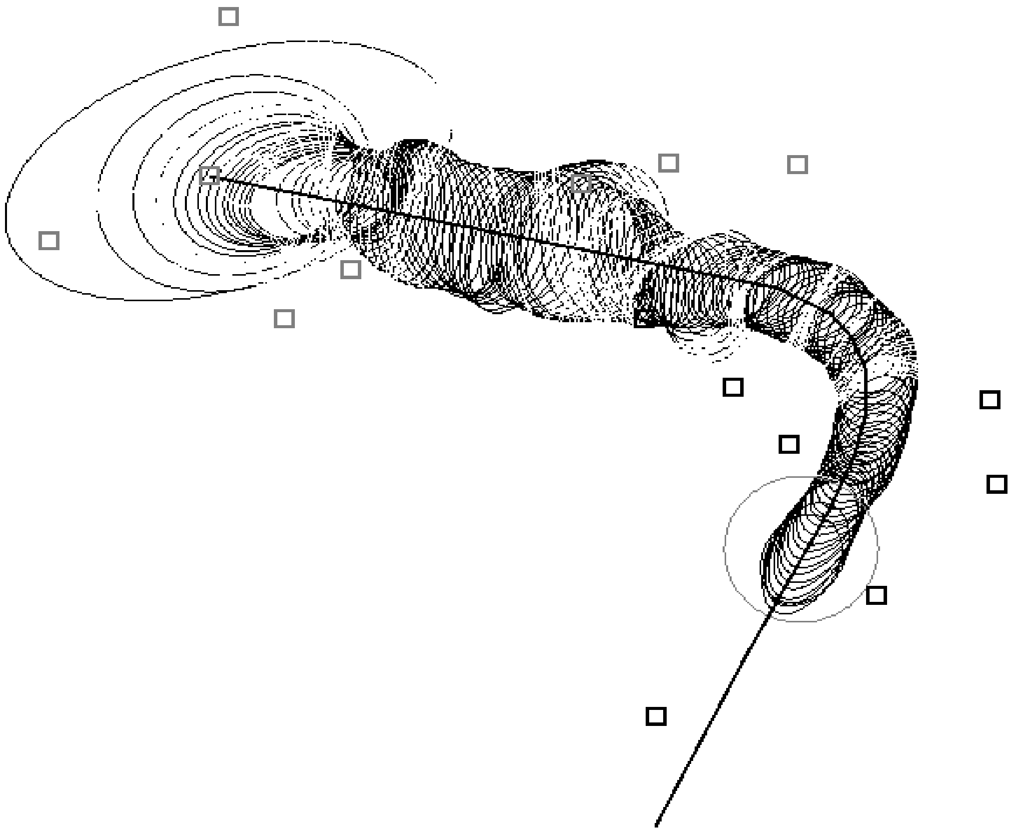
FOA-R--97-00440-505--SE
June 1997
ISSN 1104-9154

Methodology report

KRISTIAN JOHANSSON AND PER SVENSSON

Submarine Tracking by Means of Passive Sonobuoys

II. POSITION ESTIMATION AND BUOY DEPLOYMENT PLANNING



Div. of Command and Control Warfare Technology
SE-581 11 LINKÖPING, SWEDEN

Defence Research Establishment
Division of Command and Control Warfare Technology
P. O. Box 1165
SE-581 11 LINKÖPING
SWEDEN

FOA-R--97-00440-505--SE
June 1997
ISSN 1104-9154

Kristian Johansson and Per Svensson

Submarine Tracking by Means of Passive Sonobuoys

II. Position Estimation and Buoy Deployment Planning

Distribution: ML/Led, 2. Yaflj, 3 Yaflj, FMV:Span, FMV:MML, KTH/NADA

FOA: MTC/OA, 61, 62, 64, 65, 73

Dokumentets utgivare

Försvarets Forskningsanstalt
Avdelningen för Ledningssystemteknik
Box 1165
581 11 LINKÖPING

Dokumentnamn och dokumentbeteckning

FOA-R--97-00440-505--SE

Dokumentets datum

Juni 1997

Projektnummer

E 1434

Projektnamn (ev förkortat)

Styrning av spaningskapacitet

Upphovsman(män)

Kristian Johansson och Per Svensson

Uppdragsgivare

FM

Projektansvarig

Per Svensson

Fackansvarig

Per Svensson

Dokumentets titel

Följning av ubåt med hjälp av passiva hydrofonbojar
II. Positionsestimering och planering av bojutläggning

Huvudinnehåll

Denna rapport behandlar problemet att följa och prediktera en ubåts rörelse med hjälp av adaptiv planering för placeringen av passiva sonarbojar. När fyra eller flera bojar är utplacerade på ett fördelaktigt sätt erhålles ett positionsestimat samt en konfidensellips för estimatet i varje tidssteg.
Målets framtida position predikteras med hjälp av Kalmanfiltrering. Osäkerheten i framtida positionsestimat beräknas och genom att hålla den största osäkerheten för ett positionsestimat inom det predikterade området under kontroll kan tidpunkten bestämmas för när nästa bojutplacering måste ske. Positionen för denna boj väljs sedan via en avvägning mellan de båda kraven minimal osäkerhet i positionsestimatet och maximalt utnyttjande av bojen.
Simuleringar visar att strategin fungerar i den förenklade tvådimensionella modell av verkligheten som programmet utgör.

Nyckelord

Ubåtsmålföljning, Prediktion, Kalmanfilter, Estimering, Sonarekvationen, Adaptiv planering, Datafusion, Sonarbojar

Övriga bibliografiska uppgifter

Språk

Engelska

ISSN 1104-9154

ISBN

Omfång

IV+53 sidor

Pris

( ) Begränsad distribution

Distributör (om annan än ovan)

Issuing organization	Document name and doc.ref.no.	
Defence Research Establishment Division of Command and Control Warfare Technology P.O. Box 1165 SE-581 11 LINKÖPING SWEDEN	FOA-R--97-00440-505--SE	
	Date of issue June 1997	**Project No.** E 1434
	Project name (abbrev. if necessary) Control of surveillance resources	
Author(s) Kristian Johansson and Per Svensson	**Initiator or sponsoring organization** FM	
	Project manager Per Svensson	
	Scient. and techn. responsible Per Svensson	

Document title

Submarine tracking by means of passive sonobuoys.
II. Position estimation and buoy deployment planning

Abstract

This report treats the problem of tracking and predicting the motion of a hostile submarine using adaptive planning for the positioning of passive sonobuoys. When four or more buoys are deployed in a favourable way, a position estimate and a confidence ellipse for the estimate are obtained at each time step.

Using a Kalman filter the future position of the target is predicted. The uncertainty in the future position estimate is determined and by keeping the maximum uncertainty for a position estimate in the predicted area under control the time point can be determined when a new buoy must be deployed. The position for this buoy is then chosen in balance between the two requirements of minimal measurement uncertainty and maximal use of the buoy.

Simulations show that the strategy works in the simple two dimensional model of the real world that the simulator knows.

Key words

Submarine tracking, Prediction, Kalman filter, Estimation, Sonar equation, Adaptive planning, Data fusion, Sonobuoys

Further bibliographic information	Language	
	English	
ISSN 1104-9154	**ISBN**	
	Pages IV+53 pages	**Price**
	( ) restricted distribution	

1 Introduction

1.1 Background

This study deals with *the optimal placement of passive sonobuoys in order to detect and track a hostile submarine*. The concept to be investigated was the use of reactive planning and data fusion to allow tracking of a hostile submarine in shallow archipelagic waters using a limited amount of sonobuoys.

The basis for this project was a computer simulator [12], [7], written in the object oriented language Eiffel, which simulates a submarine moving in a two dimensional area. The program lets the player place simulated sonobuoys at arbitrary positions and when four or more buoys hear the target, an estimate of the target's position is computed and displayed in the shape of a confidence ellipse. The estimation of the position and the calculation of the confidence ellipse are based on simulated measurements of the so-called time differences of arrival (TDOA) at the sensors of the sound emitted by the target.

The project was carried out as an M Sc project in Computer Science at NADA, KTH, with Kristian Johansson as thesis student and Per Svensson as supervisor. This report is an extended version of the M Sc thesis.

We are grateful to Erland Sangfelt, Staffan Harling and Sven-Lennart Wirkander, FOA, and to Mats Nordin, Marine technology CTH, for their expert advice during our validation of the model.

1.2 The final goal of the study

The final goal of this study is to determine the applicability of reactive planning and multi-sensor data fusion in an archipelagic anti-submarine warfare scenario. Reactive planning means in this case that recent sensor information is used in the placement of further sonobuoys and multi-sensor data fusion means that information from several buoys is combined in such a way that the combined information is more specific and therefore more useful than the unprocessed collection of separate pieces of sonobuoy information.

In other words, can the application of reactive planning and multi-sensor data fusion contribute to solve the problem of submarine detection and tracking in shallow water, archipelagic scenarios?

If the study supports this concept, can we demonstrate convincingly and quantitatively what difference it could make, including showing that these methods can outperform a human decision maker?

1.3 The first version of the simulator

The previously developed first version of the simulator system [12] models a submarine that follows a predefined polygonal path contained in a two-dimensional gaming area.

During the game, the user places sonobuoys at arbitrary locations within the area. The information acquired from the sonobuoys is used to calculate the position of the sub. This is done by use of a generalized variant of the *Hyperbolic Fix Method* [9], [12], which employs the so-called time differences of arrival (TDOA) at the sensors of the sound emitted by the target. The signal-to-noise ratios at the positions of the sonobuoys, needed to determine which of the deployed buoys can actually hear the target, is calculated by use of the *Sonar Equation* [4], [12]. The sonobuoy-position uncertainty and the sonobuoy-information uncertainty are taken into account [3], [12], thus the area of an ellipse depicts the probable location of the sub.

As our experience with the simulator demonstrated to us, it may be quite difficult to decide where to horizontally and vertically place the buoys because the volume where the sub could be located is usually far too large to be completely covered by the detection range of the buoys. As already indicated, the information provided by a single sonobuoy is not sufficient to estimate the position of a submarine. The method used for position determination in the simulator requires four hearing buoys to estimate the target's position and its uncertainty. When only three buoys are within hearing range, the model can estimate target position but not its uncertainty. In the real world, a single buoy cannot even determine a reliable distance range for the sub. The reasons for this are (1) that the source level is usually not known and (2) that the sound propagation circumstances could differ significantly between positions that may be located only some hundred meters apart. For example, in good conditions, a 100 dB source level could be heard over a distance of several kilometers while the same source level may only be heard over a distance of a few hundred meters at a position where the conditions are poor.

The placement problem is a matter of timing as well. To have a chance to detect a submarine, the buoys must be dropped into the water soon after the first alarm of the presence of a possible sub. Other timing factors are the deployment time (the amount of time required to deploy and activate a buoy at a certain position) and the lifetime of a particular sonobuoy. Finally, to be able to track the target efficiently, one should deploy each buoy in such a way that it can hear the target for as long as possible, while continuously contributing to the availability of a sufficiently accurate position estimate.

To approach these issues, the submarine tracking simulator system had to be equipped with a position predictor and a buoy deployment planner, which implied that the following problems had to be studied, solved, implemented in the simulator, and validated:

- make a short-term prediction of the motion of the target using optimal filtering
- maximize the tracking time of the target under the condition that tracking quality is above a certain level
- make an evaluation of the tracking quality

Our solutions to these problems are presented and discussed in the following.

1.4 Method

A literature study was made to gain knowledge in the theories of optimal filtering and underwater acoustics and to find previous results concerning the use of passive sonobuoys, data fusion and tracking. A few articles concerning tracking of submarines with passive sonobuoys were found but none of them dealt with the problem of reactive planning for the positioning of the buoys. In the most recent one [11] a submarine, travelling through a sonobuoy field, is tracked by combining data from spatially separated sensors. The treatment is however completely different from that in this report.

The buoy configuration, seen as a measurement device, delivers a position estimate and the uncertainty in this estimate for every time step. To reduce the uncertainty and to predict the motion of the submarine, a Kalman filter was developed.

The decision of buoy deployment was divided into two parts, one which determines the point in time when a new buoy must be deployed, i.e. when the uncertainty of a measurement in the predicted area gets too big, and one that determines the most favourable position for this buoy. The influence a buoy position has on the uncertainty in the predicted area was studied to gain information on which to base the development of a positioning strategy.

An effort was made to make the simulator as realistic as possible. The different parameters have been chosen after consulting experts in their respective areas.

An “idealised” tracker was developed to evaluate the buoy consumption of the simulator in a theoretically tractable special case. A set of simulations with different presumptions was made to find the limitations of the strategy.

1.5 The sonobuoy

A sonobuoy [6], [12] is basically a submerged microphone with a small radio transmitter. The sonobuoys used in archipelagic anti-submarine warfare in Sweden can be deployed from a moving platform (aeroplane, helicopter). When the buoy hits the water it anchors up at the bottom of the sea and the microphone descends to a predefined depth (Figure 1.1). The most basic (and cheapest) ones assumed here provide information about the time varying sound level and frequency spectrum at its position. The sonobuoy register not only the sound of a possible target but also the background noise such as waves, other ships and animals. Most of the unbiased background noise can be eliminated by integrating the signal from the buoy for a certain time. To determine the position of the submarine requires at least three sonobuoys but to get a more robust position estimation more than three will be necessary.

The advantages of using a sonobuoy instead of other acoustic measuring systems are

- it is cheap
- it can be deployed quickly at an arbitrary position
- it is passive and does not reveal itself acoustically to the submarine

- it is not disturbed by noise from a platform

The buoys used in naval practice today are usually not very well calibrated but in order for the algorithms described in this report to work accurately it is important that they are. An absolute calibration to a normal is necessary for the estimation of the buoy range.

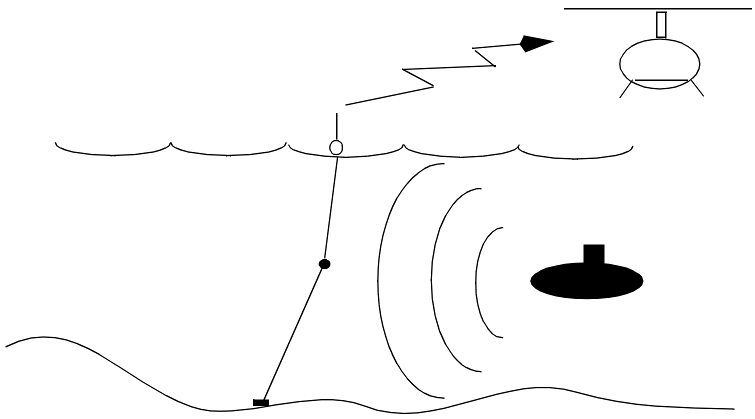


Figure 1.1 A sonobuoy transmits to a platform the time-varying acoustic signal it detects.

1.6 Structure of this report

In Chapter 2 we treat briefly the sonar equation and discuss how its various parameters have been chosen to make the model as realistic as possible.

We need a problem-solving strategy to be able to develop a technique for continually estimating the state of the target and making correct decisions about time and place for buoy deployment. Such a strategy is developed in Chapter 3.

In Chapter 4, the concept of a Kalman filter is introduced. A simple instance of such a filter is developed which proves to be quite appropriate for the tracking problem at hand.

Chapter 5 gives an account of the algorithms that were developed to solve the coupled problems of when and where to deploy the next buoy. We conclude that a closed mathematical solution of the problem is not in hand but we show how an approximate solution can be developed. In Chapter 7, this method is carefully validated and found satisfactory.

To start the tracking process in a situation where it is assumed that a report has been received of a single submarine detection made by other means, the simulator needs a procedure for rediscovering the submarine at a later point in time, i.e., for regaining contact. Such a procedure is developed in Chapter 6. This procedure will be used also to try to rediscover a submarine which was lost during tracking. Note that the length of the time period between the initial contact and regained contact using sonobuoys is probably the parameter which is most critical for the success of the proposed method.

In Chapter 8, we present our conclusions of the study. In brief, the method works well in the world of our simulation model, which is a drastic simplification of reality. One of the critical issues, which will probably become ever more important with the development of more silent, acoustically “stealthier”, submarines, is the limited validity of the assumption of a point-like target when sound emission measurements require ever shorter distances between target and buoys.

In Chapter 9 we discuss what work needs to be done to move from theory to practice, i.e., what additional facilities have to be available before the method can be tested in practice.

2 The choice of parameters in the sonar equation

This chapter treats the sonar equation and how the different parameters in it have been chosen in order to make the model as realistic as possible.

2.1 The sonar equation

The sonar equation [4], [12] states a relationship between the target sound level, the received sound level at the position of the buoy and the sonar equipment. It is a simple equation which has nevertheless proved to be useful in many practical situations.

The sonar equation states:

$$(SL - TL) - (NL - DI) = DT \tag{2.1}$$

where

SL = Source Level, in this case the sound level at the target
TL = Transmission Loss, the loss of sound level from target to buoy,
NL = Noise Level, the background noise at the location of the buoy,
DI = Directivity Index. This noise reduction models the property of certain types of buoy that can determine an angle to the target,
DT = Detection Threshold

All these factors are measured in decibel (dB).

2.2 Source level

Definition:

$$SL = 10\log\frac{\text{source intensity}}{\text{reference intensity}}[\text{dB}] \tag{2.2}$$

To estimate the source level we have to decide in which frequency region to listen to the target. Figure 2.2 indicates that it is in the lower frequency region where the conditions are most favourable. Under normal circumstances this is between 200 and 1200 Hz. Above 1200 Hz it is usually too hard to distinguish the signal from the background noise.
SL is the average value of the sound intensity over these frequencies.

The source level of the target is read into the simulator from a file (Subtrack.Init, Appendix C). It can be changed to describe different types (generations) of submarines. The older types of course generate more noise. A hostile submarine operating in an archipelago is likely to travel in *ultra silent* mode which means that in addition to travelling at low speed, all unnecessary systems are shut down. A typical curve of the sound emission vs. speed can be seen in Figure 2.1. Here the submarine is assumed to operate in ultra silent mode in the low speed region. The estimation/prediction of the source level is of importance for this model. In the current version of the simulator, the source level is modelled by using the estimated speed of the target and then the corresponding source level is found by linear interpolation in a table of source level vs. speed.

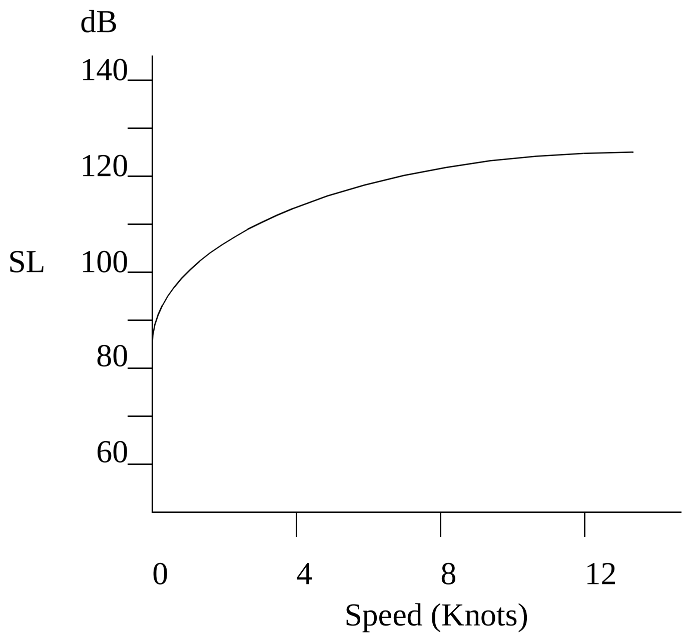


Figure 2.1 Source level vs. speed for an older type of submarine.

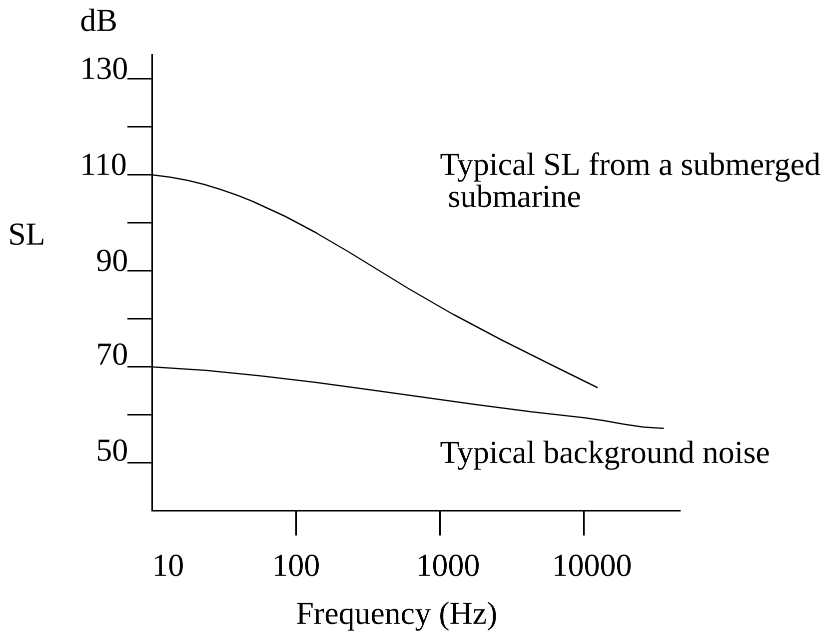


Figure 2.2 Source level vs. frequency at 10 knots.

2.3 Transmission loss

Definition:

$$TL = 10\log\frac{\text{source intensity}}{\text{receiver intensity}}[\text{dB}] \tag{2.3}$$

Here source intensity means the intensity one meter from the source when the source is seen as a point.

The transmission loss is dependent on spreading as well as on absorption, but the latter can be neglected in our case.We first assumed a spreading in between spherical and cylindrical because of the relative shallowness in the archipelago, but after consulting an expert this was changed to pure spherical spreading (power P is constant):

$$P = 4\pi r_1^2 I_1 = 4\pi r_2^2 I_2 \tag{2.4}$$

From this and the definition of TL we get

$$\underline{TL = 20\log r[\text{dB}]} \tag{2.5}$$

2.4 Noise level

Definition:

$$NL = 10\log\frac{\text{noise intensity}}{\text{reference intensity}}[\text{dB}] \tag{2.6}$$

Sea state is a measure of surface roughness. In the archipelago one often calculates with sea state two. The interested could read about the Rayleigh criterion in [4] for further information. In the frequency range in question (200 - 1200 Hz) and assuming sea state two, a typical background noise is around <u>65-70 dB</u> (Figure 2.2).

2.5 Directivity index

This is a noise reduction that is related to the ability of some buoys to determine a bearing to the target. The cheap sonobuoys simulated here do not have this property, thus <u>DI=0</u>

2.6 Detection threshold

Definition:

$$DT = 10\log\frac{\text{source intensity}}{\text{noise intensity}}[\text{dB}] \tag{2.7}$$

DT is a measure of the minimum sound level at the position of the buoy required to detect the signal. One speaks about broadband and narrowband detection thresholds. If the target has a specific sound pattern, i.e a peak in its source level vs. frequency curve, the narrow-band detection threshold should be used and one should try to listen only at this frequency. A modern submarine's sound pattern does not have this property so the broad-band detection threshold has to be used. From [4] we get:

$$DT = 5\log d - 5\log T\beta \tag{2.8}$$

Here T denotes the integration time of the signal and β denotes the bandwidth. The integration time is of central significance for the model. If it is set too short a detection is not possible and if it is set too long tracking is not possible.
To determine d we have to decide values of:

- P_{fa}, the false alarm probability and
- P_D, the detection probability

Once this has been done the corresponding value of d (signal-to-noise ratio) can be obtained from the ROC curve in Figure 2.3. We have chosen P_{fa} =0.0001, P_D =0.99, β=1000 Hz and T=4 sec. Hereby we get d=36 and finally, DT=-10

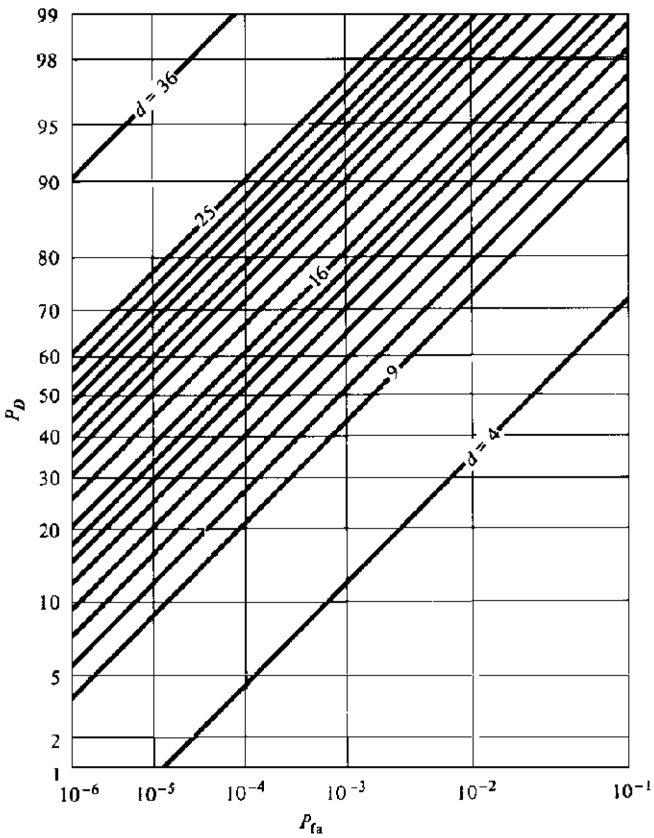


Figure 2.3 Receiver operating curve (reproduced from [4] with permission from the publisher)

3 Planning the solution of the problem

We need a strategy which enables us to continually estimate the state of the target and to make correct decisions about time and place for buoy deployment.

At each point in time the buoys have a common hearing range. If and only if the target is within this range it will be detected by the buoys. The range might change over time and it may not be possible to determine it exactly. However, a hearing range always exists and conceptually it is of importance for the development of the method. The hearing range depends on the integration time (chapter 2.6), and on the factors in the sonar equation.

Given this range, we can construct a set of hearing range areas:

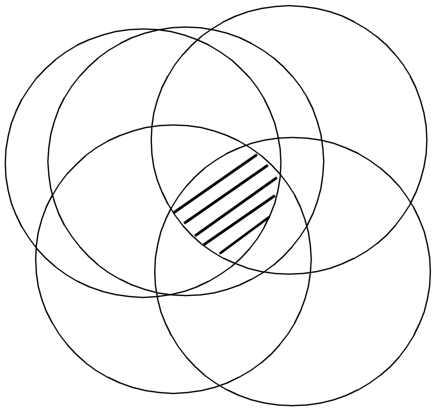


Figure 3.1 Area that is covered by (i.e, within hearing range of) n buoys (here n=5).

Figure 3.1 shows an area that more than n buoys cover. If the target is within this area its position will be measured and if n>3 a confidence ellipse for the position estimate can be calculated. The higher the value of n the better position estimates are obtained. From this we can establish a first requirement for the buoy configuration:

When tracking we must ensure that the target is covered by an area with the above property (n>3). Note that this area may be constantly changing since it is dependent on the sound emission of the target. Figure 3.2 shows an area within which the uncertainty in the position estimate is everywhere less than δ. This means that with probability *p* the submarine will be within δ meters from an estimated position within the area. If it were possible to compute this abstract area efficiently, we would have a good criterion on which to base the decision when to deploy the next buoy. Before the target leaves the area the next buoy should have been deployed in such a way that it extends the area in the direction of motion of the target. This area is also changing with the variation in the sound emission of the target.

.

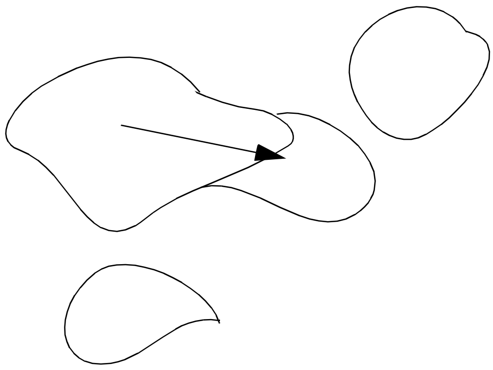


Figure 3.2 Area where the uncertainty in the position estimation is less than δ. Before the target leaves the area the next buoy must have been deployed in order to extend the area.

This can be specified as:

```
procedure
do
    predict_the_position_of_the_target_at_time_(t+T)
   gamma:=max_error_in_position_estimate_in_the_predicted_area
    quantity:=number_of_buoys_covering_the_predicted_area
    if (gamma > δ) or (quantity < n) then
        determine_where_to_deploy_next_buoy
        deploy_next_buoy
    end
end
```

The essential parts of the algorithm are:

- the prediction
- the estimation of the uncertainty of the position estimation
- the calculation of the position where to deploy a buoy

The following chapters treat these parts in detail.

4 Prediction of submarine motion

4.1 The system state and its state function

When estimating and predicting the motion of a moving target it is essential to have a model of its dynamics. In most situations this model can be supplied by a first order stochastic differential equation:

$$\dot{x}(t) = f(x(t)) + w_c(t) \tag{4.1}$$

Here x is the state vector, describing the properties of the target. The state vector could contain information such as position, speed and acceleration of the target. f is the state function describing the change over time of the properties in the state vector. w_c is the system noise and represents unpredictable events in the system.

Normally we can not observe the state directly, but only a measurement y of some function of the state:

$$y = h(x(t)) + v(t) \tag{4.2}$$

where h is the measurement function and v is the measurement noise.

4.2 The Kalman filter

To be able to handle the system in a computer we need to discretize it. One common way to do this is to linearize (4.1) and then use the sampling formula [14]:

$$\begin{gathered} \dot{x}(t) = Ax(t) + w_c(t) \\ \Rightarrow \\ x(t+T) = e^{AT}x(t) + \int_0^T e^{A\tau} w_c(t)\, d\tau \end{gathered} \tag{4.3}$$

If f in (4.1) is linear we can apply (4.3) straight away and with $h(x(t))$ linear as well, an expression of the following form can be obtained:

$$\begin{aligned} X_{k+1} &= G_k \cdot X_k + Wd_k \\ Y_k &= H_k \cdot X_k + V_k \end{aligned} \tag{4.4}$$

For an introduction to the Kalman filter, chapter 4 in [13] is recommended. [1,2,5,9] also provide valuable information.

The idea behind the Kalman filter is to compute an *a posteriori* state estimate $\hat{X}e_k$ as a sum of an *a priori* estimate $\hat{X}p_k$ and a weighted difference between the new measurement Y_k

and the predicted measurement $H_k \cdot \hat{X}p_k$. This difference is usually called the *innovation.* The process is recursive. All information about the past is stored in the state vector and its covariance matrix.

An optimal estimation of the state and a calculation of the covariance matrix of the state estimate is now made according to equation (4.5). A weight matrix K_k, or *Kalman gain*, is chosen so that if the new measurement is noisy (R_k, the covariance matrix of V_k, is large) we rely more on the predicted value $\hat{X}p_k$. On the other hand, if the predicted value has a large covariance matrix, Pp_k, the measurement is more reliable. Hereby we get the new best estimation $\hat{X}e_k$ and its corresponding covariance Pe_k.

The measurement updating or the *estimation*:

$$\begin{aligned} S_k &= H_k \cdot Pp_k \cdot H_k^T + R_k \\ K_k &= Pp_k \cdot H_k^T \cdot S_k^{-1} \\ \hat{X}e_k &= \hat{X}p_k + K_k \cdot (Y_k - H_k \cdot \hat{X}p_k) \\ Pe_k &= Pp_k - K_k \cdot H_k \cdot Pp_k \end{aligned} \tag{4.5}$$

Here $\hat{X}p_k$ and $\hat{X}e_k$ denote the predicted and the estimated state vector respectively. Pp_k and Pe_k denote the covariance of the predicted and the estimated state vector respectively. S_k is a temporary matrix, denoting the covariance of the innovation, and R_k is the covariance matrix of V_k.

Next we extrapolate the values of $\hat{X}e_k$ and Pe_k into the future to obtain the time update, or *prediction*, of the state $\hat{X}p_{k+1}$ and Pp_{k+1} according to (4.6), below.

The time updating of the state:

$$\begin{aligned} \hat{X}p_{k+1} &= G_k \cdot \hat{X}e_k \\ Pp_{k+1} &= G_k \cdot Pe_k \cdot G_k^T + Q_k \end{aligned} \tag{4.6}$$

where Q_k is the covariance for Wd_k in (4.4)

4.3 A basic model of submarine motion

Let us start the model building with a look at the acceleration components. A submarine operating under sound emission limitation has more freedom to turn than to change its speed, i.e the acceleration components normal (a_n) and parallel (a_p) to the direction of motion are of different size.

In the following, we will refer to the target's *local coordinate system,* defined as in Figure 4.1.

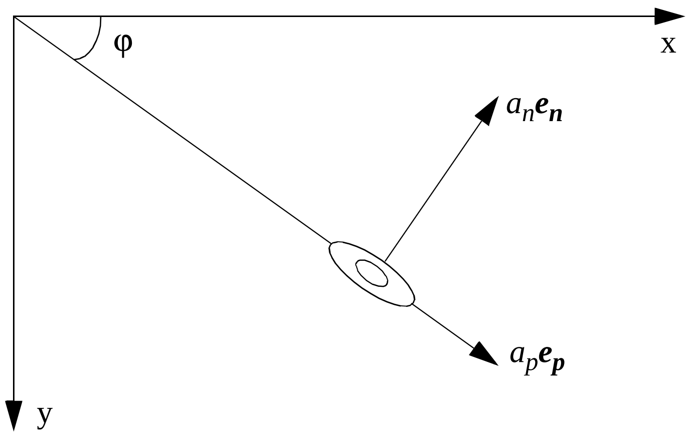


Figure 4.1 Acceleration components expressed in the local system (e_p,e_n).

With this picture in mind we can weigh the advantages of the possible coordinate systems. If we choose the polar system then the acceleration components will be linear (in space) but the rest of the model will be nonlinear. With the cartesian system the situation is the opposite, which is preferable since this allows us to use the linear Kalman filter. Therefore, the latter is chosen.

The acceleration components expressed in the cartesian system will then be:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} a_p \cos\varphi + a_n \sin\varphi \\ a_p \sin\varphi - a_n \cos\varphi \end{bmatrix} = \begin{bmatrix} \cos\varphi & \sin\varphi \\ \sin\varphi & -\cos\varphi \end{bmatrix} \begin{bmatrix} a_p \\ a_n \end{bmatrix} \tag{4.7}$$

Position and speed must of course be included in the state vector but we need to decide if the acceleration component should be included as well.

Including the acceleration components would make the state function nonlinear and the calculations would become more complicated. Since the target is assumed to move slowly so as not to reveal itself, its ability to accelerate is limited and therefore the acceleration model would probably not add much information. In fact, because of the linearization errors in the state function the uncertainty would probably increase.

Therefore the state vector has been chosen as:

$$X = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} \tag{4.8}$$

Since we strive for a linear model, all nonlinearities are placed in the noise vector. This gives us the following state function and noise vectors:

$$f(x(t)) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} = AX$$

$$w_c(t) = \begin{bmatrix} 0 \\ 0 \\ \ddot{x} \\ \ddot{y} \end{bmatrix} \tag{4.9}$$

4.3.1 Time update

According to chapter 4.2 we wish to obtain an expression of the form:

$$x(t+T) = G(T)x(t) + Wd(t, T) \tag{4.10}$$

f in (4.9) is linear so we can apply (4.3) to obtain:

$$x(t+T) = G(T)x(t) + \int_0^T G(\tau)w_c(\tau)d\tau = G(T)x(t) + Wd(t, T)$$

with

$$G(\tau) = e^{A\tau} = \begin{bmatrix} 1 & 0 & \tau & 0 \\ 0 & 1 & 0 & \tau \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.11}$$

Now we can calculate the covariance *Qd* of the discretized noise vector *Wd*(t,T).
For this we need two results from [13] (4.12-13, below):

To calculate the covariance of a function *f(x)*, the following approximate relation can be used (see for example (3.6-9) in [13]):

$$Cov[f(x)] \approx F Cov[x] F^T \tag{4.12}$$

where $F = \frac{df}{dx}$ and $x = \hat{x} = E[x]$

In continuous form, the autocorrelation function for a white noise random forcing function is given by:

$$E[w_c(t) w_c(\tau)^T] = Q_c \delta(t - \tau) \tag{4.13}$$

where Q_c is a spectral density matrix. By multiplying with the Dirac function [time^{-1}] a covariance matrix is obtained.

With (4.12) and (4.13), the covariance matrix for the noise term in (4.10) can be calculated as:

$$\begin{aligned} Q_d &= Cov[Wd(t)] \\ &= E\left[\int_0^T \int_0^T G(\tau) w_c(t) w_c^T(s) G^T(s) ds d\tau\right] \\ &= \int_0^T \int_0^T G(\tau) E[w_c(t) w_c^T(s)] G^T(s) ds d\tau \\ &= \int_0^T \int_0^T G(\tau) Q_c \delta(\tau - s) G^T(s) ds d\tau \\ &= \int_0^T G(\tau) Q_c G^T(\tau) d\tau \end{aligned} \tag{4.14}$$

To calculate Q_c the following is done:
The autocorrelation in the local system is given by:

$$E\left\{\begin{bmatrix} a_p(t) \\ a_n(t) \end{bmatrix} \begin{bmatrix} a_p(s) & a_n(s) \end{bmatrix}\right\} = \begin{bmatrix} \sigma_p^2 & 0 \\ 0 & \sigma_n^2 \end{bmatrix} \delta(t - s) \tag{4.15}$$

From this result the covariance for $(\ddot{x}, \ddot{y})$ in the cartesian coordinate system can be approximately calculated by first applying (4.12) to (4.7) and then using (4.15):

$$E\left\{\begin{bmatrix}\ddot{x}(t)\\ \ddot{y}(t)\end{bmatrix}\begin{bmatrix}\ddot{x}(s) & \ddot{y}(s)\end{bmatrix}\right\}$$

$$= \begin{bmatrix}\cos\varphi & \sin\varphi\\ \sin\varphi & -\cos\varphi\end{bmatrix} E\left\{\begin{bmatrix}a_p(t)\\ a_n(t)\end{bmatrix}\begin{bmatrix}a_p(s) & a_n(s)\end{bmatrix}\right\}\begin{bmatrix}\cos\varphi & \sin\varphi\\ \sin\varphi & -\cos\varphi\end{bmatrix}$$

$$= \begin{bmatrix}\cos\varphi & \sin\varphi\\ \sin\varphi & -\cos\varphi\end{bmatrix}\begin{bmatrix}\sigma_p^2 & 0\\ 0 & \sigma_n^2\end{bmatrix}\begin{bmatrix}\cos\varphi & \sin\varphi\\ \sin\varphi & -\cos\varphi\end{bmatrix}\delta(t-s) \tag{4.16}$$

$$= \begin{bmatrix}\cos\varphi^2\sigma_p^2 + \sin\varphi^2\sigma_n^2 & \cos\varphi\sin\varphi(\sigma_p^2 - \sigma_n^2)\\ \cos\varphi\sin\varphi(\sigma_p^2 - \sigma_n^2) & \sin\varphi^2\sigma_p^2 + \cos\varphi^2\sigma_n^2\end{bmatrix}\delta(t-s)$$

$$= \begin{bmatrix}q_{11} & q_{12}\\ q_{21} & q_{22}\end{bmatrix}\delta(t-s)$$

and hence

$$E[w_c(t)w_c^T(s)] = \begin{bmatrix}0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & q_{11} & q_{12}\\ 0 & 0 & q_{21} & q_{22}\end{bmatrix}\delta(t-s) = Q_c\delta(t-s) \tag{4.17}$$

where

$$Q_c = \begin{bmatrix}0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & q_{11} & q_{12}\\ 0 & 0 & q_{21} & q_{22}\end{bmatrix} \tag{4.18}$$

Finally, by substituting (4.11) and (4.18) into (4.14) and evaluating the integral:

$$Q_d = \begin{bmatrix} \frac{T^3}{3}q_{11} & \frac{T^3}{3}q_{12} & T^2q_{11} & T^2q_{12} \\ \frac{T^3}{3}q_{21} & \frac{T^3}{3}q_{22} & T^2q_{21} & T^2q_{22} \\ T^2q_{11} & T^2q_{12} & Tq_{11} & Tq_{12} \\ T^2q_{21} & T^2q_{22} & Tq_{21} & Tq_{22} \end{bmatrix} \tag{4.19}$$

4.3.2 Choosing the parameters in the Kalman filter

The parameters that need to be estimated are the variances σ_p^2 and σ_n^2 of the acceleration components a_p and a_n.

For a single-propeller ship the practical maximum values of the acceleration components have been assumed to be a_p=0.02 m/s^2 and a_n=0.04 m/s^2. By choosing the variances as a_p^2 and a_n^2 the filter can make good predictions of realistic manoeuvres. With lower values for the variances the risk for unreliable predictions increases, i.e., although the maximal precision of the predictions increases, their robustness decreases.

4.3.3 Measurement update

From the buoy configuration the position of the target is estimated. The measurement function simply becomes the identity:

$$h(X) = \begin{bmatrix} x \\ y \end{bmatrix} \tag{4.20}$$

Since this is a linear function we get the following expression:

$$h(X) = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} = HX \tag{4.21}$$

With each new measurement we also get a covariance matrix R, which expresses the uncertainty in this measurement:

$$R = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix} \tag{4.22}$$

Normally in a Kalman filter this matrix is constant, given a priori by the characteristics of the measuring instrument, but here it will change over time. One could see this as a succession of measurements with different instruments.

4.4 Statistical tests for consistency of the Kalman filter

In Figure 4.2, two statistical tests [2] and the error in the position estimation for one simulation run are presented.

The NEES (Normalized state Estimation Error Squared) is defined as

$$\begin{aligned} \tilde{X}e(k) &= X(k) - \hat{X}e(k) \\ \varepsilon(k) &= (\tilde{X}e(k))^T \cdot Pe^{-1}(k) \cdot \tilde{X}e(k) \end{aligned} \tag{4.23}$$

Under the hypothesis that the state estimation errors are consistent with the filter calculated covariances, ε(k) is chi-square distributed with 4 degrees of freedom. From a table (Table 1.5.4-1 in [2] for example) one sees that the hypothesis can not be rejected. This simply means that the filter parameters fit the model in this situation as we would expect. This test can only be performed when the true system is known (as it is in this simulation).

The NIS (Normalized Innovation Squared) is defined as

$$\varepsilon_z(k) = z(k)^T \cdot s(k)^{-1} \cdot z(k) \tag{4.24}$$

Under the hypothesis that the filter is consistent, $\varepsilon_z(k)$ is chi-square distributed with 2 degrees of freedom. This hypothesis can not be rejected either.

This test can be performed on-line to check the behaviour of the filter. As Figure 4.2 indicates it can be used as a warning when the model starts to behave poorly.

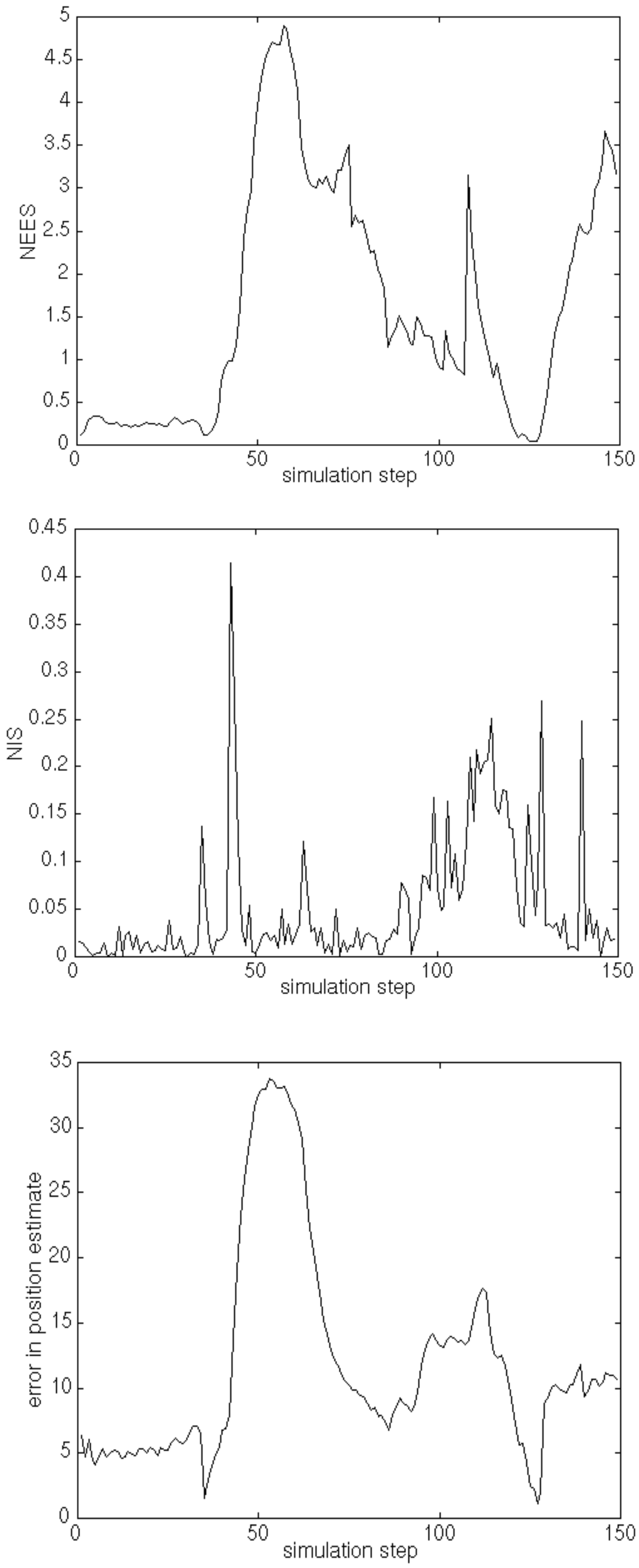


Figure 4.2 Diagrams of the NIS and NEES tests and the true error in the position estimate.

5 Optimization of buoy deployment

5.1 Estimation of the buoy range

Both when estimating the uncertainty in a future position estimation of the target and when calculating an optimal buoy position, it is important to be able to estimate the buoy range as well as possible. This range is dependent on the speed and type of the target and on the sea conditions. The better the buoy range can be estimated the more efficiently one can utilize the buoys. To understand the difficulties one should know that this range can vary from around hundred meters or perhaps less for a modern submarine operating in ultra silent mode in extremely bad weather up to many kilometres if the conditions are ideal.

After the speed of the submarine has been estimated, the corresponding source level is interpolated from a table like the one diagrammed in Figure 2.1. By using the sonar equation (chapter 2) the distance can be calculated as:

$$\text{Buoy range}=10^{(SL-NL-DT)/(TL)}$$

More sophisticated methods could be developed in a final version. See chapter 9.

5.2 Deciding when to deploy a buoy

For every time step the configuration of buoys gives us an estimated position of the target and the uncertainty in this estimate [3,11]. This is our measurement device. These values could of course be used as the final estimate of the system state, but then one would not use all available information, such as our apriori knowledge of the target properties. Therefore the information from every new measurement is fused with the information gathered until current time. This is what the Kalman filter does (chapter 4).

An obvious fact is that for a measurement to be useful it has to have a sufficiently small error. Therefore we cannot let the error covariance matrix of the measurement grow too large. In order to keep the error under control, the following has to be done in every time step:

Predicting the future position of the submarine provides us with a confidence ellipse within which the submarine will be with probability p at time $t+T$. Now, wherever the submarine will be, we want to be able to measure its position with an error less than δ. If this is possible wherever in the predicted area the submarine might be at time $t+T$, there is no need to deploy another buoy. On the other hand, if the measurement error grows too large for some location of the target at time $t+T$, we will have to deploy a new buoy.

We need to calculate the error covariance matrix for the position (x_0,y_0) which our measurement instrument would give us at time $t+T$, if the target were in this position. From this, we can calculate the confidence ellipse and, in particular, the length of the major half axis *a* of the confidence ellipse:

$$a = R\sqrt{\lambda_{max}} \tag{5.1}$$

where λ_{max} is the greatest eigenvalue of the covariance matrix and *R* is determined by solving the equation (see [3,10] for details):

$$p = 1 - e^{\frac{-R^2}{2}} \tag{5.2}$$

Here *p* is the probability that the true position of the submarine is within the ellipse.

5.3 Determining the uncertainty of a future target position estimate

To be able to predict the measurement error that the present configuration of buoys would give if the target was in position (x_v,y_v) an estimation of the error covariance matrix for a target in this position has to be done:

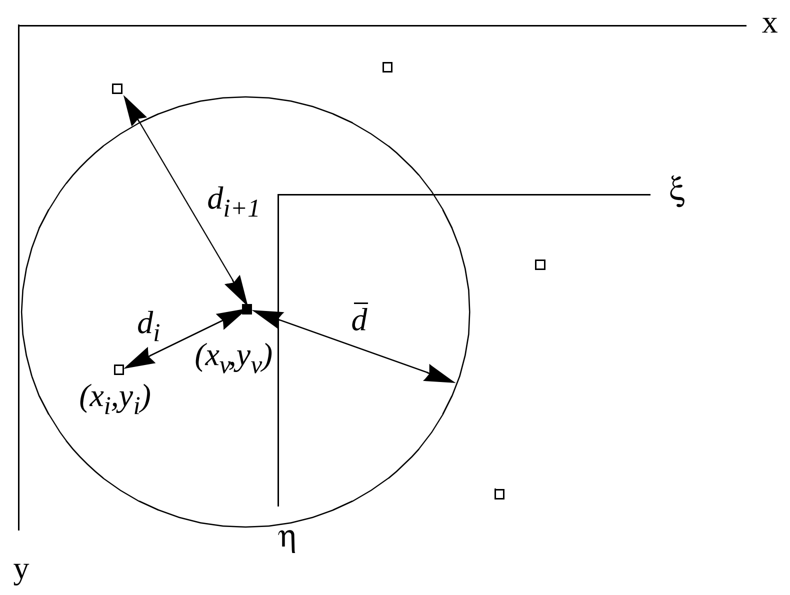


Figure 5.1 The global and the average systems.

- (x_v,y_v) is the position of a virtual target
- (x_i,y_i) is the position of a sonar buoy
- d_i is the distance between buoy *i* and the target

- $\bar{d}$ denotes the average distance between the buoys and the target
- x,y denotes the global system
- ξ,η denotes the average system for the buoy positions

This gives us the following equation system:

$$\begin{aligned} d_1 &= \sqrt{(x_1 - x_v)^2 + (y_1 - y_v)^2} \\ &\dots \\ &\dots \\ d_n &= \sqrt{(x_n - x_v)^2 + (y_n - y_v)^2} \end{aligned} \tag{5.3}$$

The solution to (5.3) is given by (see [3],[12] for details):

$$X = \frac{1}{2} M^{-1} (A^T b) \tag{5.4}$$

where

$$X = \begin{bmatrix} \xi_v \\ \eta_v \\ \delta_v \end{bmatrix} = \begin{bmatrix} x_v - \bar{x} \\ y_v - \bar{y} \\ d_v - \bar{d} \end{bmatrix} \tag{5.5}$$

and

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i,\ \bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i,\ \bar{d} = \frac{1}{n} \sum_{i=1}^{n} d_i \tag{5.6}$$

The vector X is here expressed in the average systems, one for the x,y positions with origin in the average position of the buoys and one for the absolute distances with origin at the average distance between the target and the buoys.

Furthermore

$$A = \begin{bmatrix} \xi_1 & \eta_1 & -(\delta_1) \\ \xi_2 & \eta_2 & -(\delta_2) \\ \dots & \dots & \dots \\ \xi_n & \eta_n & -(\delta_n) \end{bmatrix}, b = \begin{bmatrix} {\xi_1}^2 + {\eta_1}^2 - \delta_1^2 \\ {\xi_2}^2 + {\eta_2}^2 - \delta_2^2 \\ \dots \\ {\xi_n}^2 + {\eta_n}^2 - \delta_n^2 \end{bmatrix} \tag{5.7}$$

and

$$M = A^T A \tag{5.8}$$

Our goal is to estimate the uncertainty of a measurement in X so the next step is to Taylor expand (5.4) around the solution. Note that we regard the δ_ι's as exact values. In reality the TDOA (chapter 1.1) are indeed associated with errors but these are uncorrelated to the errors associated with the buoy positions. They depend on the sonar equipment and water conditions and are assumed to be white noise.

Approximately we get:

$$\begin{aligned} d\xi_v &= \frac{\partial \xi_v}{\partial \xi_1} d\xi_1 + \dots + \frac{\partial \xi_v}{\partial \eta_n} d\eta_n \\ d\eta_v &= \frac{\partial \eta_v}{\partial \xi_1} d\xi_1 + \dots + \frac{\partial \eta_v}{\partial \eta_n} d\eta_n \end{aligned} \tag{5.9}$$

or

$$\begin{bmatrix} d\xi_v \\ d\eta_v \end{bmatrix} = G \begin{bmatrix} d\xi_1 & d\eta_1 \\ d\xi_2 & d\eta_2 \\ \dots & \dots \\ d\xi_n & d\eta_n \end{bmatrix} \tag{5.10}$$

with

$$G = \begin{bmatrix} \frac{\partial \xi_v}{\partial \xi_1} & \frac{\partial \xi_v}{\partial \xi_2} & \dots & \frac{\partial \xi_v}{\partial \xi_n} \\ \frac{\partial \eta_v}{\partial \xi_1} & \frac{\partial \eta_v}{\partial \xi_2} & \dots & \frac{\partial \eta_v}{\partial \xi_n} \end{bmatrix} \tag{5.11}$$

Now the variance of the positions of the buoys has to be approximated.

The buoy positions are associated with white noise:

$$dx_i, dy_i \in N(0, s^2) \tag{5.12}$$

The covariance matrix C for (5.12) will then look like

$$C = \begin{bmatrix} s^2 & & & 0 \\ & s^2 & & \\ & & \dots & \\ 0 & & & s^2 \end{bmatrix} \tag{5.13}$$

Finally the covariance K for the position estimate is given by $K=GCG^T$:

$$K = \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{21} & \sigma_2^2 \end{bmatrix} \tag{5.14}$$

where $\sigma_1^2=\mathrm{var}(d\xi_v)$, $\sigma_2^2=\mathrm{var}(d\eta_v)$ and $\sigma_{12}=\sigma_{21}=\mathrm{cov}(d\xi_v,d\eta_v)$.

Now recalling chapter 5.2, the length of the major axis in the confidence ellipse is obtained from (5.1).

5.4 Deciding where to position the buoy

In every simulation step we want to gain as much information as possible from the buoy constellation. Therefore the position for the next buoy deployment should satisfy two requirements:

(1) The first requirement is related to the local optimization problem. There should be no other buoy position which would enable the buoy constellation, for any future target position within the predicted area, to add more information to the system in the next simulation step.

(2) The second requirement originates from the global optimization problem. In order to save buoys, the position should be chosen so that the buoy can be of use for as long time as possible.

These requirements are in conflict and a balance between them has to be found.

The first optimization problem can be solved as follows:

We want the next measurement to give us as much information as possible. Therefore we must choose the position for the next buoy so that the information from our next measurement will have as small uncertainty as possible.

This leads to the following optimization problem.

We know:

- A buoy must be deployed at time $t=t_{j+1}$
- The submarine will then be in the area $E(t_{j+1})$ with probability p. This is the area that the Kalman filter predicts.
- $r_i \in R(\dot{\hat{\rho}}_j, \hat{\rho}_j)$, r_i=position of buoy i, $\dot{\hat{\rho}}_j, \hat{\rho}_j$ denote the estimated target speed and position respectively. R is an area depending on $\dot{\hat{\rho}}_j, \hat{\rho}_j$, denoting where buoy i can be deployed so as to cover $E(t_{j+1})$.
- $\gamma(r_i, \hat{\rho}_{j+1})$ = the uncertainty associated with the measurement of a target in position $\hat{\rho}_j$ with buoy i deployed in position r_i.
- $G(r_i) = \max\limits_{\hat{\rho}_{j+1} \in E(t_{i+1})} \gamma(r_i, \hat{\rho}_{j+1})$ is the largest uncertainty associated with a measurement in $E(t_{j+1})$.

Now we can state our first optimization problem:

$$\text{minimize } G(r_i) \quad \text{subject to the constraint} \quad r_i \in R(\dot{\hat{\rho}}_j, \hat{\rho}_j) \tag{5.15}$$

This will satisfy the first requirement above.

Some difficulties arise when one tries to satisfy the second requirement. If we were to consider this requirement only, we would place the buoys as far away in the direction of motion as possible. But then the buoys would eventually end up in a row and the resulting confidence ellipse would grow indefinitely. This corresponds to the case d=0 in chapter 7.1. See also [9] where different buoy patterns are discussed.

To achieve a balance between the requirements the following strategy was chosen:

Fix a tolerance δ for $G(r_i)$. We say that requirement (1) is fulfilled if a buoy position results in $G(r_i) < \delta$ (Figure 5.2).

If the position for r_i which maximizes e (the distance along the direction of motion, covered by the buoy i) is chosen, we have a candidate for the optimal position but this position is still only optimal in a local sense. The position might be unfavourable for the future as Figure 5.5 indicates.

To understand what effect the positioning of a buoy has on the accuracy of the target position determination, some simulations have been made (Figure 5.3 to Figure 5.5). The position in the x,y plane for which we have a minimum in the *$G(r_i)$* direction corresponds to requirement (1) above.

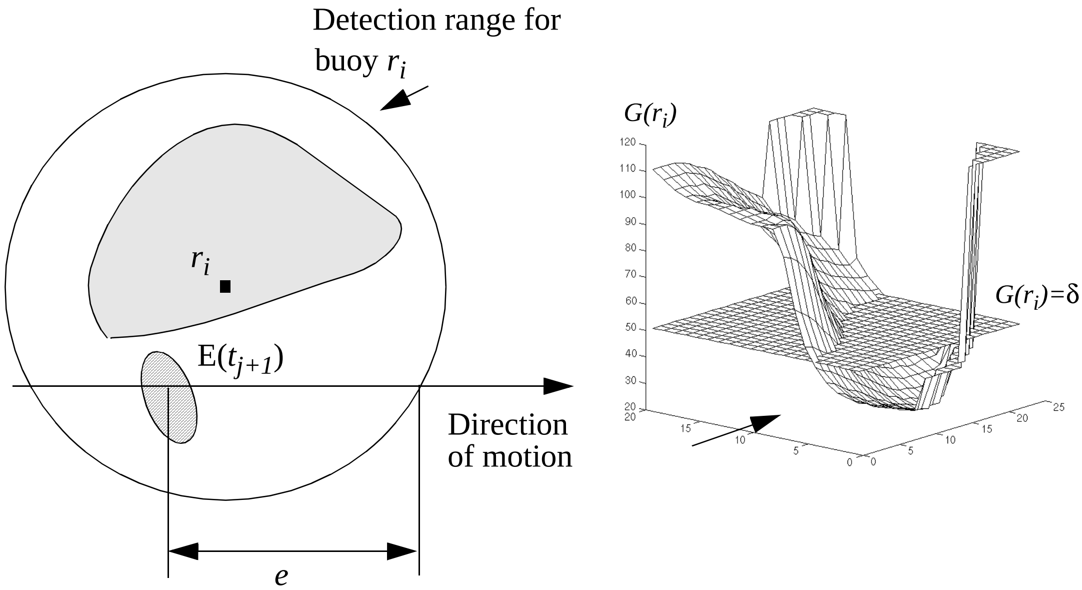


Figure 5.2 A buoy position in the grey area (corresponds to points in the xy plane with *$G(r_i)<\delta$*) with maximal e is a candidate for the optimal position.

Below, *$G(r_i)$* is plotted for a new buoy with position (x,y).

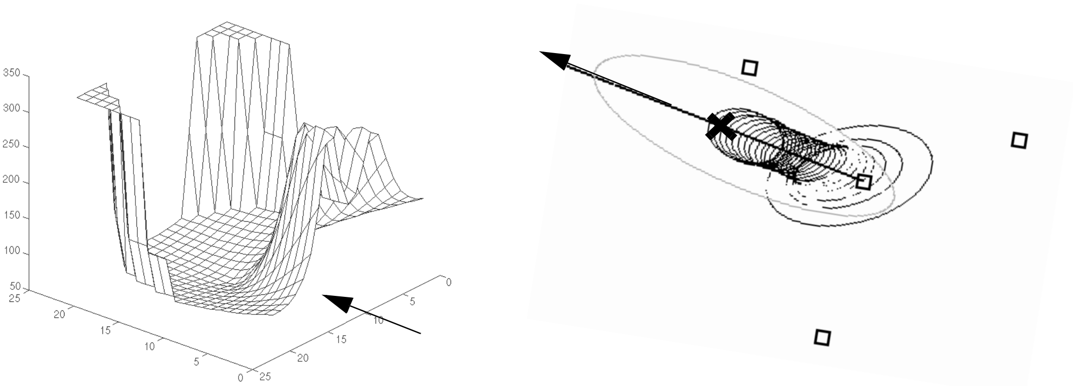


Figure 5.3 A position to the left is slightly more advantageous.

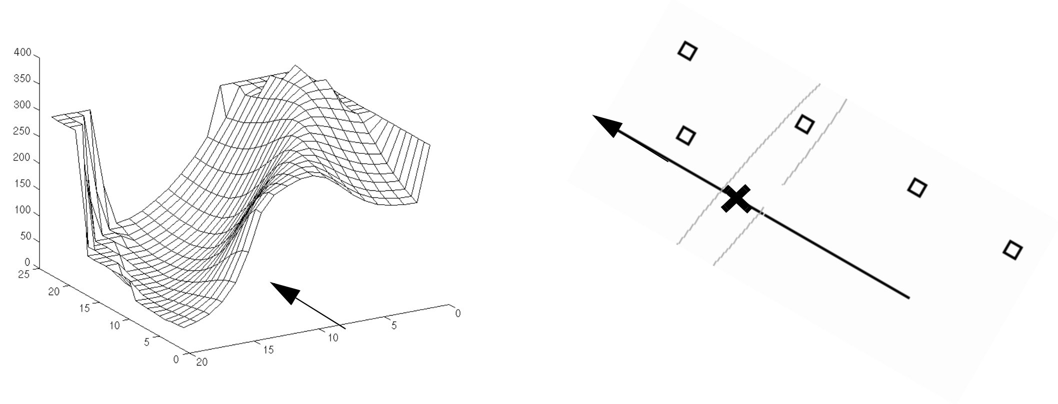


Figure 5.4 Too many buoys have been deployed to the right of the direction of motion. The best position in this case can be found to the left.

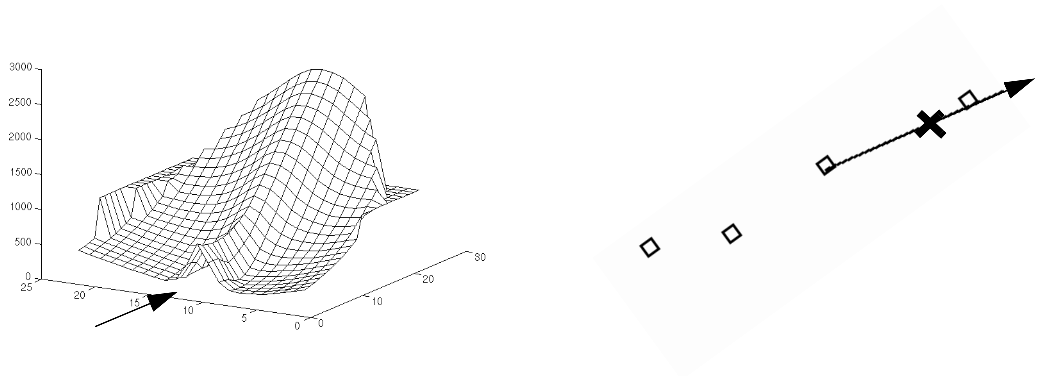


Figure 5.5 The buoys are placed almost on a line. The only possible positions for the new buoy are perpendicular to the direction of motion.

In most cases the situation will look like that shown in Figure 5.3 or Figure 5.4. The algorithm based on the theory in chapter 5.4 would choose among positions with $G(r_i)<\delta$ (Figure 5.2) and position the buoy at the hearing range in the direction of motion. This works well until we reach the situation in Figure 5.5 where the only possible buoy position is perpendicular to the direction of motion. We could let the system behave in this way, repeatedly having to “save” the situation, but this is an unstable process and the buoy consumption is higher than the one that can be achieved with a slight modification: we simply forbid buoy positions within a constant angle from the direction of motion. No optimization has been done but with an angle around 15 degrees the unstable behaviour is avoided.

One further adjustment has to be done to the algorithm because dense sampling of the function over the entire area of interest is too time consuming with the computer available to us. If the originally two-dimensional problem could be approximated by a one dimensional one, a lot of computer time would be saved. The question is how to choose this one dimensional manifold of possible buoy positions. The following ad hoc solution has been

found to work well in practice.
Simulations like those in Figure 5.3 to Figure 5.5 indicate that the angle to the buoy position is the factor of dominating significance for the uncertainty reduction. Therefore the search is restricted to a curve:

$$r(\varphi) \qquad \varphi \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \tag{5.16}$$

where φ denotes the angle to the direction of motion.

One could let r(φ) be a constant (=the hearing range of the buoy), but to get use for the buoys for a longer time, the radius is successively reduced from r(0)=hearing range to

$$r\left(-\frac{\pi}{2}\right) = r\left(\frac{\pi}{2}\right) = \frac{\text{hearing range}}{2} .$$

With a faster computer and an optimized program design this modification would probably not be necessary but with this modification the system has proven to work satisfactory.

5.5 Changing the behaviour of the tracker

A few parameters can be adjusted to change the behaviour of the tracker:

(1) The tolerance δ of the maximum uncertainty of a measurement in the predicted area. When this value is exceeded a new buoy must be deployed in such a way so that the uncertainty is reduced (<δ). By increasing this value the tracker consumes fewer buoys but the position estimation will suffer. A value around 300 has been used. A future development is suggested in chapter 9.

(2) One possible tracking strategy alternates the direction where to look for a buoy position. With this strategy, a new buoy must always be deployed on the opposite side from the one. This strategy mimics the idealised tracker described in 7.1, but it does not work well in practice.

(3) The minimum number of buoys within hearing range of the target has been set to four but increasing this value would result in a more precise, robust, and buoy consuming tracker.

6 Regaining contact

The game will start from a point in time and a position for an observation of the submarine. At the later time for the tracking to start, the travelling distance for the submarine can be estimated given its speed. The speed is unknown but one can assume that it is low since it is operating under sound emission limits. The following strategy is being used in the program:

- *dt* = time difference between the observation and the start of the tracking
- s_i = the assumed speed of the target in the first attempt
- b_i = the approximate buoy range when the target is travelling with speed s_i
- $r_i = s_i d_i$ is the travelling range of the submarine given its speed s_i

On a radius = r_i around the observation the buoys are deployed at distances < $2b_i$
This continues until one of the buoys indicates a detection. Around this buoy a few more buoys are deployed in a circle so as to get at least four hearing buoys.

If no detection is made the procedure is repeated assuming the speed $s_{i+1}=2s_i$.

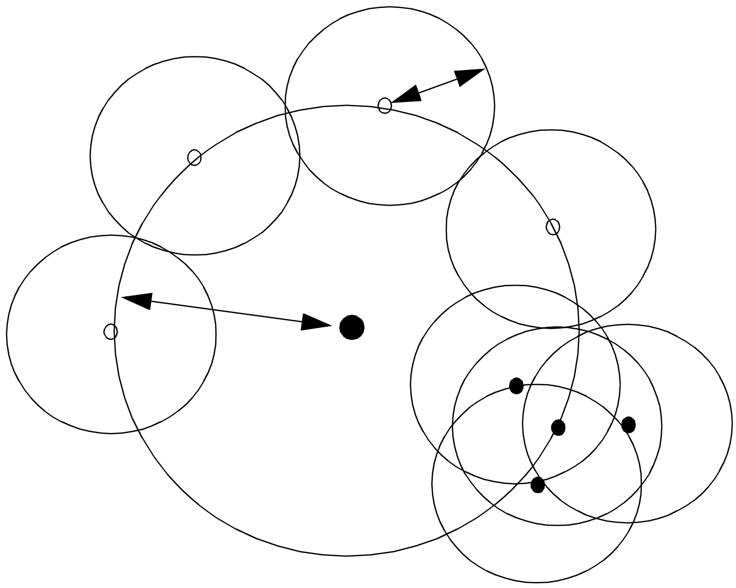


Figure 6.1 If the target moves with the assumed speed without manoeuvring it will reach the big circle and the buoys will have the assumed range.

The critical point in this strategy is the time difference *dt.* If it is too long, the assumption of constant direction of motion and speed of the submarine will be unrealistic. The error in the speed estimation will also have an increasingly negative effect on the estimated travelling range as *dt* gets bigger.

7 Model validation

7.1 An idealized tracker

Is there an ideal tracking strategy? In at least one situation we can make an attempt to construct an optimal buoy configuration. Consider a target following an infinite straight line at constant speed. Let the buoy range be known exactly. A balance between the two conflicting demands of optimal tracking quality and longest possible tracking time has to be made. Given a lower limit for the tracking quality, how should the buoys be deployed in order to spend as few as possible? How should this lower limit be chosen?

A theoretical limit of the number of buoys that have to be deployed can be constructed as follows.

If the buoy range R is known and at least four buoys within hearing range are desired, the following linear configuration can be used to find the lowest possible buoy consumption:

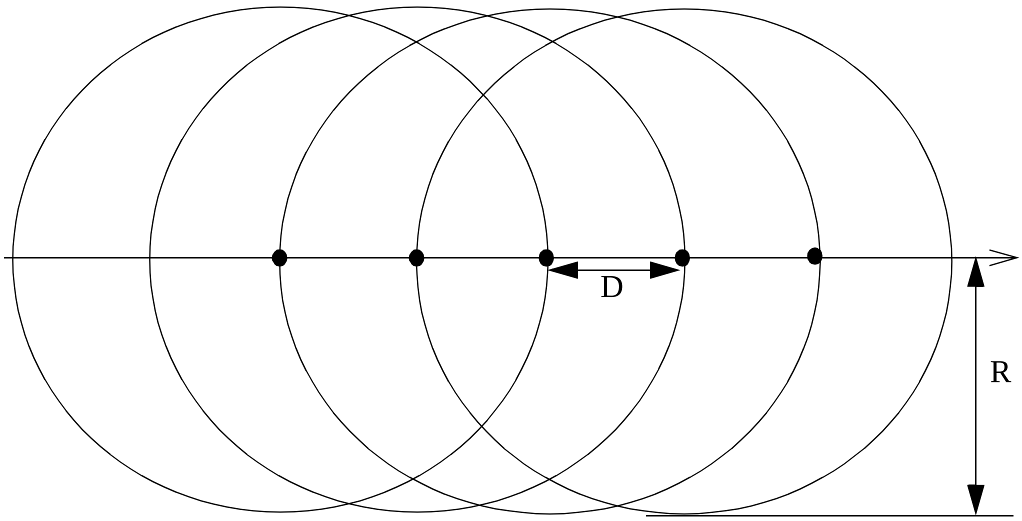


Figure 7.1 The theoretically minimal buoy consumption is reached with this configuration.

By requiring that a "hearing circle" with radius *R* must always contain at least four hearing buoys wherever the sub is along its linear path, one gets the following maximum buoy displacement (as can easily be seen from Figure 7.1):

$$D = \frac{R}{2} \quad (7.1)$$

This value of D will result in an infinite measurement error but it serves as an upper limit: it is not possible to consume fewer buoys if four buoys are required to hear the target all the time.

How close to this value can we get? If the buoys are displaced the distance d from the line of motion the following situation is obtained:

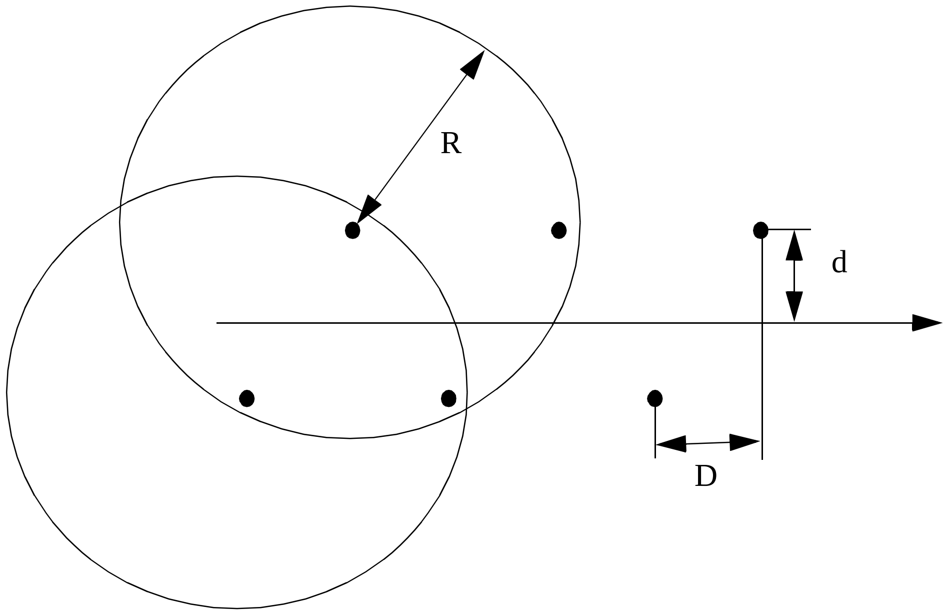


Figure 7.2 The buoys are displaced the distance d from the line of motion to get a usable configuration.

To guarantee that a minimum of four buoys are within hearing range from the target, D is chosen as

$$D \le \alpha\sqrt{R^2 - d^2} \tag{7.2}$$

In Figure 7.3 the length of the major axis in the confidence ellipse is plotted against the percentage of the theoretical maximum tracking distance for two values of the parameter α (α=1 and α=0.8).

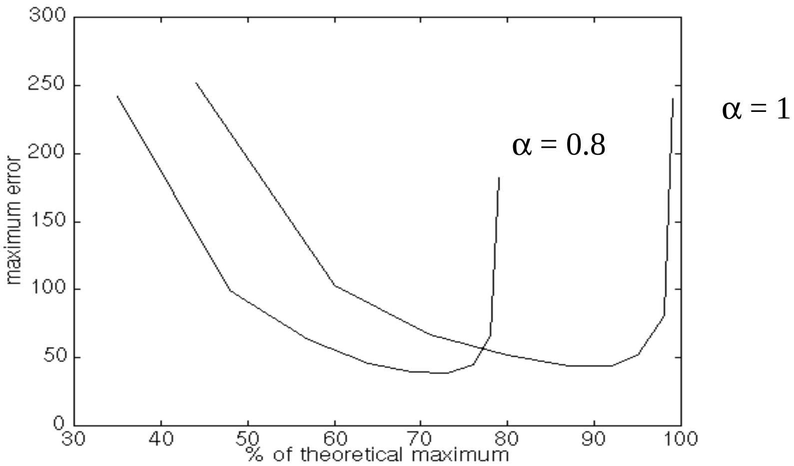


Figure 7.3 The uncertainty in the measurement vs. the reduction in efficiency compared to the theoretical limit.

If we choose to deploy the buoys as far away as possible (α=1) and also choose the displacement d that results in the smallest major axis of the confidence ellipse, the number of buoys consumed will only be slightly higher than the limit value!

Figure 7.4 shows the appearance of the confidence ellipse for different values of the displacement *d* (from left to right d=0.9R,..,0.1R).

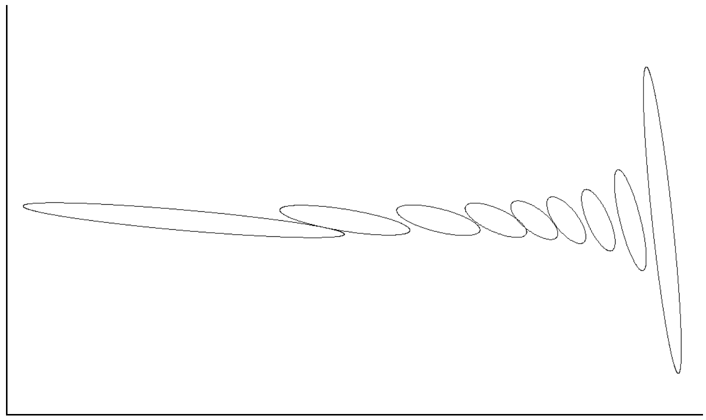


Figure 7.4 The shape of the confidence ellipses for different d. From left to right d=0.9R,..,0.1R.

Now we can compare the performance of the buoy algorithm vs. our "optimal" pattern in this specific situation:

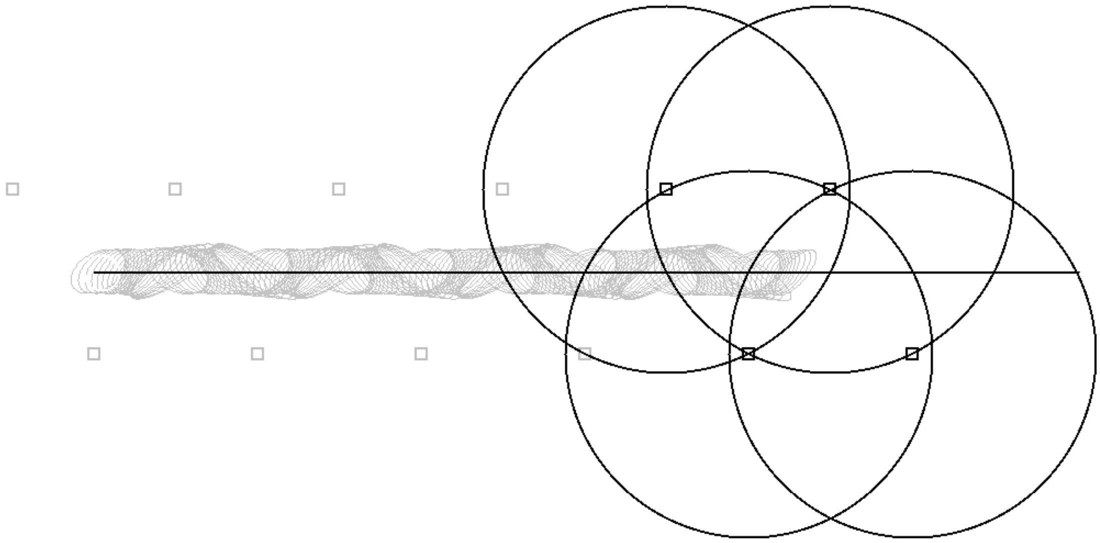


Figure 7.5 Optimal pattern.

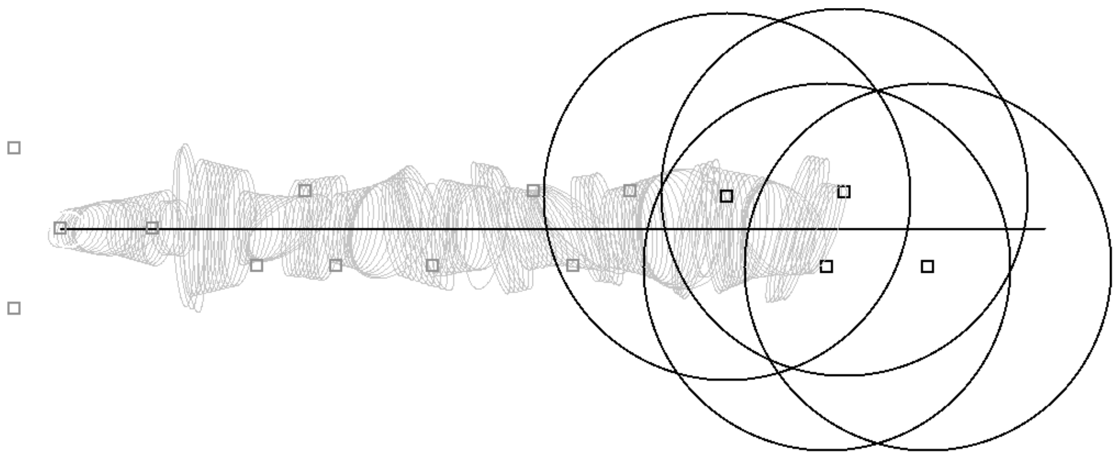


Figure 7.6 Pattern from the algorithm in a model without noise.

To be able to compare the optimal pattern with the algorithm the noise has been removed from the simulation in Figure 7.6.

Since the deployment algorithm always takes into account that the submarine might do a maximum turn in the next moment it can cope with more potential manoeuvres than the optimal one.

The reason for the “missing” buoy every fifth step is the second requirement stated in section 5.4: both the chosen position and the “missing” position satisfy the tolerance requirements. If only the first requirement in chapter 5.4 were taken into consideration the “missing” position would have been chosen since it results in a smaller error, but because of the second requirement the other position is chosen instead.

7.2 Simulation analysis

The first two pictures are examples of how the balance between the two requirements in chapter 5.4 can be chosen. The speed is 4 knots and the manoeuvre corresponds to an acceleration of approximately 0.04 m/s^2.

In Figure 7.7 the variance in the buoy position is small relative to the distances between the buoys and the manoeuvre is easily coped with.

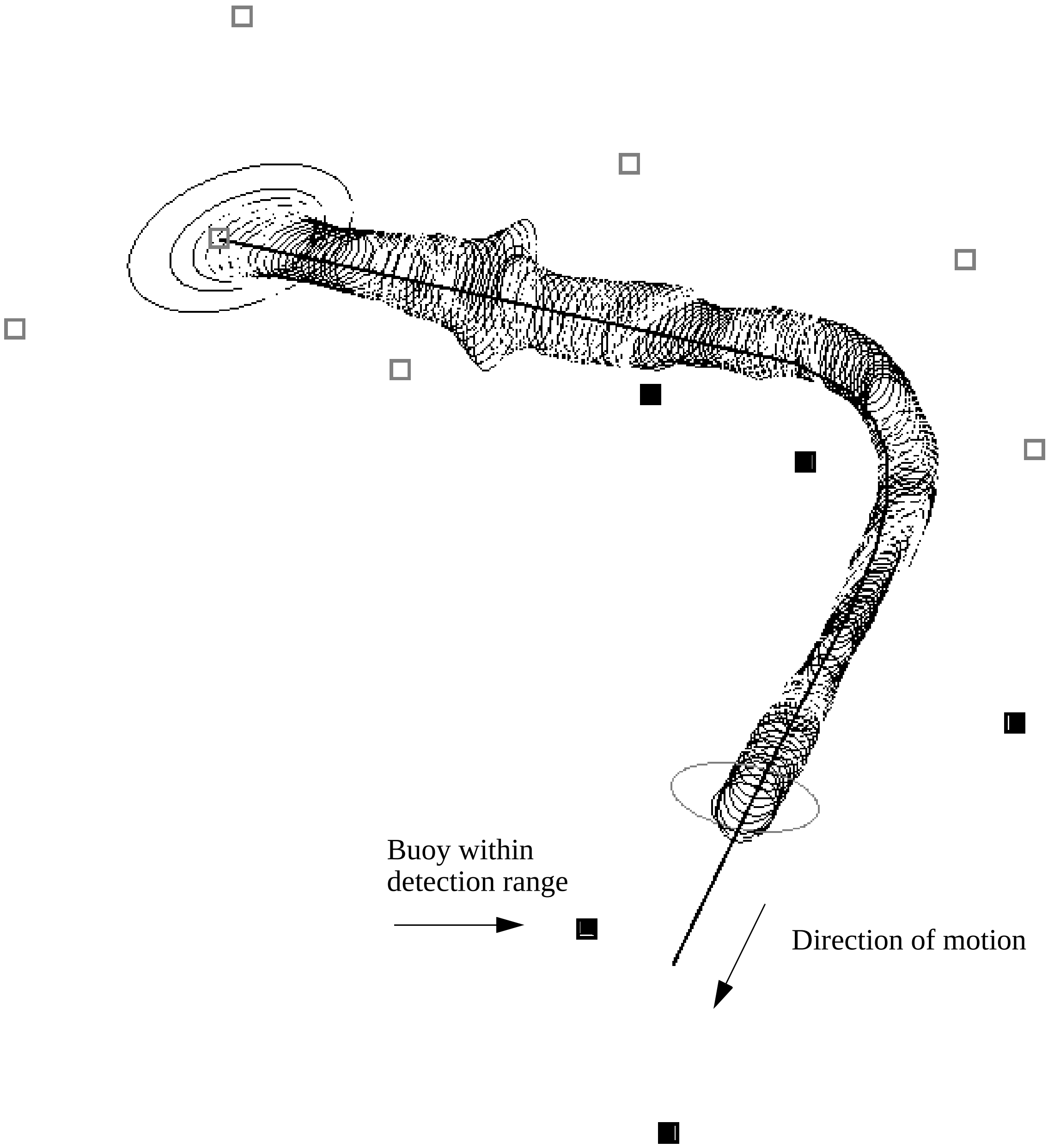


Figure 7.7 Target following a predefined path. Buoy range 350-400 meters.

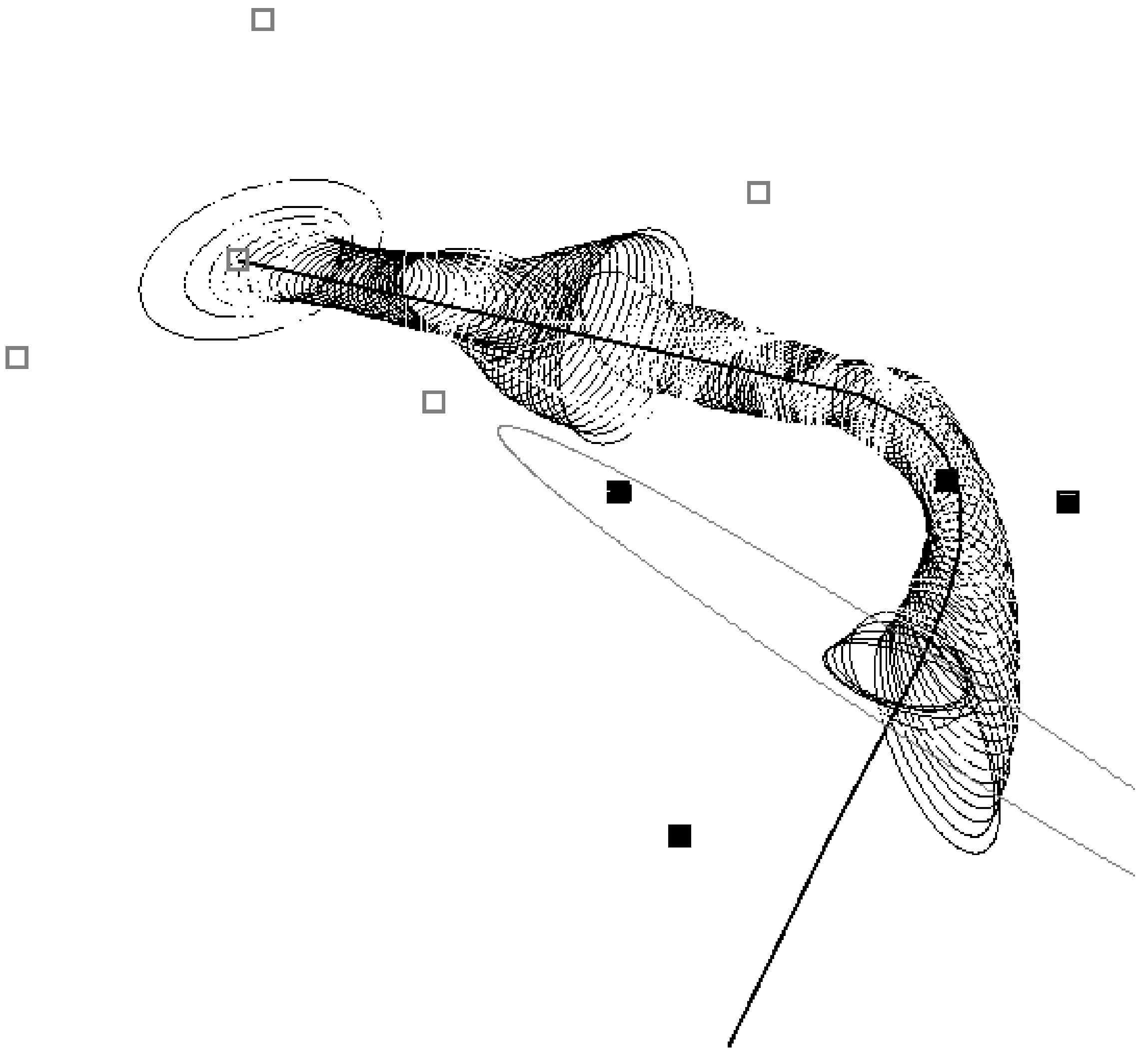


Figure 7.8 Target following a predefined path. Buoy range 350-400 meters. High tolerance for the measurement confidence ellipse.

In Figure 7.8 a simulation is made with a high value for the tolerance of the measurement uncertainty. The buoy consumption is reduced compared to the case shown in Figure 7.7 but the position estimate of the target is less precise. The Kalman filter reduces the uncertainty however.

In Figure 7.9 to Figure 7.11 the source level is successively reduced in order to find a minimum buoy range for the tracker to work properly. The acceleration in the manoeuvre in these pictures are 0.08 m/s^2.

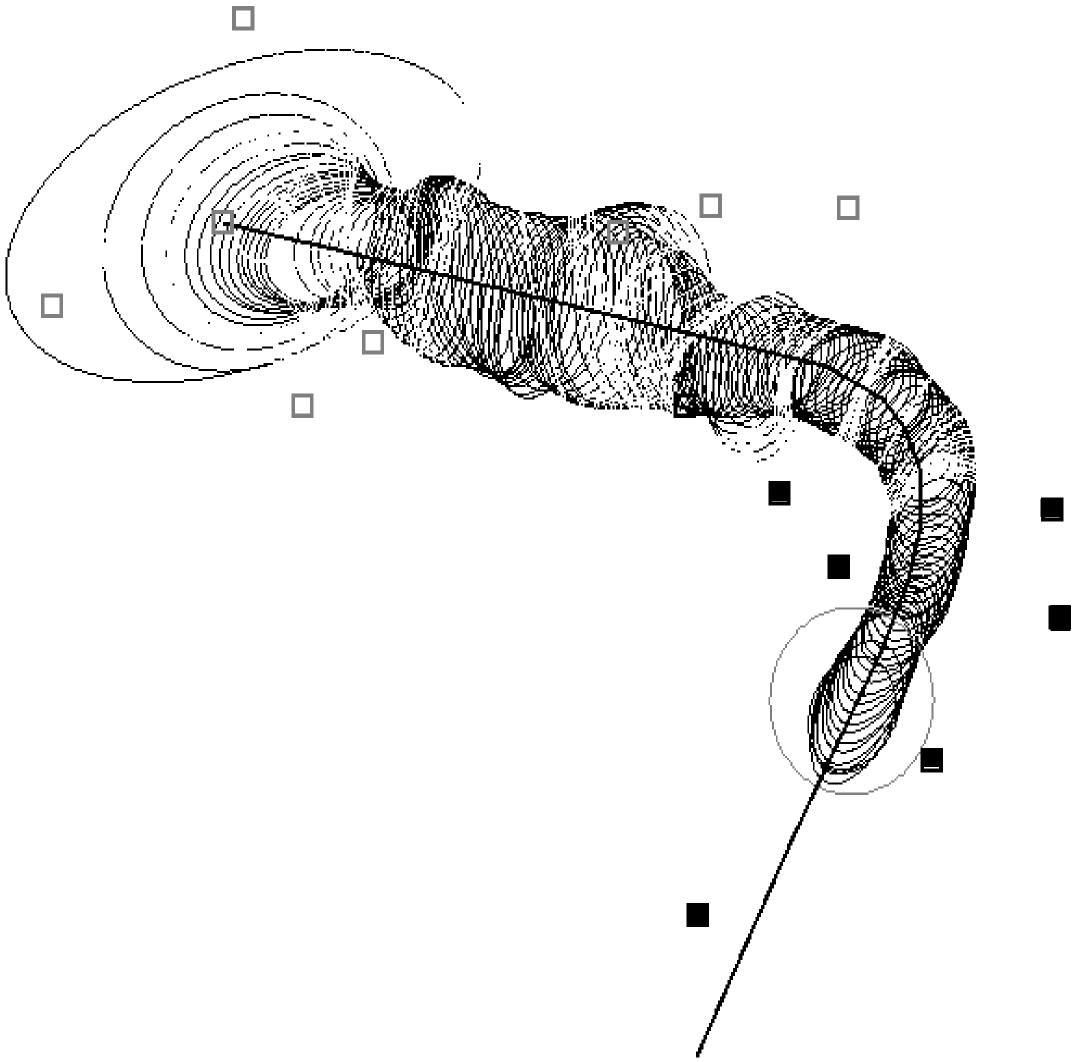


Figure 7.9 Target following a predefined path. Buoy range is around 130-160 meters

In Figure 7.9 the detection range of the buoys is reduced to 130-160 meters. The variance of the buoy positions starts to affect the tracking quality but the manoeuvre is still easily handled.

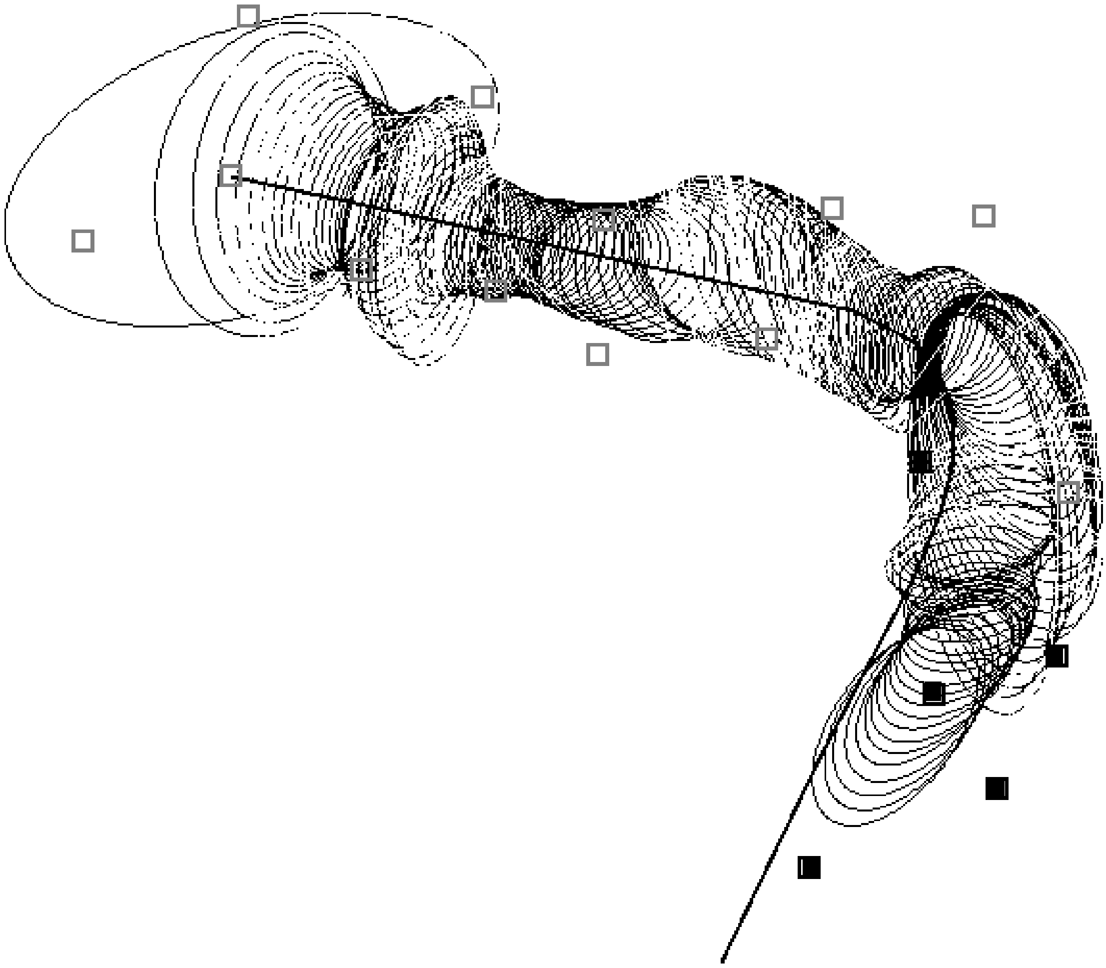


Figure 7.10 Target following a predefined path. Buoy range is around 110-130 meters.

With a buoy range around 110-130 meters the tracker has problems to follow the manoeuvre. The variance in the buoy positions starts to affect the accuracy.

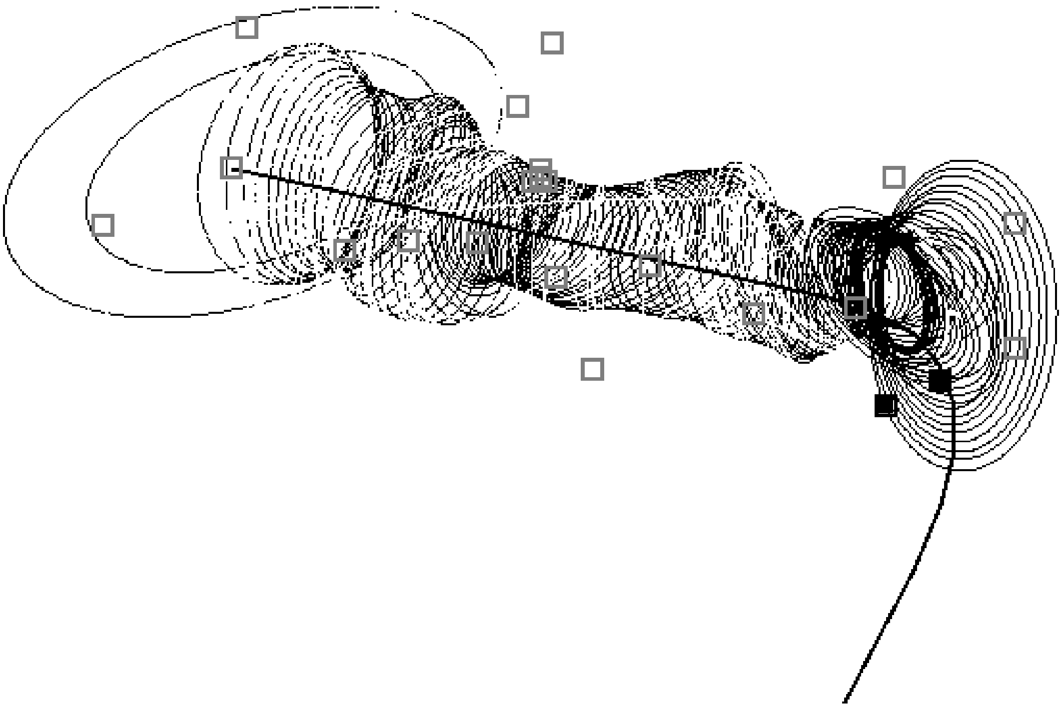


Figure 7.11 Target following a predefined path. Buoy range is around 80-110 meters.

In Figure 7.11 the tracker can no longer follow the sharp manoeuvre of the target.

8 Conclusions

We have developed a simplified two dimensional model without islands and bottom structure and *all conclusions below are related to the model and not to reality.*

The model is based on the application of simple but well-established physical models, and the substitution of numerical values for the model parameters has been done in cooperation with domain experts. Chances should therefore be good that our conclusions will be valid also if the model were applied to real sonobuoy data, but this certainly remains to be demonstrated.

Thus, in the model's world:

- it is possible to track a hostile submarine with sonobuoys;
- with four or more sonobuoys in advantageous positions the submarine's position can be estimated and a confidence ellipse for this position can be calculated;
- from a Kalman filter not only a prediction of the motion is achieved, but also a reduction of the uncertainty in the same. The more one knows about the dynamic properties of the target the larger reduction can be obtained;
- by determining the uncertainty of future measurement one can position the buoy in an advantageous way;
- the critical aspect of the simulation is the length of the time period from the initial observation until contact has been regained.

In the simulator we have used a standard deviation of 15 meters in the positioning of the buoys but we do not know with certainty if this is realistic. With this value however, the tracking works fine down to a buoy range around 100 meters.

Before the technique can be tested in a real submarine tracking scenario, one needs to find solutions to the following problems:

(1) computing on-line the TDOA for each interesting pair of sonobuoys;

(2) determining on-line good estimates for the location of each sonobuoy, at least for testing purposes; this could be done for example by measuring the sound travel time to each sonobuoy from three or four underwater sonic beacons whose positions are known.

9 Future work

9.1 Short-term improvements

The current versions of both the simulator and the path editor compute all positions with respect to a relatively coarse pixel grid. This approach causes problems due to discretization errors when specifying small speed changes with the path editor and when tracking silent submarines with the simulator. The pixel grid should only be used for purposes of data input and visualization. All positions and speeds should be represented as floating point numbers.

The path editor should be equipped with an acceleration estimator and limiter so that the user can easily avoid creating tracks which correspond to unwanted accelerations.

The algorithm used for solving the least-squares problem was taken from a C freeware library on the Internet. Its quality was later found to be inadequate but prevailing priorities did not permit its replacement. Thus, the algorithm should be replaced by an Eiffel implementation of any of the standard numerical algorithms available for this problem.

The program could create a speed vs. source level table on-line, based on information gathered during tracking. Every buoy registers the noise intensity and by making use of the sonar equation and the estimated speed, a table could be constructed and gradually refined.

The model's treatment of signal integration and buoy deployment time is coarse and should be improved, or at least further analysed.

In this version it has not been necessary to predict the time and position for the next deployment more than one simulation step ahead but in a final version this strategy could be changed. Instead of being satisfied with establishing that no buoy has to be deployed at time t+T, one could continue to check time t+T*i, i=1,2,... to determine when and then where the next buoy is likely to be deployed. This would give the platform more time to prepare for the next deployment. The position and time would then be revised as time passes. The final position and time for deployment would however be the same as in this version.

The NIS test in chapter 4.4 could perhaps be used to tune the filter during tracking. If one notices that the NIS value gets too big, it is time for an adjustment of the parameters. Perhaps an automatic algorithm that constantly makes small adjustments of the parameters could be developed.

Related to such parameter tuning is the possibility for further development of the tracking technique by use of so-called parallell Kalman filters [2] which would probably allow considerably improved position estimates in regions where the submarine has constant speed and heading, while still permitting the tracking of a maneuvering target.

The program is prepared for different values of the tolerance of the uncertainty in the predicted area (1) when a buoy is to be deployed and (2) when choosing a position for the buoy.

A high tolerance of (2) often results in a position close to the direction of motion due to requirement (2), chapter 5.4, whereas a narrow tolerance of (2) will in the limit result in a position fulfilling requirement (1) in chapter 5.4 only. How to choose the value has not been investigated in detail and it is therefore set to the same as δ (chapter 5). One possibility could perhaps be to choose the value as a function of the smallest $G(r_i)$ and δ.

9.2 Topics for future research

A deeper study of the problem may presumably result in a further improved method to achieve the global goal. In the two dimensional simulation described in this report, a prediction in the longer perspective is not meaningful but in the real world of the archipelago the situation will be different. If one fuses information about prior known submarine behaviour, islands, sea depth, possible missions and so on with the observed behaviour a better strategy might be possible.

From a theoretical viewpoint, one would prefer an approach to the modelling problem based entirely on the theory of stochastic processes. This would probably lead to increased complexity when solving the buoy placement problem but, assuming this complexity can be mastered, could improve the fidelity and robustness of the model.

When tracking those very silent submarines which may be deployed in the future, the distances between buoys and target may need to be so small that the target can no longer be represented by a point-like sound emitter. This will complicate the position determination considerably, since the problem can no longer be reliably treated as two-dimensional when the distance between buoy and submarine becomes comparable to both the length of the submarine and the depth of the archipelago. It is not obvious that the technique can be extended to handle this case. To do this, new research ideas will have be developed.

On the other hand, delivery platform, sensor, energy source, and communication system developments will probably allow precision deployment and networking of large numbers of very small and cheap measurement devices in the future.

10 References

[1] Brian D. O. Anderson, John B. Moore. 1979. *Optimal filtering*. Prentice Hall Inc.

[2] Yaakov Bar-Shalom, Xiao-Rong Li. 1993. *Estimation and tracking: Principles, Techniques and Software*. Artech House, Inc.

[3] Catarina Blixt. 1991. *Felanalys vid lokalisering av hydroakustiska transienter.* FOA Rapport C 20814-2.2.

[4] W.S Burdic. 1991. *Underwater acoustic system analysis*. Prentice Hall Inc.

[5] Patrik Johansson. 1995. *Målföljning genom fusion av data från radar och IR-sensor.* FOA-R--95-00210-3.4--se.

[6] Navy International, September 1990, 307-312.

[7] Göran Neider. 1997. *Ubåtsföljning med passiva sonobojar. Användarhandledning till program Subtrack.* FOA R-97--00447-505--SE.

[8] Jean-Marc Nerson, Bertrand Meyer, Dino Valente. 1995. *EiffelCase*. ISE Technical Report TR-EI-53/EA.

[9] Rickard Sjögren. 1990. *Lokalisering av hydroakustiska transienter.* FOA Rapport C 20813-2.2.

[10] Harold W. Sorenson. 1985. *Kalman Filtering: Theory and Application*, IEEE Press.

[11] M. Swift, W. Elliston, M. Sandys-Wunsch. 1996. *Tracking weak targets with a grid search approach using spatially distributed sensors*. First Australian data fusion symposium. IEEE, Inc.

[12] Alexander Wahlstedt, Jesper Fredriksson, Karsten Jöred and Per Svensson. 1997. *Submarine Tracking by Means of Passive Sonobuoys*. FOA-R--96-00386-505--SE.

[13] The Analytic Sciences Corporation. 1974. *Applied optimal estimation.* The M.I.T Press.

[14] Karl Johan Åström. 1968. *Reglerteori*. Almquist & Wiksell.

Appendix A Functionality of the program

A target path in shape of a piecewise linear polygon can be constructed using a separate program, the Path Editor[12].

After a predefined path has been loaded, the simulation can begin. A circle indicates the position of the target at time 0. Now it is possible to deploy simulated sonobuoys at arbitrary positions in order to try to determine the position of the target. When four or more buoys are within the detection range of the target and form a sufficiently favourable configuration a confidence ellipse is displayed.

The objective is to track the target with a certain quality, i.e. with the major axis of the confidence ellipse smaller than a given value. Several options are available:

- Viewing the predicted area. When this alternative has been selected, the program displays the predicted future position of the target in the shape of a confidence ellipse (see chapter 3).
- Viewing the range of the buoys. The detection range of the buoys will be estimated and then displayed as a circle around active buoys.
- Viewing the target path.
- Showing buoy information: number of available, deployed, active and dead buoys is displayed.
- The freeze option stops the time but the simulation continues, using the latest target source level (chapter 2.2).
- Activation of the optimal buoy deployment. The buoys are deployed automatically when necessary and the position is chosen to be optimal, i.e. a new buoy should add sufficiently accurate information to the system for as long time as possible.
- The buoys can also be deployed manually in different patterns such as pointwise, random, square, triangle etc.
- A score is calculated in the following manner: The number of simulation steps with zero, one, two and three hearing buoys are registered.
  If four or more buoys are hearing the target an evaluation of the confidence ellipse is done as follows: If the major axis α is larger than a desired maximum ρ a tracking quality variable is increased with a value $\left(1 - \frac{\rho}{\alpha}\right) \in [0, 1]$. The value presented as score is the tracking quality variable divided with the number of simulation steps with four or more hearing buoys and $\text{score} \in [0, 1]$. 0 corresponds to perfect tracking and 1 to infinitely bad tracking.

Appendix B Representation and handling of uncertainties

The following uncertainties are recognized in the program:

(1) The uncertainties in the measurement of the TDOA's (chapter 1.1). The true TDOA's are perturbed by white noise.

(2) The uncertainty in the position of the buoys. The desired position of the buoys are perturbed by two dimensional white noise.

(3) The estimated variance of (1).

(4) The estimated variance of (2).

There are two levels in the program, the underlying world and the model of the underlying world.

- (1) and (2) above exists in the underlying world. The positions and TDOA's are not stochastic, but they have been perturbed with white noise.
- In the model, all calculations are based on stochastic variables using estimations of the variances in the TDOA's and the positions in the underlying world (3), (4).

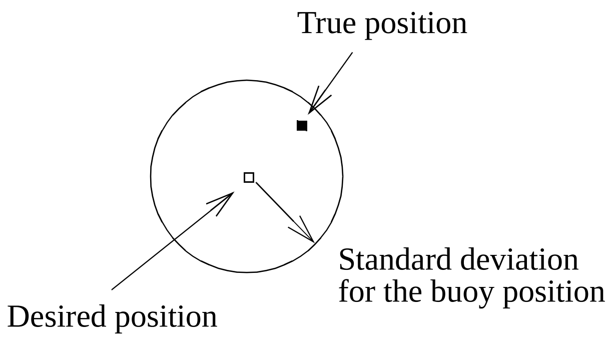


Figure B.1 All calculations are made on the desired position regarded as a stochastic variable with known variance. The TDOA's are calculated from the true positions.

Appendix C The initialization file

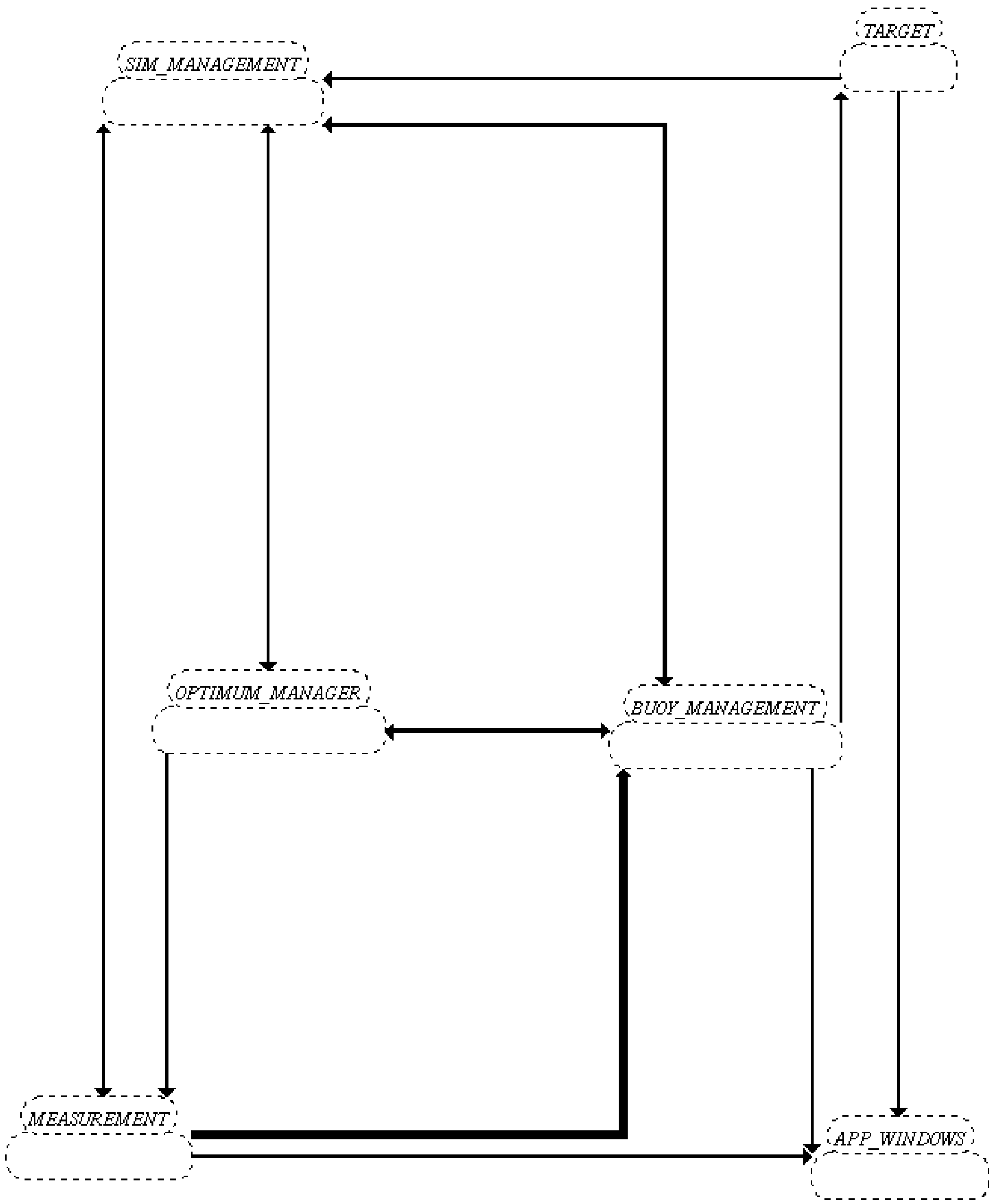
Parameter values are given in SI units when possible.

```
SubTrack.Init -- SubTrack.Init, 2.1, 96/10/17, 09:37:56 --fall 2
1996-10-23--1 parameters for elliptic sound picture
1996-09-26--1 DT-REF 1->8
1996-09-09--1 Created from subtrack.ini
------------------------------------------------------------
COLORS
Red             -- TargetColor
Gold            -- Error_Ellipse
Coral           -- FrameColor_name =>bg 1000 m, tail to target, point in buoysquare,erase
MediumBlue      -- SeaColor
LightSlateBlue  -- BuoyColor2 active and out of range or passive
White           -- PathColor
Yellow          -- BuoyColor active and within range
NavyBlue        -- Color
DarkOliveGreen -- DeployMenuColor
LimeGreen       -- BuoyColor3
Gold            -- SubMenu_Color_name
------------------------------------------------------------
SIMULATION_PARAMETERS
0.95 -- probability
3    -- scale
2    -- simulation stepsize
------------------------------------------------------------
SONAR_EQ
0         -- DI
20        -- TL_K   : K in expr: TL=K log r
-10       -- DT_REF : Detection Threshold (reference)
          -- Use Burdic equ 15-15 and fig13-10 (Bw 1000 Hz, Pfa=0.0001, 4 sec integration time Pdet=0.99)
68        -- NL     : Noise Level
{0,105} {2,120} {5,138} {10,150}$     -- SL     : Source Level {vel,SL(vel)}
------------------------------------------------------------
BUOY_PARAMETERS
15        -- StandardDeviation (position)
0.001     -- StandardDeviation (time difference of arrival)
4         -- IntegrationTime
0         -- ActivationTime
1000      -- CarrierSpeed
1000      -- AvailableBuoys
1000      -- ActionTime
1450.0    -- SpeedOfSound
------------------------------------------------------------
TARGET_SOUND_EMISSION_PARAMETERS
1         -- value for the big axis in elliptic sound picture from target
1         -- value for the small axis
0         -- displacement of ellipse
------------------------------------------------------------
OPTIMIZATION_PARAMETERS
400       -- big axis tolerance
10        -- the number of steps in the angle optimization
400       -- tolerance when positioning a buoy
------------------------------------------------------------
TARGET_MOTION_PARAMETERS
0.0004    -- linear_acceleration variance
0.0016    -- centripetal_acceleration variance
------------------------------------------------------------
```

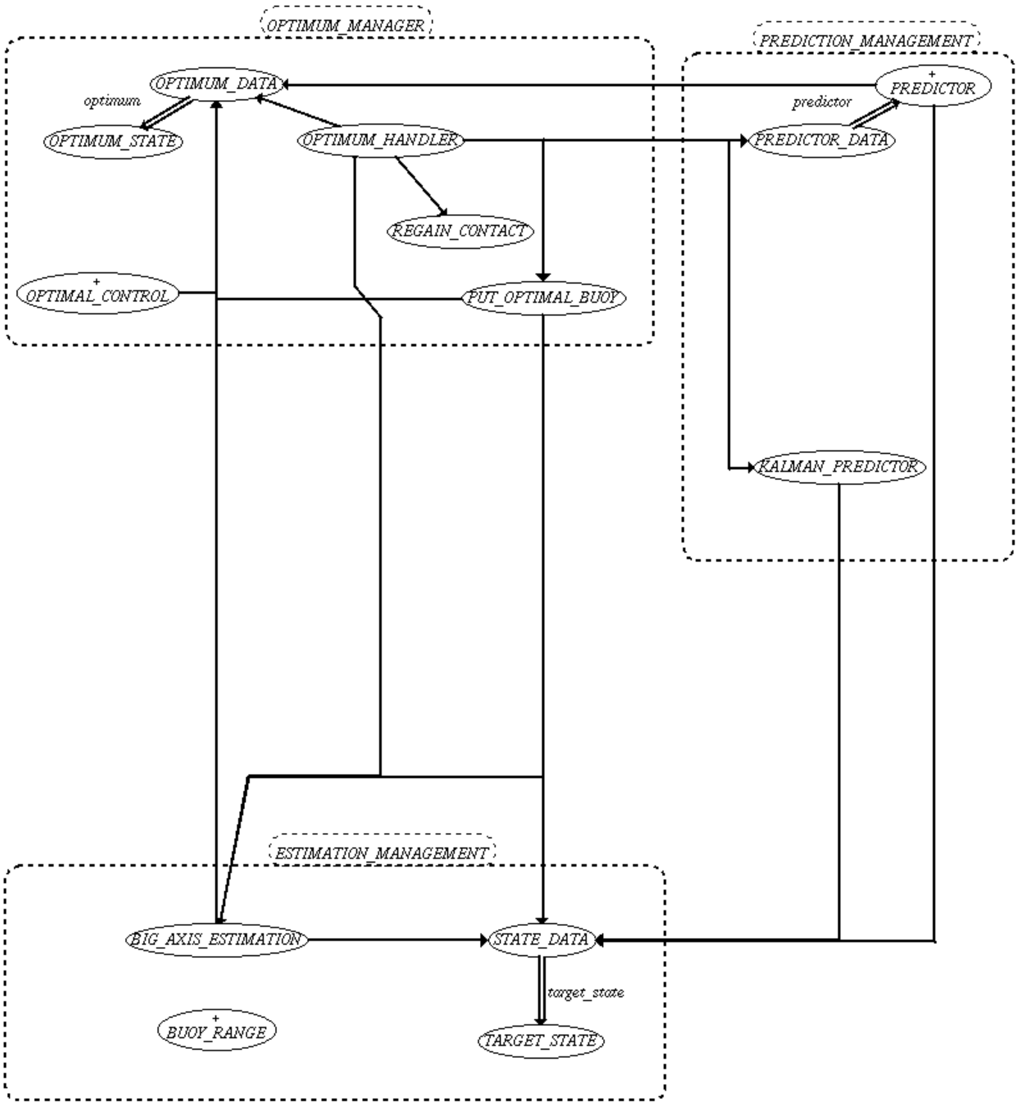
Appendix D System architecture

The program contains more than 13000 lines of code and it is not possible to depict more than selected portions.

The following architecture diagrams are made using the BON format [8].



The BON diagram describing the clusters involved in the algorithm for optimal buoy deployment is:



Parts of the implementation:

D.1 OPTIMUM_HANDLER

This class is the heart of the optimization of buoy positions. It is more or less the pseudo code in chapter 3 implemented in Eiffel.

```
class OPTIMUM_HANDLER -- %M%,%I%,%E%,%U%
inherit
	PREDICTOR_DATA
	OPTIMUM_DATA
	BIG_AXIS_ESTIMATION
	BUOY_CLUSTER
	SETUP_DATA
	PUT_OPTIMAL_BUOY
	KALMAN_PREDICTOR
	REGAIN_CONTACT
	TIME_DATA
	TARGET_DATA
	OBSERVATION_DATA
	SCORE_DATA
creation make_optimum
--------------------------------------------------------------------------------
-- This class is the heart of the automatic control mode
--------------------------------------------------------------------------------------------

feature{START}

	keep_score: BOOLEAN

	optimum_controller(t: INTEGER) is
	   local gamma: DOUBLE
		dt : INTEGER
		put_buoy : BOOLEAN
		cluster : LINKED_LIST[BUOY]
	   do
		cluster:=get_buoy_cluster
		if cluster.count>3 then optimum.is_predictable end -- At least four buoys are needed to start the prediction
		if time.get_time > 10 then keep_score:=TRUE end
		if path.path_exists and then optimum.is_placing_optimal then
			if cluster.count<3 then
				search(time.get_time-optimum.get_last_contact_time,optimum.get_last_pos)
				-- If in optimal deployment mood the contact should be lost, this routine tries to regain the contact
				optimum.set_last_lost(time.get_time)
			end
		end
		if optimum.predictable and not time.iced then
				kalman_handler   -- The Kalman filtering process
				if optimum.is_placing_optimal then
					gamma:=estimate_in_predicted_area
					if (gamma > settings.delta or gamma = 0 or cluster.count<4) and cluster.count<9 and
					    optimum.get_last_lost+settings.simulation_step<time.get_time then
					       put_optimal_buoy
					       -- If in the predicted area, the next measurement error will get too big -Deploy a buoy
					end
				end
```

```
                end
            end
        if cluster.count>3 then
            optimum.set_last_pos(observations.estimated_coord)
            optimum.set_last_contact_time(time.get_time)
        end
        if keep_score then
              score.update_score(settings.simulation_step+time.get_time) --number of steps in simulation. should be
changed
        end
    end


end -- class OPTIMUM_HANDLER
```

D.2 KALMAN_HANDLER

The following features control the Kalman filtering algorithms:

```
    kalman_handler is
        local   cluster: LINKED_LIST[BUOY]
                t:INTEGER
        do
                cluster:=get_buoy_cluster
            if cluster.count>3 then
                t:=settings.simulation_step
                kalman_est
                kalman_pred(t)
                target_state.re_set_dead_count
            else
                target_state.inc_dead_count
                io.putint(target_state.dead_counting) io.new_line
                kalman_pred(target_state.dead_counting)
            end
        end

    kalman_pred(t:INTEGER) is
        local   pp,q,xp,f: MATRIX
                pred_area: ELLIPSE_INFO

        do
            f:=create_f(t)
            q:=create_q(t)
            pp:=f*target_state.pe*f.tr+q
            xp:=f*target_state.xe
            target_state.set_pp(pp)
            target_state.set_xp(xp)
            set_predicted_area
        end

    kalman_est is
        local j:INTEGER
               s,k,z,pe,xe,I,r,temp:MATRIX
        do
            !!z.make(2,1)
            !!I.make(4,4)
```

```
        z.put(observations.estimated_coord.x-target_state.xp.item(1,1),1,1)
        z.put(observations.estimated_coord.y-target_state.xp.item(2,1),2,1)
        from j:=1 until j>4 loop I.put(1,j,j) j:=j+1 end
        r:=create_r_matrix
        s:=(target_state.h*target_state.pp)*(target_state.h.tr)+r
        k:=(target_state.pp*target_state.h.tr)/s
        pe:=(I-k*target_state.h)*target_state.pp*(I-k*target_state.h).tr+k*r*k.tr
        xe:=target_state.xp+k*z
        target_state.set_xe(xe)
        target_state.set_pe(pe)
    end
```

D.3 BIG_AXIS_ESTIMATION

The following features estimate the major axis in the predicted area:

```
set_orig (b_list: LINKED_LIST[BUOY] coord: COORD_XY_FIG) is
    local n: INTEGER
        sum_x,sum_y,sum_d,abs_dist: DOUBLE
    do
        from
            b_list.start
        until
            b_list.off
        loop
            abs_dist:=sqrt((b_list.item.get_xpos-coord.x)^2 +(b_list.item.get_ypos-coord.y)^2)
            if b_list.item.range>abs_dist then
                sum_x:=sum_x+b_list.item.get_xpos
                sum_y:=sum_y+b_list.item.get_ypos
                sum_d:=sum_d+abs_dist
                n:=n+1
            end
            b_list.forth
        end
        x_orig:=sum_x/n
        y_orig:=sum_y/n
        d_orig:=sum_d/n
        optimum.set_hearing_buoys(n)
    end


---------------------------------------------------------------------------
--uses c routines to predict an error ellipse that the present buoy configuration
--will give us if the target would be in pos coord.
--returns the big_axis value
---------------------------------------------------------------------------

estimate_in(coord: COORD_XY_FIG): DOUBLE is
    local cluster : LINKED_LIST[BUOY]
        A : ARRAY2[DOUBLE]
        x,b : ARRAY[DOUBLE]
        i,n : INTEGER
        xi,yi,di,abs_pos :DOUBLE
    do
        cluster:=get_buoy_cluster
        set_orig(cluster,coord)
```

```
            if optimum.nr_hearing_buoys < 4 then
                  result:=0
            else
                  n:=optimum.nr_hearing_buoys
                  from
                        i:=start_index
                        cluster.start
                        !!A.make(n,3)
                        !!b.make(start_index,n)
                  until
                        cluster.off
                  loop
                    abs_pos:=sqrt( (cluster.item.get_xpos-coord.x)^2 + (cluster.item.get_ypos-coord.y)^2)
                        if cluster.item.range>abs_pos then
                              xi:=cluster.item.get_xpos-x_orig
                              yi:=cluster.item.get_ypos-y_orig
                              di:=abs_pos-d_orig
                              b.put(xi^2+yi^2-di^2,i)
                              A.put(xi,i,start_index)
                              A.put(yi,i,start_index+1)
                              A.put(-di,i,start_index+2)
                              i:=i+1;
                        end
                        cluster.forth;
                  end
                  create_c_data(A.height,A.width)
                  x:=solve1(A,b)
                  predictions.compute_K_matrix(n)
                  predictions.set_estimated_coord(coord)
                  predictions.set_axes(n,settings.probability,coord)
                  predictions.set_orientation(n)
                  destroy_c_data
            result:=predictions.big_axis
            end
      end

-----------------------------------------------------------------------------
-- Calls estimate_in in positions in the predicted area and returns
-- the biggest big_axis value found there
-----------------------------------------------------------------------------
estimate_in_predicted_area : DOUBLE is
      local temp_pos : COORD_XY_FIG
            axis_array:ARRAY[DOUBLE]
            x0,y0,a,b,alpha,i,max_axis,min_axis,av_axis : DOUBLE
            sol_nr : INTEGER
      do
            !!temp_pos
            !!axis_array.make(1,9)
            x0:=target_state.last_pred_ellipse.pos.x
            y0:=target_state.last_pred_ellipse.pos.y
            a:=target_state.last_pred_ellipse.big_axis
            b:=target_state.last_pred_ellipse.small_axis
            alpha:=target_state.last_pred_ellipse.orientation/180*Pi
            from
                  i:=-1 sol_nr:=1
            until
                  i>1
            loop
                  temp_pos.set((x0+i*a*cosine(alpha)).rounded,(y0-i*a*sine(alpha)).rounded)
```

```
            axis_array.put(estimate_in(temp_pos),sol_nr)
            sol_nr:=sol_nr+1
            temp_pos.set((x0+i*b*sine(alpha)).rounded,(y0+i*b*cosine(alpha)).rounded)
            axis_array.put(estimate_in(temp_pos),sol_nr)
            sol_nr:=sol_nr+1
            i:=i+2
        end
        temp_pos.set((x0).rounded,(y0).rounded)
        axis_array.put(estimate_in(temp_pos),sol_nr)
        max_axis:=maximum(axis_array)
        result:=maximum(axis_array)
    end
```