters, 3 meters wide, and 1 meter deep. The mechanism provides ground-truth good to a few millimeters. Simulated speed and heading measurements were generated and used for dead-reckoning. Initially, the sensor dead-reckoned through a trajectory of 11 positions as shown in Figure 3-2. Upon completing this trajectory, the robot had a state vector and covariance matrix which contained only robot states, one estimate for each position. In Figure 3-3, the correlation coefficients for the x components of the trajectory are plotted. Each line represents the correlations between one timestep and all other saved timesteps. Because this is a short dead-reckoned trajectory, each curve has only one maxima; more complex trajectories may have numerous local maxima.

The assumed range measurement standard deviation was 3 centimeters for each measurement. The added process noise had a standard deviation of 1 cm per time step in x and y and 2 degrees per time step in heading.

The data processing was performed as follows. First, state projection was performed and trajectory states were created for robot 1 for time steps 1 through 11 without any measurements being processed. The three-sigma error bounds for the dead-reckoned (x, y) trajectory of robot 1 are shown in Figure 3-2. The correlation coefficients between the x coordinates of the robot trajectory states are plotted in Figure 3-3.

Having an entire trajectory of positions, robot 1 starts to construct its map. Because the robot uses range measurements only, features must be observed from multiple vantage points to be mapped. By combining returns 1 and 5 (labeled in Figure 3-4), the robot initializes the point object at the bottom of its map (Figure 3-4). Similarly, by intersecting two more arcs (returns 2 and 6), the bottom corner of the triangle is mapped. The equations for arc intersection are given in Appendix A. Next, return 4 is used in conjunction with the estimated location of the bottom corner of the triangle to add the right wall of the triangle to the map. Finally, by intersecting return 3 with the estimated line corresponding to the right wall, the top corner is added to the map. Using these observations, the point object and the side of the triangle are initialized (Figure 3-5).

The wall is represented in the map by an infinite line with two parameters, ρ and θ, which are the angle and offset of the normal with respect to the origin. The dashed lines in Figures 3-4 and 3-5 show the extension of the estimated line.

After initializing the features, all other observations are used for a batch update of the newly initialized features (Figure 3-6). Mapping and navigation are improved substantially. Robot 1 obtained 29 total measurements. Of these, six were used for initialization and 23 were used for batch updating.

Next, robot 1 tries to use information from robot 2. We assume that there is no *a priori* information for the initial location of robot 2 and that hence, robot 2 must be initialized into the map using shared measurements of features seen by both robots.

It is determined that robot 2 has observed features in robot 1's map. From the ranges to the point object and the bottom corner of the triangle, (returns 7 through 10 as labeled in Figure 3-7) robot 2's first two positions can be observed. Those two positions are initialized into the map. Their initialization function is of the form $\mathbf{g}(\mathbf{x}_{f_{r1}}, \mathbf{z}_{r2})$, meaning that robot 2 is added to robot 1's map using robot 2's observations of features that have been previously mapped by robot 1 (Figure 3-8).

Next, using the first two estimated positions for robot 2, the initial heading and velocity of robot 2 are mapped. Using this information, along with the control inputs, a dead-reckoned trajectory for robot 2 is established (Figure 3-9). Because neither compass nor velocity observations are available, and because the initial estimates of velocity and heading for robot 2 are imprecise, the trajectory is imprecise and has large error bounds.

Having a dead-reckoned trajectory for robot 2 and its measurements, robot 1 then maps the (otherwise unobservable) backside of the triangle and performs a batch update to get an improved map and an improved estimate of where robot 2 traveled (Figure 3-10) . Robot 2 obtained 22 total measurements. Of these, four measurements were used to initialize the position of robot 2 in the stochastic map of robot 1. Two measurements were used in initializing features on the left side of the triangle, and the remaining 16 returns were used in batch updating.

The error bounds for robot 2 do not exhibit the growth profile that is normally seen. Typically, since the robot starts with an initial position estimate and then moves, the uncertainty grows with time. In this case, since the second robot is mapped and localized with respect to previously mapped features, the smoothed estimate of its trajectory has the least uncertainty in the middle (Figure 3-11).

The next experiment uses data from a simple "box" environment made of plywood, demonstrating the processes of multiple vantage point initialization, batch updating, and composite feature initialization. The data association and feature modeling techniques utilize the *a priori* knowledge of the structure of the box, namely that each corner of the box was created by two walls and that each wall was bounded at each end by a corner. The input data consists of 600 sonar returns acquired from a sequence of 50 positions that form one-and-a-half loops around the inside of the box. The vehicle started in the lower left corner facing upward.

The data processing proceeded as follows. First, state projection was performed and trajectory states were created for time step 1 through time step 50 without any measurements being processed.

At each processing cycle, a new vehicle trajectory state was added to the system state vector. The dead-reckoned vehicle trajectory is shown in Figure 3-13(b). After fifty cycles, a manually-guided search strategy was performed to find nine returns that

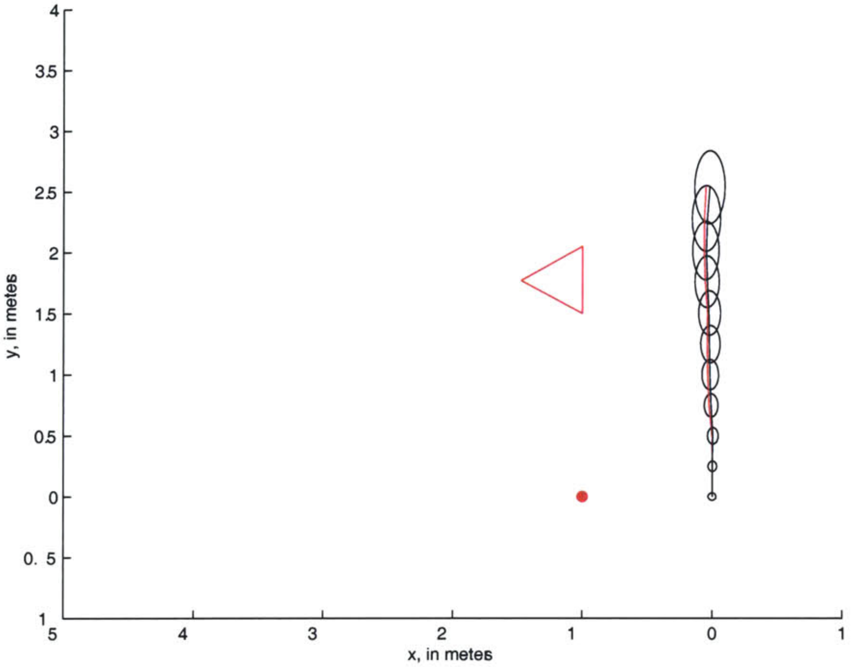


Figure 3-2: Dead-reckoned trajectory of robot 1. The robot started at the bottom and moved upwards. The ellipses represent the 99% confidence interval for each position. The triangle is an aluminum sonar target, and the small filled circle at the position x=-1.0 and y=0.0 is a fishing bobber, which served as a "point" object.

Table 3.1: Details for the ten manually-selected returns used for feature initialization.

Return number	Robot	Time Step	Odometry X (m)	Odometry Y (m)	Range (m)
1	1	1	0.0	0.0	1.0036
2	1	1	0.0	0.0	1.8058
3	1	3	.0045	.4992	1.8380
4	1	8	-.0399	1.7637	.9504
5	1	11	-.0235	2.537	2.7205
6	1	11	.0235	2.5373	1.4192
7	2	1	(unused)	(unused)	.9773
8	2	1	(unused)	(unused)	1.7851
9	2	2	(unused)	(unused)	1.0054
10	2	2	(unused)	(unused)	1.5692

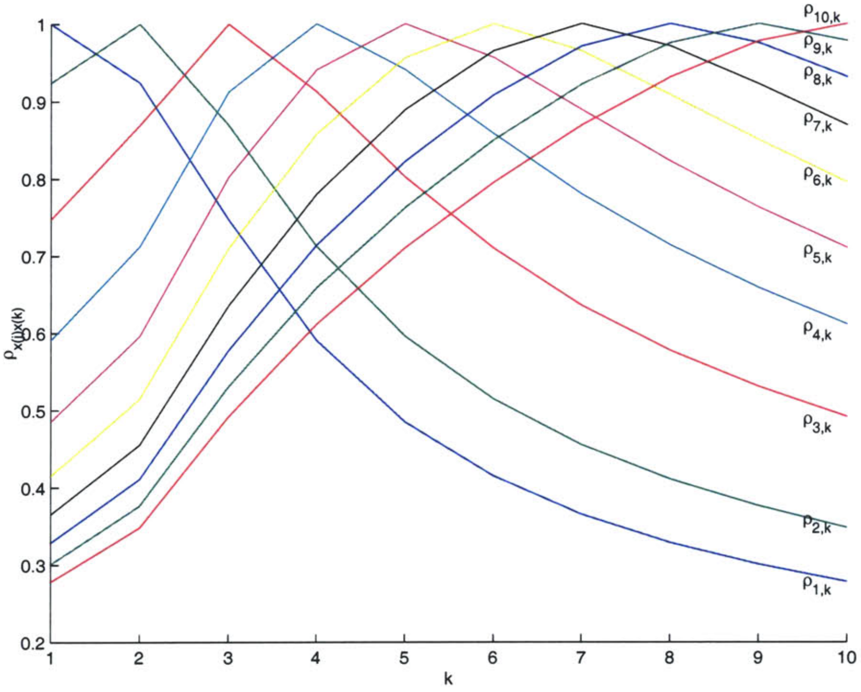


Figure 3-3: Correlation coefficients between the x components of robot 1's trajectory. Each line represents the correlations between a specific timestep and all other timesteps.

Figure 3-4: Set of observations used to initialize the side of the triangle and the point object. Two observations are needed to initialize the point target, two more are needed to initialize the corner of the triangle. Given the constraint of the corner, only one measurement was needed to map the wall. Given the wall, only one more measurement was needed to map the top corner of the triangle.

Table 3.2: Method of initialization for features.

Feature	Initialization Method
Point object	Return 1 and Return 5
Lower right vertex of triangle	Return 2 and Return 6
Right plane of triangle	Lower right vertex of triangle and Return 4
Upper right vertex of triangle	Right plane of triangle and Return 3
Position 1 for robot 2	Point object and Returns 7 and 8
Position 2 for robot 2	Point object and Returns 9 and 10

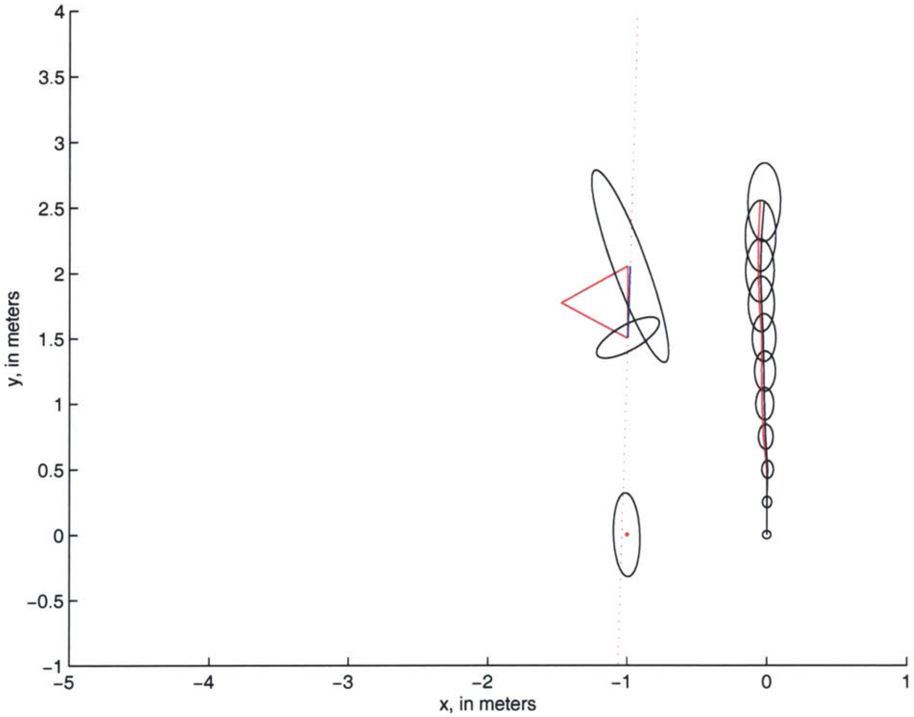


Figure 3-5: Initial map. 99% confidence intervals for the corners and the point object are shown.

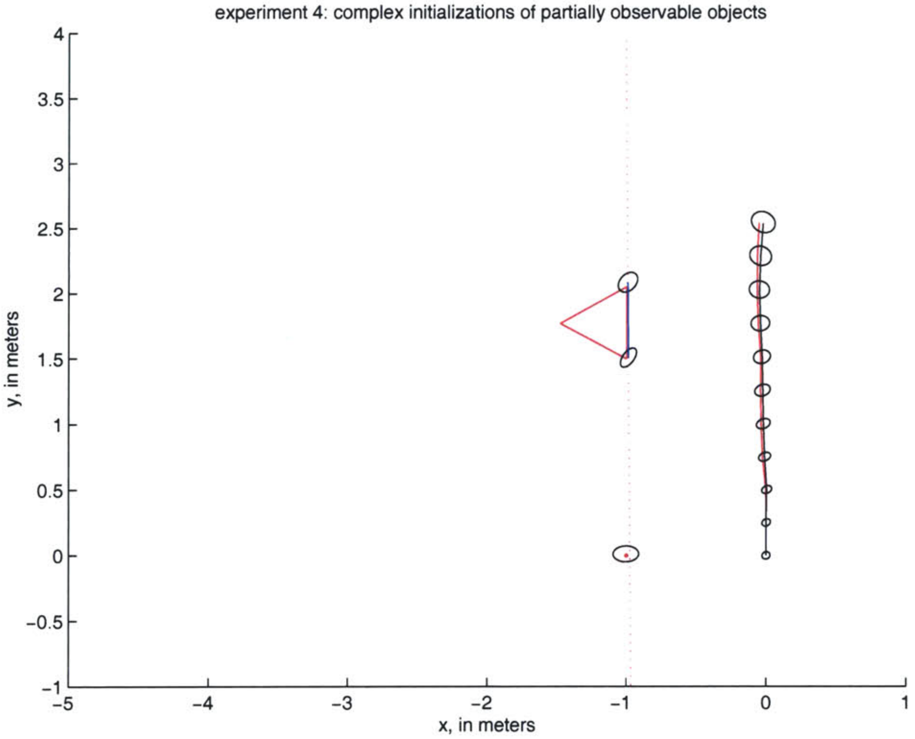


Figure 3-6: Map after a batch update. Note the improved confidence intervals for the features and the robot.

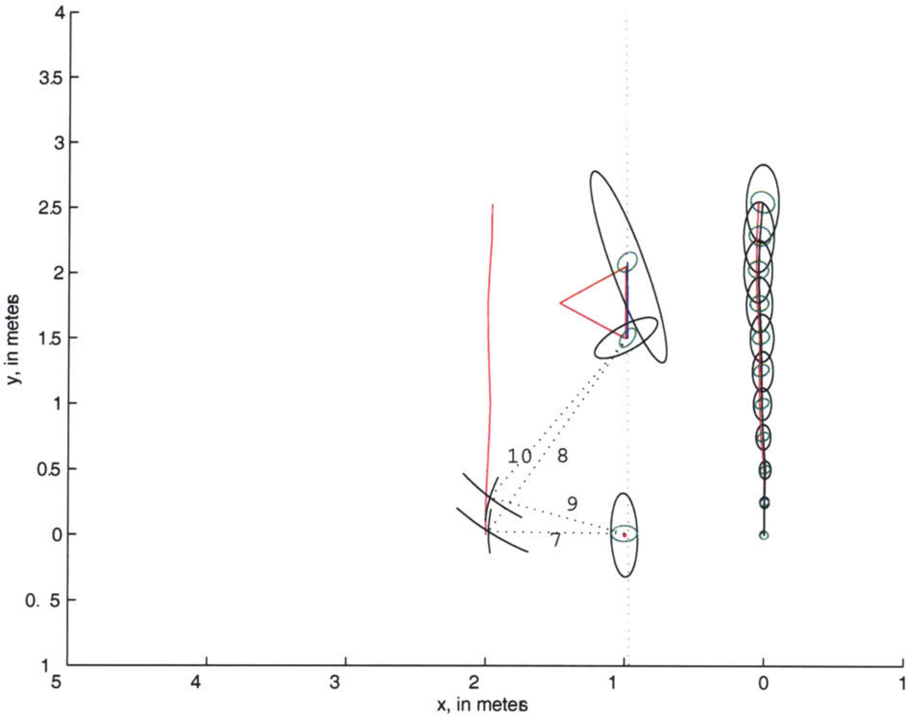


Figure 3-7: Adding in the second robot. The true trajectory for robot 2 is the line on the left. Observations of the bottom corner of the triangle and of the fishing bobber are reversed to find the first two vantage points.

Table 3.3: Comparison of hand-measured and estimated feature locations for points features (in meters).

	Hand measured		CML estimated	
Feature	x	y	x	y
Point object	-1.0	0.0	-1.0054	.0109
Lower right vertex of triangle	-1.0	1.5	-.99	1.5045
Upper right vertex of triangle	-1.0	2.05	-.9922	2.0837
Left vertex of triangle	-1.4763	1.77	-1.4908	1.7846

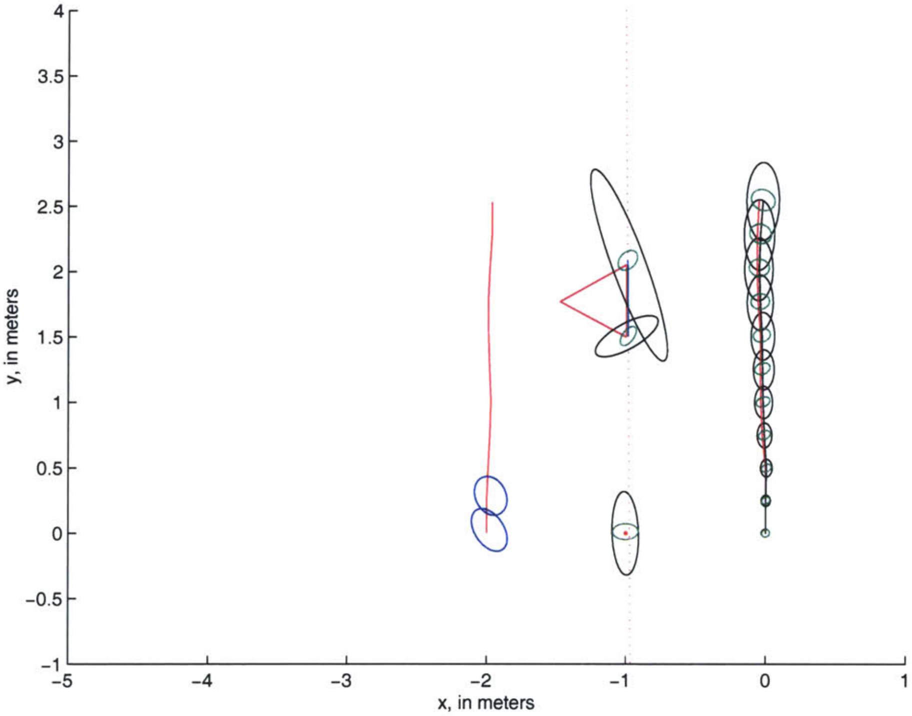


Figure 3-8: First two position for robot 2 are mapped. Using these two positions is is possible to estimate the initial heading and velocity. Confidence intervals for robot 2's positions are shown in blue.

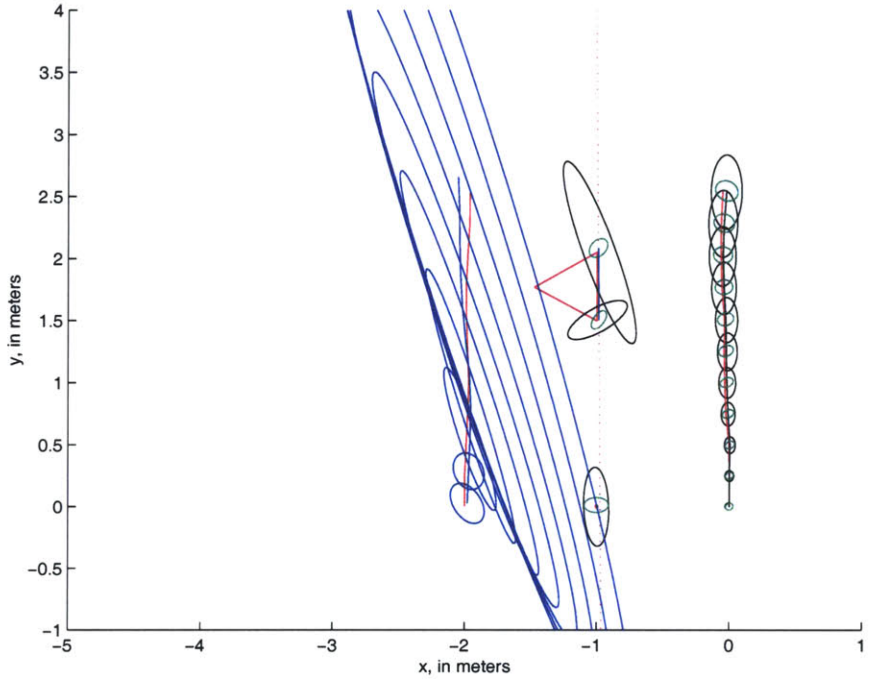


Figure 3-9: Using the estimated initial heading and velocity for robot 2, along with its control inputs, a dead-reckoned trajectory is constructed. With poor initial estimates of heading and velocity, and no compass or velocity measurements for updates, the trajectory is very imprecise.

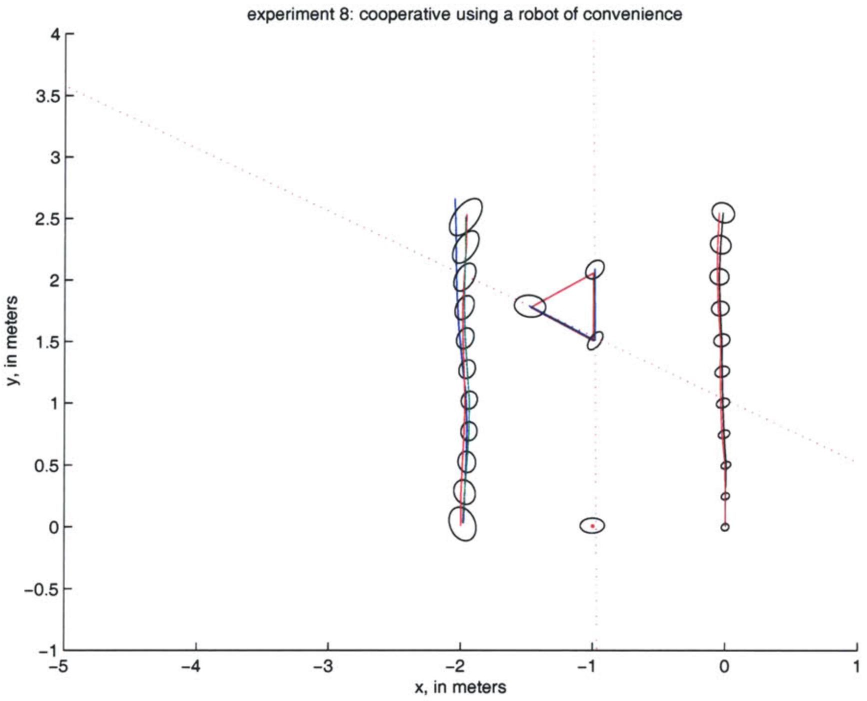


Figure 3-10: Using information from robot 2, robot 1 is able to map the back side of the triangle, which it otherwise could not observe. After the batch update its estimate of robot 2 improves considerably.

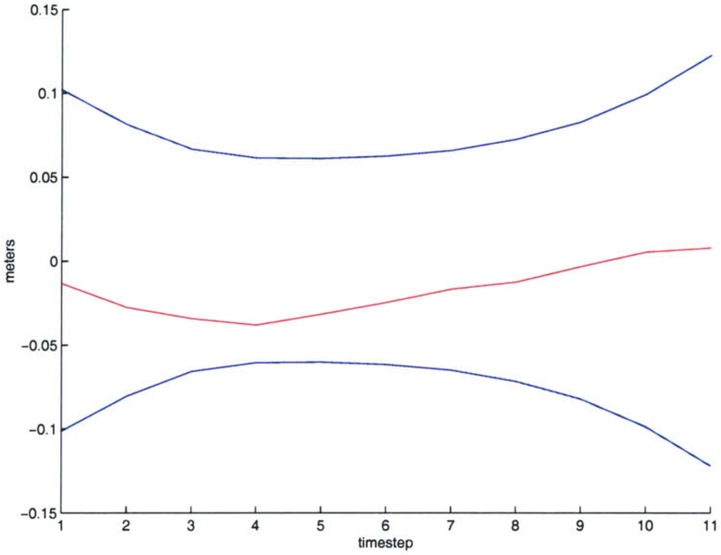
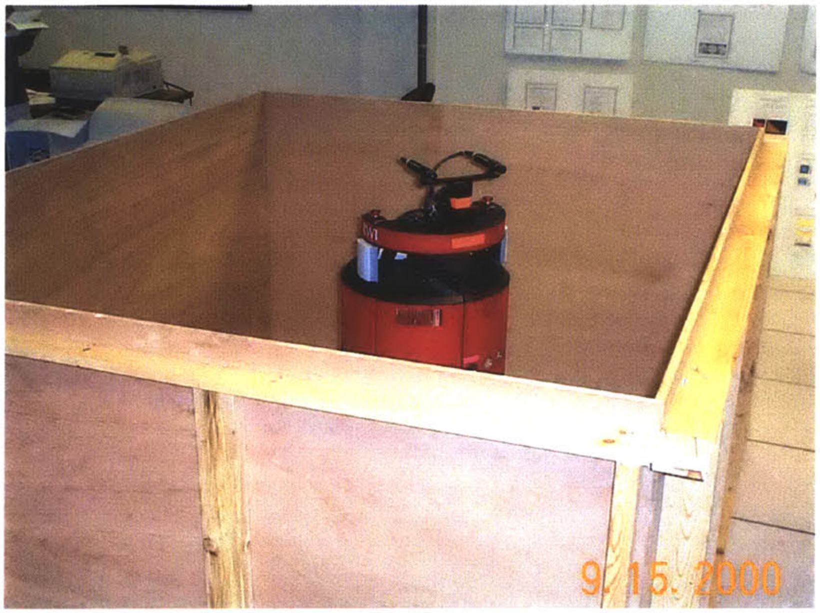


Figure 3-11: Error in the smoothed estimate and three-σ confidence interval for robot 2's x position. The minimum uncertainty is in the middle of the trajectory due to forward/backward smoothing.



9.15.2000

Figure 3-12: B21 mobile robot in the plywood box.

were used to initialize nine features (the four corners and four walls of the box and a prominent "crack" on the bottom wall"). The nine returns and the nine features are each labeled in Figure 3-14, and details of the initialization sequence are shown in Tables 3.4 and 3.5. Azimuth information from the sonar returns was used for gating but not for estimation. The processing proceeded as follows. First, returns 1 and 2 were used to initialize corner 1, using a two position range-only initialization (circle intersection). Next, return 3 was used in conjuction with the state estimate for corner 1 to initialize plane 1, and return 4 was used in conjuction with the state estimate for corner 1 to initialize plane 2. After this, return 5 was used in conjuction with the state estimate for plane 1 to initialize corner 2, and return 6 was used in conjunction with the state estimate for plane 2 to initialize corner 3. Likewise, return 7 was used in conjuction with the state estimate for corner 2 to initialize plane3, and return 8 was used in conjuction with the state estimate for corner 3 to initialize plane 4. Next, return 9 and plane 4 were used to initialized the crack on plane 4. Finally, the state estimates for plane 3 and plane 4 were used to initialize the final feature, corner 4.

After the initializations, nine constrained features (shown in Figure 3-14) were mapped using nine range measurements (shown in Figure 3-15). Once these features were initialized, nearest-neighbor gating was performed between all of the remaining sonar measurements and the newly initialized map features. A total of 217 of the

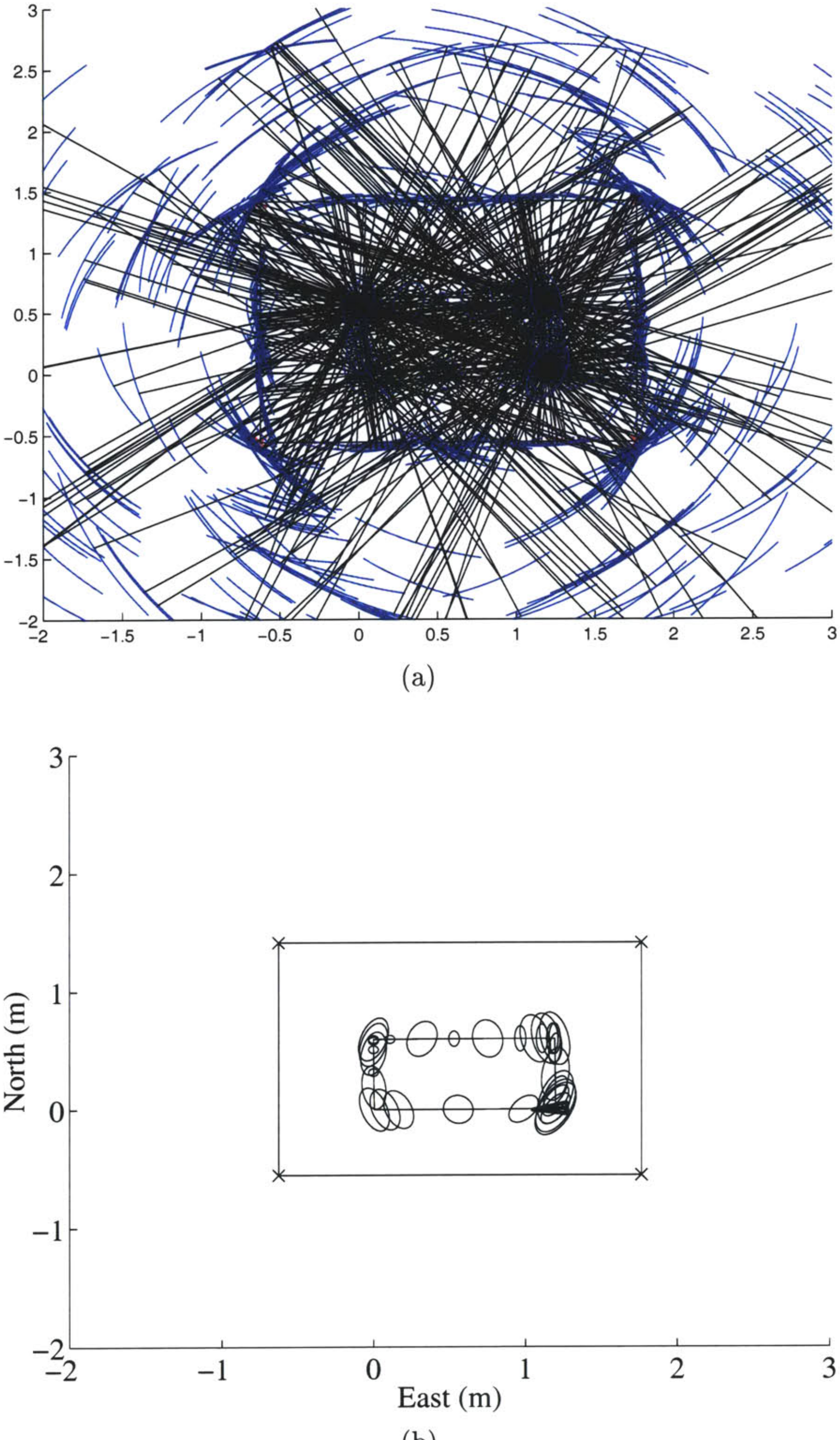
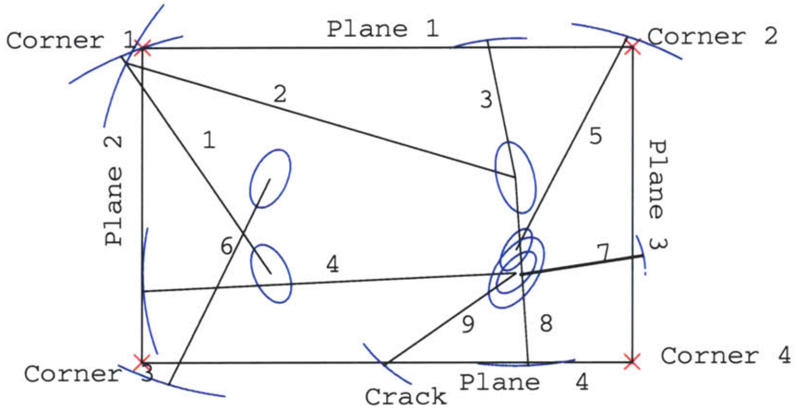


Figure 3-13: (a) Set of all measurements processed, from 50 vehicle positions. Each sonar return is shown as a circular arc with rays drawn from the center of the dead-reckoned robot position to the center of each arc. (b) Dead-reckoned vehicle trajectory with 3-σ error ellipses.



(b)

Figure 3-14: Nine measurements used to initialize nine new features, starting with the corner in the upper left of the figure, building in both directions around the room, and closing the box in the lower right hand corner. Three-sigma error ellipses are shown for the dead-reckoned vehicle positions for each of the returns.

Table 3.4: Details for the nine manually-selected returns used for feature initialization.

Return number	Time Step	Odometry X (m)	Odometry Y (m)	Odometry Heading (deg)	Range (m)	Azimuth (deg)
1	27	.0058	.0002	-3.8264	1.5486	-.3927
2	44	1.1949	.5997	-7.8248	2.0303	4.3197
3	44	1.1949	.5997	-7.8248	.8706	3.2725
4	49	1.1998	.0044	-8.9705	1.8202	-.3927
5	16	1.1990	.1490	-1.5643	1.4352	2.7489
6	36	.0004	.5950	-6.2571	1.3806	-1.9635
7	21	1.1995	.0063	-3.1013	.6280	3.272 5
8	42	1.1949	.5997	-7.1450	1.1788	-.6545
9	50	1.1998	.0044	-9.3471	.8658	.6545

Table 3.5: Method of initialization for the feature primitives and comparison of hand-measured and actual locations for the four corners of the box.

		Hand measured		CML estimated	
Feature	Initialization Method	x	y	x	y
Corner 1	Returns 1 and 2	-0.6240	1.4153	-0.6564	1.4287
Plane 1	Corner 1 and Return 3				
Plane 2	Corner 1 and Return 4				
Corner 2	Plane 1 and Return 5	1.7652	1.4153	1.7838	1.4605
Corner 3	Plane 2 and Return 6	-0.6240	-0.5550	-0.6050	-0.6243
Plane 3	Corner 2 and Return 7				
Plane 4	Corner 3 and Return 8				
Crack	Plane 4 and Return 9				
Corner 4	Plane 3 and Plane 4	1.7652	-0.5550	1.7652	-0.5550

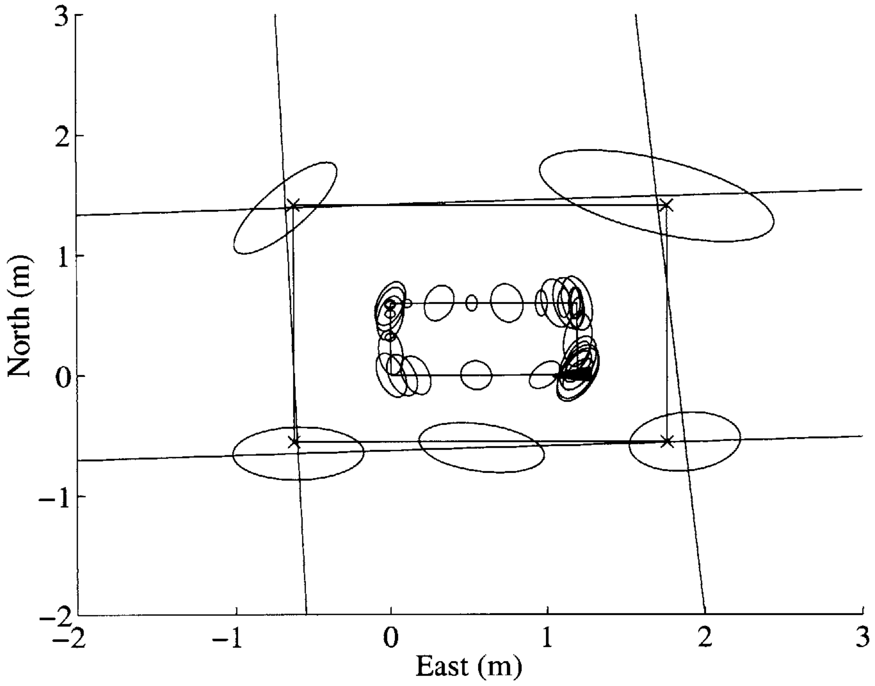


Figure 3-15: State estimates and 3-σ error ellipses for the nine DOF object.

original 600 measurements were uniquely matched to one of the nine features shown in Figure 3-16(a). Finally, Figure 3-16(b) shows the result when all these measurements were applied in a single batch update, resulting in a dramatic reduction in the uncertainty ellipses for the estimated feature locations and in the complete trajectory of the vehicle.

3.7 Integrated Concurrent Mapping and Localization Results

In collaboration with Leonard, Newman, and Bosse [62], the framework described above has been implemented as part of an integrated framework for real-time CML, which incorporates delayed state management, perceptual grouping, multiple vantage point initialization, batch updating, and feature fusion. These results utilize a Hough transform method for grouping and classifying measurements of point and line features kindly made available by J. Tardós, J. Neira, and P. Newman [90].

For these experiments, a fixed size of 40 trajectory time steps was utilized. Every

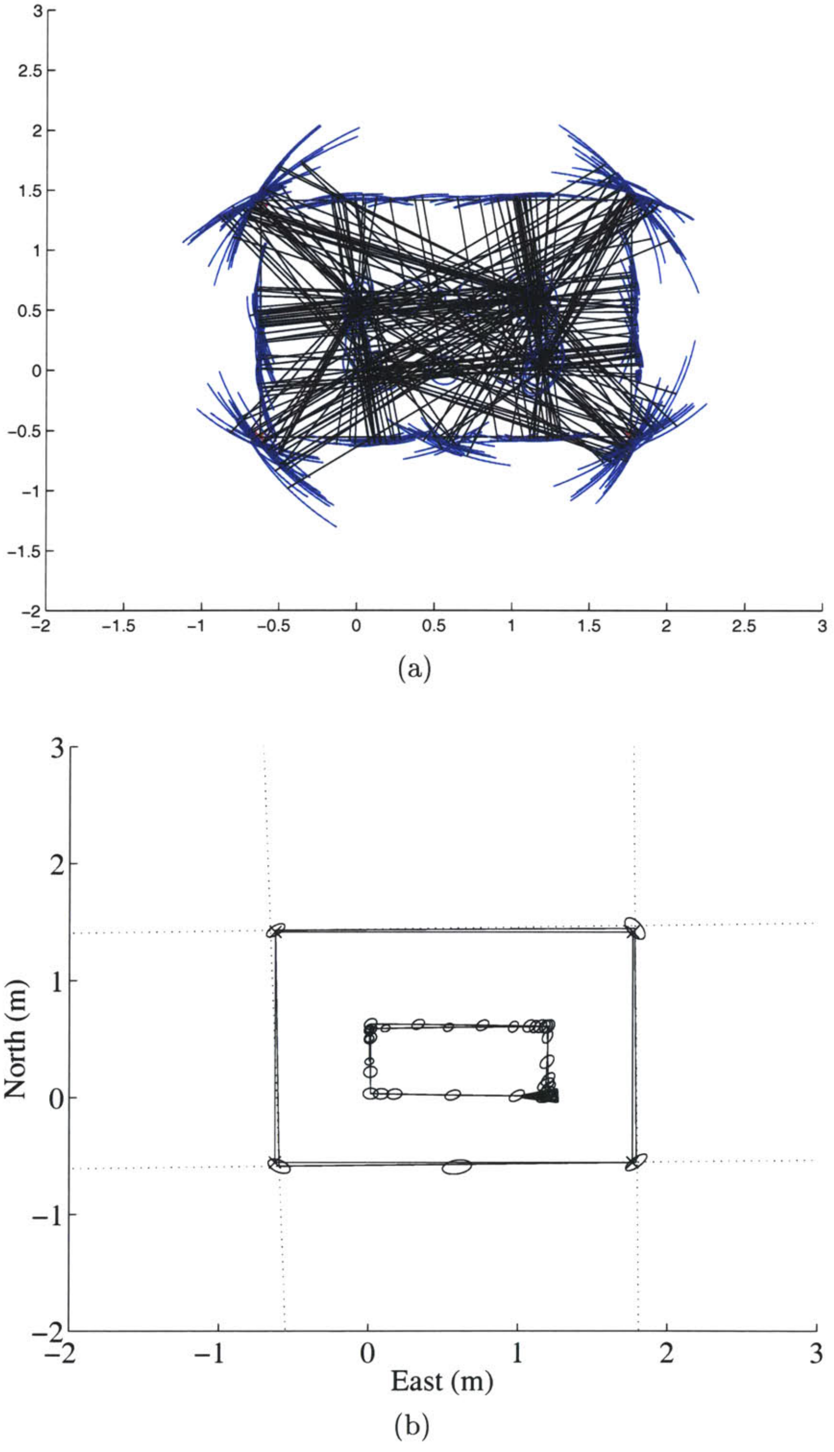
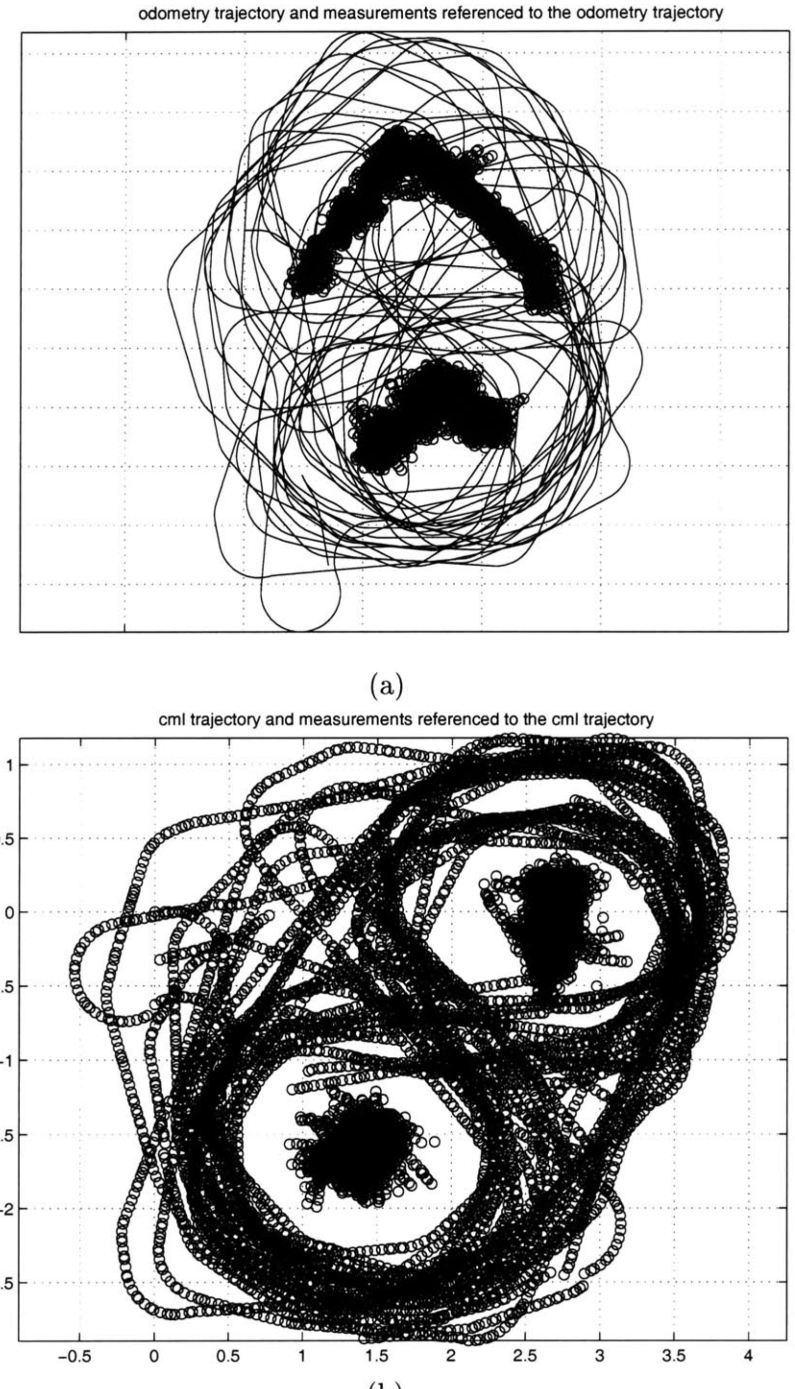


Figure 3-16: (a) Sonar measurements that uniquely gated with the nine initialized feature primitives to be used in the batch update. (b) Feature location estimates, vehicle trajectory, and error ellipses after the batch update.



(a)

(b)

Figure 3-17: (a) Raw sonar data for experiment with two point objects, referenced to odometry. (b) Sonar returns matched to the two features, referenced to the CML estimated trajectory. The experiment was 50 minutes long. The vehicle moved continuously under manual control at a speed of 0.1 meters per second, making about 15 loops of the two cylinders.

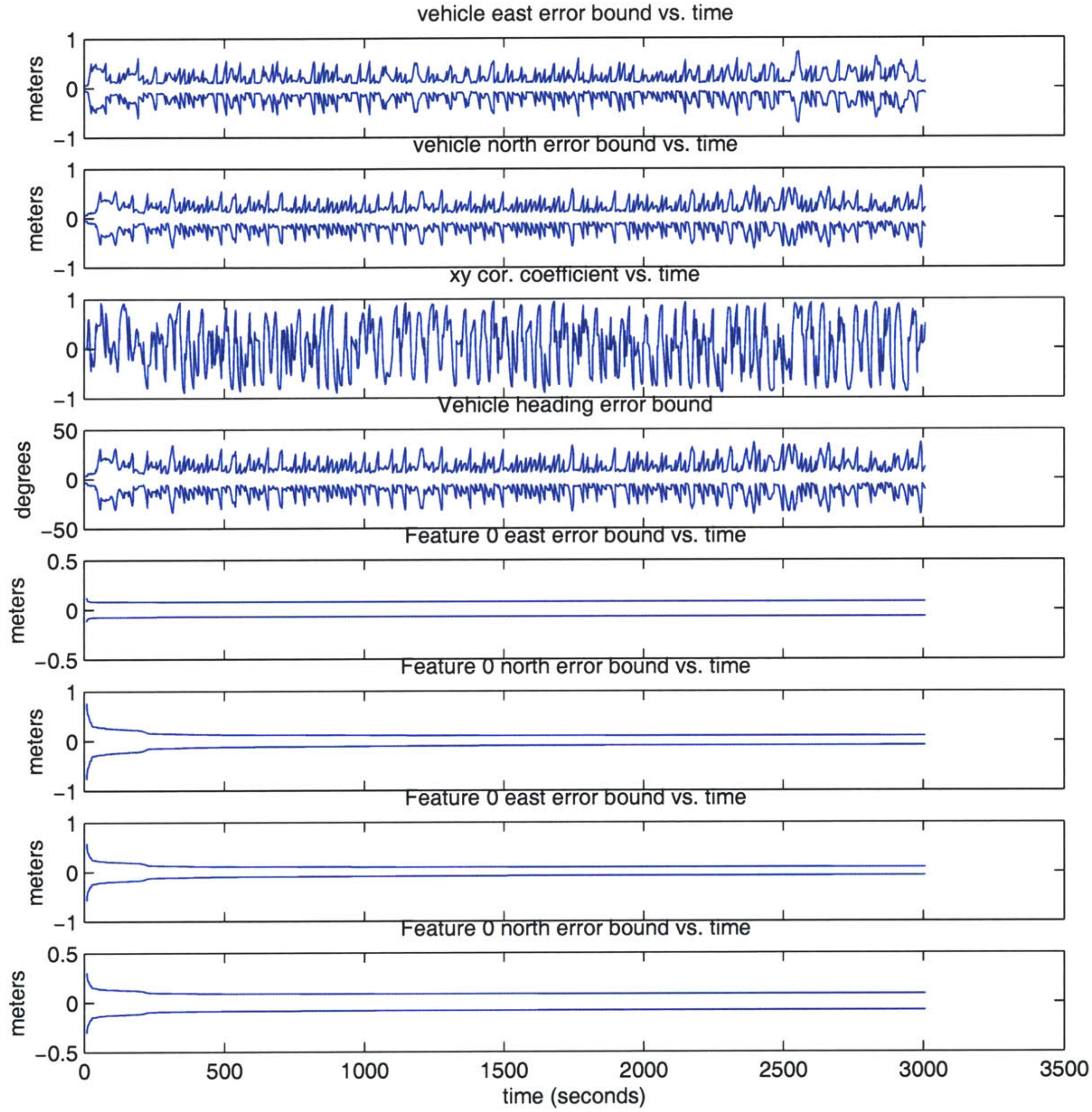


Figure 3-18: Estimated error bounds for the experiment. (Top plots: three-sigma bounds for x and y of the vehicle; next plot: x-y correlation coefficient; next plot: three-sigma bounds for vehicle heading; bottom four plots: three-sigma bounds for the x and y locations of the two features. While there is no ground-truth for this experiment, the vehicle returned to within a few inches of the start position, commensurate with the CML algorithm state estimation error.

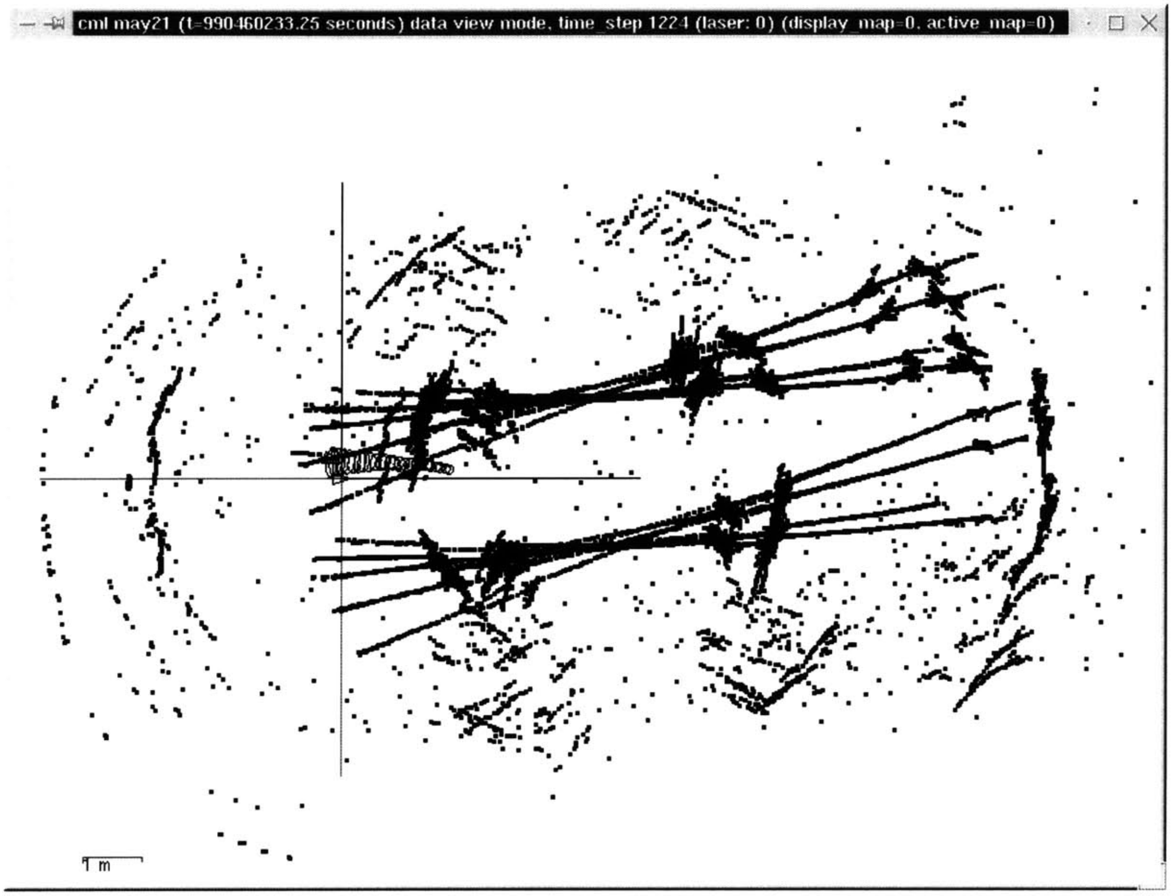
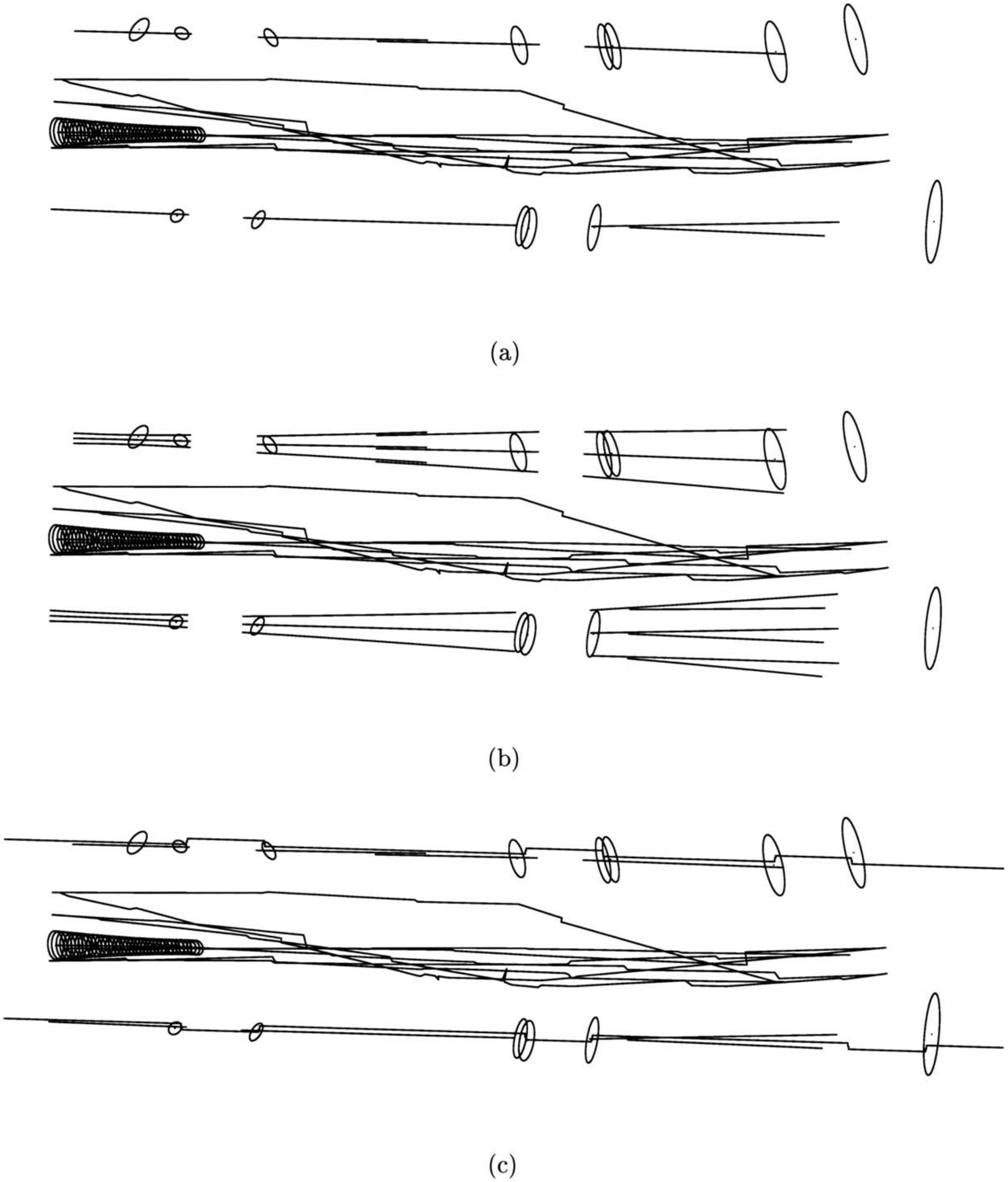

cml may21 (t=990460233.25 seconds) data view mode, time_step 1224 (laser: 0) (display_map=0, active_map=0)
1 m

Figure 3-19: Raw data for corridor experiment, referenced to odometry.



(a)

(b)

(c)

Figure 3-20: (a) CML estimated trajectory for corridor scene and estimated map consisting of points and line segments. Three-sigma error bounds are shown for the location of points. (b) Same plot as in (a), but with three-sigma error bounds for lines added. (c) Same plot as in (a), but with hand-measured model overlaid.

40 time steps, perceptual grouping was performed using the sonar returns from the past 40 time steps. In general, a wide variety of strategies for making delayed data association decisions are possible within this framework. In this chapter, we do not attempt to describe a single definitive decision-making policy, rather our goal is to illustrate the process with a few representative examples with sonar data. For the results reported here, we use a Hough transform technique documented in Tardós *et al.* [90]. A brief summary of the technique follows [63].

The data from a standard ring of Polaroid sonar sensors can be notoriously difficult to interpret. This leads many researchers away from a geometric approach to sonar mapping. However, using a physics-based sensor model, the geometric constraints provided by an individual sonar return can be formulated [60]. Each return could originate from various types of features (point, plane, etc.) or could be spurious. For each type of feature, there is a limited range of locations for a potential feature that is possible. Given these constraints, the Hough transform [6] can be used as a voting scheme to identify point and planar features. More detail on this technique is contained in Tardós *et al.* [90]. The method is similar in spirit to the TBF method of Wijk and Christensen [99], but can also directly identify specular planar reflectors from sonar data, which is vital in typical man-made environments with many smooth walls.

The output from the Hough transform gives sets of measurements with a high likelihood of originating from a single point or plane feature. Each candidate set from the Hough typically contains between 10 and 40 sonar returns hypothesized to originate from a new feature. For each candidate set, two returns are chosen to serve as "seed" features for the initialization to be used in the function $\mathbf{g}(\cdot)$, and the remaining returns are used in a batch update. The first of the two "seed" measurements is chosen to be the return in the candidate set that originates from the earliest vehicle position from the set of trajectory states. The other seed measurement is chosen to be the earliest return that, when combined with the first seed measurement, achieves a sufficient minimum baseline for feature initialization (typically 0.6 meters). Once a new feature is initialized, it is discarded if it has too small of a baseline. To successfully distinguish doors from the walls in the corridor experiment shown in Figure 2-1, a minimum valid line length of 1.2 meters is required for adding a feature into the map. (This restriction can be removed if the joint compatibility method of Neira and Tardós [75] is applied.)

For state estimation, one has a choice between two basic strategies: (1) attempt to match individual measurements to pre-existing features, or (2) use measurements exclusively for new feature initialization and batch updating, followed by feature fusion with previously mapped features to obtain error reduction. A hybrid strategy that mixes both policies is also possible. While we have had good success with either (1) only, (2) only, or a mix of both, in this chapter we focus on option (2), namely new feature mapping followed by feature fusion. See Ayache and Faugeras [5], Chong and Kleeman [22], Tardós *et al.* [90] and Williams *et al.* [101] for further discussion of feature fusion. To determine when features should be fused together, we use the Mahalanobis distance and nearest neighbor.

To illustrate the performance of the implementation, we present results from two different simplified settings: one experiment with two point objects only (cylinders of known radii), and one experiment in the corridor shown in Figures 2-1 and 2-2.

The method has also been implemented running in real-time under manual control. To our knowledge, this is the first successful feature-based CML implementation using sonar sensing for which the robot was continually in motion and the CML output was generated in real-time. (Chong and Kleeman [22] implemented sonar-based mapping with a high-performance sonar array that stopped to perform mechanical scanning for each data acquisition cycle.) The method uses the standard Polaroid sonar array on the B21 robot and could be readily ported to any B21 mobile robot. Such a result has not been achieved before because it has not been possible without the expanded representation accounting for temporal correlations described in this chapter.

The approach presented in this chapter also has been used extensively in two other experimental systems. With sonar, using RANSAC for perceptual grouping and the ATLAS framework for scalable mapping [13], Bosse *et al.* have mapped a large portion of the MIT campus and demonstrated closure of large loops, using only sonar and odometry data. Additionally, Bosse *et al.* have used a similar methodology to perform 3-D mapping of vanishing points and 3-D lines from omnidirectional video data [14].

3.8 Related Research

The notion of incorporating segments of the robot trajectory in the state vector (instead of just the current robot state) employed in this chapter is similar in some respects to the work of Thrun [91] and Gutmann and Konolige [40], which also use the vehicle trajectory as one of the key elements of the map representation. In our work, we only save partial segments of the vehicle trajectory, on an "as-needed" basis to resolve data association and feature modeling ambiguity. We believe that it is possible to pose the problem of stochastic mapping without features, using only trajectory states. The basic update operation would be to correlate the observed sensor data from one position with that observed at another position and to formulate a measurement update function $\mathbf{h}(\cdot)$ that involves only trajectory states. For example, Carpenter and Medeiros [20] have reported CML results using multibeam sonar images. Fleischer has employed smoothing to good effect in a stochastic framework for undersea video mosaicking [36].

The methods of Thrun [91] and Gutmann and Konolige [40] can compute position offsets for the robot by correlating the current laser scans with another previously obtained laser scan. A benefit of this type of approach is that the data association problem does not need to be solved for individual sensor measurements. Very impressive experimental results have been obtained with both approaches. With sonar, the raw data is usually too noisy and ambiguous for these correlation-based approaches to work.

Recent work in feature-based CML has shown the importance of maintaining spatial correlations between the state estimates for different features in order to maintain consistent error bounds [21, 28]. The representation of spatial correlations results in an $\mathcal{O}(n^2)$ growth in computational cost [71], where n is the number of features in the environment. This has motivated techniques to address the map scaling problem through spatial and temporal partitioning [27, 61, 39]. The current chapter has not addressed the map scaling problem; however, it provides a framework for increasing the reliability of local map building. It is anticipated that this will greatly expand the range of environments in which CML can be successfully performed. For a given new type of environment, it is essential to establish reliable local mapping before considering the large-scale mapping problem.

An alternative to achieve the sonar mapping results presented here is to use a custom sonar array that can classify and initialize geometric primitives from a single vantage point. The state-of-the-art in this area is exhibited by the work of Kleeman *et al.* [48, 42, 22, 49]. For example, Chong and Kleeman [23] have used custom advanced sonar arrays to very good effect in testing large-scale CML algorithms. Since this was a scanning sonar, the robot has to stop and scan at each location. However, more recently, Heale and Kleeman [42] have demonstrated a small, multi-element sensor that performs rapid classification to enable mapping while moving.

Nevertheless, attempting to perform CML with the standard ring of Polaroid sensors is an interesting and important problem from both a practical and a basic science perspective. The challenges of range-only interpretation explicitly capture many important uncertainty management problems posed by CML. The fundamental essence of sonar as a range-only sensor providing only sparse information is maintained in a manner that can be applied to alternative, more general situations, such as multi-robot mapping.

3.9 Conclusion

This chapter has described a generalized framework for CML that incorporates temporal as well as spatial correlations, allowing features to be initialized from multiple uncertain vantage points. The method has been applied to Polaroid sonar data from a B21 mobile robot, demonstrating the ability to perform CML with sparse and ambiguous data. These experiments illustrate the benefits of adding past vehicle positions to the state vector, enabling stochastic mapping to be performed in situations in which the state of a feature can only by partially observed from a single vehicle position and the ambiguity of individual measurements is high.

In this chapter, we have assumed that the association of measurements to features is understood. In the next chapter, we develop methods for tracking features from wide beam sonar data.

Chapter 4

Trajectory Sonar Perception

We want to determine the minimum information necessary to determine correspondence between measurements of static features. We would like to do this without knowing what or where the feature is. This is appropriate because the modeling problem is greatly simplified if correspondence is known. In this chapter, a method for tracking locally curved objects will be developed, and the tradeoffs between sensor design, robot dynamics, and computational complexity will be analyzed. Two experiments using this approach are presented at the end of the chapter.

4.1 Introduction

Based on prior work on the general perception problem, it is reasonable to expect that it will be extremely difficult to transform raw sonar data into a high level description of the environment in a single step. Applying lessons from the vision community, it would seem reasonable to break the sonar perception problem into a sequence of processing stages. Each stage, or competence, is heavily grounded in physics and performs a limited but well defined transformation efficiently and reliably. To solve the perception problem, the precise sequence of competences must be determined, and each competence must be rigorously established.

Since this is the first attempt at breaking sonar perception down in this manner, any proposed architecture would certainly have speculative aspects. Nevertheless, it is appropriate to propose such an architecture, so that it can be implemented, tested, and refined.

Therefore, for the purposes of this thesis, the following sequence of competences will be assumed. First, there is an active sonar, which transmits and receives some waveform. The raw signal is processed by the first competence, a matched filter. The matched filtered output is passed to the second competence, a beamformer. The beamformed signal is thresholded to create measurement tokens. These measurement tokens are higher level constructs such as range and bearing. Next, measurement tokens are used to track unknown features. In the fifth competence, tracked features are modeled using Mahalanobis testing. In the sixth stage, features are mapped and recognized. In this case, recognition is done by comparing high level features, rather than comparing measurements to features.

The first three competences are basic signal processing functions. The fifth and sixth competences, Mahalanobis testing and mapping groups of temporally separated measurements, were established in the first half of the thesis. The fourth competence, feature tracking, is not rigorously defined and is the subject of this chapter.

Prior approaches have attempted modeling and correspondence in a single step, using forms of clustering. Separate clusters are required for each possible model. Each model type has required a separate cluster. Moving beyond points and planes towards a more general environment, it will be difficult to maintain models for all possible features. It would be desirable to develop generalized techniques that are applicable to all features.

Correspondence and modeling could be considered separate processes. Correspondence can be thought of as the process of grouping measurements of common origins. Modeling can be thought of as the process of determining the properties of the common origin. Because Mahalanobis testing offers the possibility of modeling feature given accurate correspondences, it makes sense to investigate a competence for establishing correspondence - specifically, a competence that can track features without knowing the feature model.

Most prior work tries to extract features from large sets of data, assuming that a feature existence will show up in the data as some large regularity and that in some appropriate measurement space there will be a large cluster of measurements. Most algorithms try to correctly identify these clusters. Unfortunately, the act of projecting measurements into those sorts of spaces often requires the incorporation of navigational uncertainty, implicitly smeering those piles.

Rather than trying to identify clusters, rather than attempting to determine correspondence and models in a single step, and rather than maintain separate hypotheses for different feature types, a simpler approach will be attempted. This section of the thesis will investigate the minimum information needed to establish correspondence between two measurements regardless of feature model.

This chapter will assume that objects are locally curved. Echoes are assumed to reflect normally from targets. If, after moving a small amount, a second measurement is made, it is assumed that there exists a sphere that would cause those two measurements to occur. Such as sphere is described by its center of curvature and its radius of curvature. Points have zero radius of curvature; planes have a radius of curvature of $\pm\infty$. A positive radius of curvature refers to a convex target; a negative radius of curvature refers to a concave target. This model is valid for small perturbations, specular echoes, diffractive edges, and anything that is pointlike.

4.2 A Simple Feature Constraint

One easy way to track features is by overconstraining them. This is analogous to the binary constraints of [38].

We consider a two dimensional environment and a sonar that provides ranges and bearings of specular echoes from targets. If the environment consists entirely of point objects, it would be possible from an initial measurement to predict a subsequent measurement, assuming positions were known. If the sensory degrees of freedom are denoted as m, and the feature degrees of freedom are labelled as n, this is the case where $m = n$, since a two DOF sensor is observing a two DOF feature.

If, instead, there are both points and lines in the world, the features would have three degrees of freedom. The state of a point would be $(x, y, \rho = 0)^T$; the state of a line would be $(n_x, n_y, \rho = \infty)^T$, where n_x and n_y are the vector components of the normal to the line from the origin. With a 2 DOF sensor, it would not be possible to determine the third degree of freedom, point or plane, from a single measurement since $m < n$. If the sensor had a third degree of freedom, so that $m = n$, it might be possible to resolve this difference. In Chong and Kleeman [23], power is used as a third degree of freedom to differentiate walls and corners since edge diffraction is much less intense than specular echoes.

Suppose instead of points and lines, a two dimensional world only has one type of feature: circles. Circles have three degrees of freedom, $(x, y, \rho)^T$. Points are the limiting case where $\rho \to 0$ and planes are the limiting case where $\rho \to \pm\infty$. With range and angle measurements from a single location only, curvature cannot be estimated; the feature state is only partially observable from a single measurement. Because the robot can only use the two geometric degrees of freedom, $m_1 = 2$ and $n = 3$.

A second measurement with m_2 degrees of freedom must lie in some region of the m_2-dimensional measurement space. How well the second measurement can be predicted depends on $n - m_1$. If $n - m_1 = 0$, the second measurement should be perfectly predicted. If $n - m_1 > 0$, there are $n - m_1$ unconstrained degrees of freedom. If $m_2 < n - m_1$, the number of unconstrained degrees of freedom exceeds the number of degrees of second measurement, and no strong constraints can be imposed on it (exempting special circumstances, such as a stationary robot in a static environment). However, if $m_2 > n - m_1$, or $n < m_1 + m_2$, then the second measurement can be constrained to lie upon some function in measurement space. If it is assumed that $m_1 = m_2 = m$, the if $m < n < 2m$, given a prior measurement, it is possible to predict some function that a second measurement must lie on. For the most general case, if $(k-1)m < n < km$, then a minimum of $k - 1$ measurements are needed to find a constraint function for the kth measurement of a feature. In general, to avoid a combinatorial explosion, it is preferable to design a sensor such that $2m > n$. For this reason, we discourage the use of range only sonars, preferring more advanced sensors such as binaural sonars.

Given an initial robot position, an initial range and bearing measurement to a surface, and a second robot position, this constraint can be applied as follows. ρ is the unknown variable in this 2D analysis. First, the state of the circle is calculated as a function of the unknown variable ρ:

$$x_c(\rho) = x_{r1} + (r_1 + \rho)\cos(\theta_{r_1} + \theta_1) \tag{4.1}$$

$$y_c(\rho) = y_{r1} + (r_1 + \rho)\sin(\theta_{r_1} + \theta_1). \tag{4.2}$$

Then, a predicted measurement is made using $(x_c(\rho), y_c(\rho), \rho)$:

$$\hat{r}_2(\rho) = \sqrt{(x_c - x_{r2})^2 + (y_c - y_{r2})^2} - \rho \tag{4.3}$$

$$\hat{\theta}_2(\rho) = \arctan(\frac{y_c - y_{r2}}{x_c - x_{r2}}) - \theta_{r_2} \tag{4.4}$$

assuming that x_c does not equal x_{r2}.

The innovation is calculated, but instead of having a value, it is a function of the unknown variable

$$\nu(\rho) = \begin{bmatrix} r_2 \\ \theta_2 \end{bmatrix} - \begin{bmatrix} \hat{r}_2(\rho) \\ \hat{\theta}_2(\rho) \end{bmatrix}. \tag{4.5}$$

The Mahalanobis distance also becomes a function of the unknown variable ρ.

$$\gamma(\rho) = \nu(\rho)^T \mathbf{S}(\rho)^{-1} \nu(\rho). \tag{4.6}$$

If the minimum value of $\gamma(\rho)$ is below an appropriate threshold, the robot cannot rule out a possible correspondence between two measurements. At this stage, the robot is not trying to accurately assign correspondence, it is merely trying to eliminate the most unlikely correspondences.

If objects are assumed to be locally curved, then sequential observations of such objects can utilize this curvature constraint.

A two dimensional form of this equation, using globally referenced angles, is

$$r_2 = \frac{(x_2 - x_1)(\sin(\theta_{r_2} + \theta_2) - \sin(\theta_{r_1} + \theta_1)) + (y_2 - y_1)(\cos(\theta_{r_2} + \theta_2) - \cos(\theta_{r_1} + \theta_1))}{\sin(\theta_{r_1} - \theta_{r_2} + \theta_1 - \theta_2)} + r_1. \tag{4.7}$$

Unfortunately, this approach requires the feature to be fully observed. It assumes that all aspects of a feature can be observed and that measurements are a function of all aspects. However, in some cases, far fewer aspects of a feature are observable.

For instance, if the robot drives directly towards a target, the reflection point does not change. Only that reflection point is observable; however, it is also sufficient to predict future measurements provided the robot continues on its path. Processing measurements while only observing part of a target's state is examined in the next section.

4.3 Trajectory Perception

Trajectory Perception models the geometric relationship between a robot and a feature as a potential field. This is similar to the derivation for Optical Flow [44]. As a moving observer passes through a field, the rate at which measurements of the field change is known as the substantial derivative [76]. For a field Φ, the substantial derivative would be defined as

$$\frac{D\Phi}{Dt} = \frac{\partial \Phi}{\partial t} + \dot{\mathbf{x}} \cdot \frac{\partial \Phi}{\partial \mathbf{x}}. \tag{4.8}$$

Given a previous measurement of Φ_{i-1}, Φ_i could be predicted through a Taylor series expansion of substantial derivatives

$$\Phi_i = \Phi_{i-1} + \frac{D\Phi_{i-1}}{Dt}\Delta t + \frac{D^2\Phi_{i-1}}{Dt^2}\frac{\Delta t^2}{2!} + \text{h.o.t.} \tag{4.9}$$

Suppose an initial observation of a target is $[r_0\ \theta_0\ \phi_0]^T$. A subsequent measurement could be predicted through a Taylor series expansion of substantial derivatives.

$$\begin{bmatrix} r_1 \\ \theta_1 \\ \phi_1 \end{bmatrix} = \begin{bmatrix} r_0 + \Delta t \frac{Dr_0}{Dt} + \frac{\Delta t^2}{2!}\frac{D^2 r_0}{Dt^2} + h.o.t. \\ \theta_0 + \Delta t \frac{D\theta_0}{Dt} + h.o.t. \\ \phi_0 + \Delta t \frac{D\phi_0}{Dt} + h.o.t. \end{bmatrix}. \tag{4.10}$$

The Taylor series for range includes a second order term. As will be shown later, the second substantial derivative of range is equivalent to the first substantial derivatives of the angles. The second substantial derivative of range is

$$\frac{D^2 r_m}{Dt} = \frac{\partial^2 r_m}{\partial t^2} + 2\dot{\mathbf{x}} \cdot \nabla \frac{\partial r_m}{\partial t} + \ddot{\mathbf{x}} \cdot \nabla r_m + \dot{\mathbf{x}}^T(\nabla \nabla^T r_m)\dot{\mathbf{x}}. \tag{4.11}$$

4.3.1 2D Trajectory Perception

Rather than jump directly to a six degree of freedom robot, a simplified two dimensional case will be used to demonstrate the concept. Assume a simple, streamlined, non-holonomic robot. The robot's state describes its (x, y) position, its orientation θ_r, its velocity V, its yaw rate $\dot{\theta}_r$, and its acceleration $\dot{V}$.

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ \theta_r \\ V \\ \dot{\theta}_r \\ \dot{V} \end{bmatrix} . \tag{4.12}$$

The robot is assumed to have only a forward velocity and acceleration. It has neither vertical (heave) nor side (sway) velocities.

Assume it is observing a locally curved target with center at (x_c, y_c) and radius ρ. The measured range r_m is then

$$r_m = \sqrt{(x_c - x)^2 + (y_c - y)^2} - \rho \tag{4.13}$$

while the range to the center of curvature is

$$r_c = \sqrt{(x_c - x)^2 + (y_c - y)^2}. \tag{4.14}$$

The radius of curvature is unknown. So while it is possible to measure r_m, the distance $r_c = r_m - \rho$ cannot be directly measured. Nevertheless, r_c is mentioned because it arises frequently in calculations.

The target bearing θ depends on the location of the robot with respect to the target and the robot's orientation θ_r

$$\theta = \arctan(\frac{y_c - y}{x_c - x}) - \theta_r. \tag{4.15}$$

The first substantial derivative of range will be examined initially. Noting that the x and y velocities are $V\cos(\theta_r)$ and $V\sin(\theta_r)$ respectively, the first substantial derivative is

$$\frac{Dr_m}{Dt} = \frac{\partial r_m}{\partial t} + \dot{\mathbf{x}} \cdot \nabla r_m = \frac{\partial r_m}{\partial t} + V\cos(\theta_r)\frac{\partial r_m}{\partial x} + V\sin(\theta_r)\frac{\partial r_m}{\partial y}. \tag{4.16}$$

Examining the relevant derivatives, it is first assumed that $\frac{\partial r_m}{\partial t}$ is zero. This assumes the target to be static and the water column to have temporal constant properties. Examining the derivatives with respect to x and y, we immediately see the unobservable $1/r_c$ term. However, when combined with the numerator, these derivatives become trigonometric functions of the bearing, so they can be forced to drop out.

$$\frac{\partial r_m}{\partial x} = -\frac{x_c - x}{\sqrt{(x_c - x)^2 + (y_c - y)^2}} = -\frac{x_c - x}{r_c} = -\cos(\theta_r + \theta) \tag{4.17}$$

$$\frac{\partial r_m}{\partial y} = -\frac{y_c - y}{\sqrt{(x_c - x)^2 + (y_c - y)^2}} = -\frac{y_c - y}{r_c} = -\sin(\theta_r + \theta). \tag{4.18}$$

Substituting, the range derivative becomes

$$\frac{Dr_m}{Dt} = 0 - V\cos(\theta_r)\cos(\theta_r + \theta) - V\sin(\theta_r)\sin(\theta_r + \theta) \tag{4.19}$$

$$\frac{Dr_m}{Dt} = -V\cos(\theta). \tag{4.20}$$

This is intuitively correct. We would expect that the change in range would depend on the robot velocity and the target bearing, but not global quantities.

The second substantial derivative of range is more complicated. In a compact form, it is

$$\frac{D^2 r_m}{Dt} = \frac{\partial^2 r_m}{\partial t^2} + 2\dot{x} \cdot \nabla \frac{\partial r_m}{\partial t} + \ddot{\mathbf{x}} \cdot \nabla r_m + \dot{\mathbf{x}}^T(\nabla\nabla^T r_m)\dot{\mathbf{x}}. \tag{4.21}$$

As an initial simplification, assume the feature to be static and the water column to have constant properties so the $\frac{\partial}{\partial t}$ terms can be eliminated. The second substantial derivative becomes

$$\frac{D^2 r_m}{Dt} = \ddot{\mathbf{x}} \cdot \nabla r_m + \dot{\mathbf{x}}^T(\nabla\nabla^T r_m)\dot{\mathbf{x}}. \tag{4.22}$$

Expanding this yields

$$\frac{D^2 r_m}{Dt} = \ddot{x}\frac{\partial r_m}{\partial x} + \ddot{y}\frac{\partial r_m}{\partial y} + \dot{x}^2\frac{\partial^2 r_m}{\partial x^2} + 2\dot{x}\dot{y}\frac{\partial^2 r_m}{\partial x \partial y} + \dot{y}^2\frac{\partial^2 r_m}{\partial y^2}. \tag{4.23}$$

The relevant velocities and accelerations are

$$\dot{x} = V\cos(\theta_r) \tag{4.24}$$

$$\dot{y} = V\sin(\theta_r) \tag{4.25}$$

$$\ddot{x} = -V\dot{\theta}_r\sin(\theta_r) + \dot{V}\cos(\theta_r) \tag{4.26}$$

$$\ddot{y} = V\dot{\theta}_r\cos(\theta_r) + \dot{V}\sin(\theta_r). \tag{4.27}$$

The first partial derivatives are the same as those used to calculate the first substantial derivative, but three second partial derivatives are needed. They are

$$\frac{\partial^2 r_m}{\partial x^2} = \frac{(y_c - y)^2}{((x_c - x)^2 + (y_c - y)^2)^{3/2}} = \frac{\sin^2(\theta_r + \theta)}{\sqrt{(x_c - x)^2 + (y_c - y)^2}} = \frac{\sin^2(\theta_r + \theta)}{r_c} \tag{4.28}$$

$$\frac{\partial^2 r_m}{\partial y^2} = \frac{(x_c - x)^2}{((x_c - x)^2 + (y_c - y)^2)^{3/2}} = \frac{\cos^2(\theta_r + \theta)}{\sqrt{(x_c - x)^2 + (y_c - y)^2}} = \frac{\cos^2(\theta_r + \theta)}{r_c} \tag{4.29}$$

$$\frac{\partial^2 r_m}{\partial x \partial y} = -\frac{(x_c - x)(y_c - y)}{((x_c - x)^2 + (y_c - y)^2)^{3/2}} = -\frac{\cos(\theta_r + \theta)\sin(\theta_r + \theta)}{\sqrt{(x_c - x)^2 + (y_c - y)^2}} = -\frac{\cos(\theta_r + \theta)\sin(\theta_r + \theta)}{r_c}. \tag{4.30}$$

Solving for $\frac{D^2 r_m}{Dt^2}$ yields

$$\frac{D^2 r_m}{Dt^2} = \frac{V^2\sin^2(\theta)}{r_c} - V\dot{\theta}_r\sin(\theta) - \dot{V}\cos(\theta). \tag{4.31}$$

Again, the relative motion of the stationary object depends entirely on the relative position, not absolute position, which is intuitively correct.

The first substantial derivative of bearing is derived similarly to that of range. Again, the bearing θ is

$$\theta = \arctan(\frac{y_c - y}{x_c - x}) - \theta_r. \tag{4.32}$$

The first substantial derivative of bearing is

$$\frac{D\theta}{Dt} = \frac{\partial\theta}{\partial t} + \dot{\mathbf{x}} \cdot \nabla\theta. \tag{4.33}$$

Assuming the partial derivative with respect to time to be zero, the first substantial derivative is

$$\frac{D\theta}{Dt} = \frac{\partial\theta}{\partial x}\dot{x} + \frac{\partial\theta}{\partial y}\dot{y} + \frac{\partial\theta}{\partial\theta_r}\dot{\theta}_r. \tag{4.34}$$

In this case, since the substantial derivative is a chain rule expansion, and since θ_r is a state variable that is relevant to the calculation of the bearing, a partial derivative with respect to θ_r is included.

The relevant partial derivatives are

$$\frac{\partial\theta}{\partial x} = \frac{y_c - y}{(x_c - x)^2 + (y_c - y)^2} = \frac{\sin(\theta_r + \theta)}{\sqrt{(x_c - x)^2 + (y_c - y)^2}} = \frac{\sin(\theta_r + \theta)}{r_c}. \tag{4.35}$$

$$\frac{\partial\theta}{\partial y} = -\frac{x_c - x}{(x_c - x)^2 + (y_c - y)^2} = -\frac{\cos(\theta_r + \theta)}{\sqrt{(x_c - x)^2 + (y_c - y)^2}} = -\frac{\cos(\theta_r + \theta)}{r_c}. \tag{4.36}$$

$$\frac{\partial\theta}{\partial\theta_r} = -1. \tag{4.37}$$

Substituting, the first substantial derivative in bearing is found to be

$$\frac{D\theta}{Dt} = \frac{V\sin(\theta)}{\sqrt{(x_c - x)^2 + (y_c - y)^2}} - \dot{\theta}_r = \frac{V\sin(\theta)}{r_c} - \dot{\theta}_r. \tag{4.38}$$

Revisiting the second substantial derivative of range and moving the terms around, it is apparent that the equation contains the first substantial derivative of bearing. Fortuitously, it is the part that contains the unobservable $1/r_c$ term.

$$\frac{D^2 r_m}{Dt^2} = (\frac{V\sin(\theta)}{r_c} - \dot{\theta}_r)V\sin(\theta) - \dot{V}\cos(\theta). \tag{4.39}$$

Substituting, a second trajectory perception constraint is found

$$\frac{D^2 r_m}{Dt^2} = V\sin(\theta)\frac{D\theta}{Dt} - \dot{V}\cos(\theta). \tag{4.40}$$

4.3.2 2D Trajectory Constraint

Having established the equations governing the evolution of measurements, we want to test whether a set of measurements are internally consistent.

Assume three measurements (r_0, θ_0), (r_1, θ_1), and (r_2, θ_2). The two constraints are

$$\frac{Dr_m}{Dt} = -V\cos(\theta) \tag{4.41}$$

and

$$\frac{D^2 r_m}{Dt^2} = V\sin(\theta)\frac{D\theta}{Dt} - \dot{V}\cos(\theta). \tag{4.42}$$

By fitting a parabola to the three range measurements and using least squares to fit a line through the three bearing measurements, $\frac{Dr_m}{Dt}$, $\frac{D^2 r_m}{Dt^2}$, and $\frac{D\theta}{Dt}$ are shown to be

$$\frac{Dr_m}{Dt} = \frac{t_1 - t_2}{(t_0 - t_1)(t_0 - t_2)}r_0 + \frac{t_2^2 - 2t_1t_2 + 2t_0t_1 - t_0^2}{(t_0 - t_1)(t_0 - t_2)(t_2 - t_1)}r_1 + \frac{t_0 - t_1}{(t_2 - t_1)(t_0 - t_2)}r_2. \tag{4.43}$$

$$\frac{D^2 r_m}{Dt^2} = 2\frac{r_2(t_0 - t_1) + r_1(t_2 - t_0) - r_0(t_2 - t_1)}{(t_2 - t_1)(t_0 - t_1)(t_2 - t_0)}. \tag{4.44}$$

$$\frac{D\theta}{Dt} = \frac{(2t_0 - t_1 - t_2)\theta_0 + (2t_1 - t_0 - t_2)\theta_1 + (2t_2 - t_0 - t_1)\theta_2}{2(t_0^2 + t_1^2 + t_2^2 - t_0t_1 - t_0t_2 - t_1t_2)}. \tag{4.45}$$

This leads to a hypothesis test of the form

$$\nu = \begin{bmatrix} \frac{Dr_m}{Dt} + V\cos(\theta) \\ \frac{D^2r}{Dt^2} - V\sin(\theta)\frac{D\theta}{Dt} \end{bmatrix}. \tag{4.46}$$

4.3.3 3D Trajectory Perception

Unlike the two dimensional case, for the three dimensional case, the target will be placed in body coordinates. Its motion with respect to the robot will be modeled. The robot will be at the origin of the frame of reference, and the target will be assumed to be static. The center of curvature of the locally curved surface is at

$$\mathbf{x_c} = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}. \tag{4.47}$$

The center of curvature will appear to move as the robot translates and rotates. The apparent velocity of the center will be denoted as

$$\dot{\mathbf{x}}_\mathbf{c} = \begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{z}_c \end{bmatrix}. \tag{4.48}$$

The apparent velocity due to translation is

$$\mathbf{v_a} = \begin{bmatrix} u_a \\ v_a \\ w_a \end{bmatrix}. \tag{4.49}$$

The apparent velocity $\mathbf{v_a}$ is the negative of the robot velocity $\mathbf{v}$.

The apparent roll, pitch, and yaw of the particle with respect to the vehicle (rotation around the x_c, y_c and z_c axes) are denoted as p_a, q_a, and r_a, and the vector $\omega_\mathbf{a}$ will be the vector of apparent rotation rates

$$\omega_\mathbf{a} = \begin{bmatrix} p_a \\ q_a \\ r_a \end{bmatrix}. \tag{4.50}$$

The apparent rotation rate $\omega_{\mathbf{a}}$ is the negative of the true robot angular rate ω.

The apparent velocity of the center of curvature due to rotation is $\omega_{\mathbf{a}} \times \mathbf{x_c}$. Combined with the apparent translation velocity, this yields the apparent velocity

$$\dot{\mathbf{x}}_{\mathbf{c}} = \mathbf{v_a} + \omega_{\mathbf{a}} \times \mathbf{x_c}. \tag{4.51}$$

Expanded, the apparent velocities are

$$\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{z}_c \end{bmatrix} = \begin{bmatrix} u_a \\ v_a \\ w_a \end{bmatrix} + \begin{bmatrix} q_a z_c - r_a y_c \\ r_a x_c - p_a z_c \\ p_a y_c - q_a x_c \end{bmatrix}. \tag{4.52}$$

The apparent acceleration of the center of curvature can be similarly derived.

$$\ddot{\mathbf{x}}_{\mathbf{c}} = \dot{\mathbf{v}}_{\mathbf{a}} + \omega \times \mathbf{v_a} + \dot{\omega}_{\mathbf{a}} \times \mathbf{x_c} + \omega_{\mathbf{a}} \times (\omega_{\mathbf{a}} \times \mathbf{x_c}) \tag{4.53}$$

Expanded, the accelerations become

$$\begin{bmatrix} \ddot{x}_c \\ \ddot{y}_c \\ \ddot{z}_c \end{bmatrix} = \begin{bmatrix} \dot{u}_a + q_a w_a - r_a v_a + \dot{q}_a z_c - \dot{r}_a y_c + (q_a y_c + r_a z_c) p_a - (q_a^2 + r_a^2) x_c \\ \dot{v}_a + r_a u_a - p_a w_a + \dot{r}_a x_c - \dot{p}_a z_c + (p_a x_c + r_a z_c) q_a - (p_a^2 + r_a^2) y_c \\ \dot{w}_a + p_a v_a - q_a u_a + \dot{p}_a y_c - \dot{q}_a x_c + (p_a x_c + q_a y_c) r_a - (p_a^2 + q_a^2) z_c \end{bmatrix}. \tag{4.54}$$

4.3.4 3D First Substantial Derivative Range

The measured range to the target, r_m, is the distance to the center of curvature, r_c, minus the radius ρ.

$$r_m = r_c - \rho = \sqrt{x_c^2 + y_c^2 + z_c^2} - \rho. \tag{4.55}$$

The rate at which the measured range changes, the first substantial derivative, is

$$\frac{Dr_m}{Dt} = \frac{\partial r_m}{\partial t} + \dot{\mathbf{x}}_{\mathbf{c}} \cdot \nabla r_m = \frac{\partial r_m}{\partial t} + \dot{x}_c \frac{\partial r_m}{\partial x_c} + \dot{y}_c \frac{\partial r_m}{\partial y_c} + \dot{z}_c \frac{\partial r_m}{\partial z_c}. \tag{4.56}$$

By assuming a static target and a water column with constant properties, the first term can be set to zero

$$\frac{\partial r_m}{\partial t} = 0. \tag{4.57}$$

The denominator of the partial derivatives of r_m with respect to x_c, y_c, and z_c is $\sqrt{x_c^2 + y_c^2 + z_c^2}$, which is unobservable, since the distance to the center of curvature is the measured velocity plus the unknown radius of curvature. The respective numerators are also unobservable. However, combined, we can easily recognize each quantity as a trigonometric functions of the azimuth and elevation, which are observable.

$$\frac{\partial r_m}{\partial x_c} = \frac{x_c}{\sqrt{x_c^2 + y_c^2 + z_c^2}} = \cos(\theta)\sin(\phi). \tag{4.58}$$

$$\frac{\partial r_m}{\partial y_c} = \frac{y_c}{\sqrt{x_c^2 + y_c^2 + z_c^2}} = \sin(\theta)\sin(\phi). \tag{4.59}$$

$$\frac{\partial r_m}{\partial z_c} = \frac{z_c}{\sqrt{x_c^2 + y_c^2 + z_c^2}} = \cos(\phi). \tag{4.60}$$

Combining these terms, the first substantial derivative in range becomes

$$\frac{Dr_m}{Dt} = \dot{x}_c\cos(\theta)\sin(\phi) + \dot{y}_c\sin(\theta)\sin(\phi) + \dot{z}_c\cos(\phi). \tag{4.61}$$

.

Substituting in the equations for the velocities, the rotational rates drop out, yielding

$$\frac{Dr_m}{Dt} = u_a\cos(\theta)\sin(\phi) + v_a\sin(\theta)\sin(\phi) + w_a\cos(\phi). \tag{4.62}$$

Intuitively, it makes sense that the first substantial derivative of range is independent of rotational rates. If the robot is stationary, turning should bring it no closer to a static target.

4.3.5 3D First Substantial Derivative in Azimuth

The azimuth angle

$$\theta = \arctan(\frac{y_c}{z_c}) \tag{4.63}$$

has the first substantial derivative

$$\frac{D\theta}{Dt} = \frac{\partial\theta}{\partial t} + \dot{\mathbf{x}}_{\mathbf{c}} \cdot \nabla\theta = \frac{\partial\theta}{\partial t} + \dot{x}_c\frac{\partial\theta}{\partial x_c} + \dot{y}_c\frac{\partial\theta}{\partial y_c} + \dot{z}_c\frac{\partial\theta}{\partial z_c}. \tag{4.64}$$

Again, by assuming a static target, the first term can be set to zero.

$$\frac{\partial\theta}{\partial t} = 0. \tag{4.65}$$

The other partial derivatives are partially observable. Like the partial derivatives of range, trigonometric functions of observable angles can be substituted into the equations. However, the x and y partial derivatives are left with the unobservable radius of curvature in the denominator. Because θ is independent of z, the z partial derivative is zero.

$$\frac{\partial\theta}{\partial x_c} = -\frac{y_c}{x_c^2 + y_c^2} = -\frac{\sin(\theta)}{\sin(\phi)\sqrt{x_c^2 + y_c^2 + z_c^2}} = -\frac{\sin(\theta)}{\sin(\phi)(r_m + \rho)}. \tag{4.66}$$

$$\frac{\partial\theta}{\partial y_c} = \frac{x_c}{x_c^2 + y_c^2} = \frac{\cos(\theta)}{\sin(\phi)\sqrt{x_c^2 + y_c^2 + z_c^2}} = \frac{\cos(\theta)}{\sin(\phi)(r_m + \rho)}. \tag{4.67}$$

$$\frac{\partial\theta}{\partial z_c} = 0. \tag{4.68}$$

Consequently, the substantial derivative of bearing contains the unobservable radius of curvature ρ

$$\frac{D\theta}{Dt} = -\frac{\sin(\theta)u_a - \cos(\theta)v_a}{\sin(\phi)(r_m + \rho)} + \frac{p_a\cos(\theta)\cos(\phi) + q_a\sin(\theta)\cos(\phi)}{\sin(\phi)} - r_a. \tag{4.69}$$

4.3.6 3D First Substantial Derivative of Elevation

The elevation angle

$$\phi = \arccos(\frac{x_c}{\sqrt{x_c^2 + y_c^2 + z_c^2}}) \tag{4.70}$$

has the first substantial derivative

$$\frac{D\phi}{Dt} = \frac{\partial \phi}{\partial t} + \dot{\mathbf{x}}_{\mathbf{c}} \cdot \nabla \phi = \frac{\partial \phi}{\partial t} + \dot{x}_c \frac{\partial \phi}{\partial x_c} + \dot{y}_c \frac{\partial \phi}{\partial y_c} + \dot{z}_c \frac{\partial \phi}{\partial z_c}. \tag{4.71}$$

The relevant partial derivatives are

$$\frac{\partial \phi}{\partial t} = 0. \tag{4.72}$$

$$\frac{\partial \phi}{\partial x_c} = \frac{z_c x_c}{(x_c^2 + y_c^2 + z_c^2)(x_c^2 + y_c^2)^{\frac{1}{2}}} = \frac{\cos(\theta)\cos(\phi)}{r_m + \rho}. \tag{4.73}$$

$$\frac{\partial \phi}{\partial y_c} = \frac{z_c y_c}{(x_c^2 + y_c^2 + z_c^2)(x_c^2 + y_c^2)^{\frac{1}{2}}} = \frac{\sin(\theta)\cos(\phi)}{r_m + \rho}. \tag{4.74}$$

$$\frac{\partial \phi}{\partial z_c} = -\frac{\sqrt{x_c^2 + y_c^2}}{(x_c^2 + y_c^2 + z_c^2)} = -\frac{\sin(\phi)}{r_m + \rho}. \tag{4.75}$$

Substituting, the substantial derivative is

$$\frac{D\phi}{Dt} = \frac{\cos(\phi)\cos(\theta)u_a + \cos(\phi)\sin(\theta)v_a - \sin(\phi)w_a}{r_m + \rho} - p_a \sin(\theta) + q_a \cos(\theta). \tag{4.76}$$

4.3.7 Second Substantial Derivative of Range

The second substantial derivative of range is

$$\frac{D^2 r_m}{Dt} = \frac{\partial^2 r_m}{\partial t^2} + \frac{\partial r_m}{\partial t}(\dot{\mathbf{x}}_{\mathbf{c}} \cdot \nabla r_m) + \ddot{\mathbf{x}}_{\mathbf{c}} \cdot \nabla r_m + \dot{\mathbf{x}}_{\mathbf{c}}^T \nabla \nabla^T r_m \dot{\mathbf{x}}_{\mathbf{c}}. \tag{4.77}$$

Expanded, while assuming a static feature and constant water column, this becomes

$$\begin{aligned} \frac{D^2 r_m}{Dt^2} &= \frac{\partial r_m}{\partial x_c}\dot{u}_a + \frac{\partial r_m}{\partial y_c}\dot{v}_a + \frac{\partial r_m}{\partial z_c}\dot{w}_a \\ &+ \frac{\partial^2 r_m}{\partial x_c^2}u_a^2 + \frac{\partial^2 r_m}{\partial y_c^2}v_a^2 + \frac{\partial^2 r_m}{\partial z_c^2}w_a^2 \\ &+ 2\frac{\partial^2 r_m}{\partial x_c \partial y_c}u_a v_a + 2\frac{\partial^2 r_m}{\partial x_c \partial z_c}u_a w_a + 2\frac{\partial^2 r_m}{\partial y_c \partial z_c}v_a w_a. \end{aligned} \tag{4.78}$$

The relevant second derivatives are

$$\frac{\partial^2 r_m}{\partial x_c^2} = \frac{y_c^2 + z_c^2}{(x_c^2 + y_c^2 + z_c^2)^{\frac{3}{2}}} = \frac{\sin(\theta)^2 \sin(\phi)^2 + \cos(\phi)^2}{r_m + \rho}. \tag{4.79}$$

$$\frac{\partial^2 r_m}{\partial y_c^2} = \frac{x_c^2 + z_c^2}{(x_c^2 + y_c^2 + z_c^2)^{\frac{3}{2}}} = \frac{\cos(\theta)^2 \sin(\phi)^2 + \cos(\phi)^2}{r_m + \rho}. \tag{4.80}$$

$$\frac{\partial^2 r_m}{\partial z_c^2} = \frac{x_c^2 + y_c^2}{(x_c^2 + y_c^2 + z_c^2)^{\frac{3}{2}}} = \frac{\sin(\phi)^2}{r_m + \rho}. \tag{4.81}$$

$$\frac{\partial^2 r_m}{\partial x_c \partial y_c} = -\frac{x_c y_c}{(x_c^2 + y_c^2 + z_c^2)^{\frac{3}{2}}} = -\frac{\cos(\theta)\sin(\theta)\sin(\phi)^2}{\sqrt{x_c^2 + y_c^2 + z_c^2}}. \tag{4.82}$$

$$\frac{\partial^2 r_m}{\partial x_c \partial z_c} = -\frac{x_c z_c}{(x_c^2 + y_c^2 + z_c^2)^{\frac{3}{2}}} = -\frac{\cos(\theta)\sin(\phi)\cos(\phi)}{\sqrt{x_c^2 + y_c^2 + z_c^2}}. \tag{4.83}$$

$$\frac{\partial^2 r_m}{\partial y_c \partial z_c} = -\frac{y_c z_c}{(x_c^2 + y_c^2 + z_c^2)^{\frac{3}{2}}} = -\frac{\sin(\theta)\sin(\phi)\cos(\phi)}{\sqrt{x_c^2 + y_c^2 + z_c^2}}. \tag{4.84}$$

Substituting these terms, the resulting form of the second substantial derivative contains the unobservable radius of curvature.

$$\frac{D^2 r_m}{Dt^2} = \frac{u_a^2(\sin(\theta)^2\sin(\phi)^2 + \cos(\phi)^2) + v_a^2(\cos(\theta)^2\sin(\phi)^2 + \cos(\phi)^2)}{r_m + \rho} \tag{4.85}$$

$$+\frac{w_a^2\sin(\phi)^2 - 2u_a v_a\cos(\theta)\sin(\theta)\sin(\phi)^2 - 2u_a w_a\cos(\theta)\sin(\phi)\cos(\phi) - 2v_a w_a\sin(\theta)\sin(\phi)\cos(\phi)}{r_m + \rho}$$

$$+(\cos(\theta)\sin(\phi)\dot{u}_a + \sin(\theta)\sin(\phi)\dot{v}_a + \cos(\phi)\dot{w}_a$$

$$+\cos(\theta)\sin(\phi)(r_a v_a - q_a w_a) + \sin(\theta)\sin(\phi)(p_a w_a - r_a p_a) + \cos(\phi)(q_a u_a - p_a v_a)).$$

4.3.8 3D Trajectory Constraint

The first substantial derivative in range is observable from state information and measurements. It can be used as a constraint.

$$\frac{Dr_m}{Dt} = u\cos(\theta)\sin(\phi) + v\sin(\theta)\sin(\phi) + v\cos(\phi) \tag{4.86}$$

The first substantial derivatives of azimuth and elevation and the second substantial derivative of range are not fully observable; they are functions of the unknown radius of curvature. However, by combining the three equations, the unknown quantity can be removed. The resulting equation for the second substantial derivative of range is

$$\frac{D^2 r_m}{Dt^2} = \dot{u}_a\cos(\theta)\sin(\phi) + \dot{v}_a\sin(\theta)\sin(\phi) + \dot{w}_a\cos(\phi) \tag{4.87}$$

$$+(-u_a\sin(\theta)\sin(\phi)+v_a\cos(\theta)\sin(\phi))\frac{D\theta}{Dt}+(u_a\cos(\theta)\cos(\phi)+v_a\sin(\theta)\cos(\phi)-w_a\sin(\phi))\frac{D\phi}{Dt}.$$

This equation is a function of the angles θ and ϕ, the apparent velocity $\mathbf{v_a}$, and the apparent acceleration vector $\dot{\mathbf{v}}_\mathbf{a}$. The feature is assumed static. Once again, the apparent velocity $\mathbf{v_a}$ is the negative of the actual robot velocity.

For a non-holonomic, streamlined robot, it may be reasonable to assume that the heave velocity $v = -v_a$, the sway velocity $w = -w_a$, the heave acceleration $\dot{v} = -\dot{v}_a$, and the sway acceleration $\dot{w} = -\dot{w}_a$ are zero. Substituting the robot's forward velocity V for the apparent object velocity $-v_a$ yields a non-holonomic form of the equations

$$\frac{Dr_m}{Dt} = -V\cos(\theta)\sin(\phi) \tag{4.88}$$

and

$$\frac{D^2 r_m}{Dt^2} = -\dot{V}\cos(\theta)\sin(\phi) + V\sin(\theta)\sin(\phi)\frac{D\theta}{Dt} - V\cos(\theta)\cos(\phi)\frac{D\phi}{Dt}. \tag{4.89}$$

Assume three measurements (r_0, θ_0, ϕ_0), (r_1, θ_1, ϕ_1), and (r_2, θ_2, ϕ_2). By fitting a parabola to the three range measurements, and using least squares to fit a line through the three azimuth and elevation measurements, $\frac{Dr_m}{Dt}$, $\frac{D^2 r_m}{Dt^2}$, $\frac{D\theta}{Dt}$, and $\frac{D\phi}{Dt}$ are shown to be

$$\frac{Dr_m}{Dt} = \frac{t_1 - t_2}{(t_0 - t_1)(t_0 - t_2)} r_0 + \frac{t_2^2 - 2t_1 t_2 + 2t_0 t_1 - t_0^2}{(t_0 - t_1)(t_0 - t_2)(t_2 - t_1)} r_1 + \frac{t_0 - t_1}{(t_2 - t_1)(t_0 - t_2)} r_2. \tag{4.90}$$

$$\frac{D^2 r_m}{Dt^2} = 2\frac{r_2(t_0 - t_1) + r_1(t_2 - t_0) - r_0(t_2 - t_1)}{(t_2 - t_1)(t_0 - t_1)(t_2 - t_0)}. \tag{4.91}$$

$$\frac{D\theta}{Dt} = \frac{(2t_0 - t_1 - t_2)\theta_0 + (2t_1 - t_0 - t_2)\theta_1 + (2t_2 - t_0 - t_1)\theta_2}{2(t_0^2 + t_1^2 + t_2^2 - t_0 t_1 - t_0 t_2 - t_1 t_2)}. \tag{4.92}$$

$$\frac{D\phi}{Dt} = \frac{(2t_0 - t_1 - t_2)\phi_0 + (2t_1 - t_0 - t_2)\phi_1 + (2t_2 - t_0 - t_1)\phi_2}{2(t_0^2 + t_1^2 + t_2^2 - t_0 t_1 - t_0 t_2 - t_1 t_2)}. \tag{4.93}$$

This leads to a hypothesis test of the form

$$\nu = \begin{bmatrix} \frac{Dr_m}{Dt} + V\cos(\theta)\sin(\phi) \\ \frac{D^2 r_m}{Dt^2} + \dot{V}\cos(\theta)\sin(\phi) - V\sin(\theta)\sin(\phi)\frac{D\theta}{Dt} + V\cos(\theta)\cos(\phi)\frac{D\phi}{Dt} \end{bmatrix}. \tag{4.94}$$

4.4 Bat Tones

Certain bats have been observed to transmit a constant frequency tone while flying. By transmitting a long tone, they guarantee themselves very high frequency resolution, but little temporal resolution. In other words, they detect Doppler shifts very well, but range poorly if at all.

Muller et al [72, 73] demonstrated that a bat, using the first and second derivatives of the Doppler shift, could determine the range to the target. We will present a method for estimating range from only the first derivative of the Doppler shift.

The Doppler shift is the first substantial derivative of range multiplied by a constant. For the two dimensional non-holonomic case the Doppler shift f_d would be

$$f_d = \frac{f}{c} V\cos(\theta) = -\frac{f}{c}\frac{Dr_m}{Dt} \tag{4.95}$$

where c is the sound speed and f is the frequency of the transmitted tone. Likewise, the substantial derivative of the Doppler shift is

$$\frac{Df_d}{Dt} = \frac{f}{c}(\dot{V}\cos(\theta) - V\sin(\theta)\frac{D\theta}{Dt}) = -\frac{f}{c}\frac{D^2 r_m}{Dt^2} \qquad (4.96)$$

Assuming the Doppler shift rate $\frac{Df_d}{Dt}$ can be measured accurately, the bearing rate $\frac{D\theta}{Dt}$ can be predicted as

$$\frac{D\theta}{Dt} = \frac{\dot{V}\cos(\theta) - \frac{c}{f}\frac{Df_d}{Dt}}{V\sin(\theta)}. \qquad (4.97)$$

Alternatively, the Doppler shift rate can be used to infer the distance to the center of curvature of objects r_c

$$r_c = \frac{V^2\sin^2(\theta)}{\dot{V} + V\dot{\theta}_b - \frac{Df_d}{Dt}\frac{c}{f}} \qquad (4.98)$$

where $\dot{\theta}_b$ is the bat's yaw rate. If points are tracked, the range to the target surface is the same as the range to the center of curvature. In addition to measuring the Doppler shift rate and the bearing to the target, it is assumed that the perceiver knows its velocity. The should be a reasonable assumption. The perceiver ought to be able to judge its velocity based on the highest observed Doppler shift or the Doppler shift of targets directly in its path.

4.5 Non-dimensional Analysis

We wanted to find the minimum information necessary to track targets. What we found was the information necessary to predict the rates at which measurements change. Next we will establish when those rates are necessary and how robot dynamics and sensor configuration affect what is needed to determine those rates.

One immediate idea is to predict that the next measurement of an object will be the same as the last. If the update rate is high, the measurement may not move enough to be observable. For instance, suppose that the sensor's standard deviation in range is σ_r and that measurements are made at a sampling frequency f_s. If $\mid \frac{Dr}{Dt} \mid < \sigma_r f_s$, then the change in the measurement will not be observable. Similar constraints apply to the bearing rates and to the second substantial derivative.

We will assume that the minimum resolvable change in range is a quarter wavelength, $\frac{\lambda}{4}$. If this is true, and if $\mid r_2 - r_1 \mid < \frac{\lambda}{4}$, then no derivatives are necessary and $r_2 \approx r_1$. On the other hand, if $\mid r_2 - r_1 \mid > \frac{\lambda}{4}$, then those higher order terms need to be modeled. Alternatively, we may ask if $\Delta t \frac{Dr}{Dt} > \frac{\lambda}{4}$. Typically, the maximum value of $\frac{Dr}{Dt}$ occurs for objects directly in front of the observer, when $\frac{Dr}{Dt}$ is equal to the negative of the forward velocity. This leads to the nondimensional number N_1, which is the ratio of the distance traveled between pings and the wavelength of the signal, where f_s is the frequency at which the robot pings.

$$N_1 = \frac{V}{\lambda f_s}. \tag{4.99}$$

If the quarter wavelength resolution assumption is valid, and if $N_1 > \frac{1}{4}$, then the first substantial derivative of range should be used. A representative velocity for a dolphin is $1\frac{m}{s}$, a representative ping rate is 10 Hz, and a representative frequency is 100kHz, which has a wavelength of .015m. So, for a dolphin, $N_1 \approx 7 \gg \frac{1}{4}$. One might object that $1\frac{m}{s}$ is slow for a dolphin, but a faster dolphin would only have a larger N_1, pushing it further into this regime. For a dolphin to have $N_1 = \frac{1}{4}$, the velocity would have to be $.0375\frac{m}{s}$, which is very slow.

Similarly, the second substantial derivative of range will be necessary when its contribution exceeds a quarter wavelength.

$$\frac{\Delta t^2}{2!}\frac{D^2 r}{Dt^2} > \frac{\lambda}{4}. \tag{4.100}$$

The second substantial derivative of range can be thought of as having three parts. The first applies to a constant velocity robot driving straight. The second describes a turning robot. The third describes an accelerating robot. For a streamlined non-holonomic robot, the first part is maximized when the target is at broadside. Normalizing with respect to the wavelength, a non-dimensional number describing this quantity would be

$$N_2 = \frac{V^2}{2 f_s^2 \lambda r_c}. \tag{4.101}$$

If N_2 is less than $\frac{1}{4}$, the second derivative of range due to pure translation is small. Unfortunately, this depends on the distance to the center of curvature, which depends on the target. It is probably wiser to calculate this quantity for points at the minimum range of the sonar. This changes the quantity to

$$N_2 = \frac{V^2}{2f_s^2 \lambda r_{min}}. \tag{4.102}$$

If the robot had a forward looking sonar with a maximum beamwidth θ_{max}, this quantity would more accurately be

$$N_2 = \frac{V^2 \sin^2(\theta_{max})}{2f_s^2 \lambda r_{min}}. \tag{4.103}$$

This may be a significant change. For instance, if the beam half angle is 30°, the N_2 value will be reduced by $\frac{1}{4}$. If we assume that a dolphin has a beam half angle of 30°, using the previously mentioned numbers, we find that the translational component of second order effects is observable for targets closer than roughly $\frac{1}{3}$m. This is for a velocity of $1\frac{m}{s}$. If $4\frac{m}{s}$ were chosen this would jump to roughly 5m. If the dolphin could see objects at broadside, these numbers would jump to $1\frac{1}{3}$m and 21m respectively.

The second part, which depends on the vehicle's turning rate, is also maximized for a target at broadside. Since most vehicles yaw, the maximum turning rate will be labelled Υ. A nondimensional description of the turning contribution is

$$N_3 = \frac{V\Upsilon}{2f_s^2 \lambda}. \tag{4.104}$$

A robot with a forward looking sonar with a fixed beam width would be better described by

$$N_3 = \frac{V\Upsilon \sin(\theta_{max})}{2f_s^2 \lambda}. \tag{4.105}$$

For the dolphin we have described, the maximum turning rate for which $N_3 < \frac{1}{4}$ is $85\frac{degrees}{s}$, if $V = 1\frac{m}{s}$. If $V = 4\frac{m}{s}$, then the maximum yaw rate is roughly $20\frac{degrees}{s}$. Even the second yaw rate may seem very high, but it is reasonable to expect that a dolphin can turn 360° in less than 18s.

The last contribution to the second substantial derivative of range comes from acceleration. This is maximized for a target directly in front of the robot. Essentially, as the robot accelerates, can it track the object directly in front of itself. This is described by

$$N_4 = \frac{\dot{V}_{max}}{2f_s^2\lambda}. \tag{4.106}$$

If a dolphin accelerates at greater than $.75\frac{m}{s}$, the acceleration will observably contribute to the change in range measurements.

Finally, we examine the angular rates. The angular rates can be broken into two parts. The first is for linear motion, it describes how the bearing to a target changes as the robot moves in a straight line. The second describes how the bearing to the target changes as the robot turns.

One difficulty in non-dimensionalizing the bearing rate is determining an appropriate measure of angular resolution. For a given array, the angular resolution is a function of angle. Resolution is typically highest at broadside and lowest at endfire. For simplicity, the broadside resolution $\frac{\lambda}{4L}$ will be used, which assumes once again that a quarter wavelength can be resolved. For uniformity, the $\frac{1}{4}$ will be left out of the non-dimensional quantities, so that if the user wishes two change the resolution criteria they may. In addition to simplicity, the broadside resolution is chosen because it is the most conservative, because robots often use broadside arrays because they allow the largest aperture, and because it is possible to design mechanically scanned planar arrays so that the broadside resolution is appropriate at any angle once fixation has occurred.

The first non-dimensional number describing the first substantial derivative of bearing depends on the velocity, the target bearing, and the range to the center of curvature. It is maximized at broadside.

$$N_5 = \frac{VL}{r_{min}\lambda f_s}. \tag{4.107}$$

With a fixed beam width, this would be

$$N_5 = \frac{VL\sin(\theta_{max})}{r_{min}\lambda f_s}. \tag{4.108}$$

A crude guess for the size of a dolphins head is .2m. If the dolphin is moving at

$1\frac{m}{s}$, the bearing rate is observable for targets inside of about 3m. For a $4\frac{m}{s}$ dolphin, the transition occurs at around 11m.

The non-dimensionalized yaw rate contribution is

$$N_6 = \frac{\Upsilon L}{\lambda f_s}. \tag{4.109}$$

For a dolphin, the maximum yaw rate without entering the bearing rate regime would be about $10\frac{degrees}{second}$.

4.6 Effects of Ocean Waves

When the robot is near the surface, it will be perturbed by ocean waves. It is important to understand the magnitude of those effects and their impact on the selection of a kinematic model.

Consider a simple, linear, planar wave with amplitude A and wavelength Λ in deep water. (Surface waves are described by many of the same quantities as acoustic waves, such as wavelength and angular frequency. To avoid confusion, the "little" acoustic waves will be described by lowercase variables, while the "big" ocean waves will be described in the uppercase.) If the wave is propagating in the x direction, the surface perturbation $\eta(x, t,)$ is

$$\eta(x, t) = A\cos(Kx - \Omega t) \tag{4.110}$$

where K is the wave number $K = \frac{2\pi}{\Lambda}$ and Ω is the angular frequency. From the dispersion relationship [76] for deep water we know that $\Omega = \sqrt{Kg}$, where $g = 9.8\frac{m}{s^2}$. The dispersion relationship relates the group and phase velocity to the wavelength; the velocity of an ocean wave does not depend on its amplitude. The fluid velocities are

$$u = \Omega A e^{Kz}\cos(Kx - \Omega t) \tag{4.111}$$

and

$$w = \Omega A e^{Kz}\sin(Kx - \Omega t). \tag{4.112}$$

There are only velocities in the x and z directions, not in the y direction. Because this is a plane wave, there are no variations in the y direction to drive such a flow. The amplitude of u and w scale with the amplitude of the wave A. They occur at the same frequency as the surface wave. The horizontal velocity u peaks at the lowest and highest point on the wave. The vertical velocity is highest when $\eta = 0$. Both velocities decay exponentially with depth. At a depth of $-\Lambda$ the velocity magnitude has decayed to $\Omega A e^{-2\pi}$, which is considered small. Perhaps most interesting, the two velocities are 90° out of phase, leading to what are known as "particle orbits". Over the period of a wave, a particle in the water column will traverse a circular orbit radius Ae^{Kz}. It is important to remember that the horizontal displacement is comparable to the vertical displacement (This is for deep water. In shallow water the horizontal displacement may be much greater than the vertical displacement.). The components of the fluid acceleration are

$$\dot{u} = -\Omega^2 A E^{Kz} \cos(Kx - \Omega t) \tag{4.113}$$

and

$$\dot{w} = -\Omega^2 A E^{Kz} \sin(Kx - \Omega t). \tag{4.114}$$

In shallow water with depth h, or bottom at $-h$, the velocity and accelerations are

$$\begin{bmatrix} u \\ w \end{bmatrix} = \begin{bmatrix} \frac{gAK}{\Omega} \frac{\cosh(K(z+h))}{\cosh(zh)} \cos(Kx - \Omega t) \\ \frac{gAK}{\Omega} \frac{\sinh(K(z+h))}{\cosh(zh)} \sin(Kx - \Omega t) \end{bmatrix} \tag{4.115}$$

and

$$\begin{bmatrix} \dot{u} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} gAK \frac{\cosh(K(z+h))}{\cosh(zh)} \cos(Kx - \Omega t) \\ -gAK \frac{\sinh(K(z+h))}{\cosh(zh)} \sin(Kx - \Omega t) \end{bmatrix}. \tag{4.116}$$

Wave effects differ in shallow water because the flow cannot penetrate the bottom. The new form of the equations occurs when the vertical velocity is constrained to be zero at the bottom. Unlike the deep water case, in which the particle orbits were circular, in shallow water the orbits are elliptical with the horizontal velocity being greater than the vertical velocity. The shallow water equations apply when Kh is much less that one [76].

Now, one may wonder why this flow matters. Robots are big and strong, why worry about a little drag? These effects do matter, and they are significant. These effects are properly characterized as added mass effects, rather than as drag. Drag occurs on a body at steady state. Added mass is a force associated with accelerations due to potential flow effects. It is as if the body carries a portion of the water column with it. A sphere has an added mass equal to its displacement. Accelerating a neutrally buoyant sphere in water requires twice the force it would take in air. This is also true for a decelerating body. It is much harder to slow an underwater body. An underwater vehicle typically is neutrally buoyant, small compared to the wavelength Λ, and many vehicle lengths underwater. Under these circumstances, the vehicle will tend to act like a particle, taking on the accelerations of the water column. The forces necessary to overcome these accelerations may be quite high and perhaps a waste of energy.

Assuming the robot does not try to overcome the effects of surface waves, it is necessary to determine when they are relevant. A new form of N_1 will be developed to determine whether the robot moves more than a quarter wavelength due to wave effects. Since the velocity from the wave will be $A\Omega e^{Kz}$, this non-dimensionalizes as

$$N_7 = \frac{A\Omega e^{Kz}}{f_s \lambda}. \tag{4.117}$$

If N_7 is large, a robot's motion cannot be described by only its forward velocity and the holonomic model is necessary.

A similar number describes the motion in shallow water. Since the horizontal and vertical velocities have unequal amplitudes, the horizontal amplitude will be used.

$$N_8 = \frac{gAK}{\Omega} \frac{\cosh(K(z+h))}{f_x \lambda \cosh(Kh)}. \tag{4.118}$$

4.7 Complexity Analysis

Having established non-dimensional numbers to determine when the respective regimes are relevant, the computational complexity of tracking in those various regimes will be examined for various sensor and robot designs.

First, consider a range only sonar. If $N_1 > \frac{1}{4}$, and N_2, N_3, or N_4 are less that $\frac{1}{4}$, the farfield regime can be used (i.e. $\frac{D^2 r_m}{Dt^2}$ can be ignored). Without angular information, the range rate cannot be predicted, even if the velocity is known. If two measurements were known to have a common origin, they could be used to infer the range rate, but we cannot know they have a common origin from two measurements alone. From three range measurements of a target, it is possible to predict the range rate, and validate it. In the linear regime, the range flow has two degrees of freedom. If the robot is not tracking anything and making n observations per timestep, then a naive search for features is $O(n^3)$. This implies that all possible measurement combinations need to be considered. This is naive, because the range rate is bounded by the maximum robot velocity. If, instead, we say that the number of observations is proportional to the maximum range of the sensor r_{max}, then the naive search is $O(r_{max}^3)$. By bounding the maximum velocity, the general initialization complexity is reduced to $O(r_{max}\frac{V_{max}^2}{f_s^2})$. The search to determine whether a single initial measurement originated from a target is $O(\frac{V_{max}^2}{f_s^2})$. Once a measurement trajectory has been initialized, the naive search is $O(r_{max})$, and the improved search is $O(\frac{V_{max}}{f_s})$.

If N_2, N_3, or N_4 are greater than $\frac{1}{4}$, then $\frac{D^2 r_m}{Dt^2}$ is relevant as well. In the sonar only measures range, four measurements are required in the general search to initialize a trajectory, with a computational complexity of $O(r_{max}\frac{V_{max}^3}{f_s^3})$. Once the target has been initialized, the complexity remains $O(\frac{V_max}{f_s})$.

Consider now a sonar that measures range, azimuth, and elevation. If N_1 is greater than $\frac{1}{4}$, but N_2, N_3, and N_4 are all less than $\frac{1}{4}$, the farfield regime is applicable. If the robot's velocities are known, then a second measurement can be predicted from a prior measurement. The general search for correspondences is $O(r_{max}\frac{V_{max}}{f_s})$. If the velocities are unknown, one of two approaches can be taken. If there are enough features, the robot's velocity can directly estimated. For the non-holonomic case, in which the robot's motion is described entirely by its forward velocity, a minimum of two features must be observed. The robot's velocity is estimated from the first pairing and validated by the second. Any arbitrary pairing will yield a velocity. However, because the true velocity is the one that is observed from correct pairings, a codimension is necessary for validation. If the robot is holonomic, such that all three velocities must be inferred, a minimum of four features need to be tracked. In practice, a single noisy codimensional constraint may not be sufficiently precise to

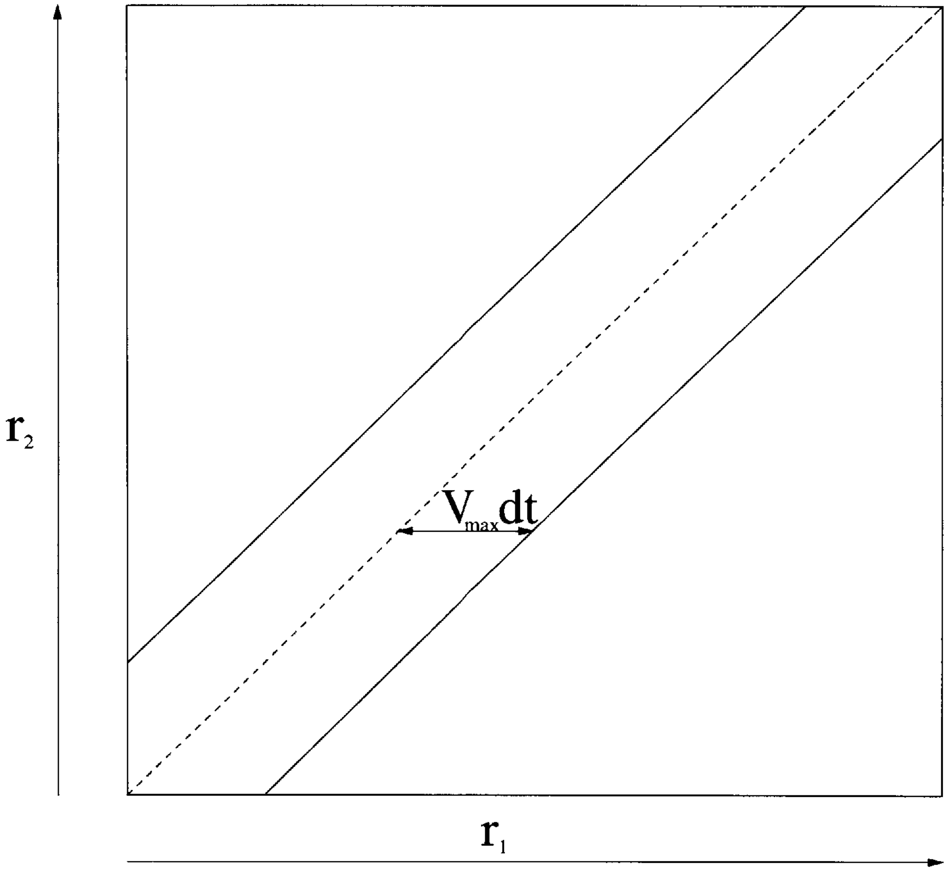


Figure 4-1: A naive search for correspondence compares a given range measurement to all other ranges at a subsequent timestep. A better approach is to only search the diagonal region, since the range rate typically does not exceed the maximum robot velocity. This can be massively beneficial. For instance, given an initial range measurement, the naive search to initialize it is $O(r_{max}^2)$, the more restrictive search is $O(\frac{V^2}{f_s^2})$. If the maximum range is 50m, and the robot moves .5m between pings, the naive technique is $\frac{50^2}{.5^2} = 10000$ times as computationally intensive.

infer that many dimensions, so more features should probably be tracked. If sufficient features are unavailable, then the range rate can be inferred from a temporal sequence of measurements, allowing tracking. Once trajectories are initialized, the search for subsequent observations remains small.

If N_1 is greater than $\frac{1}{4}$, and either N_2, N_3, or N_4 are greater than $\frac{1}{4}$, then $\frac{D^2r}{Dt^2}$ needs to be used. Ideally, N_5 or N_6 is also greater than $\frac{1}{4}$. Given two measurements, a third can be predicted, so the initial search is $O(r_{max}\frac{V_{max}^2}{f_s^2})$. If, for the non-holonomic case, the velocity and acceleration are unknown, they can be inferred by tracking a minimum of two features. For the holonomic case, the minimum is four features.

If both N_5 and N_6 are less that $\frac{1}{4}$, it may still be possible to track a feature using the second substantial derivative of range. Essentially, it is estimated by comparing the difference between measured ranges to the predicted slope. The error should be primarily due to second order effects if solidly in the second order regime. If that error is consistent, then the prediction of the second substantial derivative is essentialy predicted error in the first derivative.

4.8 Experimental Results

Two experiments were conducted in a testing tank. A binaural sonar [52] was used to measure the range and bearing to targets. A binaural sonar is a simple line array with a central transmitting transducer and two receiving "ear" transducers. Detection was performed using the method employed by Kuc [52]. First, the signal was rectified, logarithmically compressed, and low-pass filtered to yield the filtered log-envelope. Then, the signal was thresholded. Upcrossings that were preceded by a period without upcrossings were used to determine the times of flight (TOFs) for the left and right sensors, denoted T_l and T_r. Correspondence between sensors was found by constraining the difference in TOFs to be less than $2D/c$, where $2D$ is the separation between the sensors and c is the speed of sound.

Using T_l and T_r, ranges and bearings were determined as in [52]:

$$r = \frac{(cT_l)^2 + (cT_r)^2 - 2D^2}{2c(T_l + T_r)} \tag{4.119}$$

$$\theta = \arcsin(\frac{(c^2T_lT_r + D^2)(cT_l - cT_r)}{D(c^2T_l^2 + c^2T_r^2 - 2D^2)}). \tag{4.120}$$

Using the above approach, the raw waveform is reduced to a set of measurement tokens with each token being a range and bearing measurement.

The minimum range of the sensor was $r_{min} = .7$m. The minimum range was defined by the intersection of the beams of the left and right transducers; the respective cones do not intersect until roughly .7m. The resonant frequency of the transducers was 500kHz, corresponding to $\lambda = 3$mm. The separation between the left and right transducers was $L = 9.4$cm.

In the first experiment, the robot drove straight at a constant velocity so $\Upsilon = 0$ and $\dot{V}_{max} = 0$. The robot velocity was $V_{max} = .1\frac{m}{timestep}$. The robot's sample rate was $f_s = \frac{1}{timestep}$. The robot was able to mechanically fixate on the targets, so the broadside form of N_2 is appropriate. Using these values, the relevant non-dimensional quantities are $N_1 = 33.33$, $N_2 = 2.381$, $N_3 = 0$, $N_4 = 0$, $N_5 = 4.48$, and $N_6 = 0$. Since $\#_1 > \frac{1}{4}$, $\frac{Dr}{Dt}$ is necessary. Since $N_2 > \frac{1}{4}$, $\frac{D^2r}{Dt^2}$ was also relevant. Had the robot slowed, the sample rate been increased, and/or the minimum range at which features were observed been increased, this would not have been necessary. The first derivative of bearing is also observable, since N_5 exceeded $\frac{1}{4}$. This was due to linear motion, not due to the yaw rate, since the robot only yawed because of added process noise.

In the second experiment, the robot moved at $.15\frac{m}{timestep}$. It had a maximum yaw rate of $\Upsilon = 15\frac{degrees}{timestep} = .26\frac{rad}{timestep}$. It drove at a constant speed, so $\dot{V}_{max} = 0$. Otherwise, the quantities were the same as in the previous experiment. The six non-dimensional quantities were $N_1 = 50$, $N_2 = 5.36$, $N_3 = 6.5$, $N_4 = 0$, $N_5 = 6.71$, and $N_6 = 8.15$. Since the only insignificant contribution was from acceleration (N_4 was zero since the robot never changed speed), all three substantial derivatives were necessary.

The data for experiment 1 is shown in Figures 4-2 and 4-3. In this experiment, the sensor moved past a triangular target and two point objects (fishing bobbers). The side of the triangle was mapped, and the two fishing bobbers were tracked and mapped, as shown in Figure 4-5.

The data for experiment 2 is shown in Figures 4-6 and 4-7. This experiment was fully "autonomous" in the sense that the desired sensor motion was computed online in response to the sensor data. The robot adaptively fixated on a cylinder using

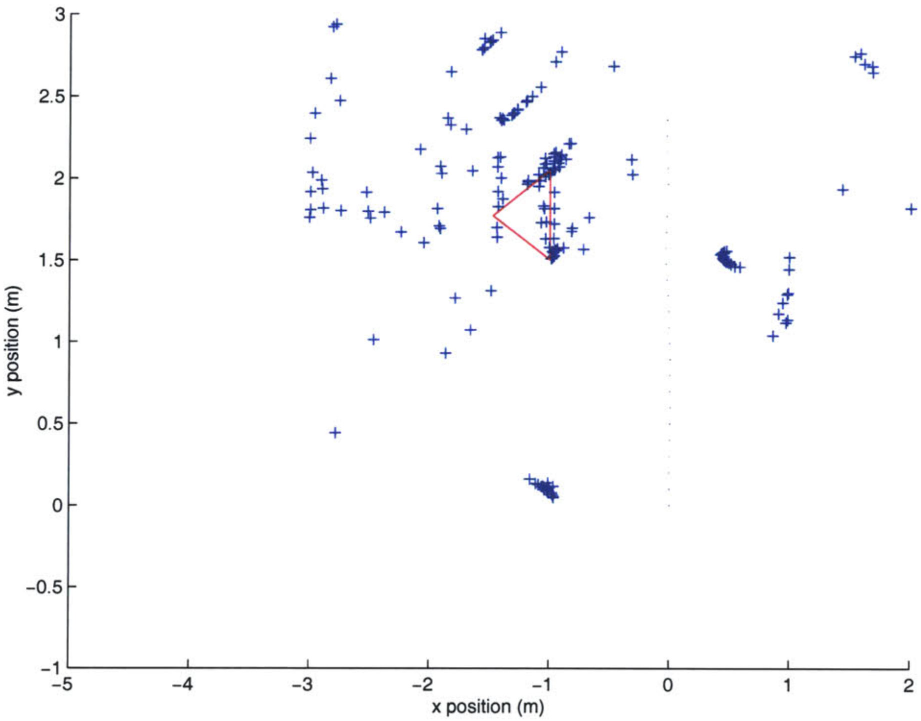


Figure 4-2: Sonar measurements displayed in a cartesian projection. Detected returns are back-projected along the sensor line of site to create a map from time-of-flight values. The correspondence of different returns is unclear.

a reactive sensing strategy [17, 18]. After detecting the cylinder, the sensor drove toward it and began to circle it, while periodically scanning backwards to map two point objects located behind it. The extracted measurement trajectories are shown in Figure 4-8. The algorithm concurrently estimated the curvature and center position of the cylinder, the (x, y) positions of the point objects, and the sensor trajectory (Figure 4-9).

4.9 Conclusion

This chapter has investigated the sonar perception problem. We have developed a measurement flow model for wide beam sonar data obtained from a moving observer in an environment with static objects. Using data from an experiment in a testing tank, we have demonstrated the effectiveness of this model for associated measurements obtained from multiple vantage points and using these correspondences for CML. In the next chapter, we apply this methodology to oceanic sonar data from the GOATS

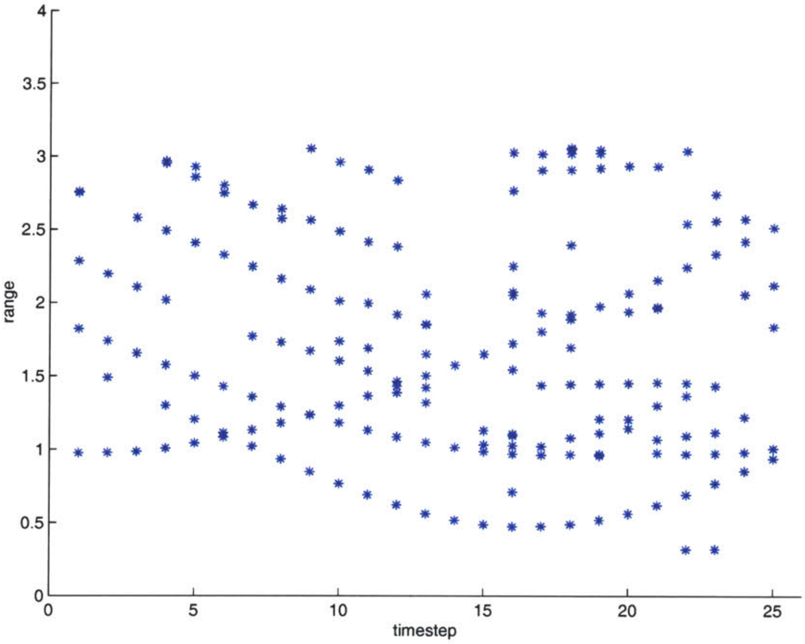


Figure 4-3: Detected returns displayed in a range vs. angle plot. In comparison to Figure 4-2, the human eye easily picks out "measurement trajectories" when viewed in this manner.

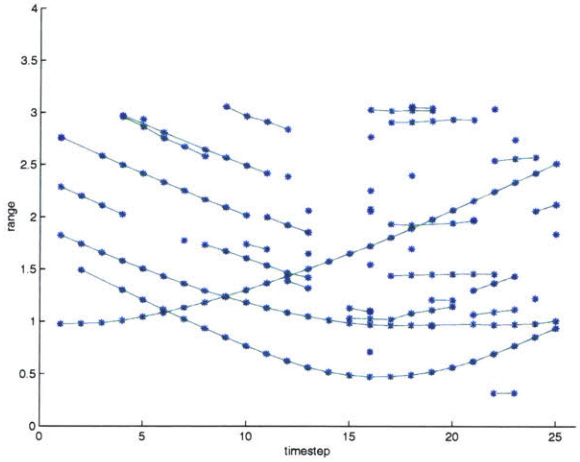


Figure 4-4: Extracted measurement trajectories for experiment 1. (Compare with Figure 4-3).

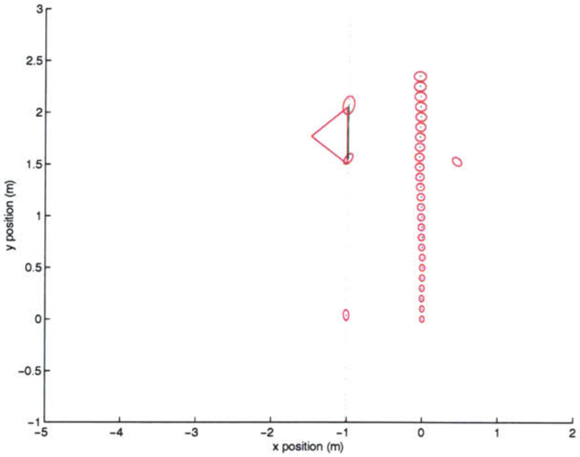


Figure 4-5: Estimated trajectory and map for experiment 1.

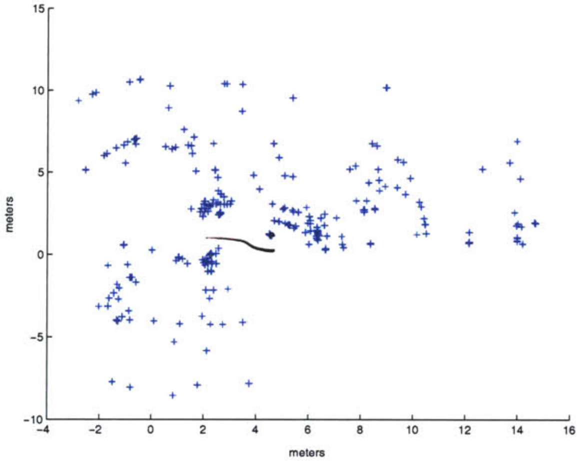


Figure 4-6: Cartesian projection of detected returns for experiment 2.

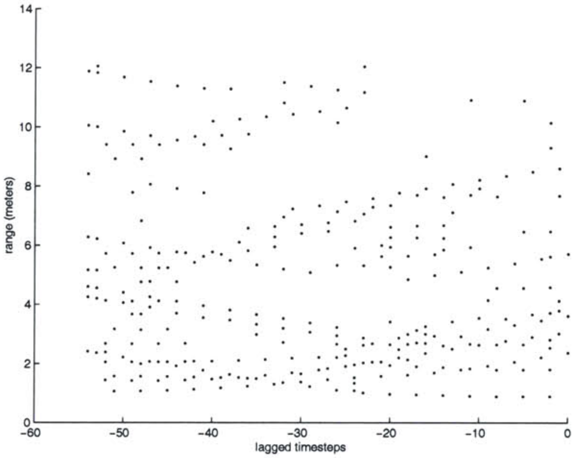


Figure 4-7: Range vs. time for detected returns for experiment 2. Returns that are not grouped into a trajectory are discarded.

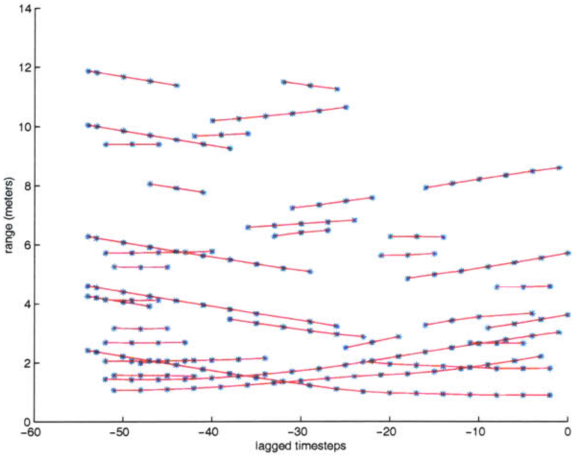


Figure 4-8: Extracted measurement trajectories for experiment 2.

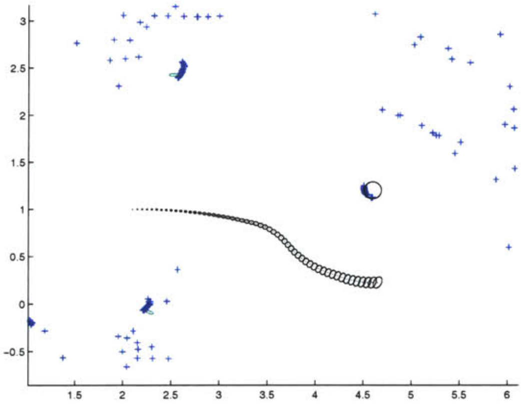


Figure 4-9: Raw data, estimated sensor trajectory, and object locations for experiment 2.

2002 experiment.

Chapter 5

Results From an Ocean Experiment

In this chapter, results from an ocean experiment using an AUV are presented. Using the MIT Odyssey III synthetic aperture sonar system, we demonstrate the process of extracting measurement trajectories from in water sonar data. These trajectories are processed to generate a preliminary CML result.

5.1 Experiment description

In June 2002, in conjunction with the NATO SACLANT research center in La Spezia, Italy, we conducted a set of experiments to investigate using AUVs for mine countermeasures (MCM). The experiment we will describe was conducted using the Odyssey III AUV Caribou, built by Bluefin Robotics (Figure 5-1). The Odyssey III is a streamlined, non-holonomic vehicle design with a vectored thruster. Its sensory payload was an active sonar, consisting of a broadbeam transmitter aimed at broadside and a 16 element tuning fork receiving array. The sonar was originally designed for synthetic aperture (SAS) missions. In post-processing we used its data to demonstrate feature tracking and CML. Figure 5-2 shows a representative set of received signals obtained from the sonar. Figure 5-3 shows a detector output for these waveforms. The receiving array was constructed by the engineering staff of the NATO SACLANT Undersea Research Centre. The SAS sonar system was developed and integrated into Caribou by D. Eickstedt, W. Xu, T.C. Liu, P. Newman, R. Damus, S. Desset, and J. Morash.

5.2 Selection of the Trajectory Sonar Equations

For the experiment, a representative robot velocity was $1\frac{m}{s}$, a representative acoustic wavelength was $.1m$, and the sampling frequency was $3Hz$. This made $\#_1 = \frac{10}{3}$, which exceeds $\frac{1}{4}$, necessitating the first order range term. The maximum velocity at which first order terms could be neglected was $\frac{\lambda f_s}{4} = .075\frac{m}{s}$. Heave and sway velocities between $.1\frac{m}{s}$ and $.25\frac{m}{s}$ were routinely observed, necessitating the modeling of all three velocities.

Although ranges that were less than the twenty meter water depth were discarded, a minimum range of $r_{min} = 10$m will be used in calculations. The maximum yaw rate and acceleration were approximately $\Upsilon = 10\frac{degrees}{s} = .1745\frac{rad}{s}$ and $\dot{V}_{max} = .1\frac{m}{s}$. The non-dimensional numbers relevant to the second substantial derivative were $\#_2 = .13$, $\#_3 = .15$, and $\#_4 = .06$. The contributions were all under a quarter wavelength, and the minimum range was artificially low; consequently, we could safely ignore second order effects.

The non-dimensional quantities relevant to the angular rates were $\#_6 = .7$ and $\#_7 = .4072$. They were both slightly greater than the threshold value of $\frac{1}{4}$, possibly implying that angular rates were observable. Unfortunately, the angular measurements were very noisy, and angular rates could not be estimated from pairs of measurements. In fact, the raw angles were too noisy for the first order range derivative, so the median of the last 5 angles was used.

5.3 Experimental Results

Data from the experiment was post-processed. First, using a two-dimensional delay and sum beamformer, each ping was beamformed over 1500 angles. The range, azimuth, and elevation of targets were constructed by thresholding the output.

It should be noted that the beamformer was very slow. The sonar transmitted three pings per second. After each transmission, the robot recorded for a tenth of a second at 100 kHz on 16 channels, meaning each ping led to 16 vectors of 10,000 samples. Forming this data over 1500 angles was extremely computationally intensive; processing a ping took roughly 10 minutes on a 1 GHz PC. Processing an entire mission on a single PC took roughly a month. It is expected that this process can be expedited using DSPs, faster computers, and parallel processing.

Once the signals had been processed to provide range and two angles, features were easily tracked using the first order variant of the trajectory sonar perception equations. The measurement trajectories in Figure 5-4 reflect the measurement triggers from Figure 5-3. A larger set of trajectories from a larger time frame is shown in Figure 5-5. The entire set of measurement trajectories is shown in Figure 5-6.

To truly perform concurrent mapping and localization, high level feature modeling and object recognition is necessary. This thesis stopped short of that, only investigating early tracking. As such, features were mapped as if they were points. Recognition was performed based on proximity. If two features appeared to be "close" using a Mahalanobis test, they were fused. The sets of measurements used to map two features are shown in Figure 5-8 and Figure 5-11. Using the entire set of measurement trajectories, the CML was performed. The robot's map and trajectory are shown in Figure 5-14.

Figure 5-16 shows a comparison of the CML mapped target locations with independent measurements of the same objects performed by the AUV REMUS. A high-resolution image of the target area for this map is shown in Figure 5-17, which was obtained by GESMA using a Klein DS5000 sidescan sonar system. One can see that our CML algorithm succeeds in detecting objects in locations that are generally correct; however, the algorithm fails by generating multiple features in its map that correspond to a single feature in reality.

The competence of trajectory perception has been developed solely for the "early" perception problem of associating measurements that originate from the same feature. Our results are encouraging because they demonstrate this capability extremely well with real ocean data. Future research is necessary, however, to address the higher-level problem of acoustic object recognition — to decide autonomously that a revisited object is the same as an object that was previously observed. This task remains for future work.

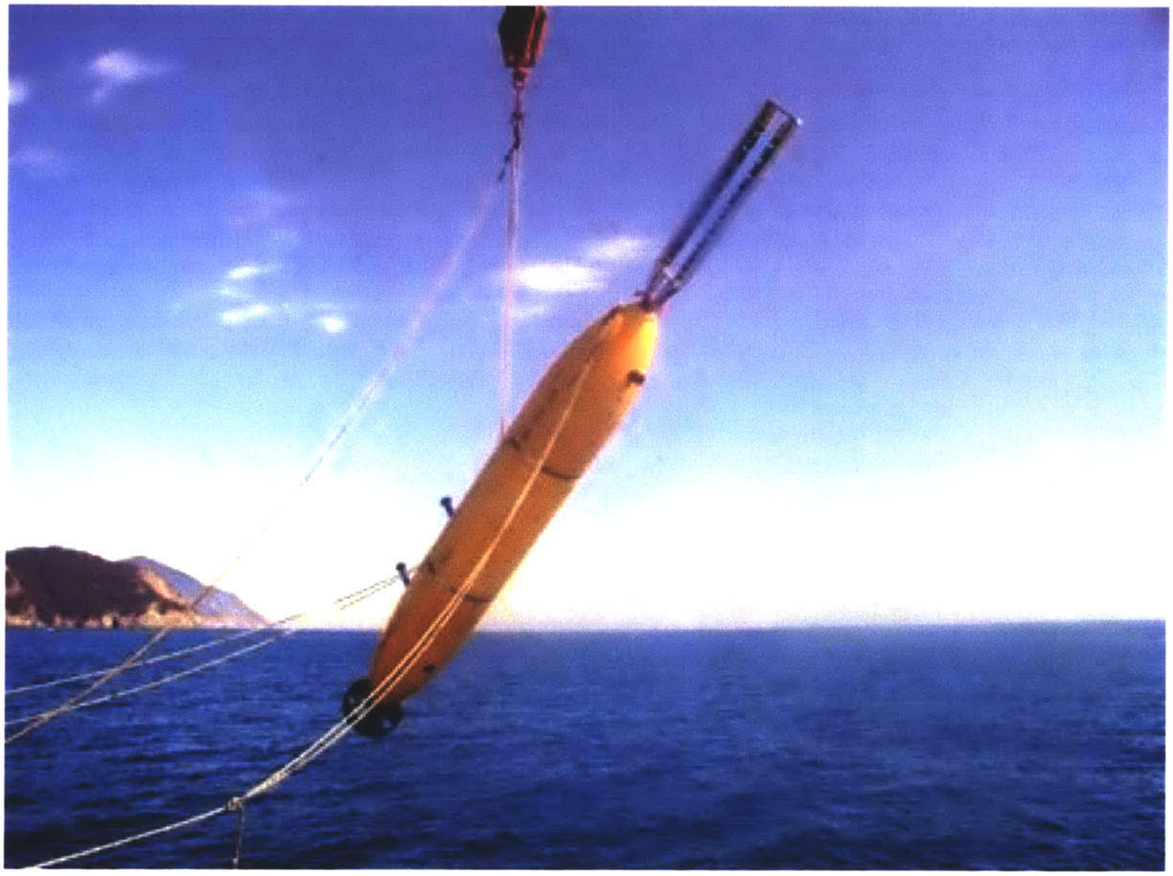


Figure 5-1: Odyssey III AUV with SAS sensor.

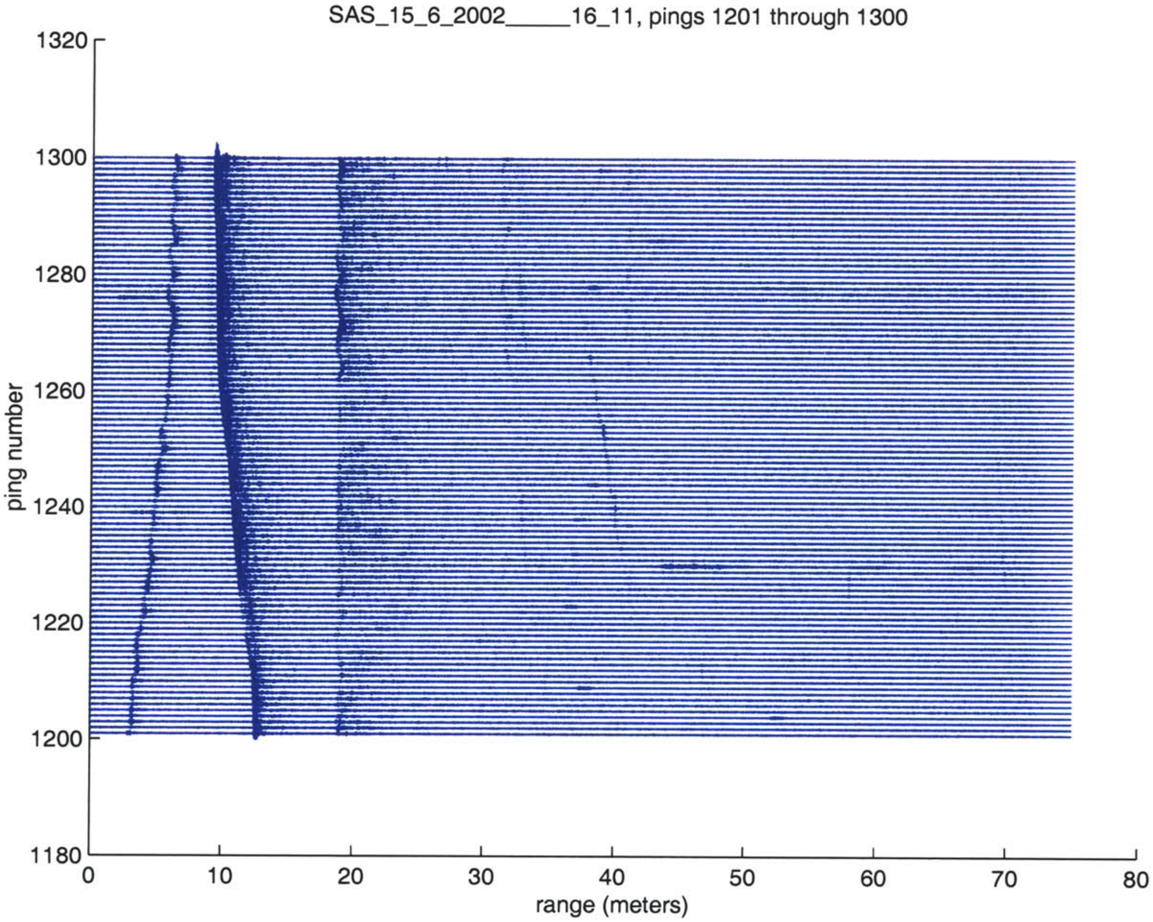


Figure 5-2: 100 pings of input data for an eight-element line array (broadside beamformer output). Pings occurred approximately three times per second. The data was acquired by the Odyssey III AUV Caribou operating in about 20 meters water depth. Most of the visible echoes in this image correspond to reflections from the bottom and/or sea surface. However, there are also many echoes corresponding to objects on the sea bottom (supports for an undersea cable, shown in Figure 1-5).

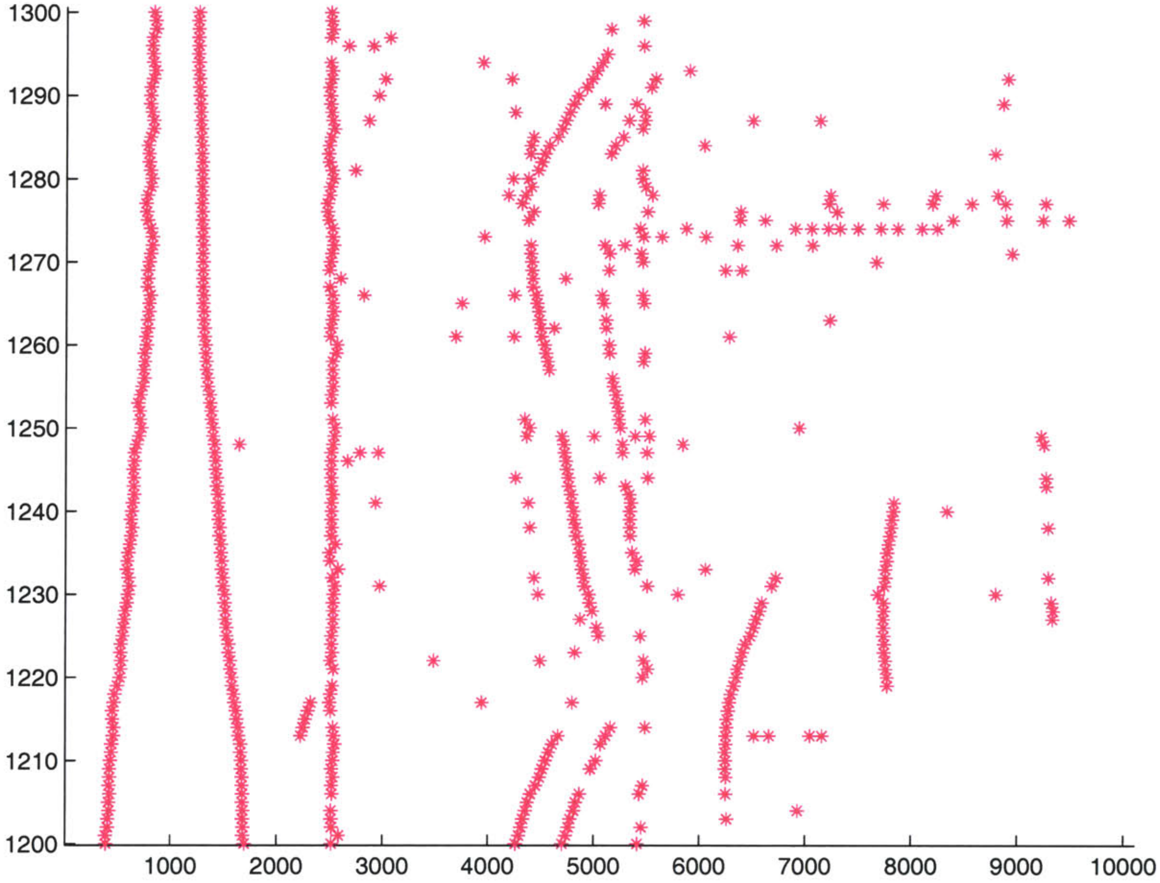


Figure 5-3: Measurement tokens extracted from raw data by thresholding the low-pass filtered, rectified output of a beamformer.

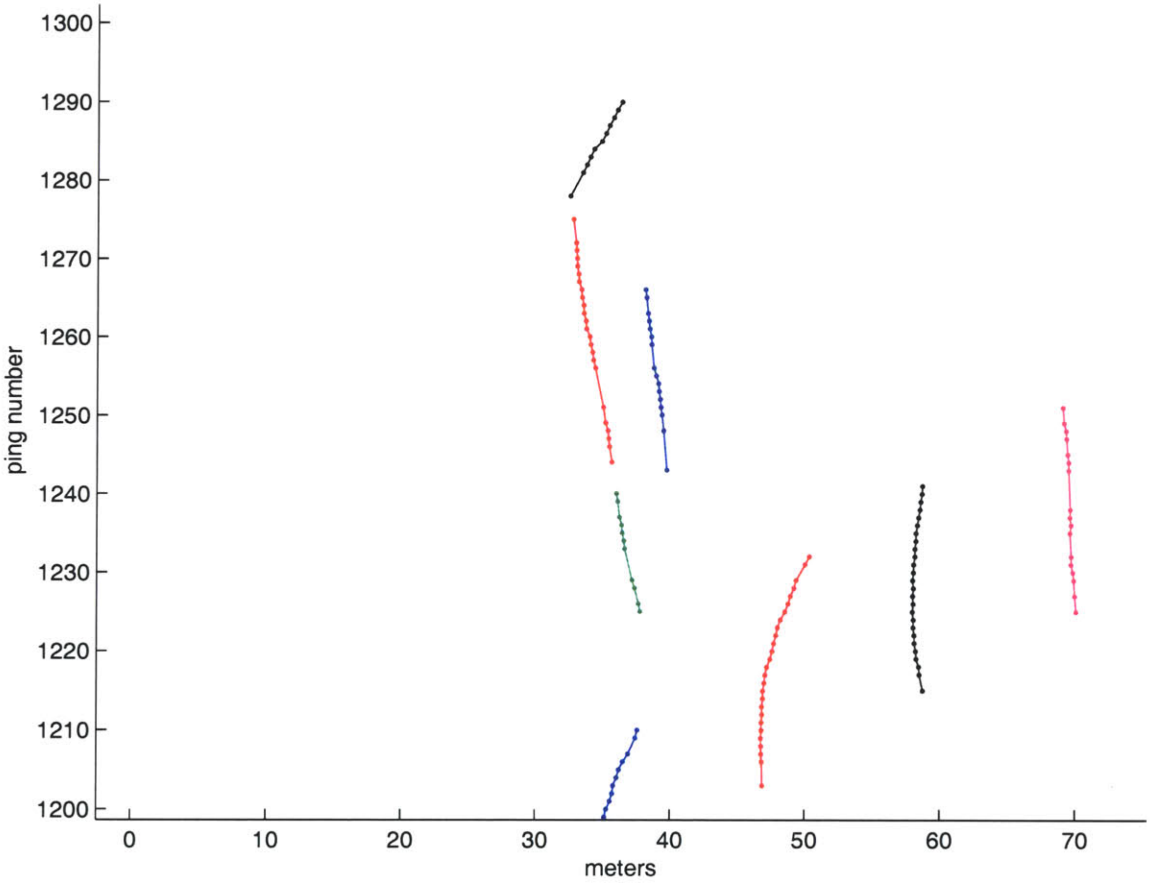


Figure 5-4: Measurement trajectories.

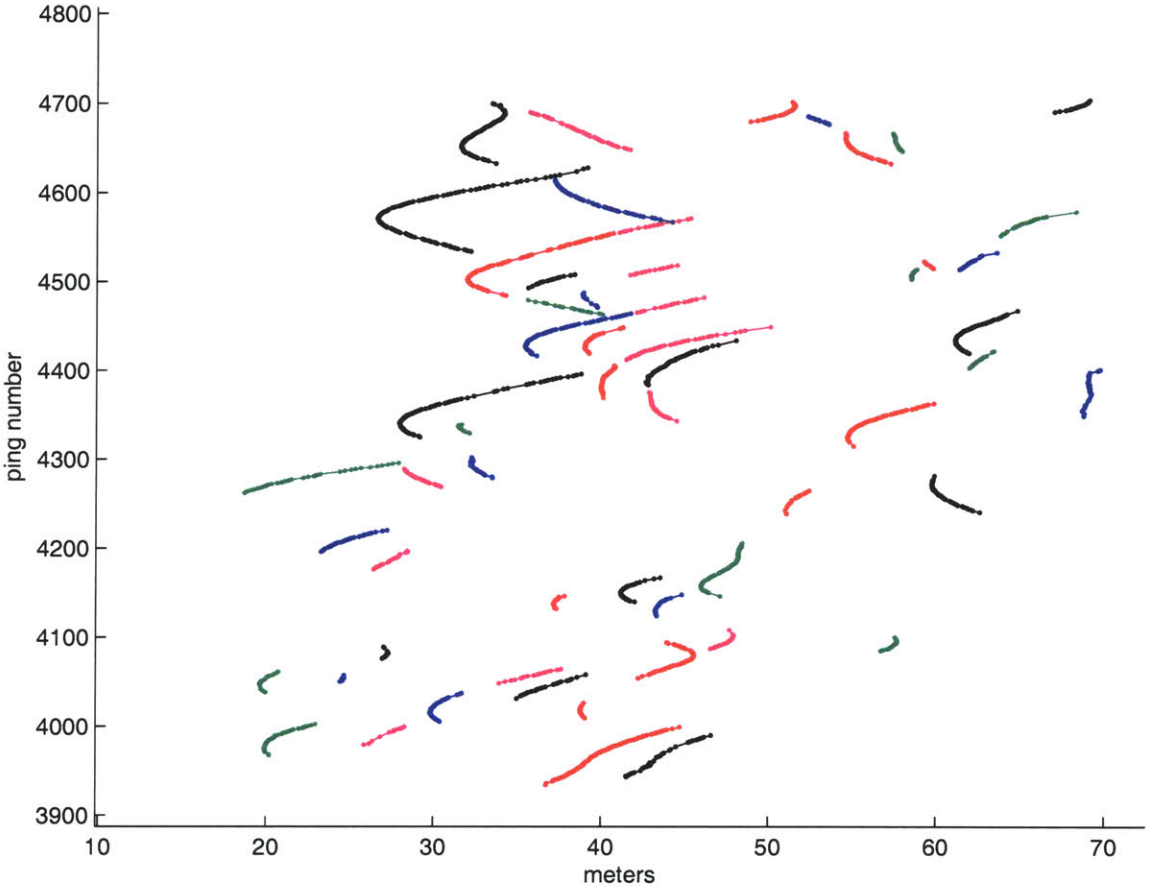


Figure 5-5: Measurement trajectories.

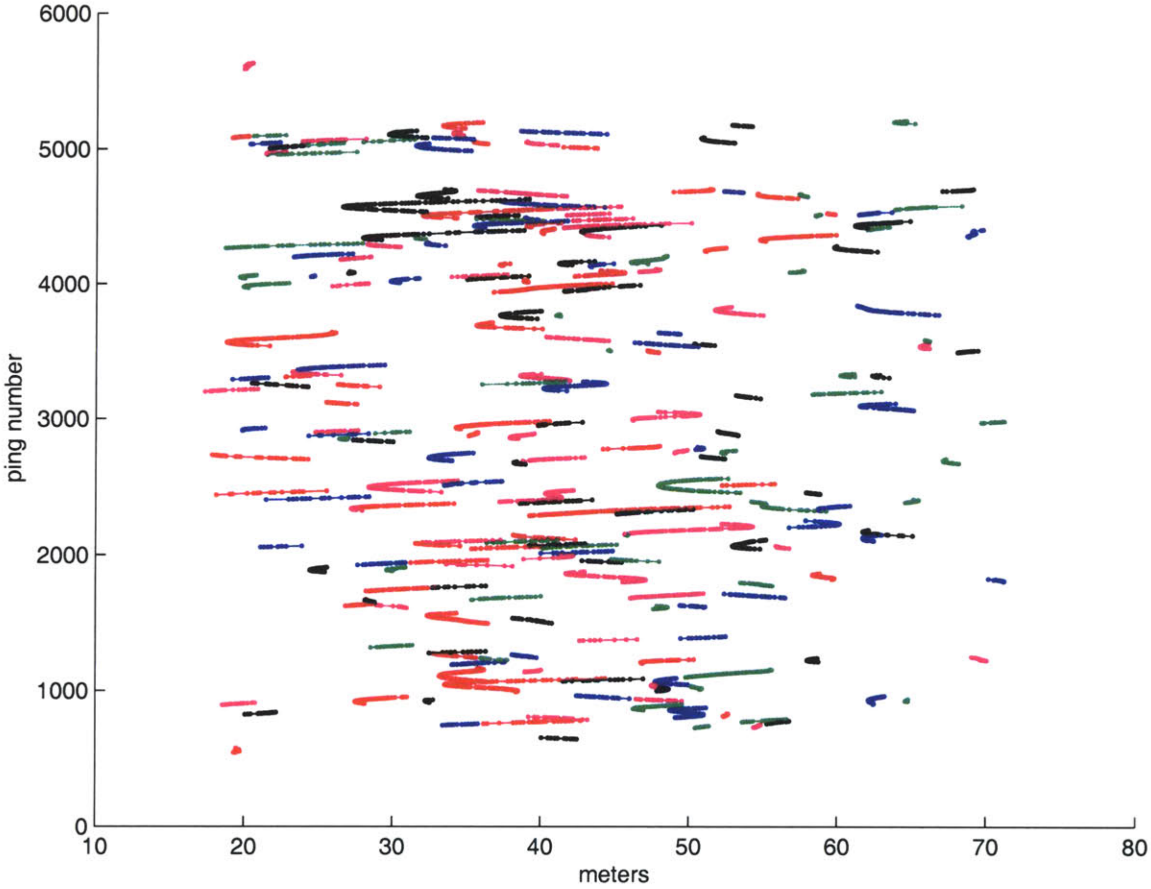


Figure 5-6: Measurement trajectories.

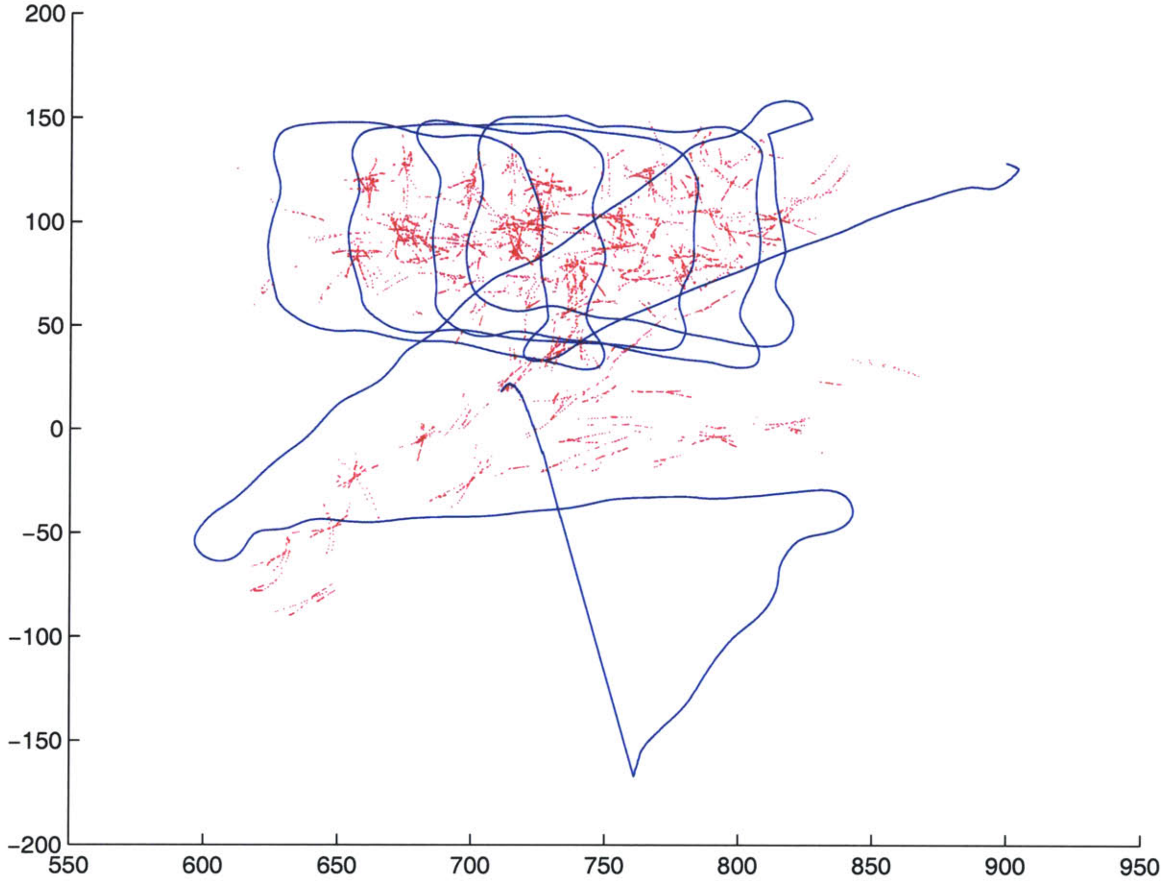


Figure 5-7: Measurement trajectories projected into cartesian coordinates from a dead reckoned robot. The spurious measurements have been removed. These are only the "good" measurements. Notice the substantial degradation of the structure.

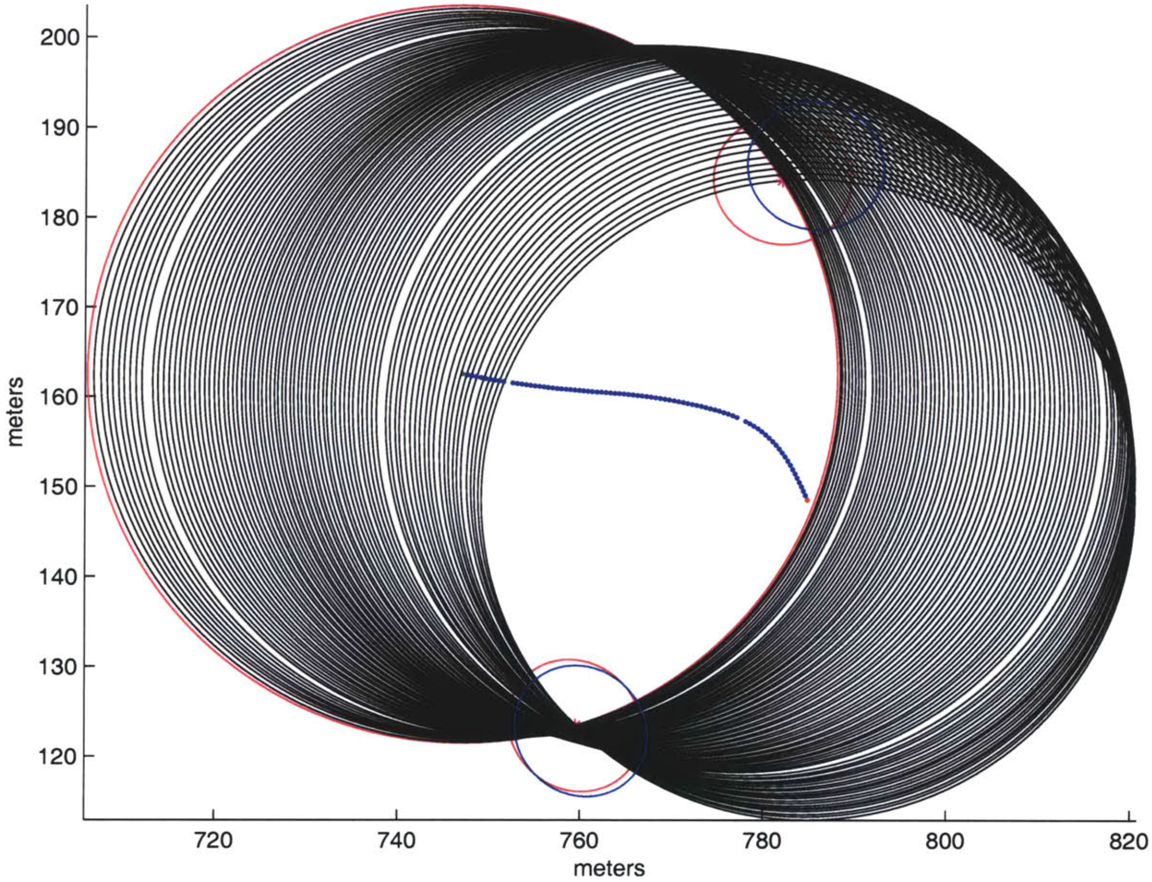


Figure 5-8: Initializing a feature based on measurement trajectory 89.

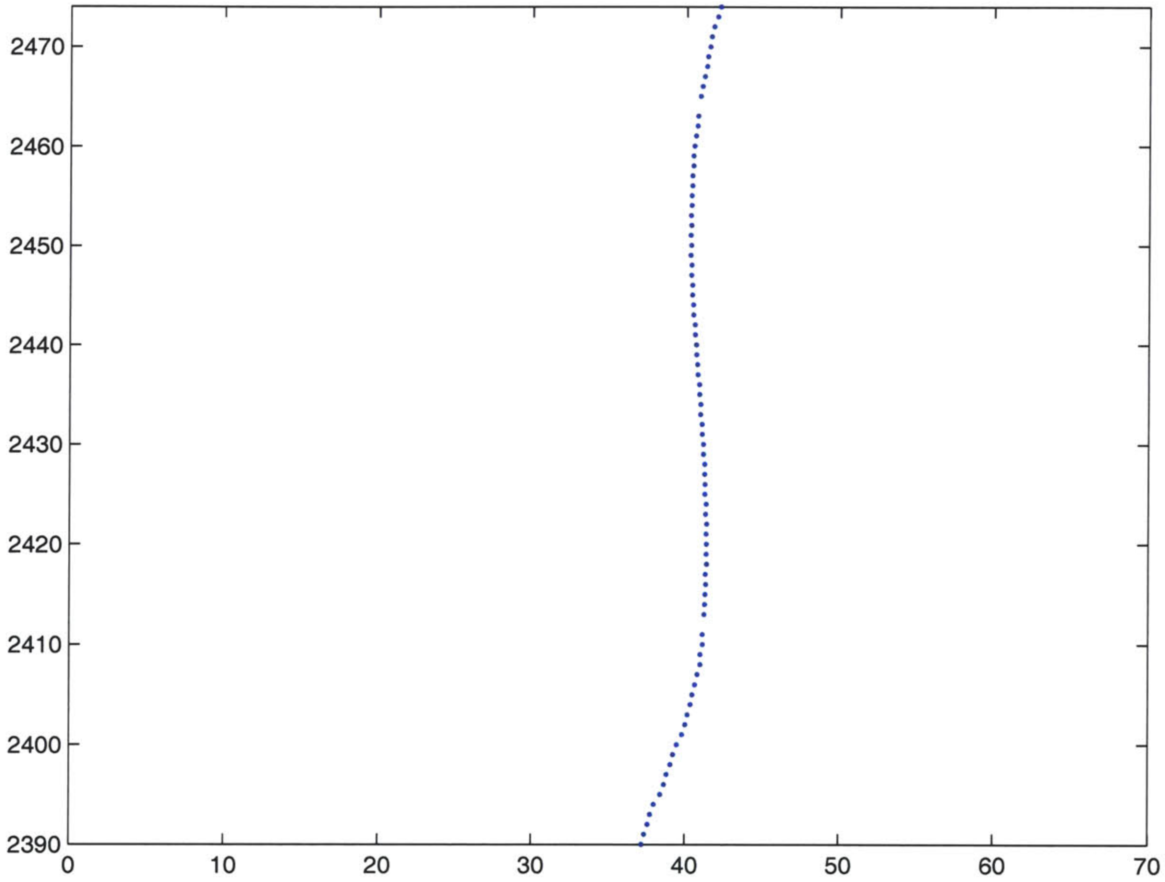


Figure 5-9: Measurement trajectory 89.

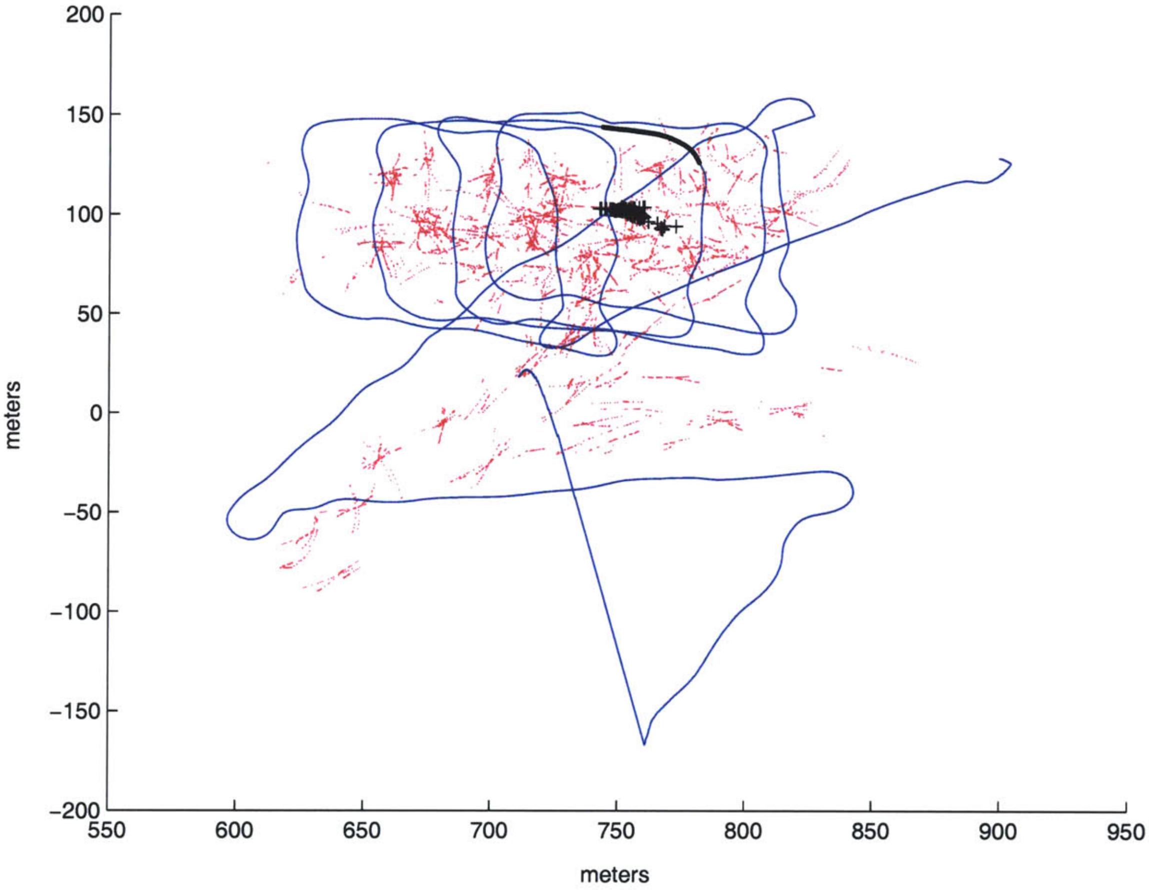


Figure 5-10: Global projection of the measurements in trajectory 89, based on a dead-reckoned robot path. The black dots along the robot path correspond to locations where the robot observed the target. The black +'s are the measurements from the trajectory projected into cartesian coordinates.

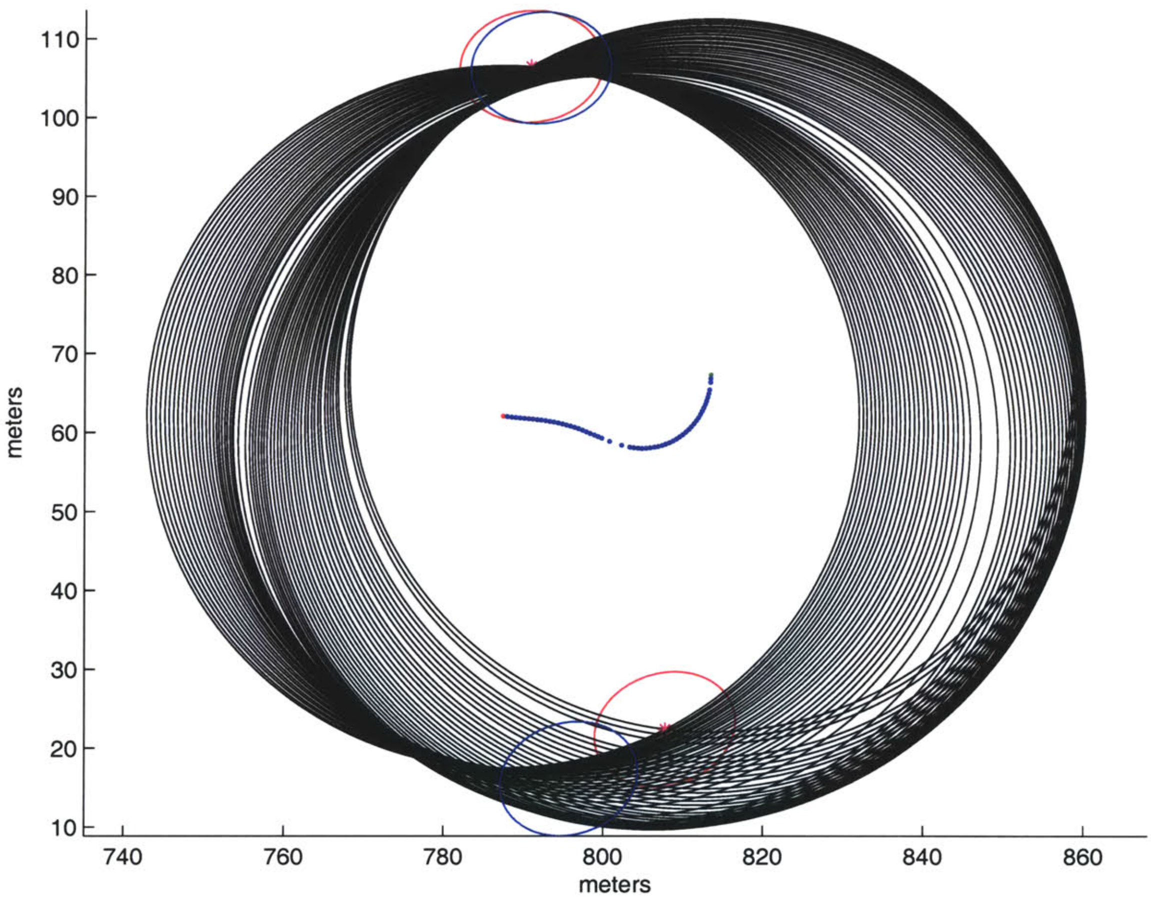


Figure 5-11: Target initialization.

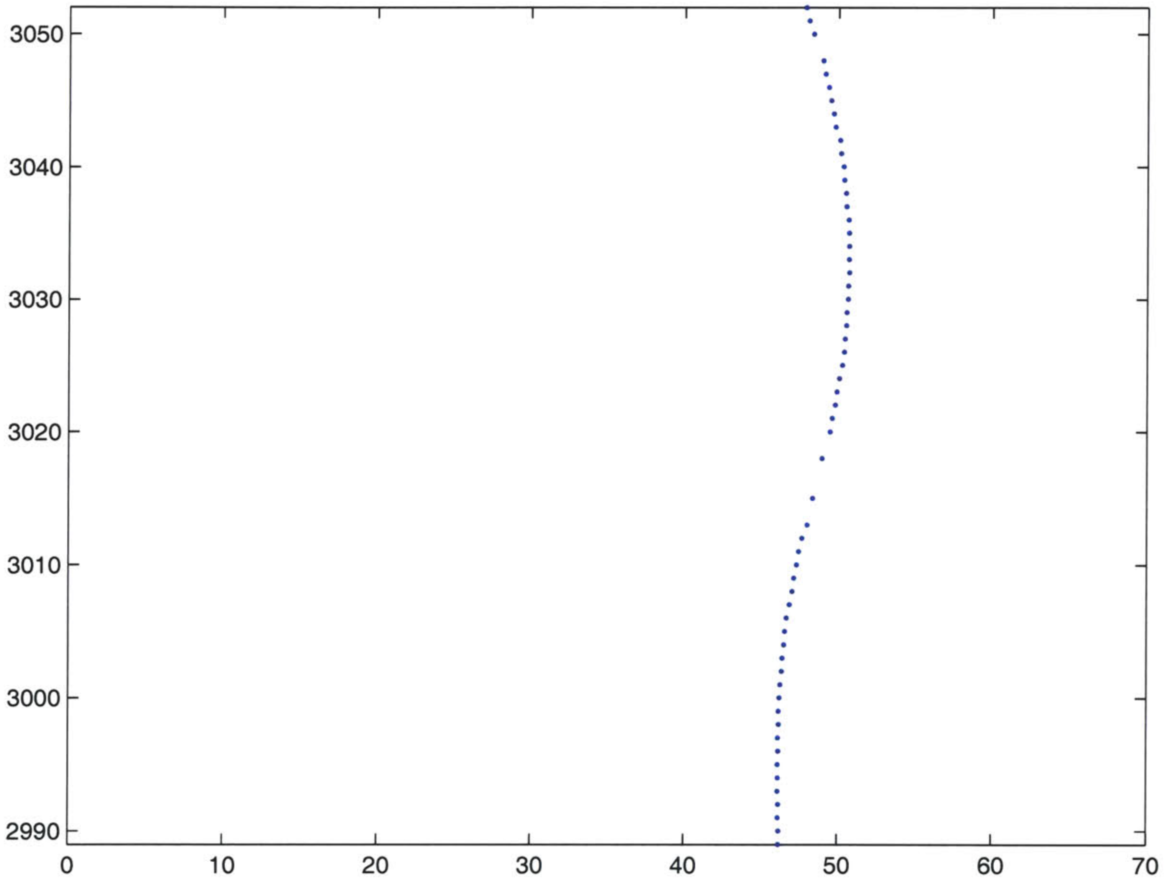


Figure 5-12: Target initialization for feature 89.

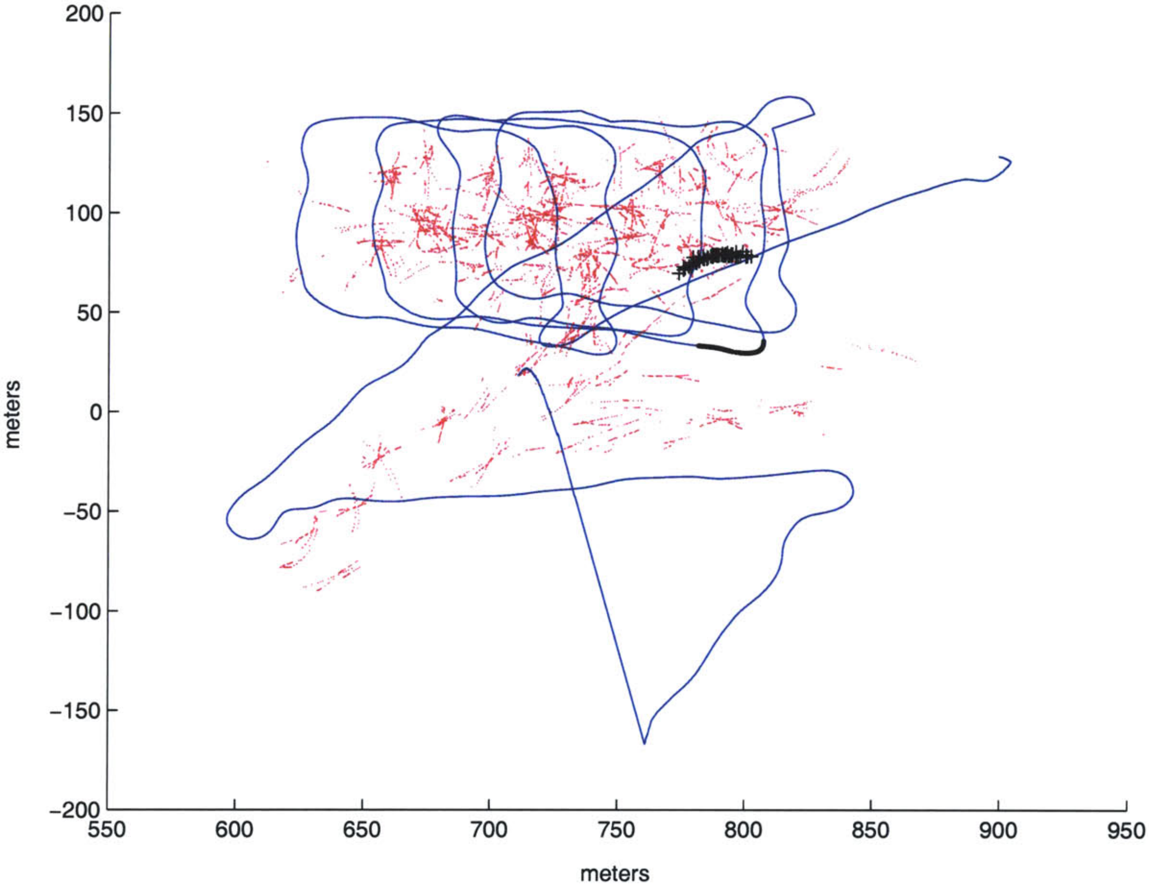


Figure 5-13: Target initialization for feature 89.

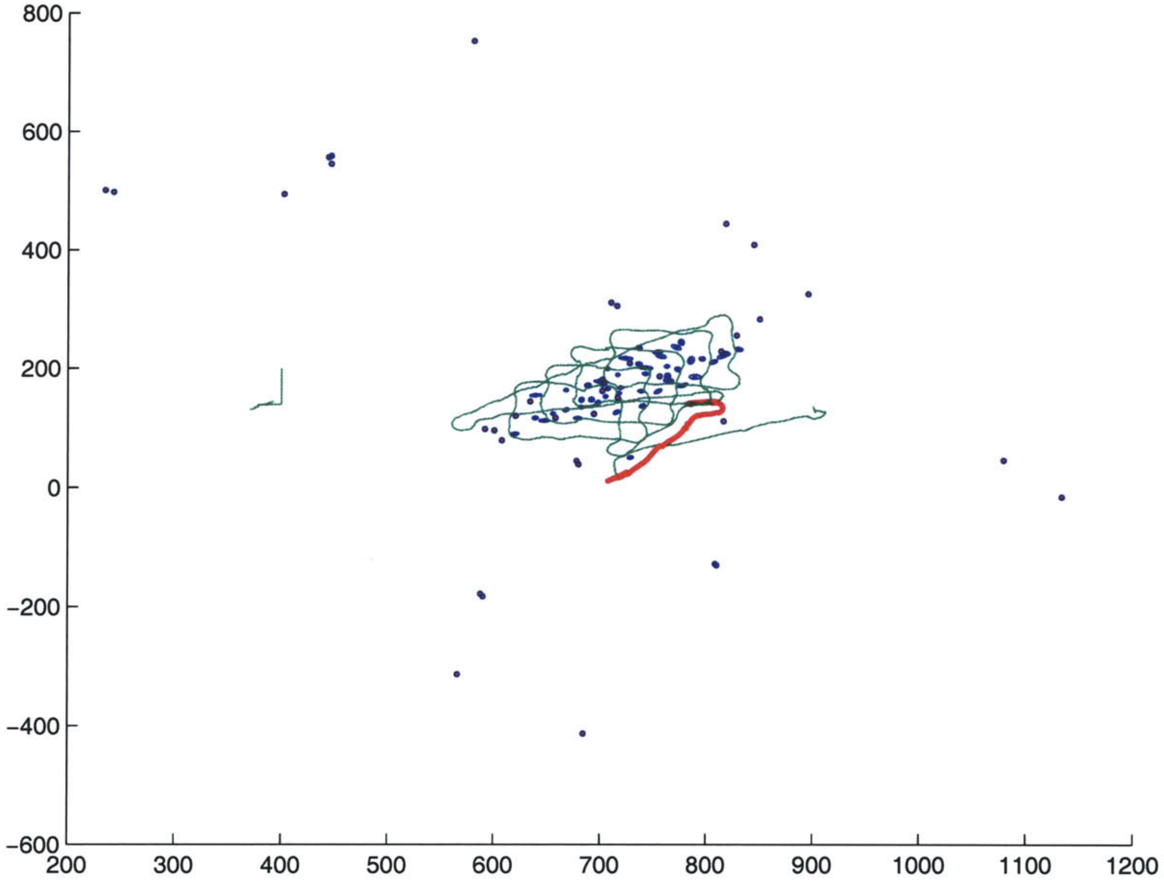


Figure 5-14: Mapped targets.

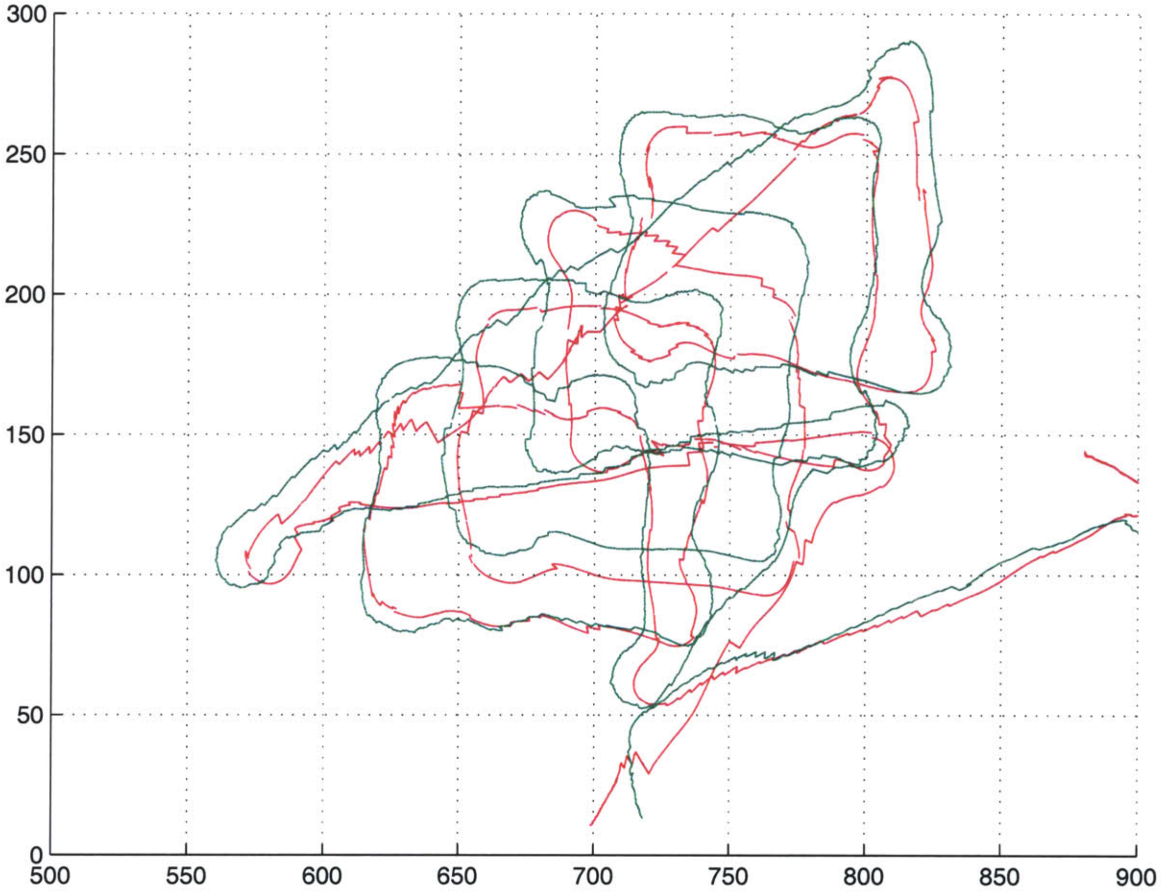


Figure 5-15: The green path corresponds to the estimate the vehicle used during the experiment. The red path is what was calculated by the CML algorithm without smoothing.

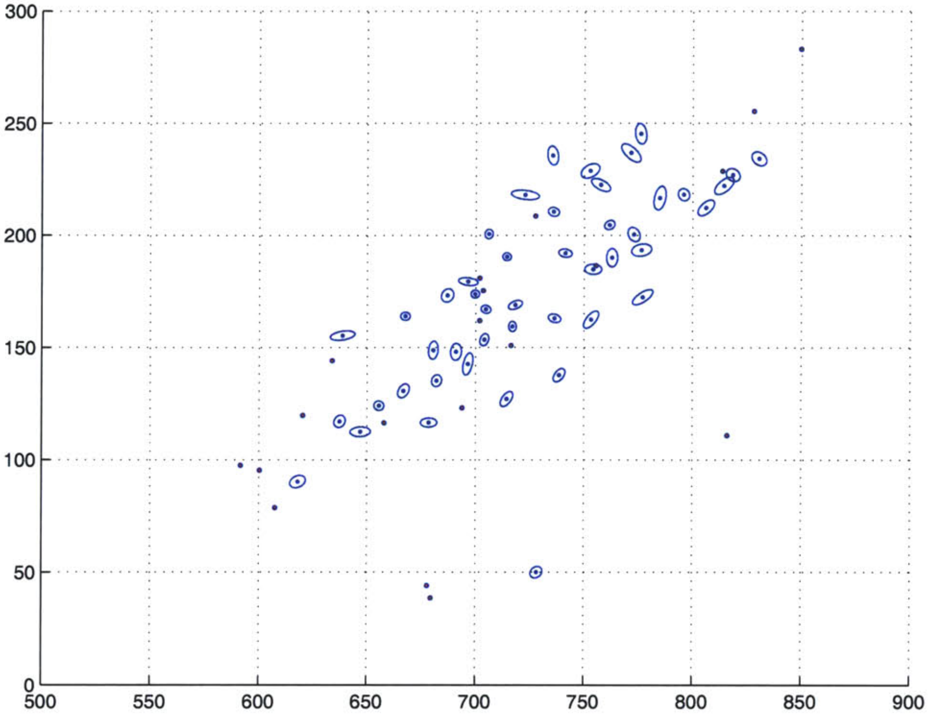


Figure 5-16: The ellipses are the 3-σ bounds for the mapped targets. The dots at the center of the ellipses are the estimates of the target locations. The points without ellipses correspond to where the AUV REMUS mapped targets using a sidescan sonar.

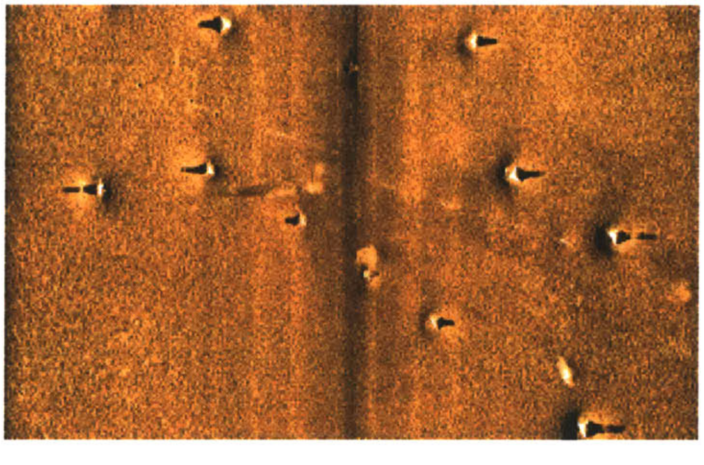


Figure 5-17: A sidescan image of the CML targets observed during GOATS 2002 (Klein DS5000 500 kHz sonar system image, courtesy of GESMA). The orientation of the linear array of targets is rotated 90 degrees counter-clockwise in relation to Figure 5-16.

Chapter 6

Conclusion

6.1 Summary

This thesis has presented a framework for CML and a method of preparing sonar data for CML. Most prior CML work has been concerned with stability, convergence, and computational complexity. The ability to process sensory data is typically assumed. Unfortunately, by designing an estimator while assuming perception, in some cases estimators have been designed which precluded perception. Early work implicitly assumed that a robot could instantaneously establish measurement to feature correspondences; we know, however, that this is not always true. It was also implicitly assumed that features were fully observable from a single measurement or vantage point. This is certainly not true for all features and types of sensors. By providing the robot with a "working memory" or history of robot positions, we have established a capability to perform CML in more generic circumstances.

Having developed a framework that better meets the constraints of perception, a new approach to sonar processing was created. First, a new representation was presented. Rather than processing measurements in a cartesian or feature space, measurements were left in a raw form with their temporal structure intact. Next, a computational theory to enable a dynamic observer to track locally curved features was developed. Based on continuity arguments, it was shown that a feature could be tracked without an explicit model. An algorithm was developed for tracking features without an explicit model.

The representation, computational theory, and algorithm were validated using data taken by a robot in the ocean. Our experiments demonstrate the capability to extract measurement trajectories from wide beam AUV sonar data. A preliminary CML result was generated to illustrate the potential of this approach for feature-based AUV navigation. Additional experiments are necessary, however, to fully "close the loop" to demonstrate real-time CML on an AUV using trajectory sonar perception.

6.2 Conclusions

We have demonstrated that robots need some sort of working memory. The move-sense-update cycle of the Kalman filter is too restrictive. Robots need more flexibility for perception, especially in light of the partially observable nature of the world. They need to be able to sit back and let the world tell its own story, making a final determination about sensory data only once there is a preponderance of evidence. Robots should act deliberately and judiciously, rather than haphazardly. Working memory allows the robot to put its observations into a broader context.

We have also demonstrated that a streamlined non-holonomic robot with a forward looking sonar which measures range, azimuth, and elevation needs the least information to track objects. As a rule of thumb, robots should be designed to maximize the sensory degrees of freedom and minimize the dynamic degrees of freedom. We have also shown that the traditional cartesian representation used to process sonar data obscures most of the structure and should not be used. Data should be left in a raw form for as long as possible.

In regards to how one should approach robotics research, it should be noted that most of the insights occurred when the robot was viewed as a complete system. By investigating navigation, we were forced to investigate perception. By investigating perception, we discovered that we needed to change the form of the estimators used in navigation, and we found that robot dynamics influenced the minimum information needed for perception. What one might conclude is that, at least occasionally, robots need to be viewed holistically.

6.3 Recommendations for future work

First and foremost, the approaches from this thesis need to be implemented in real time in the ocean. The expense of ocean experimentation makes it appropriate to initially test techniques in post-processing. However, nothing is as convincing as a real-time robot performance in the ocean.

Also, the techniques need to be tested in natural terrain. Although we certainly had no prior knowledge regarding the exact properties of the targets, they were still man made objects. It will be interesting to see how the algorithm performs with rocks. (Note to future researchers: be careful what you wish for. Apparently the Navy has a "standard rock".)

Methods for operating in dynamic environments need to be explored. Using the approach in this thesis, governing equations for dynamic objects can easily be derived. For instance, consider a locally curved target moving at velocity V_t with heading θ_t in a two dimensional environment. For a robot moving at velocity V with heading θ_r and with the target at bearing θ, the first substantial derivative of range in the non-holonomic case can be shown to be

$$\frac{Dr}{Dt} = -V\cos(\theta) + V_t\cos(\theta + \theta_r - \theta_t) = -V\cos(\theta) + V_t\cos(\theta + \Delta\theta_{rt}). \qquad (6.1)$$

Obviously, the dimensionality increases, but the problem can still be set up to guarantee codimensionality. The difficulty occurs with complex shapes. How can a robot recognize a moving target?

It was always accepted that this thesis was simply a stepping stone to object recognition. Although many have tried single step algorithms, it was felt that it would be wise to step back and create a solid foundation for higher level work. If one assumes that such a foundation has been created, then one might suggest moving on to higher level feature modeling and object recognition. However, at the level of early correspondence, a very important line of research has not been exhausted. Very little work has been done analyzing the importance of waveform selection. The Doppler shift is proportional to the first substantial derivative of range, and the Doppler shift rate is proportional to the second substantial derivative. If a perceiver can directly measure those Doppler quantities, the contribution of dynamics can be abstracted away. This can simplify tracking, if the range to the target can still be measured. Similarly, the constraints of perceiver design can be relaxed. If the perceiver/target geometry is such that it is in the second order regime (i.e. $\frac{D^2r}{Dt^2}$ is observable) then angular rates need to be observable for tracking. This may require a large aperture. With good Doppler information, much smaller apertures may be allowable, permitting smaller sonar perceivers. Bats have developed elaborate waveforms, featuring both chirps and tones. Many bats clearly use Doppler information for hunting; some identify moths by the Doppler shift of their beating wings. How Doppler information is used in navigation is less well established, but worthy of further study.

Towards modeling and object recognition, it will be important to evaluate the coupling between large scale motion and sonar sensing (this thesis has concerned small scale motion). Much dolphin sonar work has concerned their performance during "bite plate" experiments. In these experiments, the abilities of an immobilized dolphin are evaluated. In nature, dolphins are not stationary (Nevertheless, from the perspective of this thesis, they are well suited to early feature tracking. They are streamlined with a forward looking sonar, and their ear separation is sufficient for second order feature tracking, assuming quarter wavelength resolution). The paths dolphins follow while observing targets can be quite elaborate. Further investigation of the relationship between path selection and target modeling is warranted.

Finally, one might suggest designing future robots around perception. Too often, perception is an after thought, because it is difficult. We have shown in this thesis that perception is central to robot design. As such, why not make it the central design criterion?

> Artificial intelligence researchers are fond of pointing out that AI is often denied its rightful successes. The popular story goes that when nobody has any good idea of how to solve a particular sort of problem (e.g. playing chess) it is known as an AI problem. When an algorithm developed by AI researchers successfully tackles such a problem, however, AI detractors claim that since the problem was solvable by an algorithm, it wasn't really an AI problem after all. Thus AI never has any successes. But have you heard of an AI failure?
>
> I claim that AI researchers are guilty of the same (self) deception. They partition the problems they work on into two components. The AI component, which they solve, and the non-AI component which they don't

> solve. Typically, AI ”succeeds” by defining the parts of the problem that are unsolved as not AI. The principal mechanism for the partitioning is abstraction. Its application is usually considered part of good science, not, as it is in fact used in AI, as a mechanism for self delusion. In AI, abstraction is usually used to factor out all aspects of perception and motor skills. I argue below that these are the hard problems solved by intelligent systems, and further that the shape of solutions to these problems constrains greatly the correct solutions of the small pieces of intelligence which remain. -*Rodney Brooks* [18]

If, through this thesis, we have truly gained a toehold on the sonar perception problem, having developed an understanding of early perception as a function of sensing and dynamics, then perhaps it is time to build the world's first pure sonar perceiver.

Appendix A

Initialization Functions

A.1 Functions for Initialization of Features from Multiple Vantage Points

A.1.1 Initializing a Point from Two Range Measurements

Observing the range from the robot to a point defines a circle which the point must lie upon. Two observations define two circles, the intersection of which defines two points. Hence we can observe the location of a point object, subject to the ambiguity inherent to any quadratic equation. The ambiguity is resolved through the use of additional information, usually a third range measurement or a beamwidth constraint.

If the robot observes the point (x, y) from vantage points (x_1, y_1) and (x_2, y_2), measuring ranges r_1 and r_2, two circles can be defined:

$$(x - x_1)^2 + (y - y_1)^2 - r_1^2 = 0 \tag{A.1}$$

and

$$(x - x_2)^2 + (y - y_2)^2 - r_2^2 = 0 \tag{A.2}$$

Expanded, these equations become:

$$x^2 + y^2 - 2x_1x - 2y_1y + x_1^2 + y_1^2 - r_1^2 = 0 \tag{A.3}$$

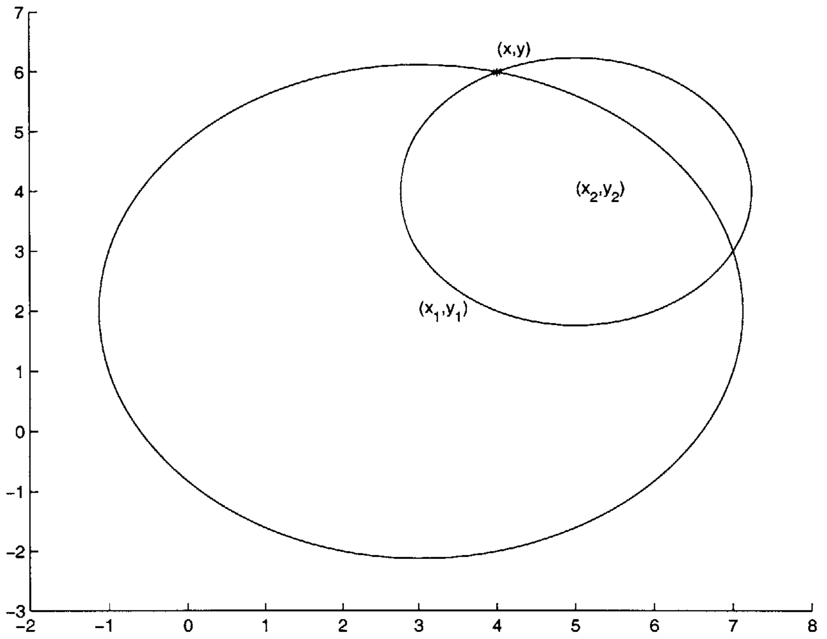


Figure A-1: Two range observations of the point (x,y).

and

$$x^2 + y^2 - 2x_2x - 2y_2y + x_2^2 + y_2^2 - r_2^2 = 0 \tag{A.4}$$

Taking the difference of these two equations yields

$$2x(x_2 - x_1) + 2y(y_2 - y_1) + x_1^2 - x_2^2 + y_1^2 - y_2^2 + r_2^2 - r_1^2 = 0. \tag{A.5}$$

How one proceeds from here depends on which denominator is chosen. For now, the derivation will continue as if $(x_2 - x_1)$ is not equal to zero. The case of it equaling zero, but $(y_2 - y_1)$ not equaling zero, will be derived afterwords. If both $(x_2 - x_1)$ and $(y_2 - y_1)$ are equal to zero there is no real solution, because a point object is only partially observable from a single vantage point.

Presuming that $(x_2 - x_1)$ is not equal to zero, we can further reduce Equation A.5:

$$x = \frac{y_2 - y_1}{x_1 - x_2} y + \frac{x_1^2 - x_2^2 + y_1^2 - y_2^2 + r_2^2 - r_1^2}{2(x_1 - x_2)}. \tag{A.6}$$

Reduced further, this becomes

$$x = \alpha_1 y + \alpha_2 \tag{A.7}$$

where

$$\alpha_1 = \frac{y_2 - y_1}{x_1 - x_2} \tag{A.8}$$

and

$$\alpha_2 = \frac{x_1^2 - x_2^2 + y_1^2 - y_2^2 + r_2^2 - r_1^2}{2(x_1 - x_2)}. \tag{A.9}$$

Substituting Equation A.7 into Equation A.1 yields

$$y^2(1+\alpha_1^2) + 2y(\alpha_1\alpha_2 - \alpha_1 x_1 - y_1) + \alpha_2^2 - 2\alpha_2 x_1 + x_1^2 + y_1^2 - r_1^2 = 0 \tag{A.10}$$

which can be reduced to

$$\lambda_1 y^2 + \lambda_2 y + \lambda_3 = 0 \tag{A.11}$$

where

$$\lambda_1 = (1+\alpha_1^2) \tag{A.12}$$

$$\lambda_2 = 2(\alpha_1\alpha_2 - \alpha_1 x_1 - y_1) \tag{A.13}$$

$$\lambda_3 = \alpha_2^2 - 2\alpha_2 x_1 + x_1^2 + y_1^2 - r_1^2. \tag{A.14}$$

Solving this quadratic and substituting the result into Equation A.7 results in two possible locations for the point:

$$y = \frac{-\lambda_2 \pm \sqrt{\lambda_2^2 - 4\lambda_1\lambda_3}}{2\lambda_1} \tag{A.15}$$

and

$$x = \alpha_1 \frac{-\lambda_2 \pm \sqrt{\lambda_2^2 - 4\lambda_1\lambda_3}}{2\lambda_1} + \alpha_2. \tag{A.16}$$

If $(x_2 - x_1)$ is equal to zero, but $(y_2 - y_1)$ is not, a second derivation is needed:

$$\beta_1 = \frac{x_2 - x_1}{y_1 - y_2}. \tag{A.17}$$

$$\beta_2 = \frac{x_1^2 - x_2^2 + y_1^2 - y_2^2 + r_2^2 - r_1^2}{2(y_1 - y_2)}. \tag{A.18}$$

$$\gamma_1 = (1 + \beta_1^2). \tag{A.19}$$

$$\gamma_2 = 2(\beta_1\beta_2 - \beta_1 y_1 - x_1). \tag{A.20}$$

$$\gamma_3 = \beta_2^2 - 2\beta_2 y_1 + x_1^2 + y_1^2 - r_1^2. \tag{A.21}$$

$$x = \frac{-\gamma_2 \pm \sqrt{\gamma_2^2 - 4\gamma_1\gamma_3}}{2\gamma_1}. \tag{A.22}$$

$$y = \beta_1 \frac{-\gamma_2 \pm \sqrt{\gamma_2^2 - 4\gamma_1\gamma_3}}{2\gamma_1} + \beta_2. \tag{A.23}$$

A.1.2 Initializing a line from two range measurements

There can be as many as four lines which are tangent to two circles. Considering only the cases in which the two circles are tangent to the same side of the line, there are two possible lines.

Given two circles (x_1, y_1, r_1) and (x_2, y_2, r_2), we want to find the two lines (ρ_1, θ_1) and (ρ_2, θ_2).

First, calculate the distance between the centers of the two circles d and the bearing θ_{12} of the second circle with respect to the first.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}. \tag{A.24}$$

$$\theta_{12} = \arctan 2(y_2 - y_1, x_2 - x_1). \tag{A.25}$$

The line connecting the centers of the two circles bisects the angle formed by the

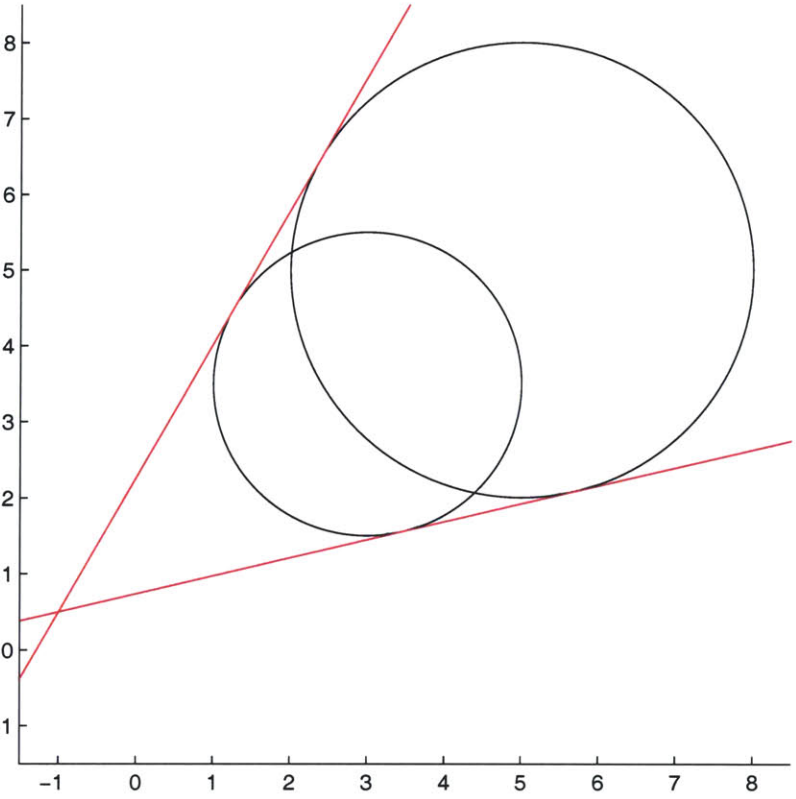


Figure A-2: Two lines tangent to two circles.

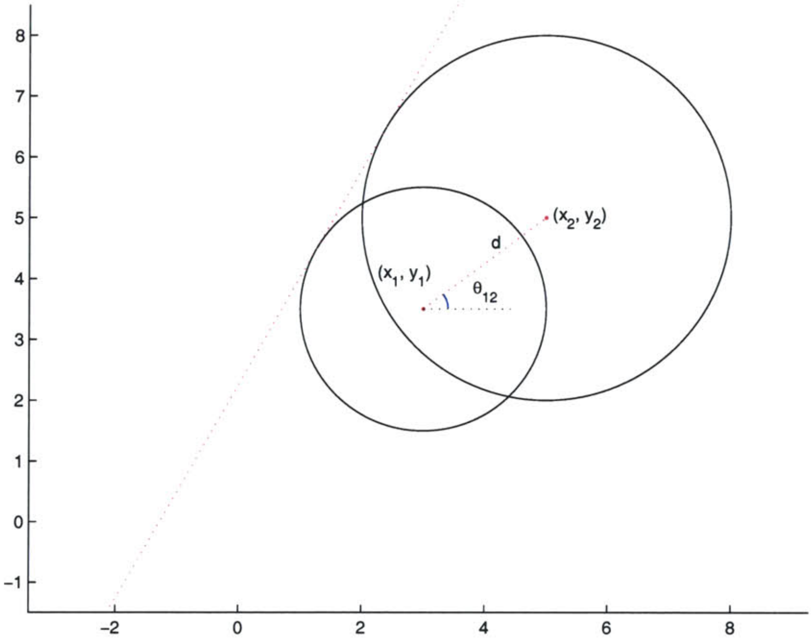


Figure A-3: Two range observations of the point (x,y).

two lines. Half of that angle is

$$\theta = \arccos \frac{r1 - r2}{d}. \tag{A.26}$$

The normals to the two lines are therefore

$$\theta_1 = \theta_{12} + \theta. \tag{A.27}$$

$$\theta_2 = \theta_{12} - \theta. \tag{A.28}$$

Next, the offset from the origin is determined. First, contact points on the first circle are found.

$$\begin{bmatrix} x_{c1} \\ y_{c1} \end{bmatrix} = \begin{bmatrix} x_1 + r_1 * \cos\theta_1 \\ y_1 + r_1 * \sin\theta_1 \end{bmatrix}. \tag{A.29}$$

$$\begin{bmatrix} x_{c2} \\ y_{c2} \end{bmatrix} = \begin{bmatrix} x_1 + r_1 * \cos\theta_2 \\ y_1 + r_1 * \sin\theta_2 \end{bmatrix}. \tag{A.30}$$

Then, the distance of the contacts from the origin is calculated.

$$\alpha_1 = \sqrt{x_{c1}^2 + y_{c1}^2}. \tag{A.31}$$

$$\alpha_2 = \sqrt{x_{c2}^2 + y_{c2}^2}. \tag{A.32}$$

Similarly, their bearings from the origin are calculated.

$$\beta_1 = \arctan 2(y_{c1}, x_{c1}). \tag{A.33}$$

$$\beta_2 = \arctan 2(y_{c2}, x_{c2}). \tag{A.34}$$

Having constructed two right triangles with hypotheni and angles of (α_1, β_1) and (α_2, β_2), it is straightforward to calculate the length of the legs which are normal to the desired lines.

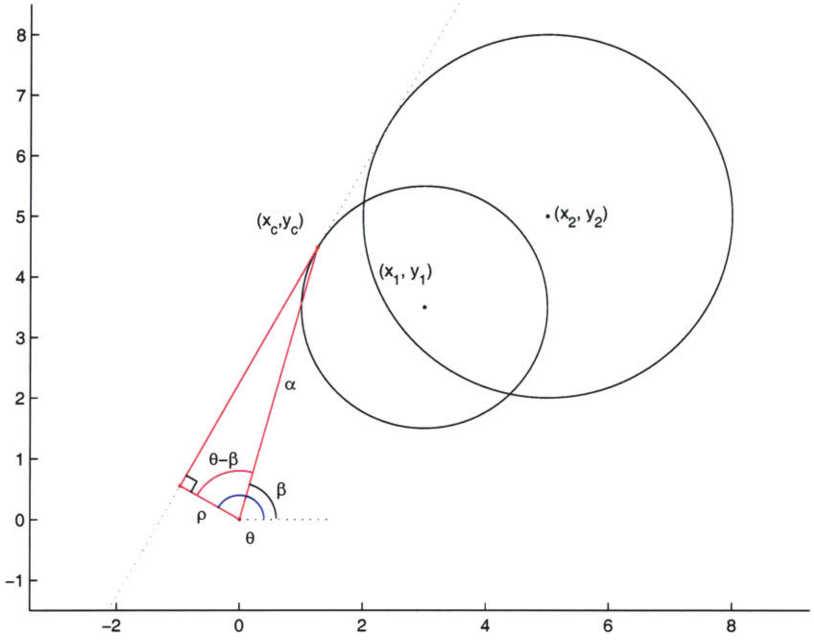


Figure A-4: Two range observations of the point (x,y).

$$\rho_1 = \alpha_1 \cos(\theta_1 - \beta_1). \tag{A.35}$$

$$\rho_2 = \alpha_2 \cos(\theta_2 - \beta_2). \tag{A.36}$$

A.1.3 Initializing a Line from a Range Measurement and a Colinear Point

Initializing the line which is tangent to circle (x_1, y_1, r_1) and passes through point (x_2, y_2) is equivalent to finding the line which is tangent to two circles when one of the circles has zero radius. There are two solutions. Without proof, the two results, (ρ_1, θ_1) and (ρ_2, θ_2), are

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}. \tag{A.37}$$

$$\theta_{pc} = \arctan 2(y_2 - y_1, x_2 - x_1). \tag{A.38}$$

$$\beta = \arccos(\frac{r}{d}). \tag{A.39}$$

$$\theta_1 = \alpha + \beta. \tag{A.40}$$

$$\theta_2 = \alpha - \beta. \tag{A.41}$$

$$\begin{bmatrix} x_{c1} \\ y_{c1} \end{bmatrix} = \begin{bmatrix} x_1 + r_1 * \cos\theta_1 \\ y_1 + r_1 * \sin\theta_1 \end{bmatrix}. \tag{A.42}$$

$$\begin{bmatrix} x_{c2} \\ y_{c2} \end{bmatrix} = \begin{bmatrix} x_1 + r_1 * \cos\theta_2 \\ y_1 + r_1 * \sin\theta_2 \end{bmatrix}. \tag{A.43}$$

$$\alpha_1 = \sqrt{x_{c1}^2 + y_{c1}^2}. \tag{A.44}$$

$$\alpha_2 = \sqrt{x_{c2}^2 + y_{c2}^2}. \tag{A.45}$$

$$\beta_1 = \arctan 2(y_{c1}, x_{c1}). \tag{A.46}$$

$$\beta_2 = \arctan 2(y_{c2}, x_{c2}). \tag{A.47}$$

$$\rho_1 = \alpha_1 \cos(\theta_1 - \beta_1). \tag{A.48}$$

$$\rho_2 = \alpha_2 \cos(\theta_2 - \beta_2). \tag{A.49}$$

A.1.4 Initializing a Point from a Line and a Range Measurement

A robot position and a range define a circle (x, y, r). Provided that the circle intersects the line (ρ, θ), we want to find the intersection points.

First, we find the distance from the center of the circle to the line, which is defined as

$$d = |\rho - x\sin(\theta) - y\cos(\theta)|. \tag{A.50}$$

This is the first leg of a right triangle. The hypotenuse is the range measurement. The angle between two is

$$\beta = \arccos(\frac{d}{r}). \tag{A.51}$$

Knowing the bearing to the line is, by definition, θ, the two intersections are therefore

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} x + \rho\cos(\theta - \beta) \\ y + \rho\sin(\theta - \beta) \end{bmatrix} \tag{A.52}$$

and

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} x + \rho\cos(\theta + \beta) \\ y + \rho\sin(\theta + \beta) \end{bmatrix} \tag{A.53}$$

A.1.5 Initializing a sphere from four spheres

An LBL beacon in the ocean has four degrees of freedom. If it is to be mapped, one must determine its position and its turnaround time. If a beacon is treated as a sphere, it is as if the robot measures the far side. If only range information is available, to estimate the state of a beacon the robot must determine the sphere which is tangent to four spheres.

Assume four robot positions $[x_1\ y_1\ z_1]^T$, $[x_2\ y_2\ z_2]^T$, $[x_3\ y_3\ z_3]^T$, and $[x_4\ y_4\ z_4]^T$. Also, assume four ranges r_1, r_2, r_3, and r_4. Calculate the following temporary variables.

$$\begin{bmatrix} A_1 \\ B_1 \\ C_1 \\ D_1 \end{bmatrix} = \begin{bmatrix} \frac{x_2-x_1}{r_1-r_2} \\ \frac{y_2-y_1}{r_1-r_2} \\ \frac{z_2-z_1}{r_1-r_2} \\ \frac{x_1^2-x_2^2+y_1^2-y_2^2+z_1^2-z_2^2+r_2^2-r_1^2}{2(r_1-r_2)} \end{bmatrix}. \tag{A.54}$$

$$\begin{bmatrix} A_2 \\ B_2 \\ C_2 \\ D_2 \end{bmatrix} = \begin{bmatrix} \frac{x_3-x_1}{r_1-r_3} \\ \frac{y_3-y_1}{r_1-r_3} \\ \frac{z_3-z_1}{r_1-r_3} \\ \frac{x_1^2-x_3^2+y_1^2-y_3^2+z_1^2-z_3^2+r_3^2-r_1^2}{2(r_1-r_3)} \end{bmatrix}. \tag{A.55}$$

$$\begin{bmatrix} A_3 \\ B_3 \\ C_3 \\ D_3 \end{bmatrix} = \begin{bmatrix} \frac{x_4-x_1}{r_1-r_4} \\ \frac{y_4-y_1}{r_1-r_4} \\ \frac{z_4-z_1}{r_1-r_4} \\ \frac{x_1^2-x_4^2+y_1^2-y_4^2+z_1^2-z_4^2+r_4^2-r_1^2}{2(r_1-r_4)} \end{bmatrix}. \tag{A.56}$$

Using these temporary variables, calculate a new set of temporary variables.

$$\begin{bmatrix} E_1 \\ F_1 \\ G_1 \end{bmatrix} = \begin{bmatrix} \frac{B_1-B_2}{A_2-A_1} \\ \frac{C_1-C_2}{A_2-A_1} \\ \frac{D_1-D_2}{A_2-A_1} \end{bmatrix}. \tag{A.57}$$

$$\begin{bmatrix} E_2 \\ F_2 \\ G_2 \end{bmatrix} = \begin{bmatrix} \frac{B_1-B_3}{A_3-A_1} \\ \frac{C_1-C_3}{A_3-A_1} \\ \frac{D_1-D_3}{A_3-A_1} \end{bmatrix}. \tag{A.58}$$

Using these temporary variables, create more temporary variables.

$$\begin{bmatrix} H \\ J \end{bmatrix} = \begin{bmatrix} \frac{F_1-F_2}{E_2-E_1} \\ \frac{G_1-G_2}{E_2-E_1} \end{bmatrix}. \tag{A.59}$$

$$\begin{bmatrix} K \\ L \end{bmatrix} = \begin{bmatrix} E_1H + F_1 \\ E_1J + G_1 \end{bmatrix}. \tag{A.60}$$

$$\begin{bmatrix} M \\ N \end{bmatrix} = \begin{bmatrix} A_1K + B_1H + C_1 \\ A_1L + B_1J + D_1 \end{bmatrix}. \tag{A.61}$$

Finally, we set up a quadratic to solve for the z coordinate of the center. The quadratic has the following coefficients.

$$\begin{bmatrix} P \\ Q \\ R \end{bmatrix} = \begin{bmatrix} 1 + H^2 + K^2 - M^2 \\ -2z_1 + 2HJ - 2y_1H + 2KL - 2x_1K - 2MN - 2r_1M \\ z_1^2 + y_1^2 - 2y_1J + J^2 + x_1^2 - 2x_1L + L^2 - r_1^2 - 2r_1N - N^2 \end{bmatrix}. \tag{A.62}$$

Solving this quadratic yields two values of the center of the sphere, z_{c1} and z_{c2}.

$$\begin{bmatrix} z_{c1} \\ z_{c2} \end{bmatrix} = \begin{bmatrix} \frac{-Q+\sqrt{Q^2-4PR}}{2P} \\ \frac{-Q-\sqrt{Q^2-4PR}}{2P} \end{bmatrix}. \tag{A.63}$$

From these values, we can determine the states of the two possible spheres. Since this is a quadratic, there are two solutions.

$$\begin{bmatrix} x_{c1} \\ y_{c1} \\ z_{c1} \\ \rho_1 \end{bmatrix} = \begin{bmatrix} Kz_{c1} + L \\ Hz_{c1} + J \\ z_{c1} \\ Mz_{c1} + N \end{bmatrix}. \tag{A.64}$$

and

$$\begin{bmatrix} x_{c2} \\ y_{c2} \\ z_{c2} \\ \rho_2 \end{bmatrix} = \begin{bmatrix} Kz_{c2} + L \\ Hz_{c2} + J \\ z_{c2} \\ Mz_{c2} + N \end{bmatrix}. \tag{A.65}$$

Bibliography

[1] Michael Adams, Wojtek Halliop. Aluminum energy semi-fuel cell systems for underwater applications: The state of the art and the way ahead. In *Oceans 2002*, pages 199–202, 2002.

[2] John D. Altringham. *Bats: Biology and Behavior.* Oxford University Press, 1996.

[3] B. D. O. Anderson and J. B. Moore. *Optimal filtering.* Englewood Cliffs, N.J.: Prentice-Hall, 1979.

[4] W. Au. *The Sonar of Dolphins.* New York: Springer-Verlag, 1993.

[5] N. Ayache and O. Faugeras. Maintaining representations of the environment of a mobile robot. *IEEE Trans. Robotics and Automation*, 5(6):804–819, 1989.

[6] D. Ballard and C. Brown. *Computer Vision.* Prentice-Hall, 1982.

[7] R. Ballard. Soundings: The Newsletter of the JASON Project for Education, Waltham, MA, http://seawifs.gsfc.nasa.gov/scripts/JASON.html, 1996.

[8] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association.* Academic Press, 1988.

[9] Y. Bar-Shalom and X. R. Li. *Estimation and Tracking: Principles, Techniques, and Software.* Artech House, 1993.

[10] B. Barshan and R. Kuc. Differentiating sonar reflections from corners and planes by employing an intelligent sensor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-12(6):560–569, June 1990.

[11] J. G. Bellingham, C. A. Goudey, T. R. Consi, J. W. Bales, D. K. Atwood, J. J. Leonard, and C. Chryssostomidis. A second generation survey AUV. In *IEEE Conference on Autonomous Underwater Vehicles*, Cambridge, MA, 1994.

[12] Y. Borenstein, H. Everett, and L. Feng. *Navigating Mobile Robots: Systems and Techniques.* A. K. Peters, 1996.

[13] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller. An atlas framework for scalable mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

[14] M. Bosse, R. Rikoski, J. Leonard, and S. Teller. Vanishing points and 3d lines from omnidirectional video. In *International Conference on Image Processing*, Rochester, NY, September 2002.

[15] Albert M. Bradley. Low power navigation and control for long range autonomous underwater vehicles. In *Proceedings of the Second International Offshore and Polar Engineering Conference*, pages 473–478, 1992.

[16] A. M. Bradley, A. R. Duester, S. R. Liberatore. A fast hydrographic profiling system. In *Proceedings, Oceans 1991, Oceanic Engineering Society of IEEE*, 1991.

[17] Valentino Braitenberg. *Vehicles: Experiments in Synthetic Psychology.* The MIT Press, 1984.

[18] Rodney A. Brooks. *Cambrian Intelligence: The Early History of the New AI.* The MIT Press, 1999.

[19] R. S. Bucy and K. D. Senne. Digital synthesis of nonlinear filters. *Automatica*, 7:287–298, 1971.

[20] R. N. Carpenter. Concurrent mapping and localization and map matching on autonomous underwater vehicles. In *IEEE Oceans*, pages 380–389, Hawaii, 2001.

[21] J. A. Castellanos and J. D. Tardos. *Mobile Robot Localization and Map Building: A Multisensor Fusion Approach.* Kluwer Academic Publishers, Boston, 2000, 2000. To appear.

[22] K. S. Chong and L. Kleeman. Sonar based map building in large indoor environments. Technical Report MECSE-1997-1, Department of Electrical and Computer Systems Engineering, Monash University, 1997.

[23] K. S. Chong and L. Kleeman. Sonar feature map building for a mobile robot. In *Proc. IEEE Int. Conf. Robotics and Automation*, 1997.

[24] Howie Choset, Keiji Nagatani. Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation*, 17(2):125–137, April 2001.

[25] H. Christensen. SLAM summer school, 2002. `http://www.cas.kth.se/SLAM/toc.html`.

[26] Per Collinder. *A History of Marine Navigation.* B. T. Batsford Ltd., 1954.

[27] A. J. Davison. *Mobile Robot Navigation Using Active Vision.* PhD thesis, University of Oxford, 1998.

[28] M. W. M. G. Dissanayake, P. Newman, H. F. Durrant-Whyte, S. Clark, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. In *Sixth International Symposium on Experimental Robotics*, March 1999.

[29] M. W. M. G. Dissanayake, P. Newman, H. F. Durrant-Whyte, S. Clark, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotic and Automation*, 17(3):229–241, June 2001.

[30] A. Doucet, N. de Freitas, and N. Gordan, editors. *Sequential Monte Carlo Methods in Practice.* Springer-Verlag, 2001.

[31] Yoichiro Endo, Ronald C. Arkin. Anticipatory robot navigation by simultaneously localizing and building a cognitive map. Technical Report GIT-COGSCI-2003/01, Georgia Tech, 2003.

[32] O. Faugeras, Q-T. Luong, and T. Papadopoulo. *The Geometry of Multiple Images.* MIT Press, 2001.

[33] H. J. S. Feder. *Simultaneous Stochastic Mapping and Localization.* PhD thesis, Massachusetts Institute of Technology, 1999.

[34] H. J. S. Feder, J. J. Leonard, and C. M. Smith. Adaptive mobile robot navigation and mapping. *Int. J. Robotics Research*, 18(7):650–668, July 1999.

[35] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[36] Stephen D. Fleischer. *Bounded-Error Vision-Based Navigation of Autonomous Underwater Vehicles.* PhD thesis, Stanford University, 2000.

[37] A. Freedman. A mechanism of acoustic echo formation. *Acustica*, 12(1):10–21, 1962.

[38] W. E. L. Grimson. *Object Recognition by Computer: The Role of Geometric Constraints.* MIT Press, 1990. (With contributions from T. Lozano-Perez and D. P. Huttenlocher).

[39] J. Guivant and E. Nebot. Optimization of the simultaneous localization and map building algorithm for real time implementation. *IEEE Transactions on Robotic and Automation*, 17(3):242–257, June 2001.

[40] J-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *International Symposium on Computational Intelligence in Robotics and Automation*, 1999.

[41] D. Hähnel, D. Schulz, and W. Burgard. Map building with mobile robots in populated environments. In *Proc. IEEE Int. Workshop on Intelligent Robots and Systems*, 2002.

[42] A. Heale and L. Kleeman. Fast target classification using sonar. In *Proc. IEEE Int. Workshop on Intelligent Robots and Systems*, pages 1446–1451, 2001.

[43] Donald D. Hoffman. *Visual Intelligence.* W. W. Norton & Company Ltd., 1998.

[44] B. K. P. Horn. *Robot Vision.* MIT Press, 1986.

[45] F. Hover. *Methods for positioning deeply towed underwater cables.* PhD thesis, Massachusetts Institute of Technology, 1993.

[46] John Charles Hyland. *A Reflexive Mine Avoidance Approach for Autonomous Underwater Vehicles.* PhD thesis, University of Florida, 1993.

[47] R. E. Kalman, P. L. Falb, and M. A. Arbib. *Topics in mathematical systems theory.* Mc Graw Hill, 1969.

[48] L. Kleeman. Advanced sonar sensing. In R. Jarvis and A. Zelinsky, editors, *Robotics Research: The Tenth International Symposium*, Lorne, Australia, 2001. Proceedings to be published by Springer Verlag.

[49] L. Kleeman and R. Kuc. Mobile robot sonar for target localization and classification. Technical Report ISL-9301, Intelligent Sensors Laboratory, Yale University, 1993.

[50] J. Knight. *Towards Fully Autonomous Navigation.* PhD thesis, University of Oxford, 2003.

[51] David C. Knill and Whitman Richards, editors. *Perception as Bayesian Inference.* Cambridge University Press, 1996.

[52] R. Kuc. Fusing binaural sonar information for object recognition. In *IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 727–735, 1996.

[53] Roman Kuc. Biomimetic sonar recognizes objects using binaural information. *Journal of the Acoustical Society of America*, 102(2), August 1997.

[54] B. Kuipers and P. Beeson. Bootstrap learning for place recognition. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, Edmonton, Canada, 2002. AAAI.

[55] B. J. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 2000.

[56] M. B. Larsen. High performance doppler-inertial navigation — experimental results. In *IEEE Oceans*, 2000.

[57] J. Leonard and P. Newman. Consistent, convergent, and constant-time SLAM. In *IJCAI*, 2003. To appear.

[58] J. J. Leonard, R. N. Carpenter, and H. J. S. Feder. Stochastic mapping using forward look sonar. In *International Conference on Field and Service Robotics*, pages 69–74, Pittsburgh, Pennsylvania, August 1999.

[59] J. J. Leonard, I. J. Cox, and H. F. Durrant-Whyte. Dynamic map building for an autonomous mobile robot. *Int. J. Robotics Research*, 11(4):286–298, August 1992.

[60] J. J. Leonard and H. F. Durrant-Whyte. *Directed Sonar Sensing for Mobile Robot Navigation.* Boston: Kluwer Academic Publishers, 1992.

[61] J. J. Leonard and H. J. S. Feder. Decoupled stochastic mapping. *IEEE J. Ocean Engineering*, 26(4):561–571, 2001.

[62] J. J. Leonard, R. J. Rikoski, P. M. Newman, and M. Bosse. Mapping partially observable features from multiple uncertain vantage points. *Int. J. of Robot. Res.*, 2002. To Appear.

[63] John J. Leonard, Richard J. Rikoski, Paul M. Newman, José Neira, and Juan. D. Tardós. Towards robust data association and feature modeling for concurrent mapping and localization. In *Proceedings of the Tenth International Symposium on Robotics Research*, Victoria, Australia, November 2001.

[64] David Lewis. *We, the Navigators: the ancient art of landfinding in the Pacific.* University of Hawaii Press, 1994.

[65] J. MacCormick. *Probabilistic modelling and stochastic algorithms for visual localisation and tracking.* PhD thesis, University of Oxford, 2000.

[66] S. Majumder. *Sensor Fusion and Feature-based Navigation for Subsea Robots.* PhD thesis, University of Sydney, 2001.

[67] D. Marr. *Vision.* New York: W. H. Freeman and Co., 1982.

[68] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, Edmonton, Canada, 2002. AAAI.

[69] S. Mori, C. Chong, E. Tse, and R. Wishner. Tracking and classifying multiple targets without a priori identification. *IEEE Trans. on Automatic Control*, AC-31(5), May 1986.

[70] P. Moutarlier and R. Chatila. An experimental system for incremental environment modeling by an autonomous mobile robot. In *1st International Symposium on Experimental Robotics*, Montreal, June 1989.

[71] P. Moutarlier and R. Chatila. Stochastic multi-sensory data fusion for mobile robot location and environment modeling. In *5th Int. Symposium on Robotics Research*, pages 207–216, 1989.

[72] R. Muller, H. Schnitzler. Acoustic flow perception in cf-bats: Properties of the available cues. *J. Acoustical Society of America*, 105(5):2958–2966, May 1999.

[73] R. Muller, H. Schnitzler. Acoustic flow perception in cf-bats: Extraction of parameters. *J. Acoustical Society of America*, 108(3):1298–1307, September 2000.

[74] Keiji Nagatani, Howie Choset, and Nicole Lazar. The arc-transversal median algorithm: an approach to increasing ultrasonic sensor accuracy. In *IEEE International Conference on Robotics and Automation*, pages 644–651, 1999.

[75] J. Neira and J.D. Tardós. Data association in stochastic mapping using the joint compatibility test. *IEEE Trans. on Robotics and Automation*, 17(6):890–897, 2001.

[76] J. N. Newman. *Marine Hydrodynamics*. The MIT Press, 1977.

[77] P. M. Newman. *On the structure and solution of the simultaneous localization and mapping problem*. PhD thesis, University of Sydney, 1999.

[78] H. Peremans, K. Audenaert, and Campenhout J. M. Van. A high-resolution sensor based on tri-aural perception. *IEEE Transactions on Robotics And Automation*, 9(1):36–48, Feb. 1993. USA.

[79] G. T. Reader, J. Potter, J. G. Hawley. The evolution of auv power systems. In *Oceans 2002*, pages 191–198, 2002.

[80] D. B. Reid. An algorithm for tracking multiple targets. *IEEE Trans. on Automatic Control*, AC-24(6), Dec. 1979.

[81] Whitman Richards, editor. *Natural Computation*. The MIT Press, 1988.

[82] I. Tena Ruiz, D. M. Lane, M. J. Chantler. A comparison of inter-frame feature measures for robust object classification in sector scan sonar image sequences. *IEEE J. of Oceanic Eng.*, 24(4):458–469, October 1999.

[83] H. Schmidt. GOATS-98 — AUV network sonar concepts for shallow water mine countermeasures. Technical Report SACLANTCEN SR-302, SACLANT Undersea Research Centre, 1998.

[84] H. Singh, D. Yoerger, R. Bachmayer, A. Bradley, and W. Stewart. Sonar mapping with the autonomous benthic explorer (ABE). In *Proc. Int. Symp. on Unmanned Untethered Submersible Technology*, pages 367–375, September 1995.

[85] C. M. Smith. *Integrating Mapping and Navigation*. PhD thesis, Massachusetts Institute of Technology, 1998.

[86] R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. In *4th International Symposium on Robotics Research*. MIT Press, 1987.

[87] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In I. Cox and G. Wilfong, editors, *Autonomous Robot Vehicles*. Springer-Verlag, 1990.

[88] H. W. Sorenson. Recursive estimation for nonlinear dynamic systems. In James C. Spall, editor, *Bayesian Analysis of Time Series and Dynamic Models*, pages 127–166. New York: Marcel Dekker, 1988.

[89] R. Stokey and T. Austin. Sequentail, long baseline navigation for REMUS, an autonomous underwater vehicle. In *SPIE International Symposium on Information systems for Navy divers and autonomous underwater vehicles operating in the surf zone*, volume 3711, pages 3711–25, 1999.

[90] J.D. Tardós, J. Neira, P.M. Newman, and J.J. Leonard. Robust mapping and localization in indoor environments using sonar data. *Int. J. Robotics Research*, 21(4):311–330, April 2002.

[91] S. Thrun. An online mapping algorithm for teams of mobile robots. *Int. J. of Robot. Res.*, 20(5):335–363, May 2001.

[92] S. Thrun, D. Fox, and W. Burgard. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, 31:29–53, 1998.

[93] S. Thrun, D. Hähnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro and S. Thayer, and W. Whittaker. A system for volumetric robotic mapping of abandoned mines. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

[94] S. Thrun, D. Koller, Z. Ghahramani, H. Durrant-Whyte, and Ng A.Y. Simultaneous mapping and localization with sparse extended information filters. In *Proceedings of the Fifth Internationl Workshop on Algorithmic Foundations of Robotics*, Nice, France, 2002. Forthcoming.

[95] C. von Alt, B. Allen, T. Austin, and R. Stokey. Remote environmental monitoring units. In *AUV 94*, 1994.

[96] C.-C. Wang, C. Thorpe, and S. Thrun. Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results

from a ground vehicle in crowded urban areas. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

[97] J.A. Whitehead. Thermohaline ocean processes and models. *Annual Review Fluid Mechanics*, 24:89–113, 1995.

[98] Olle Wijk. *Triangulation Based Fusion of Sonar Data with Application in Mobile Robot Mapping and Localization.* PhD thesis, Royal Institute of Technology, Stockholm, Sweden, 2001.

[99] O. Wijk, H. Christensen. Triangulation based fusion of sonar data with application in robot pose tracking. *IEEE Trans. Robotics and Automation*, 16(6):740–752, December 2000.

[100] O. Wijk, H. I. Christensen. Localization and navigation of a mobile robot using natural point landmarks extracted from sonar data. *Robotics and Autonomous Systems*, 31:31–42, 2000.

[101] S. Williams. *Efficient solutions to autonomous navigation and mapping problems.* PhD thesis, University of Sydney, 2001.

[102] Andrew P. Witkin. Intensity-based edge classification. In Whitman Richards, editor, *Natural Computation.* MIT Press, 1988.

[103] Guido Zunino and Henrik I Christensen. SLAM in realistic environments. In M. Devy, editor, *Symposium on Intelligent Robotic Systems*, Toulouse, F, July 2001.