2

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

# SONAR LOCALIZATION OF AN AUTONOMOUS UNDERWATER VEHICLE

by

Enis Percin

December, 1993

Thesis Advisor : Roberto Cristi

|  |  |
|---|---|
| **REPORT DOCUMENTATION PAGE** | Form Approved OMB No. 0704 |

|  |  |  |
|---|---|---|
| 1. **AGENCY USE ONLY** *(Leave blank)* | 2. **REPORT DATE** December 1993 | 3. **REPORT TYPE AND DATES COVERED** Master's Thesis |

|  |  |
|---|---|
| 4. **TITLE AND SUBTITLE** SONAR LOCALIZATION OF AN AUTONOMOUS UNDERWATER VEHICLE, UNCLASSIFIED | 5. **FUNDING NUMBERS** |
| 6. **AUTHOR(S)** Enis Percin | |
| 7. **PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)** Naval Postgraduate School Monterey, CA 93943-5000 | 8. **PERFORMING ORGANIZATION REPORT NUMBER** |
| 9. **SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)** | 10. **SPONSORING/MONITORING AGENCY REPORT NUMBER** |

11. **SUPPLEMENTARY NOTES** The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Def nse or the U.S. Government.

|  |  |
|---|---|
| 12a. **DISTRIBUTION/AVAILABILITY STATEMENT** Approved for public release; distribution is unlimited. | 12b. **DISTRIBUTION CODE** A |

13. **ABSTRACT** *(maximum 200 words)*

Two different algorithms to navigate an AUV within a charted environment are presented. They use sonar returns and a local map together with the dynamic model to estimate the vehicle's position and acceleration at all times. Kalman filtering techniques are used to compute the estimates. The main difficulty is the presence of uncharted obstacles, which are identified by the potential function algorithm. Results show that first algorithm works in an environment without obstacles. Results from the application of the potential function algorithm in a pool using Tritech ST725 high resolution sonar show the feasibility and robustness of the potential function approach to the navigation problem.

|  |  |
|---|---|
| 14. **SUBJECT TERMS** AUV; Extended Kalman Filter; Nonlinear Systems; Linearization; Estimation; Potential Function; ARX Model | 15. **NUMBER OF PAGES** 74 |
| | 16. **PRICE CODE** |

|  |  |  |  |
|---|---|---|---|
| 17. **SECURITY CLASSIFICATION OF REPORT** Unclassified | 18. **SECURITY CLASSIFICATION OF THIS PAGE** Unclassified | 19. **SECURITY CLASSIFICATION OF ABSTRACT** Unclassified | 20. **LIMITATION OF ABSTRACT** UL |

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

# Sonar Localization of an Autonomous Underwater Vehicle

by

Enis Percin
Lieutenant Junior Grade, Turkish Navy
B.S., Turkish Naval Academy, 1987

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN ENGINEERING SCIENCE (EE)

from the

NAVAL POSTGRADUATE SCHOOL
December 1993

Author: [illegible]
Enis Percin

Approved by: [illegible]
Roberto Cristi, Thesis Advisor

[illegible]
Harold A. Titus, Second Reader

Michael A. Morgan
Michael A. Morgan, Chairman
Department of Electrical and Computer Engineering

## ABSTRACT

Two different algorithms to navigate an AUV within a charted environment are presented. They use sonar returns and a local map together with the dynamic model to estimate the vehicle's position and acceleration at all times. Kalman filtering techniques are used to compute the estimates. The main difficulty is the presence of uncharted obstacles, which are identified by the potential function algorithm. Results from the application of the potential function algorithm in a pool using Tritech ST725 high resolution sonar show the feasibility and robustness of the potential function approach to the navigation problem.

## TABLE OF CONTENTS

I. INTRODUCTION . . . . . . . . . . 1

A. GENERAL . . . . . . . . . . 1

B. AIM OF THE STUDY . . . . . . . . . . 2

C. METHOD OF APPROACH . . . . . . . . . . 2

II. A GEOMETRICAL MODEL OF ENVIRONMENT . . . . . . . . . . 5

A. GENERAL . . . . . . . . . . 5

B. DISCRETE KALMAN FILTER EQUATIONS . . . . . . . . . . 5

C. LINEARIZATION . . . . . . . . . . 8

D. EXTENDED KALMAN FILTER . . . . . . . . . . 9

E. MODELLING FOR PLANAR MOTION OF THE AUV . . . . . . . 11

F. SIMULATION RESULTS . . . . . . . . . . 15

III. POTENTIAL FUNCTION APPROACH FOR ENVIRONMENT . . . . . . 21

A. GENERAL . . . . . . . . . . 21

B. MODELLING ASSUMPTIONS . . . . . . . . . . 21

C. SIMULATION RESULTS . . . . . . . . . . 23

D. ESTIMATION OF SYSTEM PARAMETERS . . . . . . . . . . 32

1. Model Structures . . . . . . . . . . . . . . . . . . . . . . . . . . . 32

2. ARMAX Models and Difference Equations . . . . . . . . . . . . . 33

3. Recursive Estimation . . . . . . . . . . . . . . . . . . . . . . . . . 34

4. Simulation Results . . . . . . . . . . . . . . . . . . . . . . . . . . 36

IV. SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS . . . . . . . 47

A. SUMMARY . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 47

B. CONCLUSIONS . . . . . . . . . . . . . . . . . . . . . . . . . . . 47

C. RECOMMENDATIONS . . . . . . . . . . . . . . . . . . . . . . . . 48

APPENDIX A . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 49

A. PROGRAM STRUCTURE . . . . . . . . . . . . . . . . . . . . . . 49

B. PROGRAM LISTING FOR SIMULATIONS IN CHAPTER II . . . 50

APPENDIX B . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 56

A. PROGRAM STRUCTURE . . . . . . . . . . . . . . . . . . . . . . 56

B. PROGRAM LISTING FOR SIMULATIONS IN CHAPTER III . . . 57

LIST OF REFERENCES . . . . . . . . . . . . . . . . . . . . . . . . . 65

INITIAL DISTRIBUTION LIST . . . . . . . . . . . . . . . . . . . . . 67

## ACKNOWLEDGEMENTS

I would like to express my thanks to Professor Roberto Cristi for his help and encouragement during this study. His technical expertise and insight were invaluable to me. His patience, maturity, and understanding of human nature helped me to overcome many difficulties.

I would like to offer special thanks to my wife Dilek, and my son Onuralp for their patience, support, encouragement and sacrification. I dedicate this thesis to you.

# I. INTRODUCTION

## A. GENERAL

Stated most simply, the problem of navigation can be summarized by the following three questions: "where am I?", "where am I going?", and "how should I get there?". The first question is one of localization: "how can I work out where I am in a given environment, based on what I can see and what I have previously been told?". The second and the third questions are essentially those of specifying a goal and being able to plan a path that results in achieving this goal. Investigation of the latter two questions usually comes under the domain of path planning and obstacle avoidance [Ref. 1]. In this thesis, we are principally concerned with the first question, localization, since a robust and reliable solution to this problem is essential to answering the remaining two questions.

An Autonomous Underwater Vehicle is a type of Unmanned Underwater Vehicle (UUV), that is not limited by the need for local human control. The freedom from requiring an external control interface theoretically allows this type of vehicle to perform a great range of missions [Ref. 2]. To widen the range of application of an AUV, it is necessary to develop systems with high levels of autonomy and to operate in unstructured environments with little a priori information. To achieve this degree of independence, the AUV must have an understanding of its surroundings, by acquiring and manipulating a rich model of its environment of operation [Ref. 3]. While autonomy has clear advantages, it requires a sophisticated level of on board processing ability.

## B. AIM OF THE STUDY

This thesis is concerned with the navigation problem of an AUV in the horizontal plane. For any navigation system to be useful, the accuracy of the instruments has to be such that the navigator's error is within the required tolerance of the vehicle which relies on the navigation system. However, accurate sensors such as high quality inertial navigation systems and doppler sonars tend to be large and expensive, and unsuitable to a small vehicle. It is the aim of this thesis to study the feasibility of combining measurements from the sonar device on board to generate relatively good position estimates over a short time interval.

## C. METHOD OF APPROACH

This thesis is concerned with the short-range navigation problem for the NPS testbed AUV. Being limited in complexity and cost, the navigator has to rely on the available hardware. Also, because the system needs only to operate over short ranges, the effects of the earth's orbit and rotation can be neglected. This approximation also has the effect of simplifying the model to be built.

The approach taken in this thesis to solve the navigation problem in the horizontal plane is to use a nonlinear dynamic model of the vehicle to filter the sonar range returns for estimation of the vehicle's position. The measurements obtained from sonar and nonlinear model are run through the Kalman filter to calculate the Kalman gain. Then, the Kalman gain is used to make the next position estimation of nonlinear model. This approach is diagramed in Figure 1.1. The filter to be used is an Extended Kalman filter.

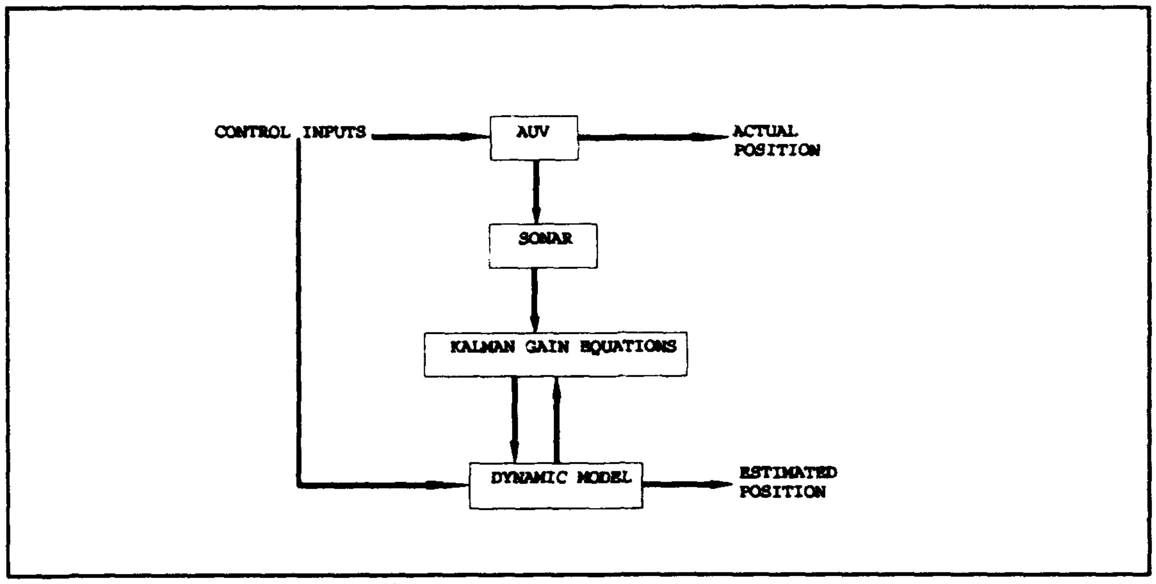

Figure 1.1 Kalman Filtering Concept

The Kalman filter theory, which appeared in the early 1960's, was recognized as an ideal solution to the navigation data processing problems for navigation. These data processing problems can be adapted nicely into the necessary Kalman filter assumptions, which means that the estimates of the desired navigation outputs are, in fact, very nearly optimum. This gives the engineer confidence that the system is the best that can reasonably be expected. Also, the recursive form of the filter is convenient. A new optimum estimate can be made very shortly after each new measurement is obtained. Nor is it necessary to store a great amount of data or invert a large matrix, as might be necessary in more conventional, least-squares fitting techniques [Ref. 4].

Navigation systems are among the most popular areas for the application of Kalman filtering. Most of the major navigation system manufacturers have developed or proposed systems based on Kalman filtering techniques, and they are being used in several vehicles in operational use. Kalman filtering has now become an integral part of almost any navigation system. The reasons for the popularity of Kalman filtering in navigation systems are not hard to find. There are at least three major complementary factors that have come together at the proper time. These factors are an increased need for self contained navigation systems, the mathematical tools to design, and the necessary equipment to implement [Ref. 4].

This thesis is presented in four parts. Chapter II discusses background and the theory behind the navigator design and presents some results for a model designed to operate in a pool without any obstacles in the operation environment. Chapter III discusses the details of potential function technique used in simulations and the autoregressive extended (ARX) model based parameter estimation. Finally, Chapter IV summarizes the results of this research and contains conclusions and recommendations for application and further study.

# II. A GEOMETRICAL MODEL OF ENVIRONMENT

## A. GENERAL

In this chapter, we set up the nonlinear model for the dynamics of an AUV and the model for sonar measurements.Then, we define the experiment in a pool without obstacles and implement Kalman filtering techniques for estimating the position of the vehicle. In the estimation process, we use the sonar range measurements only.

## B. DISCRETE KALMAN FILTER EQUATIONS

The Kalman filter estimates the state of a system given a set of known inputs to the system and a set of measurements [Ref. 5]. The system is assumed to be driven by both a known input and an unknown random input:

$$x(k+1)=\Phi x(k)+\Delta_1 u(k)+\Delta_2 w(k) \ , \qquad (2.1)$$

where u(k) is the known input, w(k) is the random input called the plant driving noise, x(k) is the state vector, $\Phi$ is state transition matrix, $\Delta_1$ is known input matrix, and $\Delta_2$ is random input matrix. We assume that w(k) is white noise.

The measurements of the system are related to the state by

$$y(k)=Hx(k)+v(k) \ , \qquad (2.2)$$

where y(k) is the measurement, H is the measurement matrix, and v(k) is the random measurement noise. We assume that v(k) is white noise.

The solution of the estimation problem can be shown to have the following form:

$$\hat{x}(k+1|k+1)=\hat{x}(k+1|k)+G(k+1)[y(k+1)-\hat{y}(k+1)] \ , \qquad (2.3)$$

where G(k+1) is the Kalman filter gain at the given specific time.

The Kalman filter gain G(k+1) is found by applying the orthogonality principle which leads to the recursive equations:

$$\begin{aligned} &P(k+1|k)=\Phi P(k|k)\Phi^T+\Delta_2 W\Delta_2{}^T \\ &G(k+1)=P(k+1|k)H^T[HP(k+1|k)H^T+V]^{-1} \\ &P(k+1|k+1)=[I-G(k+1)H]P(k+1|k) \ , \end{aligned} \qquad (2.4)$$

where P is the covariance matrix of the estimation error. The following is the summary of the Kalman filter equations in the order in which they are implemented throughout this thesis:

**The Gain Equations:**

$$\begin{aligned} &P(k+1|k)=\Phi P(k|k)\Phi^T+\Delta_2 W\Delta_2{}^T \\ &G(k+1)=P(k+1|k)H^T[HP(k+1|k)H^T+V]^{-1} \\ &P(k+1|k+1)=[I-G(k+1)H]P(k+1|k) \end{aligned} \qquad (2.5)$$

**The Filter Equations:**

$$\begin{aligned} &\hat{x}(k+1|k)=\Phi\hat{x}(k|k)+\Delta_1 u(k) \\ &\hat{y}(k+1|k)=H\hat{x}(k+1|k) \\ &\hat{x}(k+1|k+1)=\hat{x}(k+1|k)+G(k+1)[y(k+1)-\hat{y}(k+1|k)] \end{aligned} \qquad (2.6)$$

This is known as the predictor-corrector form of the Kalman filter in which the next state is predicted using the state space model and the measurement is used to correct the prediction.[Ref. 5]

The **W** matrix is the covariance of the system's noise, while the $\mathbf{\Delta}_2$ matrix describes the way this noise enters the system. If **W** is diagonal, then the disturbances are assumed to be independent, otherwise they are correlated, as described by the off-diagonal terms in **W**. [Ref. 5]

The **V** matrix describes measurement noise from the sensors. The Kalman filter considers the measurement noise to enter all the individual measurements. As with the measurement noise covariance matrix, if **V** is large, then the measurements are e :pected to deviate more from the states being measured, and the Kalman filter will rely more on the predicted state than on the measurements.[Ref. 5]

The initial **P** matrix affects the dependence the Kalman filter has on the initial conditions. Large values in initial **P** mean that the filter will not depend on the initial conditions, but will instead give more weight to the measurements. This allows the estimated state to change rapidly as the filter goes through its transient stage. In steady state, however **P** has no effect on the Kalman filter because it approaches a constant value that is dependent only on the system and the noise covariance matrices. [Ref. 5]

Because the gain equations do not depend directly on time or on the state trajectory in a LTI system, the gain can be calculated a priori and recalled from memory as needed. There is no need for real-time gain computation. Moreover, in the time invariant case the gain matrix approaches a steady-state value, which is determined by the system equations and the W and V matrices through the associated Kalman filter equations. In many cases, using the steady-state gain matrix instead of the time-varying gain matrix gives satisfactory results in a reduced-complexity algorithm which does not take into account prior knowledge of the initial conditions. [Ref. 5]

## C. LINEARIZATION

Thus far, the discussion has been limited to linear systems; however, many systems in which the control engineer is interested are nonlinear. Nonlinear system models are more general than linear models and can contain a wide variety of nonlinear characteristics for which limited analytic tools exist. In general, nonlinear systems do not exhibit the properties of homogeneity or superposition, and many include transcendental, trigonometric or other nonlinear functions [Ref. 6]. Such is the case with the model chosen for this thesis.

A method for working with nonlinear systems is to linearize them using a truncated Taylor series approximation. The Taylor series approximates a function around a given point as an infinite sum of weighted, analytically determined, partial derivatives. A general Taylor series around the point $x_o$ is given by:

$$f(x)=\sum_{n=1}^{\infty}\frac{d^n f(x_o)}{n!dx^n}(x-x_o) \qquad (2.7)$$

A truncated Taylor series only uses a few of the terms of the sum. In order to realize a linear function from the Taylor series expansion of a nonlinear one, the series must be truncated at the first term. This model is general and can be applied to any nonlinear system, and it will be valid in a region surrounding the linearization point [Ref. 7].

For the case of a continuous-time dynamic system represented in state-space form, which, in general, is not a function of a single variable, but is rather a function of time, the input vector, and the past state vector, the Taylor series is defined over the partial derivatives of each of the independent variables. In the case of a time-invariant system, the series is expanded about an operating point described by the state, $x_o$, and a corresponding input, $u_o$. A nonlinear state space equation can then be approximated as

$$\dot{x}=f(x_o,u_o)+\frac{\partial f(x_o,u_o)}{\partial x}(x-x_o)+\frac{\partial f(x_o,u_o)}{\partial u}(u-u_o)\ , \qquad (2.8)$$

where f is the nonlinear system function. This notation implies that the partial derivatives are constant terms, calculated analytically and evaluated at $x_o$, and $u_o$ [Ref. 8].

Just as a nonlinear system equation can be expanded using a Taylor series, and linearized, a nonlinear measurement equation can also be expanded and linearized. The formulation for this expansion is similar to the equation 2.8.

## D. EXTENDED KALMAN FILTER

The Kalman filter is derived for linear systems with linear measurement equations; however, using the linearization techniques described in the last section, this filter can find application to general, nonlinear systems. This is known as the Extended Kalman filter. The Extended Kalman filter is suboptimal and may suffer convergence and stability problems, but it has been shown to be useful in variety of applications. [Ref. 5]

The form of the Extended Kalman filter equations are essentially similar to those of the Linear Kalman filter. The nonlinear model is used to predict the state and the nonlinear measurement is used to correct the prediction. The most significant difference between the Extended Kalman filter and the Linear Kalman filter is the gain equations. Rather than using $\Phi$ (the state transition matrix), $\Delta$ (the input matrix), and H (the measurement matrix) matrices to calculate the gain, the Extended Kalman filter uses the linearized model of the nonlinear system. It uses the partial derivatives of the nonlinear state equations and the nonlinear measurement equations. These partials are evaluated at each estimated state. As such, the gain, G, cannot be computed in advance, because it is dependent on both the state trajectory and the input history.

Calculating partial derivatives of the nonlinear system and measurement equations for each measurement as well as calculating the Kalman filter gain matrix is a computational burden. In order to overcome this problem, it may be possible to use gain matrices calculated in advance and by choosing discrete points in the system's state space about which to linearize the nonlinear system. Practically, in choosing the points, it is convenient that only some of the system states are varied while others are kept constant. Using the chosen points, several linear approximations to the nonlinear system are calculated and then used to calculate the steady-state Kalman filter gain matrix associated with those particular points. Once the gains are calculated, the Extended Kalman filter is implemented by determining which of the chosen linearization points are closest to the current estimated state and the corresponding gain matrix is used in the Extended Kalman filter equation as given as the last of Equations 2.6. [Ref. 5]

$\theta$ In using the Extended Kalman filter, nonlinearities are modeled as plant noise. This means that the $\Delta_2$ matrix is usually the identity matrix. Furthermore, because the nonlinear effects are not included in the gain equations, the Extended Kalman filter can be very sensitive to the **W** and **V** matrices. Choosing improper values can make an Extended Kalman filter unstable. The values chosen for these matrices should not necessarily correspond to the actual noise expected in the system or in the measurements, but rather they must be made large enough to provide a robust prediction in spite of the nonlinear effects.

## E. MODELLING FOR PLANAR MOTION OF THE AUV

The basic component of the estimation process is a state vector whose value at any time t is given by:

$$x = \begin{bmatrix} x_v \\ y_v \\ v \\ \theta \\ \dot{\theta} \end{bmatrix}, \qquad (2.9)$$

where $(x_v, y_v)$ defines the position of the vehicle with speed v, heading $\theta$, yaw rate $\dot{\theta}$. The subscript "v" is used to distinguish the position of vehicle from the state vector x and measurement y.

Input to the system are propeller speed, $u_1$, and vertical rudder command, $u_2$.

$$u=\begin{bmatrix}u_1\\u_2\end{bmatrix}. \tag{2.10}$$

The differential equations forming to our model are as follows:

$$\dot{x}=\begin{bmatrix}v\cos\theta\\v\sin\theta\\-\alpha\ v+\beta\ u_1\\\dot{\theta}\\-\gamma\ \dot{\theta}+\sigma\ u_2\end{bmatrix}, \tag{2.11}$$

where $\alpha$, $\beta$, $\gamma$, and $\sigma$ are constants depending on the dynamics of the vehicle.

As we can see, equations 2.11 are nonlinear and they have to be linearized for further analysis. Now let us apply truncated Taylor series expansion to the first two equations. In the following equations the subscript "o" indicates the current value of the particular state around which we linearize the model,

$$\dot{x}_v=v_o\cos\theta_o+\frac{\partial(\dot{x}_v(v_o,\theta_o))}{\partial v}(v-v_o)+\frac{\partial(\dot{x}_v(v_o,\theta_o))}{\partial\theta}(\theta-\theta_o) \tag{2.12}$$

$$\dot{y}_v=v_o\sin\theta_o+\frac{\partial(\dot{y}_v(v_o,\theta_o))}{\partial v}(v-v_o)+\frac{\partial(\dot{y}_v(v_o,\theta_o))}{\partial\theta}(\theta-\theta_o) \tag{2.13}$$

where $v_o$ and $\theta_o$ are the current speed and heading of AUV respectively. Now if we follow the algebra involved in equations 2.12 and 2.13 we can find the following results:

$$\dot{x}_v = v\cos\theta_o - \theta v_o \sin\theta_o + \theta_o v_o \sin\theta_o \qquad (2.14)$$

$$\dot{y}_v = v\sin\theta_o + \theta v_o \cos\theta_o - \theta_o v_o \cos\theta_o \qquad (2.15)$$

The last two terms in the equations 2.14 and 2.15 are almost identical; they differ in one sign. Now the following approximations can be made:

$$\dot{x}_v \cong v\cos\theta_o \qquad (2.16)$$

$$\dot{y}_v \cong v\sin\theta_o \qquad (2.17)$$

In the model introduced above, sonar range is the only measurement that depends on the current position and orientation of the AUV with respect to the environment. The $\gamma$ parameter is sonar heading which is measured with respect to the x-axis of the vehicle and defines the angle between x-axis and the sonar beam at the measurement time.

$$y = g(x_v, y_v, \gamma) \qquad (2.18)$$

In this study, we address the problem of navigating the vehicle in a geometric environment, like a pool or tank with known dimensions $L_1$ by $L_2$. Figure 2.1 shows the setup for range measurements . The $\eta$ parameters define the particular angle computed between x-axis and corners of the pool, taking the current position of the vehicle as the origin. Range r, sonar heading $\gamma$, and vehicle position $(x_v, y_v)$ are related by the geometry of the environment as

$$r=-\frac{x_v}{\cos\gamma}\quad;\qquad \eta_2\le\gamma<\eta_1 \tag{2.19}$$

$$r=\frac{(L_2-y_v)}{\sin\gamma}\quad;\qquad \eta_3\le\gamma<\eta_2 \tag{2.20}$$

$$r=\frac{(L_1-x_v)}{\cos\gamma}\quad;\qquad \eta_4\le\gamma<\eta_3 \tag{2.21}$$

$$r=-\frac{y_v}{\sin\gamma}\quad;\qquad \eta_1\le\gamma<\eta_4 \tag{2.22}$$

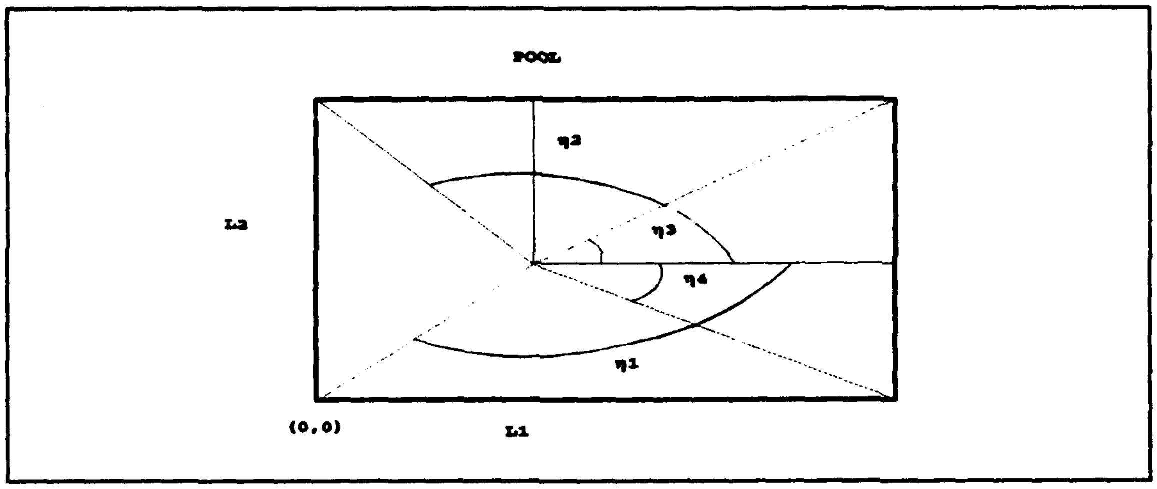

**Figure 2.1 Determination of sonar range in the pool**

If we call H the measurement matrix defined as

$$H=\begin{bmatrix}\frac{dr}{dx} & \frac{dr}{dy} & 0 & 0 & 0\end{bmatrix} \tag{2.23}$$

then we can compute it from the equations 2.19 through 2.22 as

$$H=\begin{bmatrix}-\frac{1}{\cos\gamma} & 0 & 0 & 0 & 0\end{bmatrix} \quad ; \qquad \eta_2 \leq \gamma < \eta_1 \ ; \ \eta_4 \leq \gamma < \eta_3 \qquad (2.24)$$

$$H=\begin{bmatrix}0 & -\frac{1}{\sin\gamma} & 0 & 0 & 0\end{bmatrix} \quad ; \qquad \eta_3 \leq \gamma < \eta_2 \ ; \ \eta_1 \leq \gamma < \eta_4 \qquad (2.25)$$

In the light of the preceding discussion, the state space representation of the particular model can be defined as:

$$\dot{x}=\begin{bmatrix}0 & 0 & \cos\theta_o & 0 & 0\\ 0 & 0 & \sin\theta_o & 0 & 0\\ 0 & 0 & -a & 0 & 0\\ 0 & 0 & 0 & 0 & 1\\ 0 & 0 & 0 & 0 & -c\end{bmatrix}x+\begin{bmatrix}0 & 0\\ 0 & 0\\ b & 0\\ 0 & 0\\ 0 & d\end{bmatrix}u+w \qquad (2.26)$$

$$y=Hx+v \qquad (2.27)$$

## F. SIMULATION RESULTS

Figures 2.2 through 2.10 show the simulated vehicle trajectory, as determined by the nonlinear model. In all the cases, the vehicle was given a constant propeller speed of 140 RPM. It starts from an estimated position when it is at rest initially. After the convergence of the Kalman filter to the simulated trajectory, a rudder command is given for a turn to be executed. In smooth turns, like shown in Figures 2.2 through 2.10, the model is able to track the actual simulated trajectory.

The lower left hand corner of the pool is accepted as the position reference point for all the runs. There are no obstacles present in the pool, which yields to clear sonar returns from the borders of the pool.

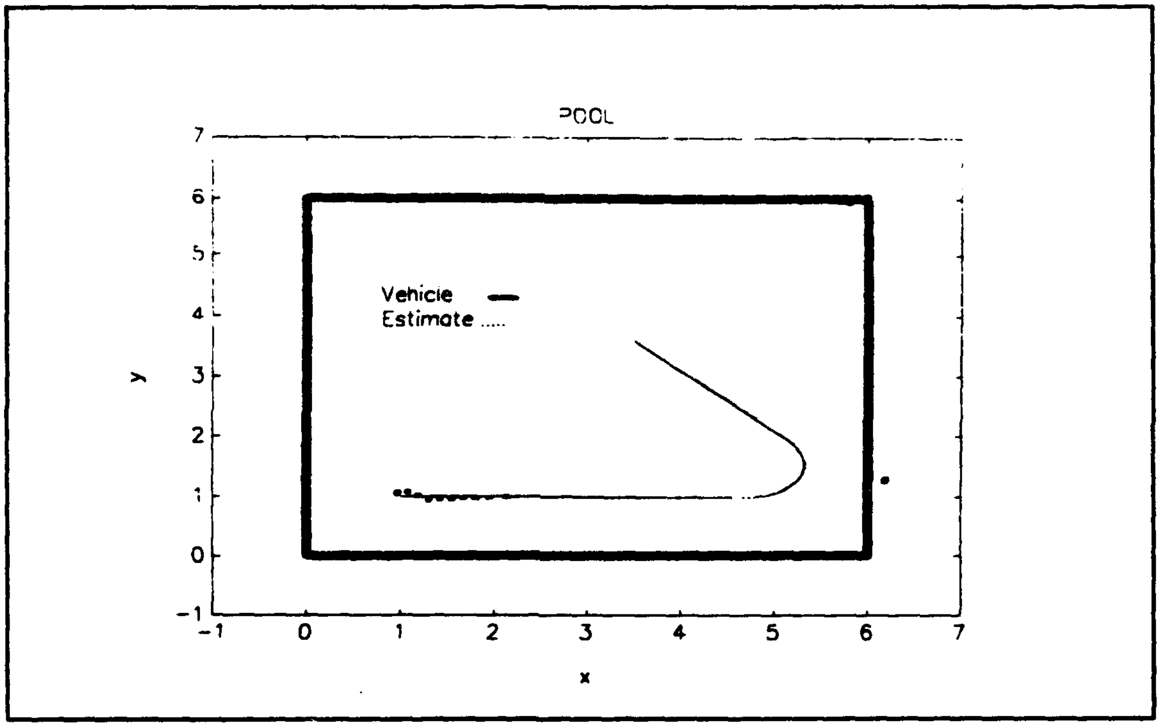

Figure 2.2 A sample run with initial heading 0°

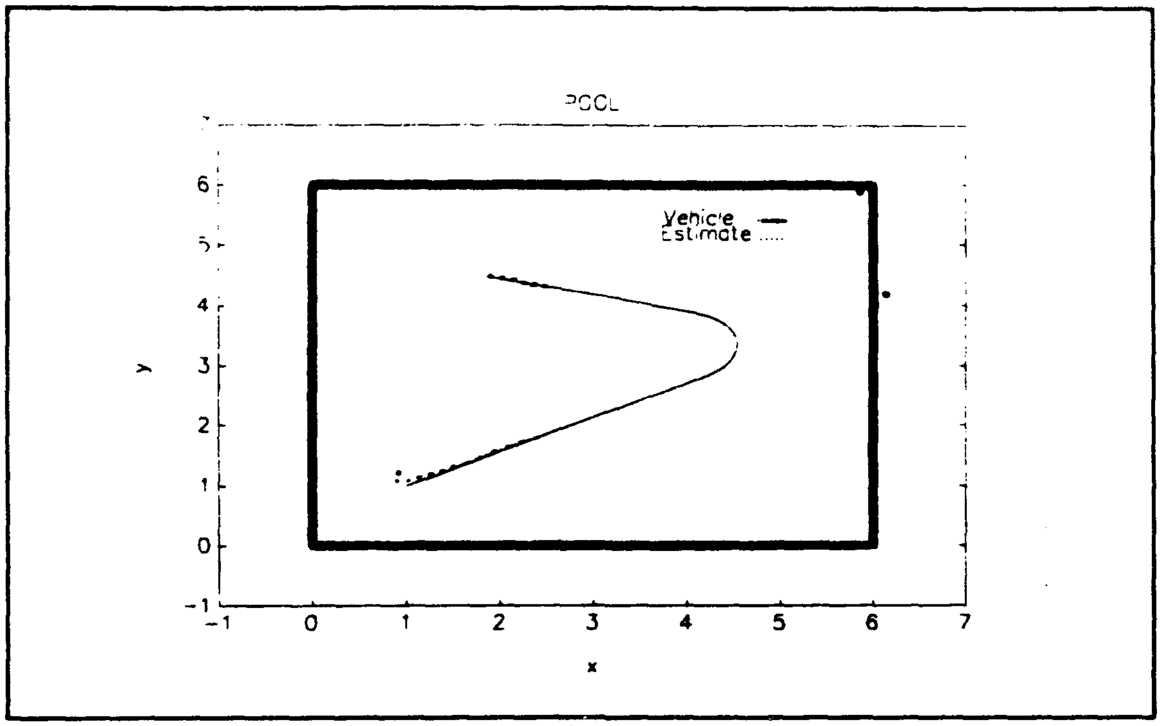

**Figure 2.3 A sample run with initial heading 30°**

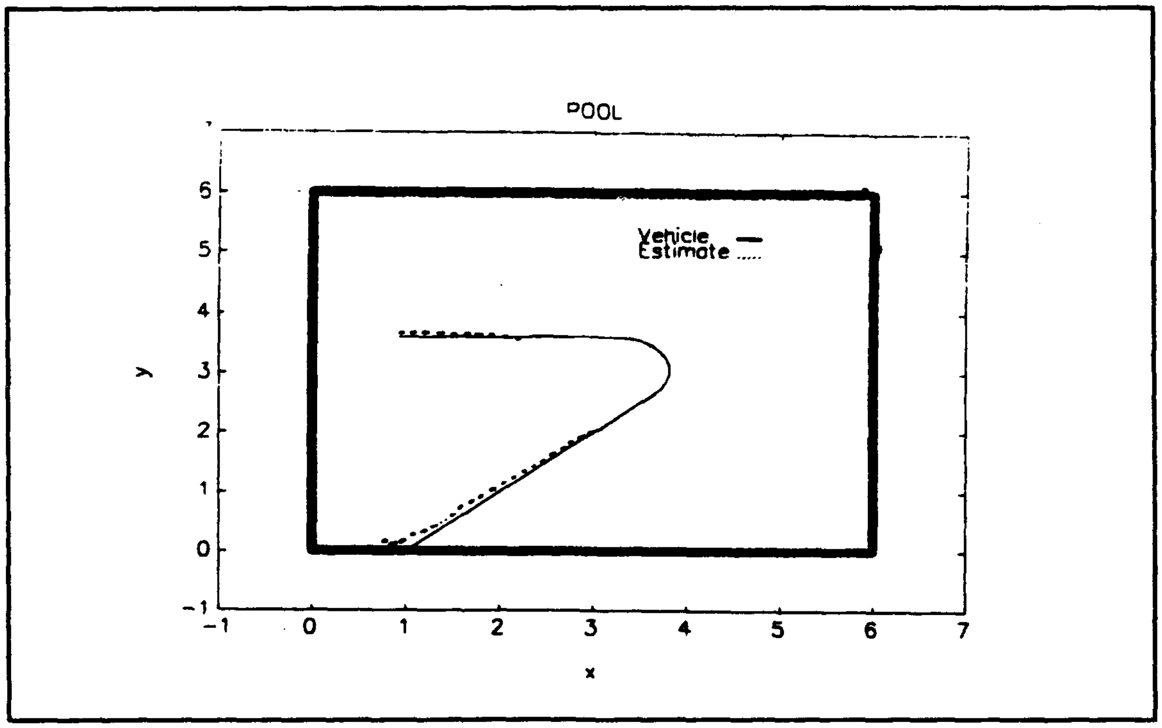

**Figure 2.4 A sample run with initial heading 45°**

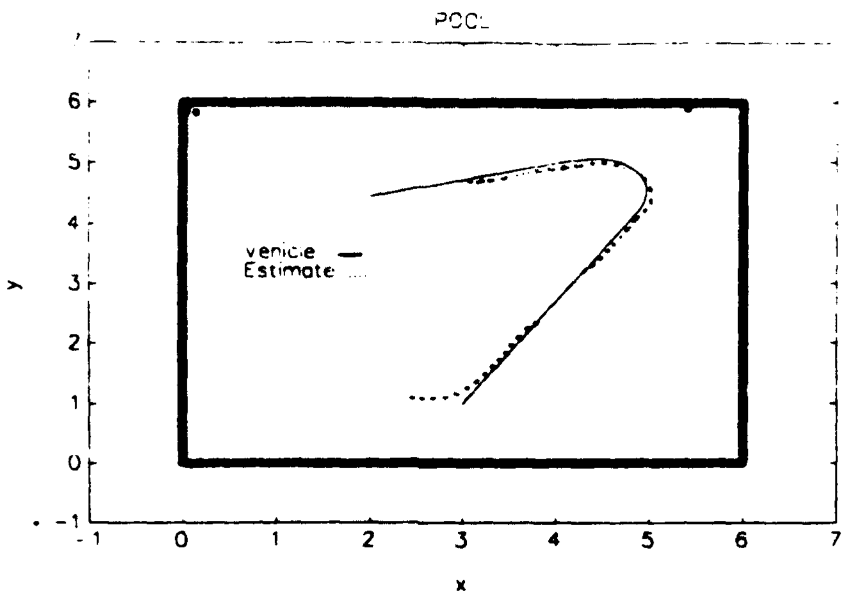

**Figure 2.5 A sample run with initial heading 60°**

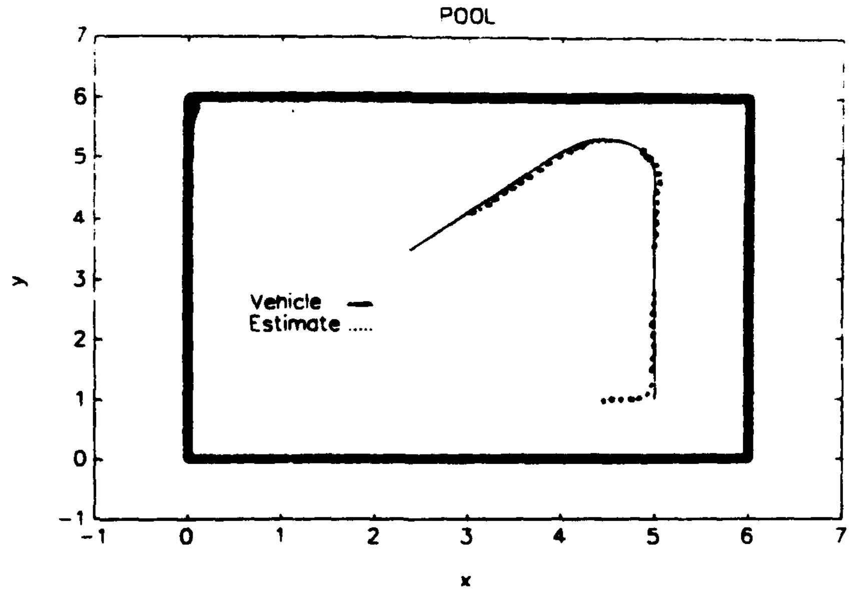

**Figure 2.6 A sample run with initial heading 90°**

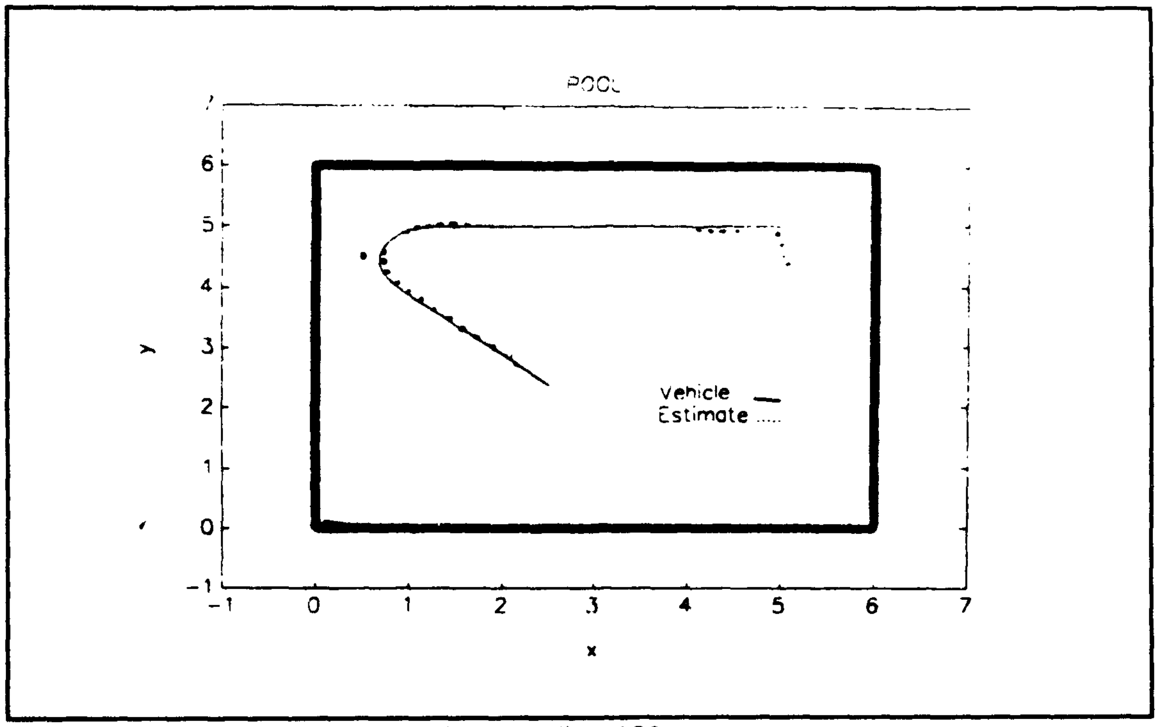

**Figure 2.7 A sample run with initial heading 180°**

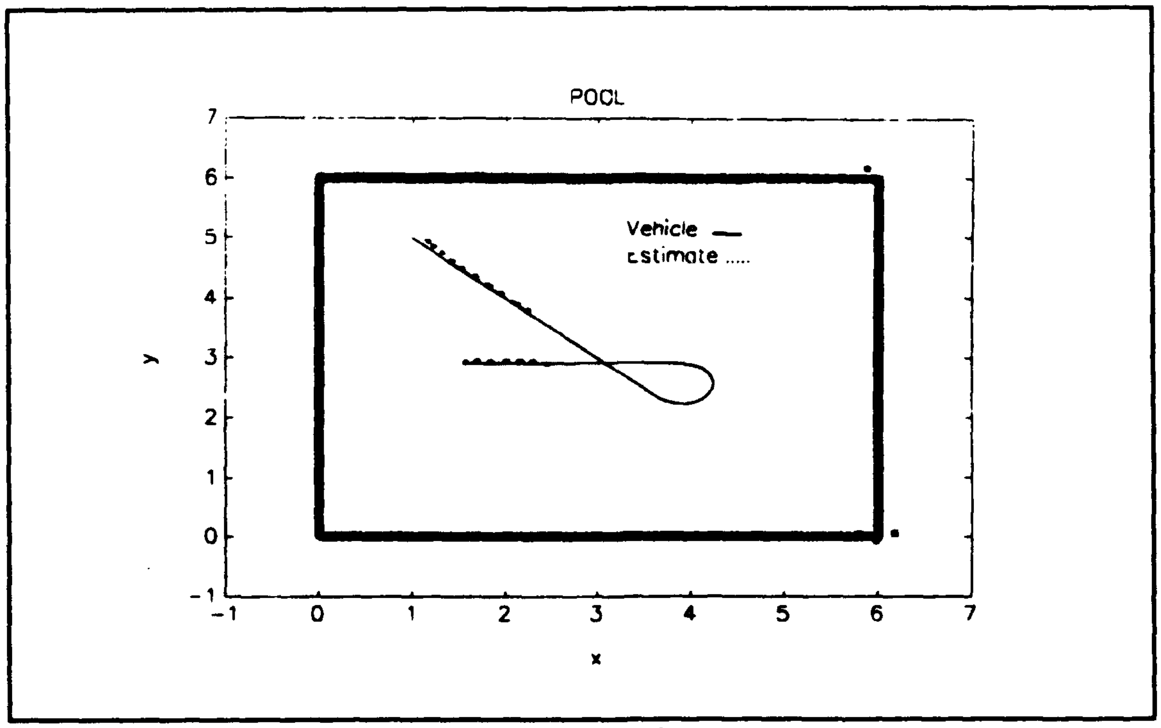

**Figure 2.8 A sample run with initial heading 315°**

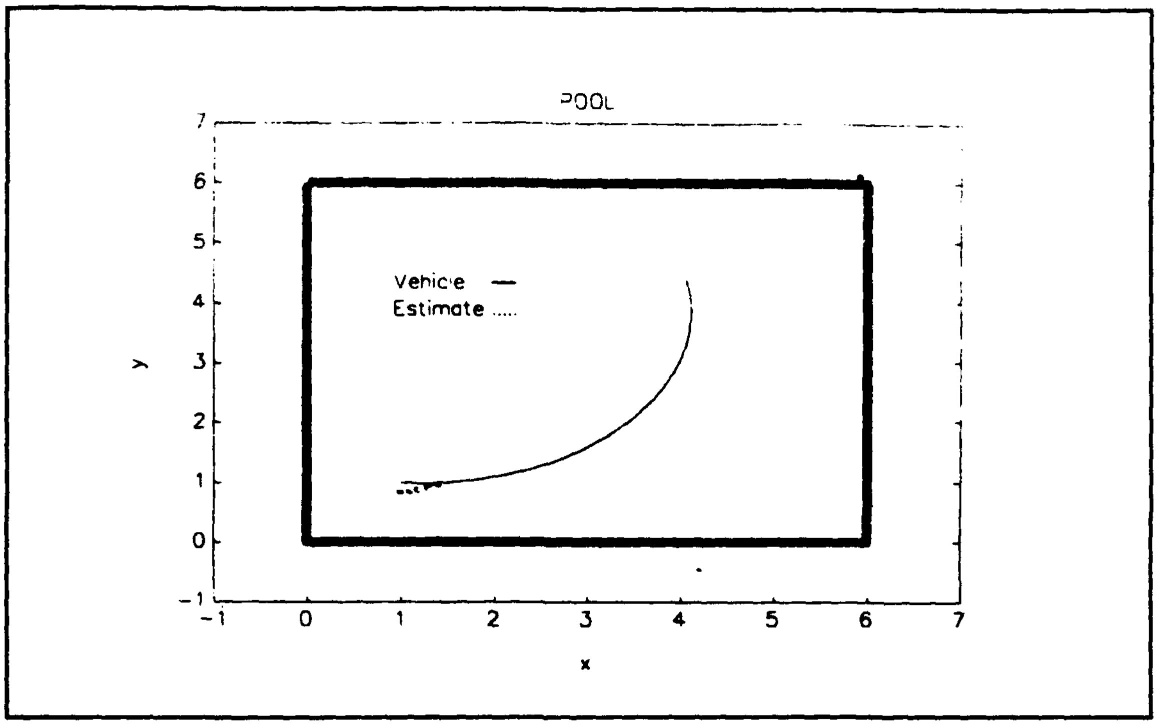

Figure 2.9 A sample run with a steady turn

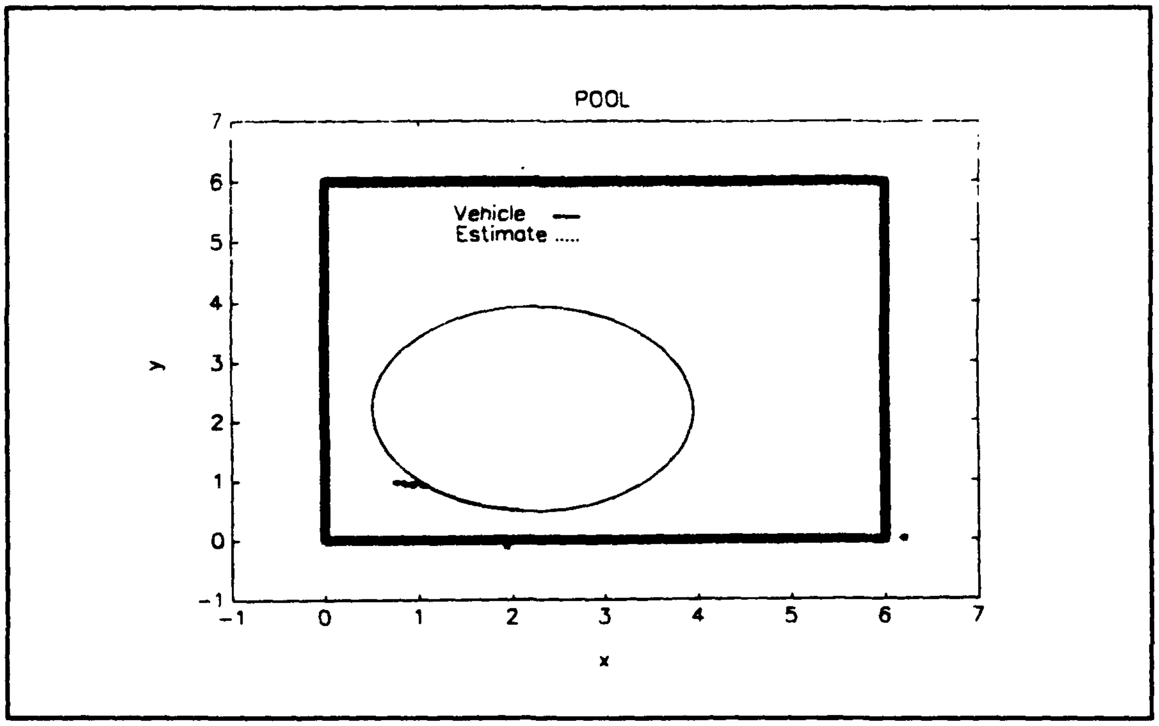

Figure 2.10 A sample run with a complete steady turn

# III. POTENTIAL FUNCTION APPROACH FOR ENVIRONMENT

## A. GENERAL

In this chapter, we will approach the problem of navigation with the same model used in Chapter II by using a potential function, which is intended to define the environment.

## B. MODELLING ASSUMPTIONS

The basic component of the estimation process is based on the state vector which is defined in equation 2.15. This leads to a dynamic model of the form

$$\dot{z}_v = f(z_v, u, w) \tag{3.1}$$

with $z_v \in \mathbb{R}^5$ the state of the vehicle, $u \in \mathbb{R}^2$ the vector of external commands (RPM and RUDDER ANGLE) and $w_v$ disturbances, modelling errors between model and the actual physical dynamics. [Ref. 9]

The measurement vector consists of dynamic observations of sonar range $\rho$ at an angle $\alpha$ with respect to the longitudial axis of the vehicle with v indicating measurement noise

$$y = g(z_v, \rho, \alpha, v) \; . \tag{3.2}$$

We assume the noise sources w and v to be of zero mean, white, and Gaussian, as is standard in this class of problems. The function g is defined by the map of the environment, and apart from measurement noise, it is zero when the sonar return information $(\rho, \alpha)$ is consistent with the vehicle position with respect to the map. The criterion we use to choose g is critical for this thesis, and it is based on the use of a potential function properly defined. For a vehicle moving on a plane and located by position (x,y) and heading $\theta$, the function g is defined as

$$g(x,y,\theta,\rho,\alpha)=V(x+\rho\cos(\theta+\alpha),y+\rho\sin(\theta+\alpha)) \tag{3.3}$$

The vector

$$[x_o,y_o]=[x+\rho\cos(\theta+\alpha),y+\rho\sin(\theta+\alpha)] \tag{3.4}$$

denotes the location of the tip of the sonar range vector in the given reference frame. With this definition, function V has to satisfy the following necessary conditions.

- V is continuous and differentiable
- $V(x_o,y_o)=0$ if the point $(x_o,y_o)$ is on a reflecting surface defined on the map
- V is uniformly bounded over $\mathbb{R}^2$

Considering a pool of rectangular shape with sides $L_1$ by $L_2$ units, and by taking the lower left corner as the origin of our map and two adjacent sides as axis, we can write the potential function V as

$$V(x,y)=F_\lambda(x(x-L_1)y(y-L_2)) \tag{3.5}$$

We choose $F_\lambda$ in equation 3.5 in order to provide boundedness for V. In particular, the choice of a "squashing" function such as the sigmoid

$$F_\lambda(z) = \frac{1 - e^{(-z/\lambda)}}{1 + e^{(-z/\lambda)}} \tag{3.6}$$

for some positive λ, will make the potential V satisfy the conditions stated above.

The parameter λ in the squashing function has to be chosen as a compromise between two conflicting necessities:

- large enough to provide a sufficient region of attraction to correct for errors in estimates
- small enough so that objects which are in the field of operation and not on the map are outside the region of attraction

In the application shown below we use a time varying λ decreasing with time in order to accommodate larger errors at the beginning of the estimation processes. The estimate of the state z is just a standard Extended Kalman filtering operation, which is basically defined in Chapter II.

## C. SIMULATION RESULTS

The navigation algorithm has been tested on a rectangular pool with sides $L_1$ by $L_2$. Figure 3.1 shows typical contours of potential function, while Figure 3.2 is the three dimensional plot of the potential function over the pool.

The sonar returns are recorded from several scans spanning the whole 360° circle at intervals of 0.9°, thus resulting in 400 sonar returns per scan. The source codes used during the simulations are listed in Appendix-B. The programs SCAN5.M, SIMUL5.M, VP.M, SIG.M, and DSIG.M are main files listed. The numbers used in the names of codes indicates the version number of a main change in program structure.

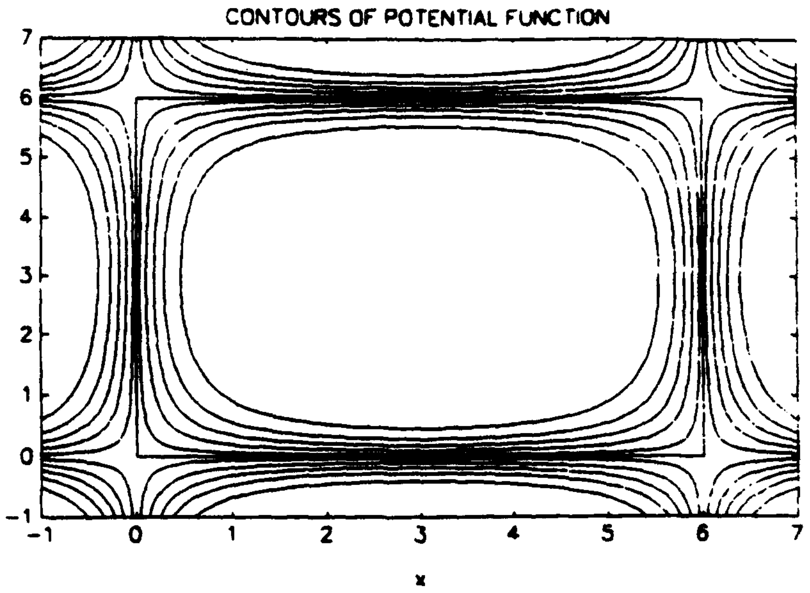

**Figure 3.1 The contours of the potential function**

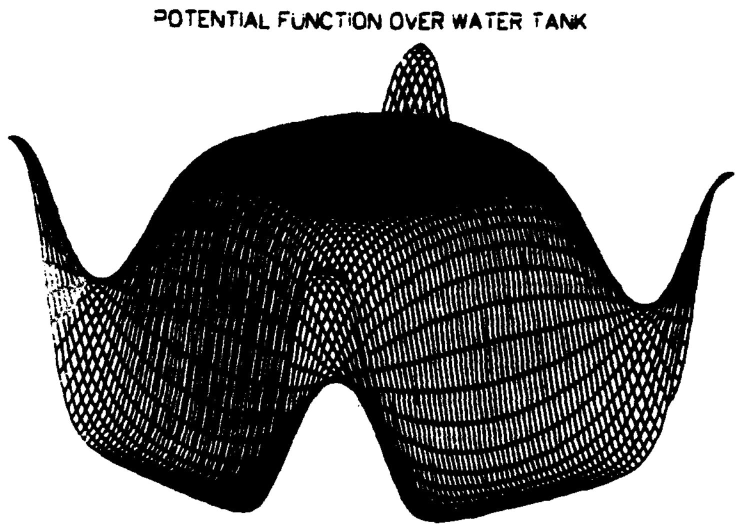

**Figure 3.2 Three dimensional potential function**

The forward velocity estimate, shown in Figure 3.3, approaches a constant value. The estimation error seen in the beginning of this run is induced by the random disturbances added to the initial state.The yaw rate estimate, shown in Figure 3.4, and the yaw estimate, shown in Figure 3.5, exhibits fairly good estimates. The initial errors are due to the initial estimates of the system states and added random disturbances. The yaw estimate approaches its actual value after 30 seconds of simulation time,which shows us the sensivitivity of the system to the heading estimates.

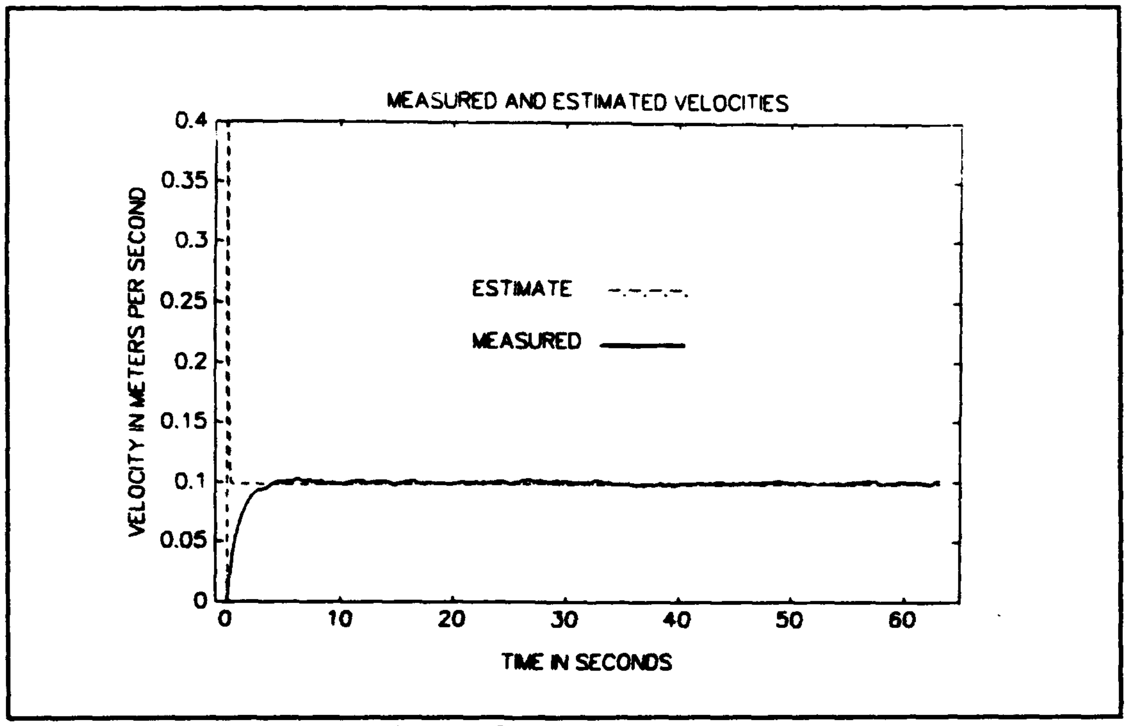

**Figure 3.3 Measured and estimated forward velocities**

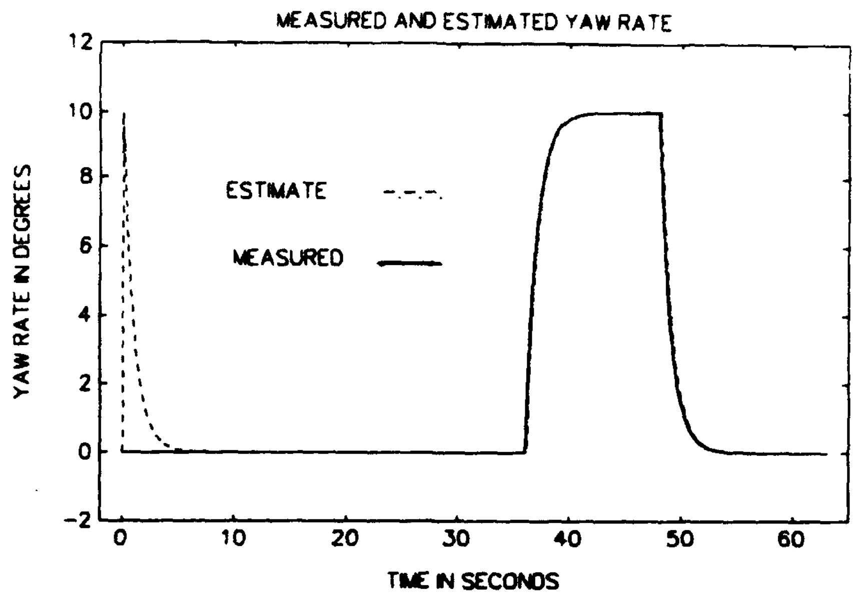

**Figure 3.4 Measured and estimated yaw rate**

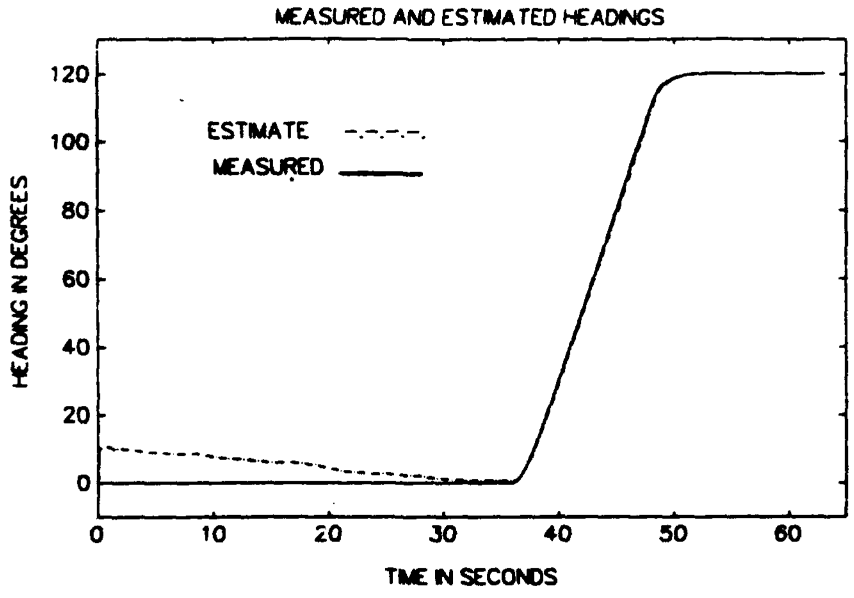

**Figure 3.5 Measured and estimated yaw angle of the AUV**

The results of the position estimation are shown in Figures 3.6 through 3.9, where we used the data collected for seven successive $360^\circ$ scans for each case. In Figures 3.6 and 3.7, we assume the initial position and orientation of the AUV to be known. In these figures the sequence of location of the estimated reflective surfaces is shown for each scan, superimposed to the contour of the potential function. The estimated trajectory is also shown, which coincides with the actual induced motion of the sonar head. The estimates of the reflection from charted objects can easily be distinguished from the charted regions, since the value of the potential function is close to one for the uncharted and zero for the charted. In Figures 3.8 and 3.9 we repeat the same simulation with a significant error in the estimate of the initial location of the sonar head. When the original location is unknown, we start with a large enough value of the parameter $\lambda$ and make it decrease exponentially. In Figure 3.9, we notice that until the convergence of the Kalman filter we might get some estimates that are strictly outside the map defined by the potential function. Then these initial estimates eventually converge to the original trajectory of the vehicle.

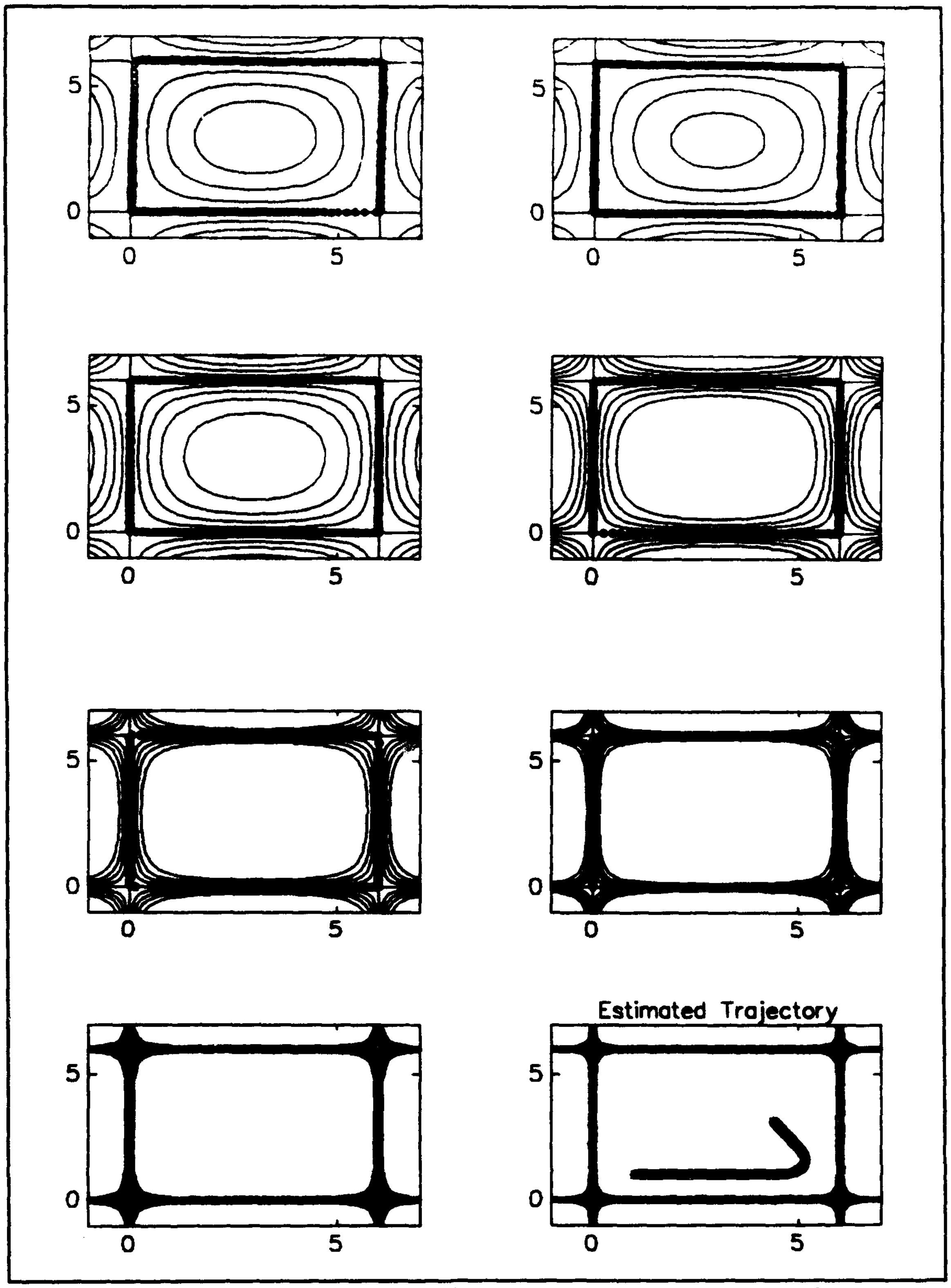

**Figure 3.6 Estimation with known initial conditions and zero degrees heading as a start**

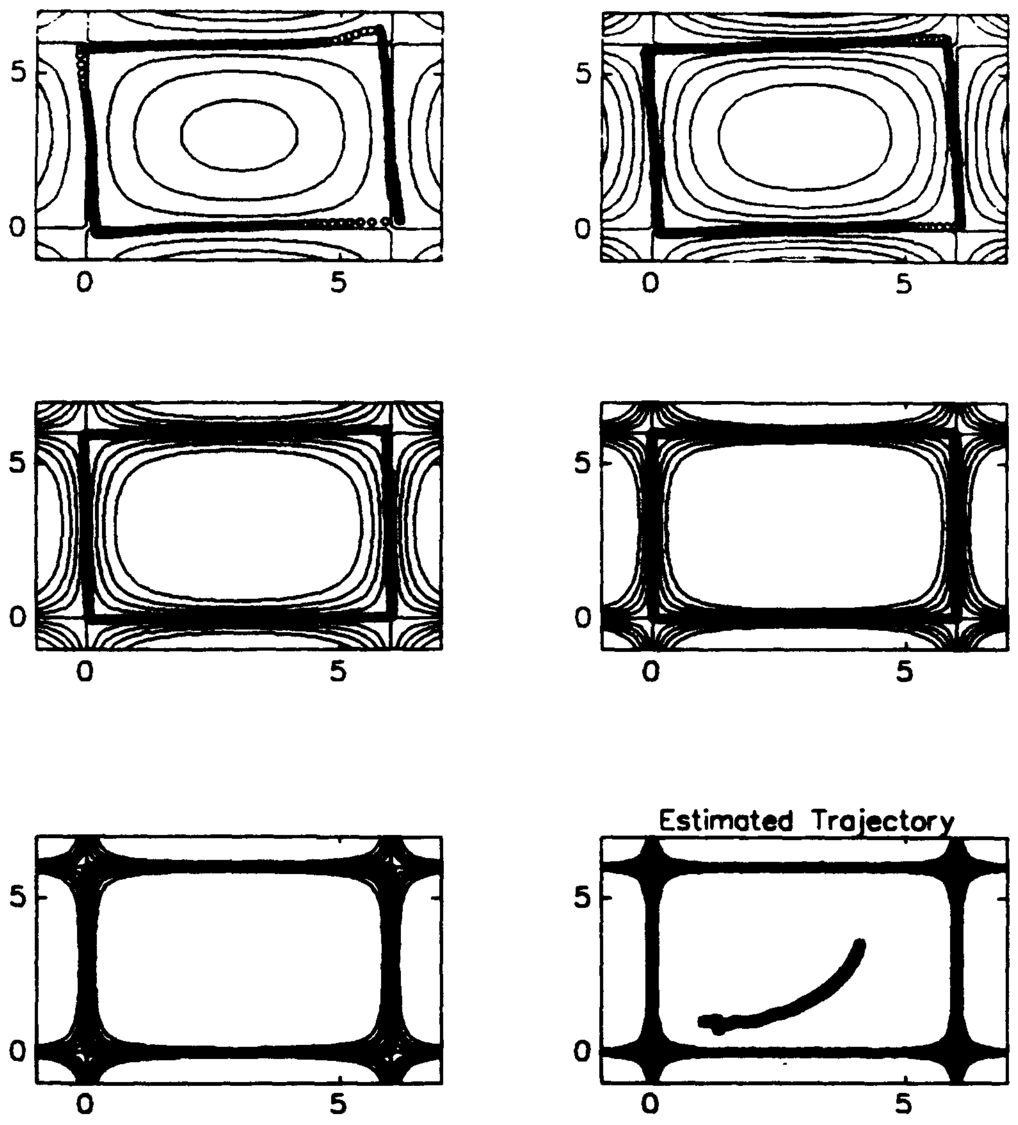

**Figure 3.7 Estimation with known initial conditions and a steady turn to portside**

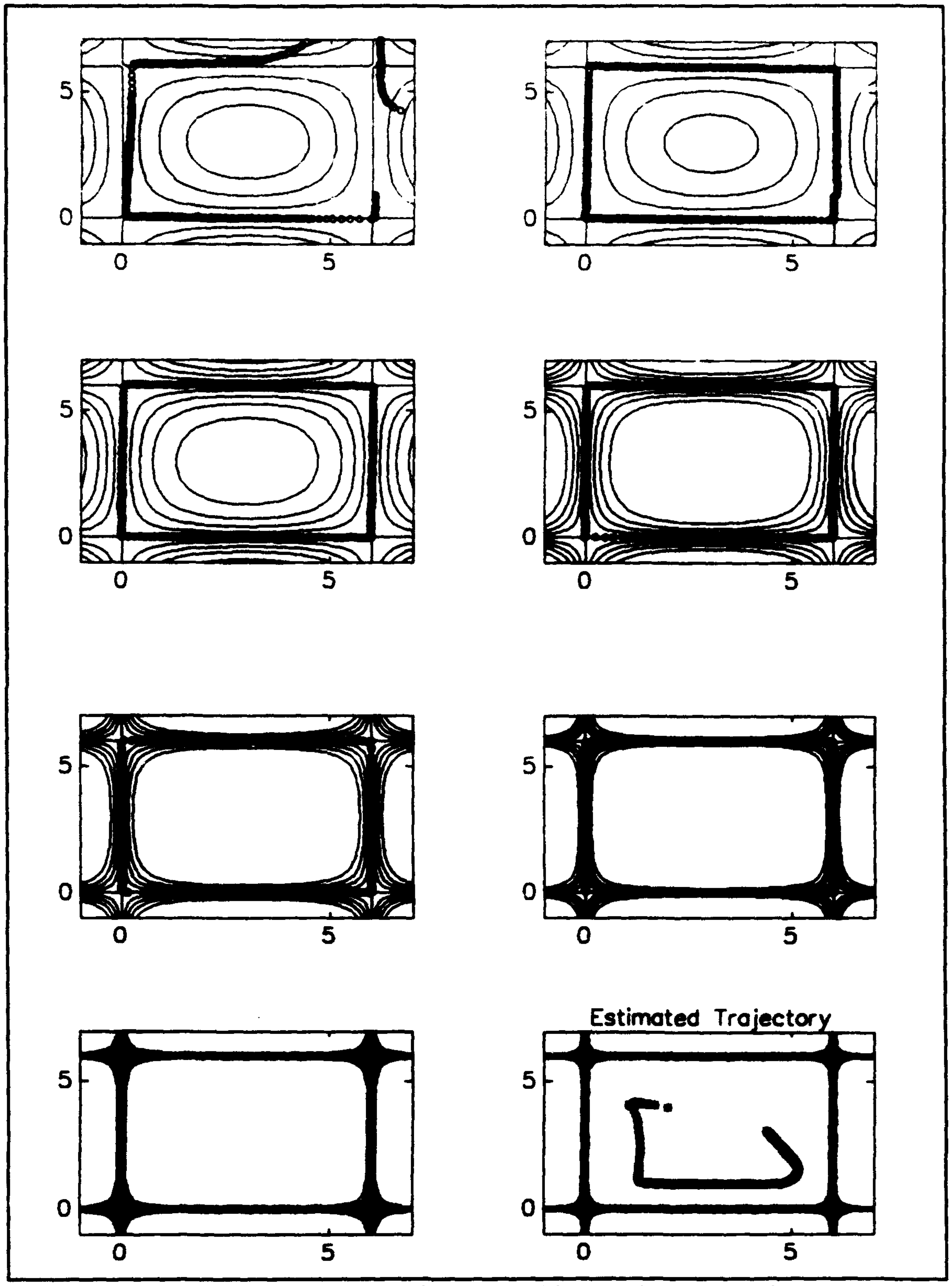

**Figure 3.8** Estimation with unknown initial conditions and steady motion

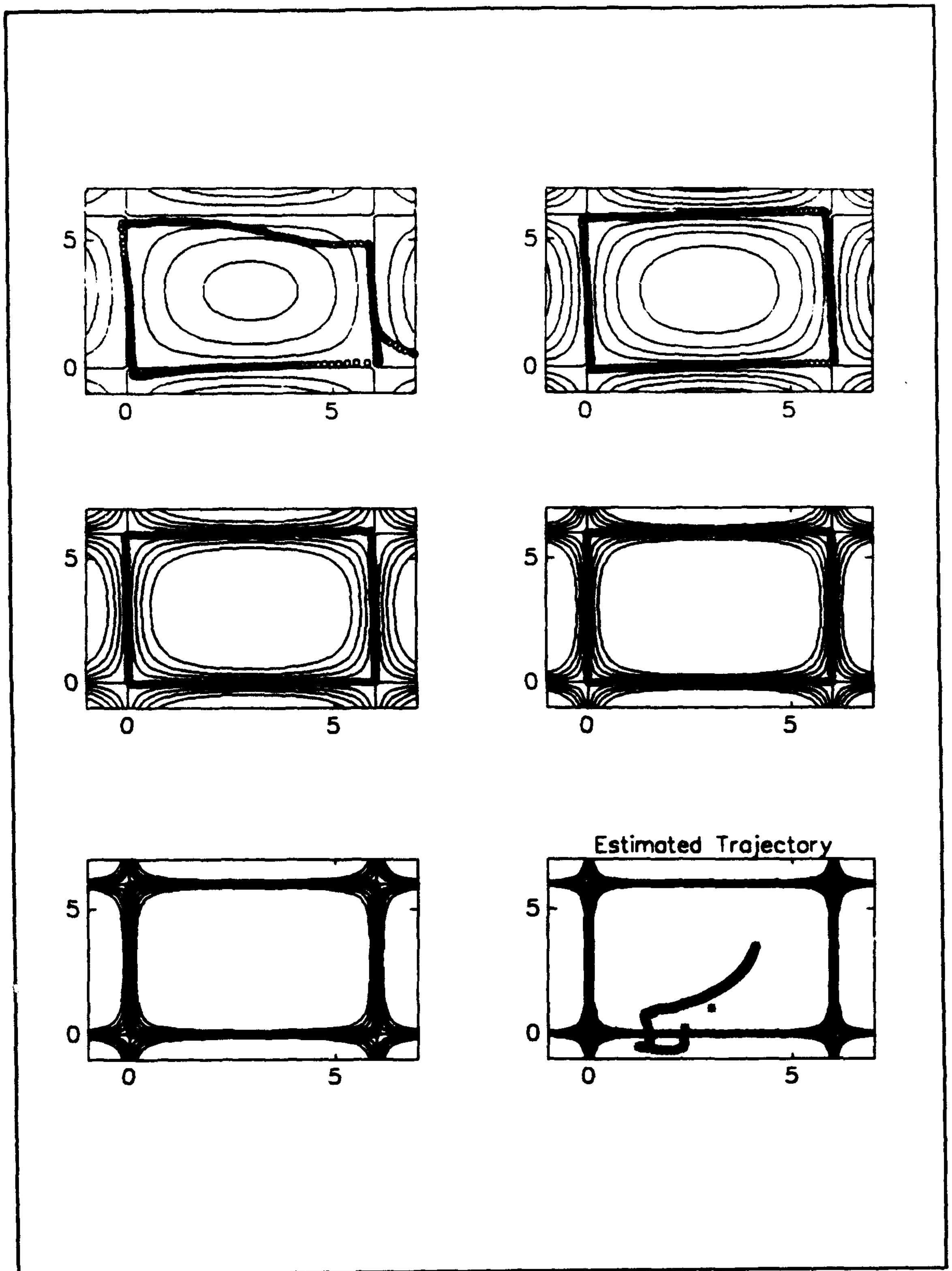

**Figure 3.9 Estimation with unknown initial conditions and a steady turn**

## D. ESTIMATION OF SYSTEM PARAMETERS

The estimation of the system dynamics is the subject of system identification. Identification has many aspects and phases, and it is customary to organize identification by considering a certain number of steps each time we encounter an identification problem. It is often natural to restrict the complexity of modeling to a certain model structure. The class of models thus adopted often belong to some standard category of models such as linear systems or ARMAX model associated with certain mathematical properties. Another standard approach often used is to make assumptions as to the physical nature of the system or other restrictions that define physical parameterization.

### 1. Model Structures

The approaches to modelling of linear systems and linear regression models yields to the following equation, in the regression form as

$$y_k=\phi^T{}_k+v_k \tag{3.7}$$

Linear regression identification consists in reformulating various estimation and prediction problems, in the form of equation 3.7, which for suitable definitions of the observations $y_k$, regressors $\phi_k$, and disturbances $v_k$ also applies directly to the model

$$A(z^{-1})y_k=B(z^{-1})u_k+v_k\ , \tag{3.8}$$

where the discrete time series $y_k$ and $u_k$ provide data. The term time series here means a sequence of data ordered in time [Ref. 10]. The corresponding identification problem consists in determining the model structure and parametric estimation of the polynomials involved. In many cases, such parameters can be identified by using a linear regression approach. The least-squares solution to the linear regression problems exhibits excellent properties in cases where the disturbances at different sampling times are uncorrelated. The systematic errors in cases with more complicated spectral disturbance characteristics constitute a significant problem. Conversely, the presence of correlated disturbances of composition other than white noise renders bias reduction necessary. Efforts to solve such problems have given rise to several extensions of linear regression models. In particular, Recursive Least-Squares (RLS) Method applied to the estimation of Autoregressive Moving Average Extended (ARMAX) models is of central importance in this thesis.

### 2. ARMAX Models and Difference Equations

The ARMAX models constitute an important class of difference equations on the form

$$A(z^{-1})y(t)=B(z^{-1})u(t)+C(z^{-1})e(t) \ , \qquad (3.9)$$

where $z^{-1}$ is a time delay and A,B,C are polynomials

$$\begin{aligned} A(z^{-1})&=1+a_1z^{-1}+...+a_nz^{-1} \\ B(z^{-1})&=b_o+b_1z^{-1}+...+b_nz^{-1} \\ C(z^{-1})&=1+c_1z^{-1}+...+c_nz^{-1} \end{aligned} \qquad (3.10)$$

The ARMAX model, in equation 3.9, is completely defined by the polynomials A, B, and C. So we can group all unknown coefficients in an array

$$\underline{\theta}=[a_1,...,a_n,b_1,...,b_n,c_1,...,c_n]^T \ , \qquad (3.11)$$

which completely defines the model.

The special case of ARMAX model that admits a reformulation to the linear regression model is the controlled autoregressive extended model (ARX)

$$A(z^{-1})y_k = B(z^{-1})u_k + w_k \, , \tag{3.12}$$

where $w_k$ is white noise. In this thesis, we deal with first order ARX models.

### 3. Recursive Estimation

Real-time application of identification algorithms is interesting for various purposes such as supervision, tracking of time-varying parameters for adaptive control, filtering, prediction, signal processing, detection, diagnosis and artificial neural networks [Ref. 9]. However, most identification methods based on a set of measurements are not suitable for real-time application. It is, therefore, desirable to make a suitable reformulation of the algorithms in order to provide efficient procedures.

There are several attractive features of recursive estimation. It is obviously suitable for real-time applications, and only a few data points need to be stored. It is thus a method which is attractive also as a computational method of off-line algorithms. In particular, it provides a method for identification of systems with time-varying parameters.

There are also certain drawbacks such as the fact that the model structure is determined a priori and the fact that iterative solutions based on large data sets may be difficult to organize. Thus, it is of some interest to consider the desirable modifications for real-time application of algorithms originally stated as off-line methods. The following study shows one such simple derivation.

Consider an ARX model of the form

$$A(z^{-1})y(t)=B(z^{-1})u(t)+e(t) \tag{3.13}$$

with u(t), y(t) input,output sequences, and e(t) zero mean white noise. Also, A and B are polynomials in the delay operator $z^{-1}$ as

$$\begin{aligned} A(z^{-1}) &= 1+a_1z^{-1}+\ldots+a_nz^{-1} \\ B(z^{-1}) &= b_1z^{-1}+\ldots+b_nz^{-1} \end{aligned} \tag{3.14}$$

The problem is to estimate the parameters $a_i$ and $b_j$ from the data {u} and {y} in a recursive fashion. In order to do so, we put the model, in equation 3.13, in Regression form as

$$\begin{aligned} y(t) &= \underline{\Phi}^T(t)\underline{\theta}+e(t) \\ \underline{\Phi}^T(t) &= [-y(t-1),\ldots,-y(t-n),u(t-1),\ldots,u(t-n)] \\ \underline{\theta} &= [a_1,\ldots,a_n,\ldots,b_n]^T \end{aligned} \tag{3.15}$$

In the light of equation 3.15 we can apply Kalman filtering techniques to estimate $\underline{\theta}$. If we assume, $\underline{\theta}$ is a constant vector, we can write the following state space equations

$$\begin{aligned} \underline{\theta}(t+1) &= \underline{\theta}(t) \\ y(t) &= \underline{\Phi}^T(t)\underline{\theta}(t)+e(t) \end{aligned} \tag{3.16}$$

Notice that equation 3.26 is a standard state space equation, where e(t) is white noise, and we can easily apply Kalman filtering for the estimation of $\underline{\theta}$(t) as

$$\hat{\underline{\theta}}(t+1)=\hat{\underline{\theta}}(t)+K(t)[y(t)-\underline{\Phi}^T\hat{\underline{\theta}}(t)]$$

$$K(t)=\frac{\overline{P}(t)\underline{\Phi}(t)}{\lambda^2+\underline{\Phi}^T(t)\overline{P}(t)\underline{\Phi}(t)} \tag{3.17}$$

$$\overline{P}(t+1)=\overline{P}(t)-\frac{\overline{P}(t)\underline{\Phi}(t)\underline{\Phi}^T(t)\overline{P}(t)}{\lambda^2+\underline{\Phi}^T(t)\overline{P}(t)\underline{\Phi}(t)}$$

with $\lambda^2 = E(e(t)^2]$. It seems that in equation 3.17 we need to know $\lambda^2$; the covariance of the error e(t), which is usually not known. However, in practice we do not need $\lambda^2$. Just define

$$P(t)=\frac{\overline{P}(t)}{\lambda^2} \tag{3.18}$$

The rest is an easy exercise to show that the recursive least squares estimation method can be written as

$$\hat{\underline{\theta}}(t+1)=\hat{\underline{\theta}}(t)+K(t)[y(t)-\underline{\Phi}^T\hat{\underline{\theta}}(t)]$$

$$K(t)=\frac{P(t)\underline{\Phi}(t)}{1+\underline{\Phi}^T(t)P(t)\underline{\Phi}(t)} \tag{3.19}$$

$$P(t+1)=P(t)-\frac{P(t)\underline{\Phi}(t)\underline{\Phi}^T(t)P(t)}{1+\underline{\Phi}^T(t)P(t)\underline{\Phi}(t)}$$

Therefore, P(t) is the error covariance matrix,and it is initialized as $P(0)=p_o I$, with I identity matrix, $p_o$ a large positive value.

## 4. Simulation Results

So far, we assumed the system dynamics introduced in the following differential equations are known.

$$\begin{aligned}\dot{v} &= -\alpha v + \beta u_1 \\ \ddot{\theta} &= -\gamma\dot{\theta} + \sigma u_2\end{aligned} \tag{3.20}$$

At this point, we try to estimate the parameters $\alpha$, $\beta$, $\gamma$, and $\sigma$ by means of RLS, explained above. During the estimation process, we try to fit our model to a first order ARX model by using the sampled values of V, $u_1$, $\theta$, and $u_2$. After off-line estimation of discrete paranieters (see Fig. 3.10), we transform discrete time parameters to continuous time parameters by Laplace Transforms and obtain our $\alpha$, $\beta$, $\gamma$, and $\sigma$ parameters, which will be used in further simulations. The source code used to do the simulations described above are written in MATLAB and listed in Appendix-B.

The results of the estimation for different initial conditions are shown in Figures 3.11 through 3.18 , where we used the data collected for seven successive 360° scans. In all the figures, the sequence of the estimated reflected surfaces is shown for each of the scans superimposed to the contours of the potential function. In Figures 3.11 through 3.14, we assume the initial position and orientation of the sonar head to be known, but the velocity and direction have uncertainties. In Figures 3.15 through 3.18, we repeat the same simulations with a significant error in the initial estimate of location. In Figures 3.13 and 3.17, we introduce one slow moving object on portside of our vehicle, and two slow moving objects in Figures 3.14 and 3.18 on both sides of the vehicle. As a result of these simulations, we conclude that our model is able to recover by using potential function approach even when reasonable numbers of obstacles, which are not on the map, are present in the operation area.

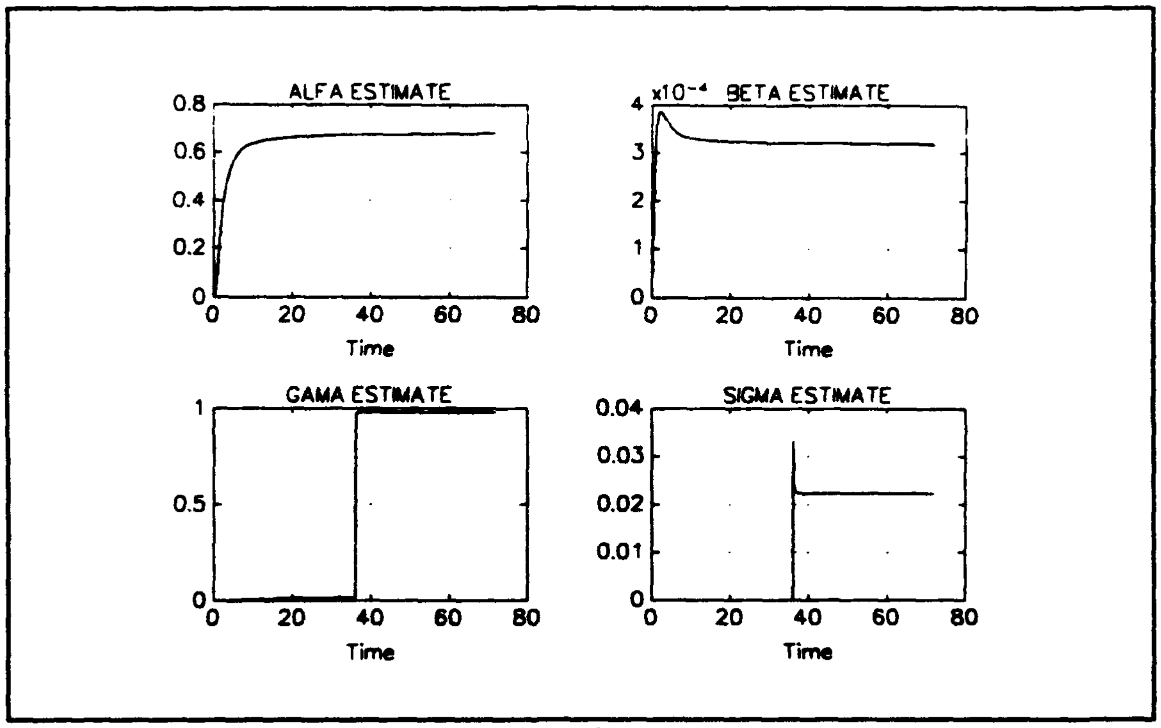

**Figure 3.10 Off-Line Estimated Discrete Time Parameters**

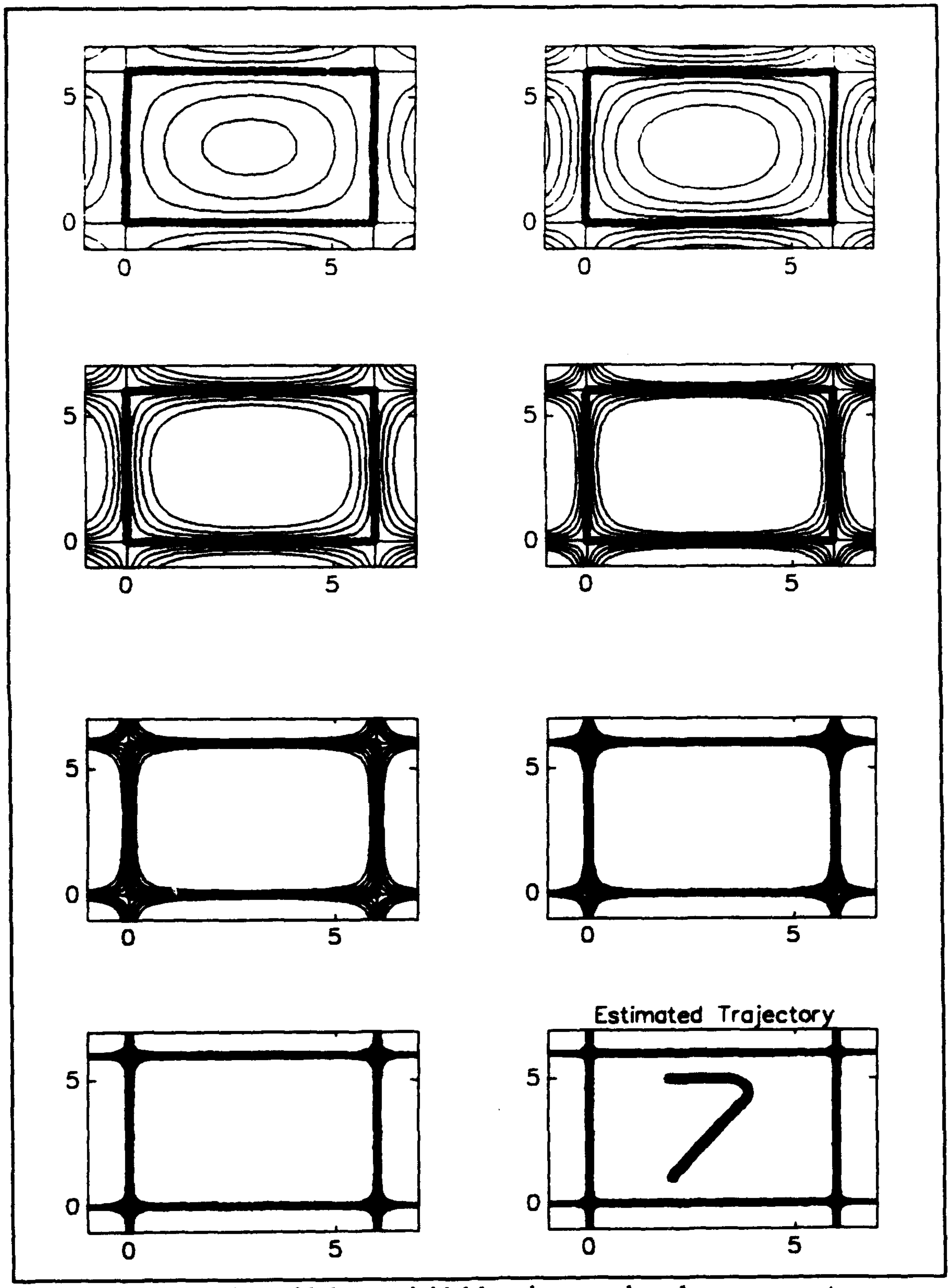

**Figure 3.11 Estimation with known initial location, no obstacles are present.**

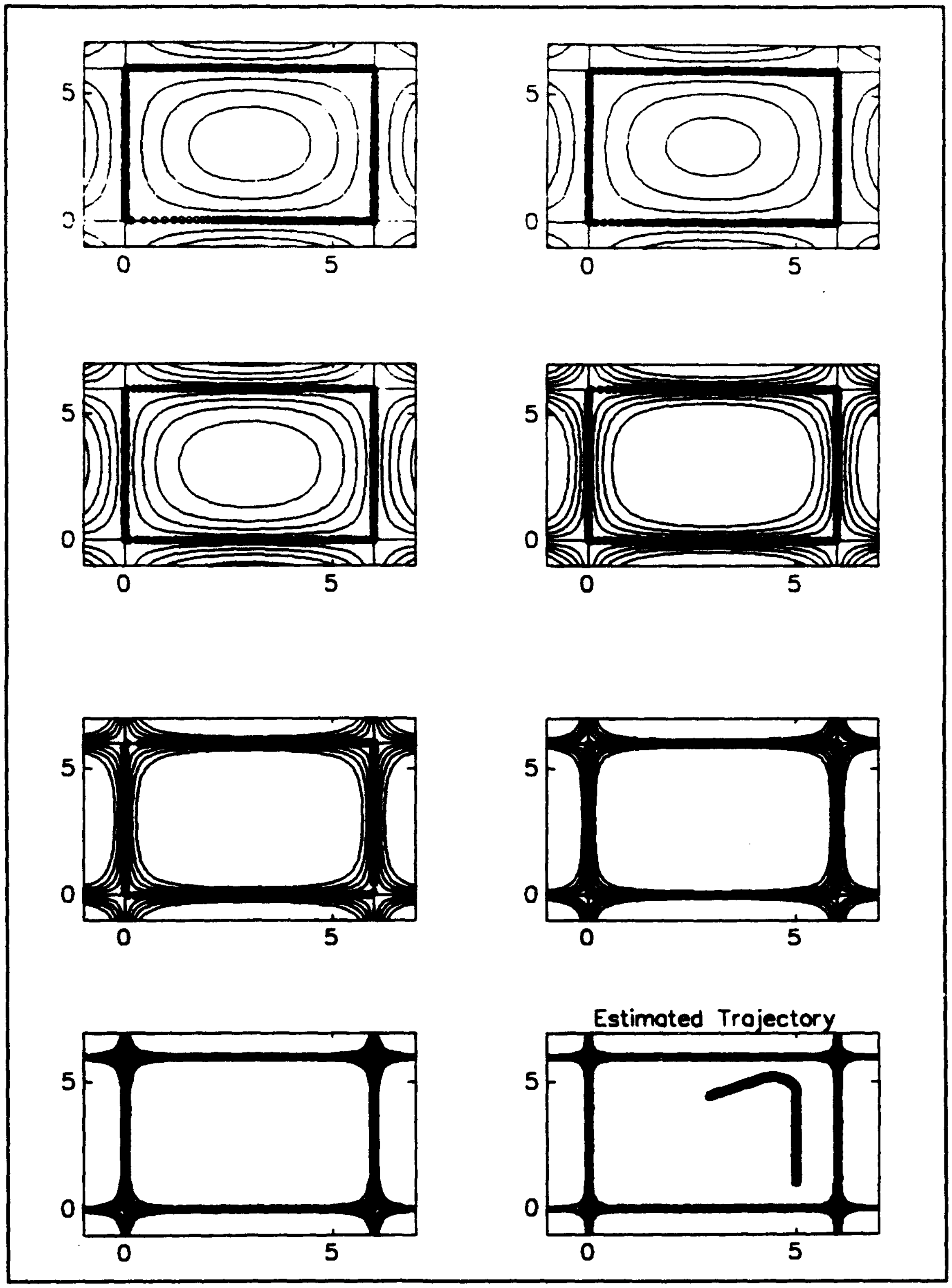

**Figure 3.12 Estimation with known initial location, no obstacles are present.**

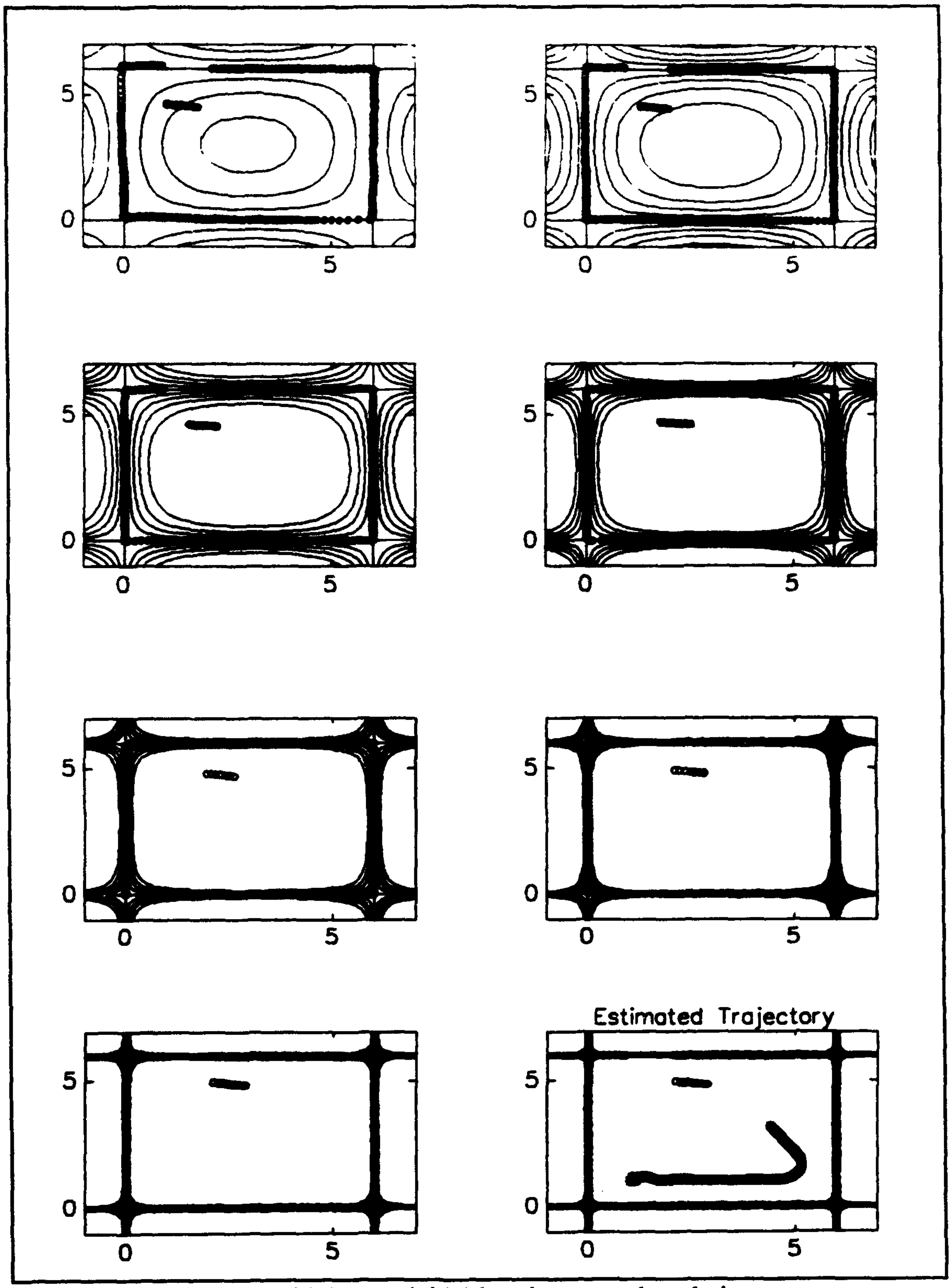

**Figure 3.13 Estimation with known initial location, one obstacle is present.**

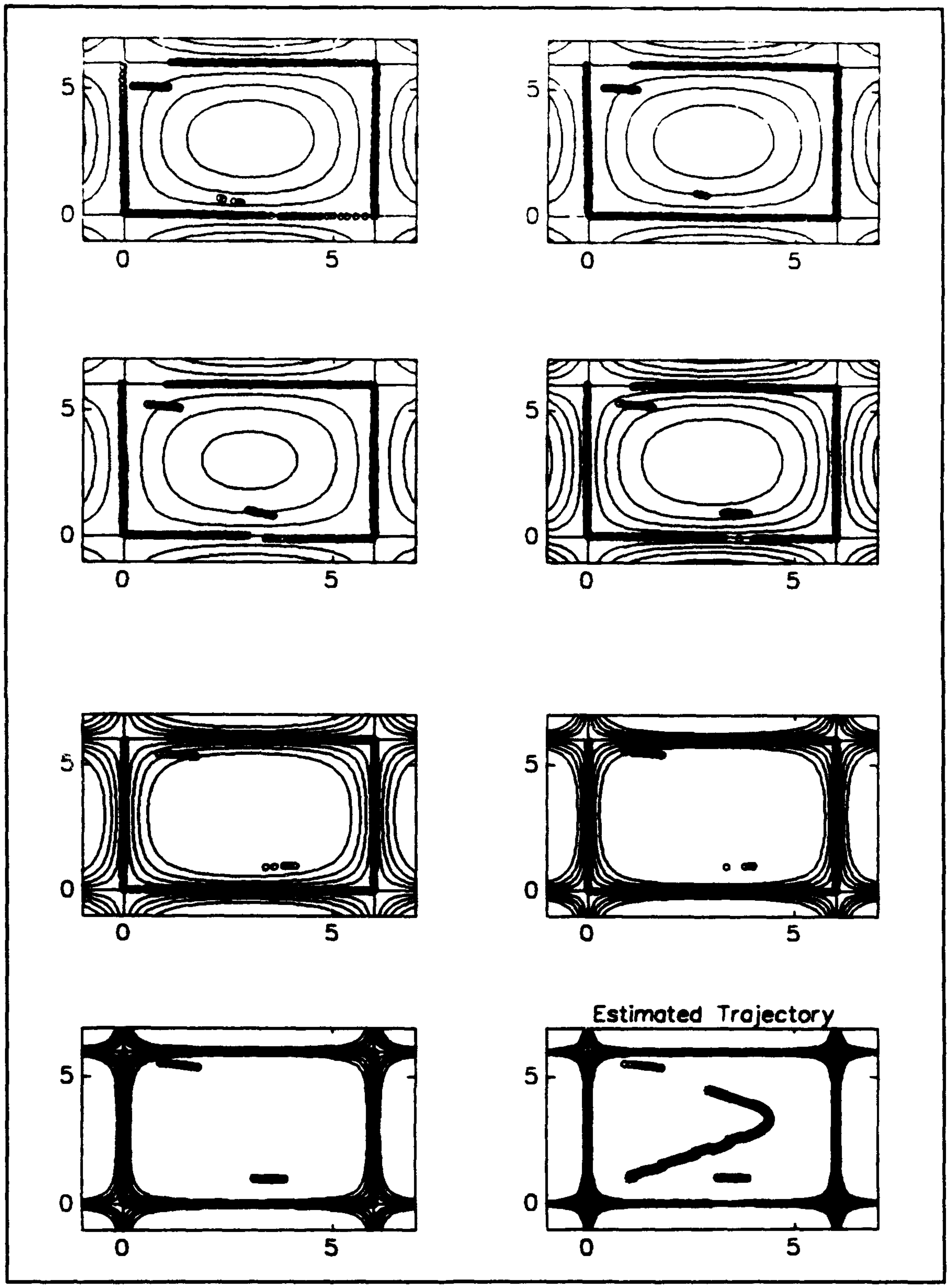

**Figure 3.14 Estimation with known initial location, two obstacles are present.**

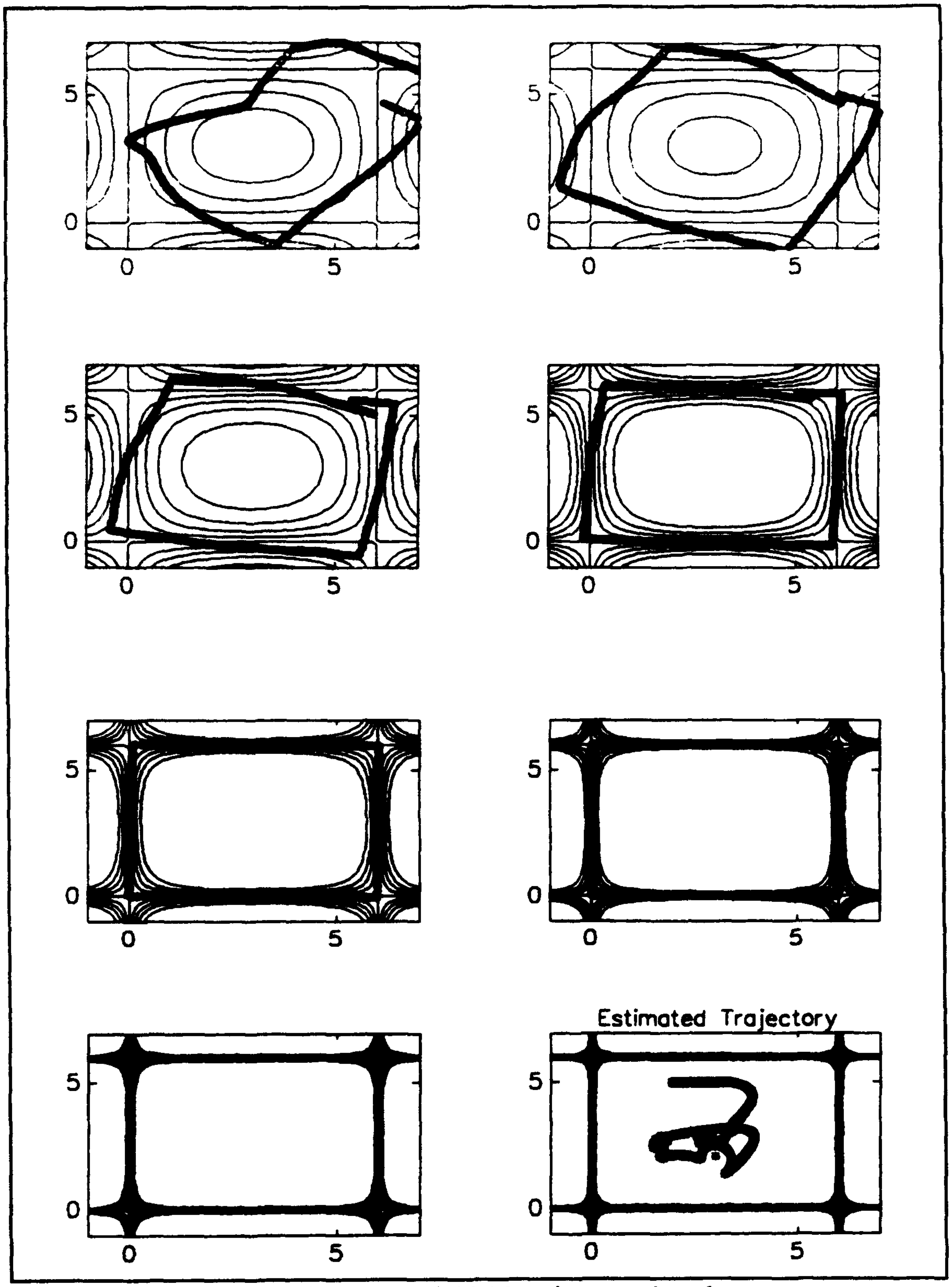

Figure 3.15 Estimation with unknown initial location, no obstacles are present.

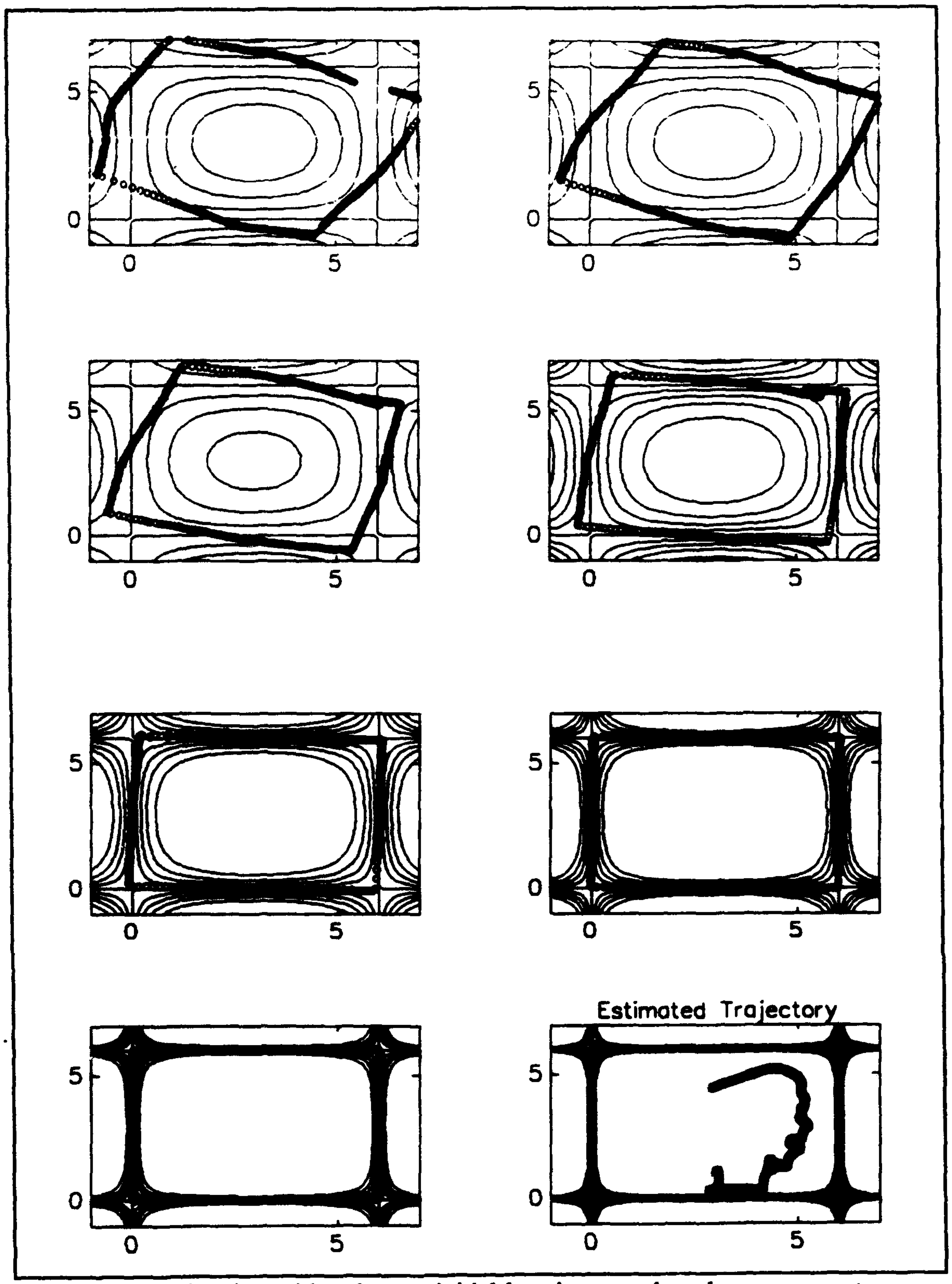

**Figure 3.16 Estimation with unknown initial location, no obstacles are present.**

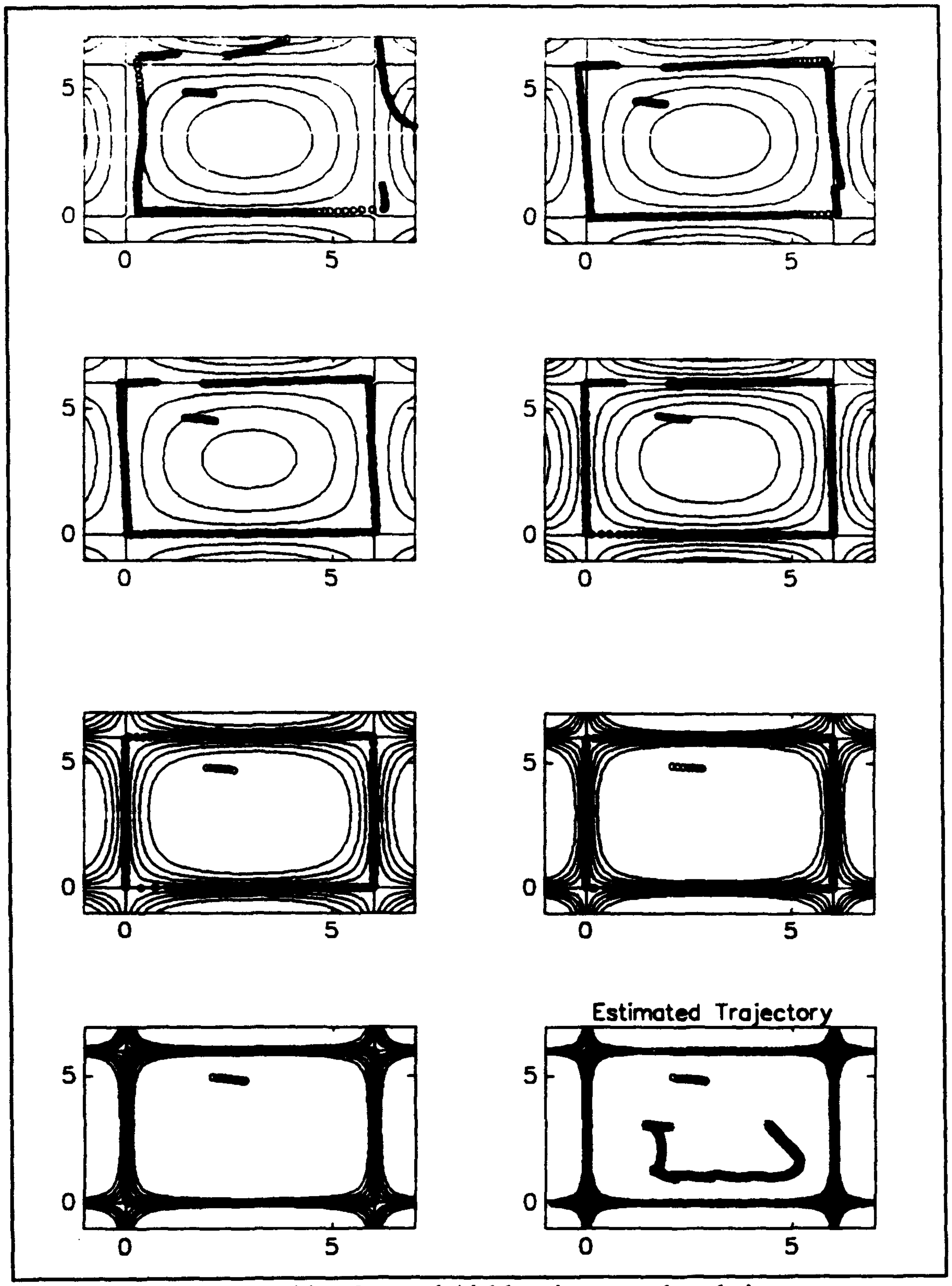

**Figure 3.17** Estimation with unknown initial location, one obstacle is present.

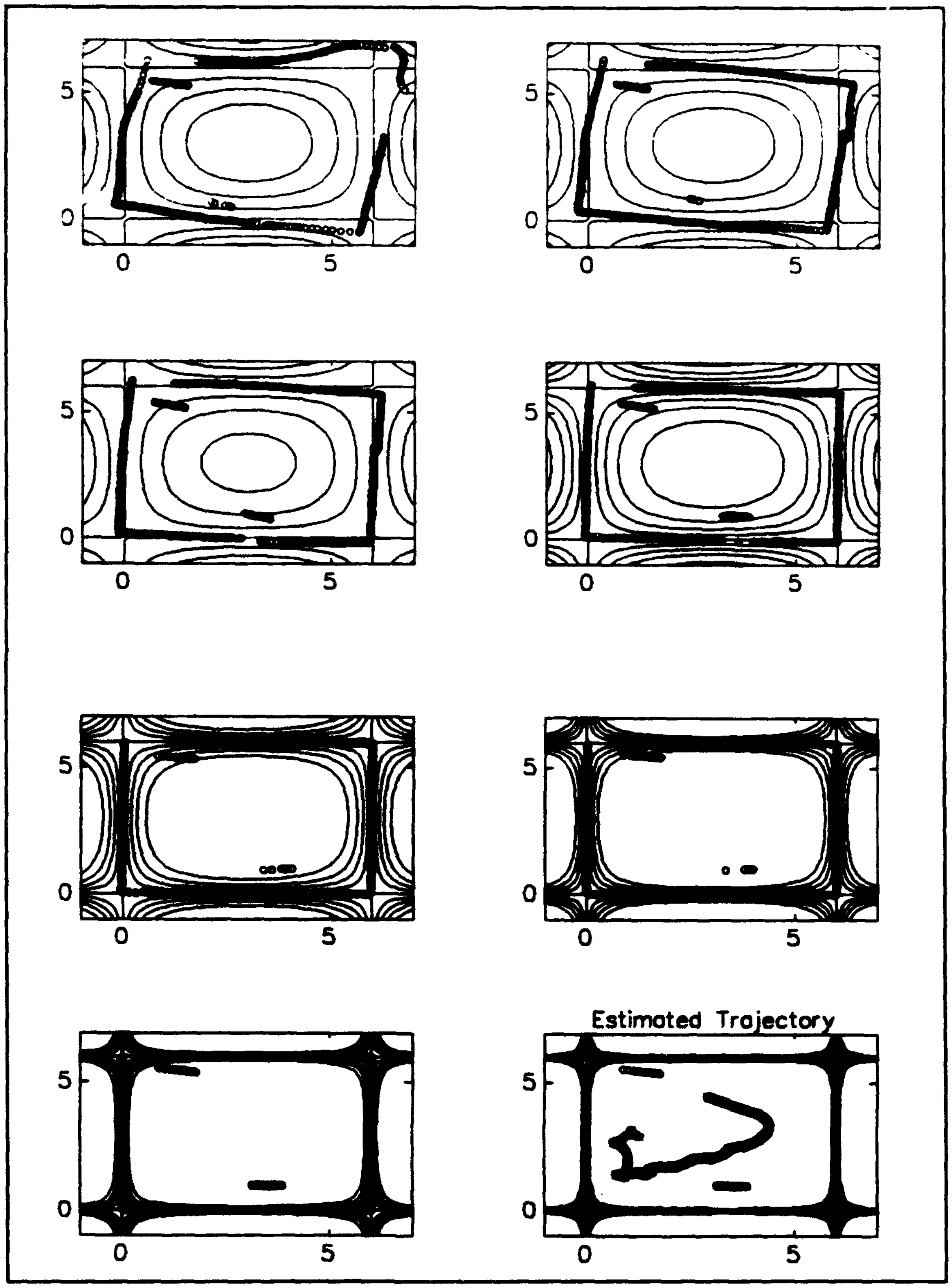

Figure 3.18 Estimation with unknown initial location, two obstacles are present.

# IV. SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

## A. SUMMARY

This thesis presents a study of model-based estimator for small autonomous underwater vehicles. The approach taken in the design and testing of the estimator included:

- The development of linearized models
- The development of potential function for the environment
- The estimation of parameters by means of an ARX model
- The programming in MATLAB
- Simulation studies using additive white Gaussian noise

## B. CONCLUSIONS

In this study, an estimator which uses knowledge of the vehicle dynamics is developed. In particular, position estimation is obtained by combining the sonar range information with inertial measurements.

The following conclusions can be drawn from the results of this study:

- Position, velocity and pitch rate estimation is possible.
- A piecewise constant Extended Kalman gain is adequate for estimator.
- Estimation of states by Potential function technique is possible.

## C. RECOMMENDATIONS

The algorithm explored in this thesis could be used in the interim for the NPS AUV II during pool missions. It is therefore recommended that the following be accomplished to facilitate implementation of the estimator:

- Convert the estimator's main loop and other MATLAB functions to C and compile the entire program for use in the NAPS AUV II computer.
- Find a better way of identifying potential functions for non-geometrical environments.
- Implement neurol networks to identify potential functions.
- Try to eliminate an extra controller by using potential functions in obstacle avoidance problem.

# APPENDIX A

## A. PROGRAM STRUCTURE

The programs are PRDATA4.M, RANGEH.M, and AUV5.M. The names of the source code files for different versions of the programs are appended with the version number, for example, PRDATA4.M.

AUV5.M is the main source code, which initializes the filter parameters, the simulated vehicle state, the Extended Kalman filter state, the control input, and the $\Phi$, $\Delta_1$, and H matrices of the linearized model. The program reads the data file as a standard MATLAB matrix instead of running a separate vehicle model to generate simulated measurements.

PRDATA4.M is the source code, which generates the simulated measurements. It also saves the measurements into a data file as a standard MATLAB matrix.

RANGEH.M is the source code, which generates the estimated nonlinear measurements of range. It also calculates the gradient of range with respect to heading and position of the vehicle. RANGEH.M is called by both PRDATA4.M and AUV5.M.

## B. PROGRAM LISTING FOR SIMULATIONS IN CHAPTER II

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   File  AUV5.M%
%         This little program is designed for estimating the position of an AUV which
%         moves in horizontal plane. We assume that AUV starts from a known
%         position with a constant heading and constant speed. In this model we also
%         use RPM and RUDDER angle as the inputs to the system.
%
%   Calls           RANGEH.M
%
%   Modified        26 Jun 93
%
%   Ver.5
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear
clg
clc

L1=6;L2=6;          %pool dimensions

Ts=0.0225;          %entering sampling time
load datae25.dat
clear D
D=datae25;
kmax=length(D)+1;          %discrete time
xo=[.8,1.2,0,1,0]';          %initilization of states
xk1k=zeros(5,kmax);
xk1k(:,1)=x0;
P0=diag([10 10 0 10 0]);
pk1k=P0;
R=0.1;
alfa=1;
beta=0.001;
sigma=1;
gama=1;

for k=1:kmax-1
   headh=xk1k(4,k);     %heading of vehicle
   theta=D(k,1);
   rho=D(k,2);
   RPM(k)=D(k,6);
```

```
  RUDDER(k)=D(k,7);

  a=cos(headh*pi/180);
  b=sin(headh*pi/180);
  A=[     0 0 a 0 0
          0 0 b 0 0
          0 0 -alfa 0 0
          0 0 0 0 1
          0 0 0 0 -gama];
  B=[0 0;0 0;beta 0;0 0;0 sigma];
  [phi,del1]=c2d(A,B,Ts);

  shdg=theta+headh;
  if shdg > 180
    shdg=-360+shdg;
  end
  [rhoh,ho]=rangeh(xk1k(1,k),xk1k(2,k),shdg,L1,L2); %sonar range
  drdx=ho(1,2);
  drdy=ho(1,3);
  h=[drdx drdy 0 0 0];
  KG=pk1k*h'/(h*pk1k*h'+R);
  pk1k1=(eye(5)-KG*h)*pk1k;
  pk1k=phi*pk1k1*phi';
  xk1k1(:,k)=xk1k(:,k)+KG*(rho-rhoh);
  xk1k(:,k+1)=phi*xk1k1(:,k)+del1*[RPM(k) RUDDER(k)]';

  xp(k)=xk1k1(1,k)+rhoh*cos(shdg*pi/180);
  yp(k)=xk1k1(2,k)+rhoh*sin(shdg*pi/180);
end
axis([-1 L1+1 -1 L2+1])
plot(xk1k1(1,:),xk1k1(2,:),D(:,4),D(:,5),'g',xp,yp,'*'),grid
xlabel('x'),ylabel('y'),title('POOL')
!del tezg25.met
meta tezg25
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   File  PRDATA4.M
%
%         This little program is designed for creating data.We assume that AUV starts
%         from a known position with a constant heading and constant speed.We also
%         use the RPM and RUDDER angle as the input. In this program we also added
%         small noises to the states.
%
%   Calls RANGEH.M
%
%   Modified 26 Jun 93
%
%   Ver.4
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


clear
clg
clc

L1=6;L2=6;                 %pool dimensions

Ts=0.0225;                 %entering sampling time
Tf=9;                      %time required for a 360 degrees return
kmax=9*Tf/Ts;              %discrete time
x=zeros(5,kmax);
x(:,1)=[1 1 0 315 0]';
RPM=150*ones(1,kmax);
RUDDER=zeros(1,kmax);
alfa=1;
beta=0.001;
sigma=1;
gama=1;

for k=1:kmax-1
   tetha=x(4,k);
   a=cos(tetha*pi/180);
   b=sin(tetha*pi/180);
```

```
    A=[     0 0 a 0 0
            0 0 b 0 0
            0 0 -alfa 0 0
            0 0 0 0 1
            0 0 0 0 -gama];
    B=[0 0;0 0;beta 0;0 0;0 sigma];
    E=eye(5);
    [phi,del1]=c2d(A,B,Ts);
    [phi,del2]=c2d(A,E,Ts);


     RUDDER(k)=5;


    rand('normal')
    ex=0.01*rand;
    ey=0.01*rand;
    ev=0.01*rand;
    et=0.01*rand;
    etd=0.01*rand;
    x(:,k+1)=phi*x(:,k)+del1*[RPM(k) RUDDER(k)]'+del2*[ex ey ev et etd]';
    head(k)=x(4,k);

    shdg(k)=rem(head(k)+0.9*k,360);              %sonar heading
    shdg1(k)=rem(0.9*k,360);
    if shdg(k) > 180
       shdg(k)=-360+shdg(k);
    end
    if shdg1(k) > 180
       shdg1(k)=-360+shdg1(k)+0.01*rand;
    end
    [r,h]=rangeh(x(1,k),x(2,k),shdg(k),L1,L2);  %sonar range
    dist(k)=r+0.01*rand;
    temp(k,:)=[shdg1(k) dist(k) x(3,k) x(1,k) x(2,k) RPM(k) RUDDER(k) x(4,k) x(5,k)];
end
!del datae33.dat
save datae33.dat temp /ascii     %saving data file in ASCII code

axis('square')
axis([0 L1 0 L2])
plot(x(1,:),x(2,:)),grid
title('POOL')
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%    File  RANGEH.M
%
%          [r,h]=range(x,y,theta,L1,L2)
%          computes the sonar range (r) and its gradient (h)
%          at position (x,y) with heading theta
%          in the L1 by L2 pool.
%          h=[dr/dtheta, dr/dx, dr/dy]
%
%    Modified  26 Jun 93
%
%    Called by AUV5.M and PRDATA4.M
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [r,h]=rangeh(x,y,theta,L1,L2)
theta=theta*pi/180;
th1=atan2(-y,-x);
th2=atan2(L2-y,-x);
th3=atan2(L2-y,L1-x);
th4=atan2(-y,L1-x);

while (theta >th1+2*pi),
theta=theta-2*pi;
end
while (theta <=th4),
theta=theta+2*pi;
end
c=cos(theta);s=sin(theta);

if (theta>=th2)&(theta<th1+2*pi),
      r=-x/c;
      h=[-x*s/c^2,-1/c,0];
end

if (theta>=th3)&(theta<th2),
      r=(L2-y)/s;
      h=[(y-L2)*c/s^2,0,-1/s];
end


if (theta>=th4)&(theta<th3),
        r=(L1-x)/c;
        h=[(L1-x)*s/c^2,-1/c,0];
end

if (theta>=th1+2*pi)&(theta<th4+2*pi),
        r=-y/s;
        h=[-y*c/s^2,0,-1/s];
end
```

# APPENDIX B

## A. PROGRAM STRUCTURE

The programs are SIMUL5.M, SCAN5.M, VP5.M, SIGMOID.M, DSIG.M, SHOW5.M, SIMUL8.M, and SCAN8.M. The names of the source code files for different versions of the programs are appended with the version number, for example, SIMUL8.M.

SIMUL5.M is the main source code, which initializes the filter parameters, the simulated vehicle state, the Extended Kalman filter state, the control input, and the $\Phi$, $\Delta_1$, and H matrices of the linearized model. The program reads the data file as a standard MATLAB matrix instead of running a separate vehicle model to generate simulated measurements. In this source code the dynamic parameters of the vehicle are assumed to be known.

SCAN5.M is designed for estimating the states of the vehicle using the potential function approach. It scans the estimated borders of the map. VP5.M, SIGMOID.M, DSIG.M, and SHOW5.M are designed to calculate the value of the potential function, the value of sigmoid function, the derivatives of sigmoid function and to show the graphics respectively.

SIMUL8.M and SCAN8.M are basically same with the SIMUL5.M and SCAN5.M. The only difference introduced in these versions is the estimation of the dynamic parameters of the vehicle by using an ARX model and RLS method.

## B. PROGRAM LISTING FOR SIMULATIONS IN CHAPTER III

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%    File  SIMUL5.M
%
%         This source code is designed for estimating the position of an AUV which
%         moves in horizontal plane. We assume that AUV starts with a constant
%         heading and speed. In this model we use RPM and RUDDER angle as the
%         inputs to the AUV system.
%
%    Calls          SCAN5.M, SHOW5.M
%
%    Modified       9 Jul 93
%
%    Ver.5
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


clear;
clg;
hold off;
clc
subplot(221)
load c:\mat\tez\data\datae32.dat        % data of pool
data=datae32;
n=length(data);

n1=1; n2=400;
lambda=50;
zhm=zeros(5,1);
P=diag([10,10,10,10,10]); zh0=[3,1,0,0,0]';        % initialize estimate
i=0;
while  n2<n,
       [zh,P,zs]=scan5(zh0,P,data(n1:n2,:),lambda);
       xm=zs(1,:); ym=zs(2,:);
       zhm=[zhm, zh];
       i=i+1;
       show5
       if i==4
       meta tezg37
```

```
          end
          hold off
          n1=n2; n2=min([n1+400,n]);
          lambda=0.5*lambda;
          nz=length(zh); zh0=zh(:,nz);
end

hold off
show5
hold on
plot(zhm(1,:), zhm(2,:),'*g'), title(' Estimated  Trajectory')
meta
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%    File  SCAN5.M
%
%          This source code is designed for estimating the position for static point using
%          potential fuction. It scans the estimated borders of the map
%
%    Calls          VP5.M
%
%    Modified       8 Jul 93
%
%    Ver.5
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function [zh,P,zs]=scan5(zh0,P,data,lambda)
n=length(data);
zh(:,1)=zh0;
R=1;
alfa=11.2957;beta=.0111;sigma=1.0023;gama=1.0023;Ts=0.0225;
for t=1:n-1
rho=data(t,2); theta=data(t,1); headh=zh(4,t);
RPM(t)=data(t,6); RUDDER(t)=data(t,7);
a=cos(headh*pi/180); b=sin(headh*pi/180);
A=[0 0 a 0 0
   0 0 b 0 0
   0 0 -alfa 0 0
   0 0 0 0 1
   0 0 0 0 -gama];
B=[0 0;0 0;beta 0;0 0;0 sigma];
[Phi,del]=c2d(A,B,Ts);
dx0=rho*cos((theta+headh)*pi/180);
dy0=rho*sin((theta+headh)*pi/180);
x0=zh(1,t)+dx0;
y0=zh(2,t)+dy0;
zs(:,t)=[x0;y0];
[v,dvx,dvy]=VP5(x0,y0,lambda);
h=[dvx,dvy,0,(-dvx*dy0+dvy*dx0)*pi/180,0]';
s=h'*P*h+1; K=Phi*P*h/s;
e=0-v;
zh(:,t+1)=Phi*zh(:,t)+del*[RPM(t) RUDDER(t)]'+K*e;
P=Phi*P*Phi'-K*s*K';
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%    File  VP5.M, SIGMOID.M, DSIG.M, SHOW5.M
%
%         These source codes are designed for calculating the potential function, the
%         value of sigmoid, its derivative at a given point and to show graphics.
%
%    Modified        9 Jul 93
%
%    Ver.5
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function [v,dvx,dvy]=VP5(x,y,lambda);
v0=(x.*(x-6))*(y.*(y-6))';
z=(v0)/(lambda); v=sigmoid(z);
dvx=dsig(z)*((x-6)*(y.*(y-6))'+x*(y.*(y-6))')/lambda;
dvy=dsig(z)*((x.*(x-6))*(y-6)'+(x.*(x-6))*y')/lambda;
end  % VP5

function y=sigmoid(x)
x=min(x,100); x=max(x,-100);
y= (1-exp(-x))./(1+exp(-x));
end  % SIGMOID

function d=dsig(x)
% derivative of sigmoid
x=min(x,100);
x=max(x,-100);
temp=exp(-x);
d=temp ./ (1+2*temp + temp .* temp);
end  %DSIG

%SHOW5.M
x=-1:.1:7;
y=-1:.1:7;
[v,dvx,dvy]=VP5(x',y',lambda);
contour(v,x,y)
hold on
for t=1:length(xm)
plot(xm(t), ym(t), 'og')
end
end %show
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%    File  SIMUL8.M
%
%         This source code is designed for estimating the position of an AUV which
%         moves in horizantal plane. We assume that AUV starts with a constant
%         heading and speed. In this model we use RPM and RUDDER angle as the
%         inputs to the AUV system. Also we try to fit the heading rate and speed data
%         to an ARX model by using RLS.
%
%    Calls          SCAN8.M, SHOW5.M
%
%    Modified       10 Aug 93
%
%    Ver.8
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


!del tezg46.met
!del para8.met
clear; clg; hold off;clc
subplot(221)
load datae25.dat
data=datae25;
n=length(data);
Ts=0.0225;

%ESTIMATION OF SYSTEM PARAMETERS BY USING RLS AND
%FIRST ORDER ARX MODEL

%ESTIMATION FOR SPEED PARAMETERS
v1=data(:,3);          %MEASURED SPEED DATA
u1=data(:,6);          %RPM INPUT

th1=zeros(2,n);
P1=10*eye(2);
for k1=2:n-1
   phit1(k1,:)=[-v1(k1-1) u1(k1-1)];
   den1(k1)=1+phit1(k1,:)*P1*phit1(k1,:)';
   K1(:,k1)=P1*phit1(k1,:)'/den1(k1);
   P1=P1-P1*phit1(k1,:)'*phit1(k1,:)*P1/den1(k1);
   th1(:,k1+1)=th1(:,k1)+K1(:,k1)*(v1(k1)-phit1(k1,:)*th1(:,k1));
end
```

```
%ESTIMATION FOR YAW RATE PARAMETERS
v2=data(:,9);            %MEASURED HEADING DATA
u2=data(:,7);            %RUDDER ANGLE INPUT

th2=zeros(2,n);
P2=10*eye(2);
for k2=2:n-1
   phit2(k2,:)=[-v2(k2-1) u2(k2-1)];
   den2(k2)=1+phit2(k2,:)*P2*phit2(k2,:)';
   K2(:,k2)=P2*phit2(k2,:)'/den2(k2);
   P2=P2-P2*phit2(k2,:)'*phit2(k2,:)*P2/den2(k2);
   th2(:,k2+1)=th2(:,k2)+K2(:,k2)*(v2(k2)-phit2(k2,:)*th2(:,k2));
end

%TRANSFORMING PARAMETERS FROM DISCRETE TO CONTINUOUS TIME
phi1=[-th1(1,n) 0;0 -th2(1,n)];
del1=[th1(2,n) 0;0 th2(2,n)];
[A1,B1]=d2c(phi1,del1,Ts);
prms=[-A1(1,1);B1(1,1);-A1(2,2);B1(2,2)];

%PLOTING THE DISCRETE PARAMETERS
t1=0:n-1;
t=0.0225*t1;

plot(t,-th1(1,:));title('ALFA ESTIMATE');xlabel('Time');grid
plot(t,th1(2,:));title('BETA ESTIMATE');xlabel('Time');grid
plot(t,-th2(1,:));title('GAMA ESTIMATE');xlabel('Time');grid
plot(t,th2(2,:));title('SIGMA ESTIMATE');xlabel('Time');grid
pause
meta para9
clg

%SIMULATION DUE TO PARAMETERS ESTIMATED ABOVE

axis('normal')
subplot(221)
n1=1; n2=400;
lambda=200;
zhm=zeros(5,1);

P=diag([100,100,100,100,100]); zh0=[2,3,1,10,10]';        % initialize estimate
i=0;
while  n2<n,
          [zh,P,zs]=scan8(zh0,P,data(n1:n2,:),lambda,prms);
          xm=zs(1,:); ym=zs(2,:);
          zhm=[zhm, zh];
          i=i+1;
          show5
          if i==4
          meta tezg46
          end
          hold off
          n1=n2; n2=min([n1+400,n]);
          lambda=0.5*lambda;
          nz=length(zh); zh0=zh(:,nz);
end

hold off
show5
hold on
plot(zhm(1,:), zhm(2,:),'*g'), title(' Estimated  Trajectory')
meta
end     % simulation
!del den.dat
save den.dat zhm /ascii
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%    File  SCAN8.M
%
%          This source code is designed for estimating the position for static point using
%          potential fuction. It scans the estimated borders of the map
%
%    Calls          VP5.M
%
%    Modified       8 Aug 93
%
%    Ver.8
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function [zh,P,zs]=scan8(zh0,P,data,lambda,prms)
n=length(data);
zh(:,1)=zh0;
R=1;
alfa=prms(1);beta=prms(2);sigma=prms(3);gama=prms(4);Ts=0.0225;
for t=1:n-1
rho=data(t,2); theta=data(t,1); headh=zh(4,t);
RPM(t)=data(t,6); RUDDER(t)=data(t,7);
a=cos(headh*pi/180); b=sin(headh*pi/180);
A=[0 0 a 0 0
      0 0 b 0 0
      0 0 -alfa 0 0
      0 0 0 0 1
      0 0 0 0 -gama];
B=[0 0;0 0;beta 0;0 0;0 sigma];
[Phi,del]=c2d(A,B,Ts);
dx0=rho*cos((theta+headh)*pi/180);
dy0=rho*sin((theta+headh)*pi/180);
x0=zh(1,t)+dx0;
y0=zh(2,t)+dy0;
zs(:,t)=[x0;y0];
[v,dvx,dvy]=VP5(x0,y0,lambda);
h=[dvx,dvy,0,(-dvx*dy0+dvy*dx0)*pi/180,0]';
s=h'*P*h+1; K=Phi*P*h/s;
e=0-v;
zh(:,t+1)=Phi*zh(:,t)+del*[RPM(t) RUDDER(t)]'+K*e;
P=Phi*P*Phi'-K*s*K';
end
```

## LIST OF REFERENCES

1. Leonard, J. J., and Durrant-Whyte, H. F., *Directed Sonar Sensing For Mobile Robot Navigation*, Kluwer Academic Publishers, 1992.

2. Bildberg, D. R., "Time-Ordered Architecture For Knowledge-Based Guidance Of An Unmanned Untethered Submersible," paper presented at the Oceans Engineering Conference, Washington, D. C., 10 - 12 September 1984.

3. Elfes, A., "Sonar - Based Real World Mapping and Navigation," *IEEE Journal Of Robotics And Automation*, Vol. RA - 3, NO. 3, June 1987.

4. Leondes, C. T., and others, *Theory And Applications Of Kalman Filtering*, Technical Editing And Reproduction Ltd., Febuary 1970.

5. Gelb, A. , and others, *Applied Optimal Estimation*, The Massachusetts Institute of Technology Press, 1974.

6. Burl, J. B., " Derivation Of The Kalman Filter Equations," Notes For EC3310 ( Linear Optimal Estimation And Control), Naval Postgraduate School, 1990 (Unpublished).

7. Thaler, George J., *Automatic Control Systems*, West Publishing Company, 1989.

8. Miller, A. C., *An Application Of Extended Kalman Filtering To A Model - Based, Short - Range Navigator For An AUV*, Master's Thesis, Naval Postgraduate School, Monterey, California, December 1991.

9. Cristi, R. , "Sensor Based Navigation of an Autonomous Underwater Vehicle," paper presented at the Proceedings at International Symposium on Unmanned, Untethered Submersible Technology, Durham, N.H., September 1993.

10. Johansson, R. , *System Modeling and Identification*, Prentice-Hall Inc., 1993.

# INITIAL DISTRIBUTION LIST

No. Copies

1. Defense Technical Information Center 2
Cameron Station
Alexandria, VA 22304-6145

2. Library Code 52 2
Naval Postgraduate School
Monterey, CA 93943-5000

3. Chairman, Code EC 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, CA 93943-5000

4. Professor R. Cristi, Code EC/Cx 2
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, CA 93943-5000

5. Professor H. Titus, Code EC/Ts 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, CA 93943-5000

6. Deniz Kuvvetleri Komutanlığı 1
Personel Daire Başkanlığı
Bakanlıklar, Ankara, TURKEY

7. Deniz Harp Okulu Komutanlığı 2
Tuzla, İstanbul, TURKEY

8. Gölcük Tersanesi Komutanlığı 2
Gölcük, Kocaeli, TURKEY

9. Taşkızak Tersanesi Komutanlığı 2
Taşkızak, İstanbul, TURKEY