Underwater positioning of an ROV using side-mounted sonars
Underwater positioning of an ROV using side-mounted sonars

Examensarbete utfört i Reglerteknik
vid Tekniska högskolan vid Linköpings universitet
av

Erik Ferm

LiTH-ISY-EX–14/4749–SE

Handledare: Niclas Evestedt
Isak Nielsen
ISY, Linköpings universitet
Micael Derelöv
SAAB Underwater Systems

Examinator: Daniel Axehill
ISY, Linköpings universitet

Linköping, 25 februari 2014
# Title
Positionering av en undervattensfarkost med sidmonterade sonorer

# Author
Erik Ferm

# Abstract
Unmanned vehicles being used more and more for tasks that need to be done in environments that are hard to access, or dangerous for humans. Because the vehicles are unmanned they need some way of conveying information to the operator about where it is located. In some cases visual feedback to the operator might be enough, but in environments with low visibility other techniques are required.

This thesis will address the issue of localization in an underwater environment by means of side-scan sonars and an inertial measurement unit (IMU). It will explore whether it is possible to localize a remotely operated vehicle (ROV) in a known environment by fusing data from the different sensors.

A particle filter is applied to the translational motion of the ROV and an extended kalman filter is used to estimate the vehicles attitude. The focus of the thesis lies in statistical modeling and simulation of the ROV and its sensors rather than in validation and testing in the physical realm.

Results show that a particle filter localization is plausible in environments given varied enough readings. For cases where measurements are similar, such as close to the floor of a pool the filter tends to diverge.

# Keywords
Underwater, localization
Abstract

Unmanned vehicles being used more and more for tasks that need to be done in environments that are hard to access, or dangerous for humans. Because the vehicles are unmanned they need some way of conveying information to the operator about where it is located. In some cases visual feedback to the operator might be enough, but in environments with low visibility other techniques are required.

This thesis will address the issue of localization in an underwater environment by means of side-scan sonars and an inertial measurement unit (IMU). It will explore whether it is possible to localize a remotely operated vehicle (ROV) in a known environment by fusing data from the different sensors.

A particle filter is applied to the translational motion of the ROV and an extended kalman filter is used to estimate the vehicles attitude. The focus of the thesis lies in statistical modeling and simulation of the ROV and its sensors rather than in validation and testing in the physical realm.

Results show that a particle filter localization is plausible in environments given varied enough readings. For cases where measurements are similar, such as close to the floor of a pool the filter tends to diverge.
Acknowledgments

I would like to show my appreciation to all the people that have helped me in any way or form to complete this thesis. My examiner Daniel Axehill for pushing me along and not letting me get stuck in the research phase of the project. Niclas Evestedt for being a supervisor that takes time to help me with all my questions. Isak Nielsen and Micael Derelöv for helping me with information about the ROV itself and earlier work done on it.

I would also like to thank my opponent Per Svennerbrandt for giving me very helpful and constructive criticism. Thus helping me to better understand and improve on my work.

Linköping, February 2014
Erik Ferm
## Contents

Notation ix

1 Introduction 1
1.1 Background ......................................................... 1
1.2 Purpose and objective ............................................. 1
1.3 Related work ....................................................... 2
1.4 Limitations ......................................................... 3

2 Description of physical system 5
2.1 ROV .......................................................... 5
2.2 Sensors .......................................................... 6
   2.2.1 IMU ......................................................... 6
   2.2.2 Sonars ...................................................... 7
   2.2.3 Sonar resolution ........................................... 8
2.3 The environment .................................................. 9

3 State estimation 11
3.1 AUVs and localization ............................................ 11
3.2 Extended Kalman filter .......................................... 12
3.3 Particle filter .................................................... 12
   3.3.1 Marginalization .......................................... 14
3.4 Usage of filters .................................................. 14

4 Modeling 15
4.1 Dynamics ......................................................... 15
   4.1.1 Model of the ROV ....................................... 16
4.2 The map .......................................................... 19
   4.2.1 The pool ................................................... 19
   4.2.2 Digital terrain model .................................... 19
   4.2.3 Octree ..................................................... 21
4.3 Sonars .......................................................... 22
4.4 IMU ............................................................ 24
5 Localization 27
  5.1 Attitude ......................................................... 27
  5.2 Estimation of position ....................................... 30
  5.3 Dimension of filters ........................................... 30
     5.3.1 The reduced particle filter ............................... 31

6 Results 35
  6.1 2-dimensional particle filter ................................. 35
     6.1.1 The pool .................................................... 35
     6.1.2 Generated terrain ......................................... 39
  6.2 3-dimensional particle filter ................................. 39
     6.2.1 The pool .................................................... 40
     6.2.2 Generated terrain ......................................... 45

7 Conclusions and future work 47
  7.1 Conclusions ..................................................... 47
  7.2 Future work ..................................................... 48
     7.2.1 Further work with ROV sensors ......................... 48
     7.2.2 Speeding up the filter .................................... 48
     7.2.3 Improving on the models .................................. 48
     7.2.4 Higher resolution seafloor grids ....................... 48

A Hardware 51
  A.1 Sonars .......................................................... 51

B Quaternions 53

Bibliography 55
# Notation

## Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUV</td>
<td>Autonomous Underwater Vehicle</td>
</tr>
<tr>
<td>EKF</td>
<td>Extended Kalman Filter</td>
</tr>
<tr>
<td>IMU</td>
<td>Inertial Measurement Unit</td>
</tr>
<tr>
<td>ROV</td>
<td>Remotely Operated Vehicle</td>
</tr>
<tr>
<td>PF</td>
<td>Particle Filter</td>
</tr>
<tr>
<td>DTM</td>
<td>Digital Terrain Model</td>
</tr>
</tbody>
</table>
1

Introduction

This is a master thesis done at the Department of Electrical Engineering (ISY) at Linköping University. It is a part of a project done in collaboration with the Department of Management and Engineering (IEI) within a larger chain of studies and development spread across both departments. This section of the thesis will cover the purpose of the thesis and related work in the field.

1.1 Background

Remotely operated vehicles have become a good alternative for performing tasks underwater that potentially could be dangerous for a diver. Such tasks include mine-sweeping, seabed mapping or maintenance work on gas pipelines. There has been an ever increasing demand for the vehicles to be able to perform complex tasks autonomously, which also has led to an increase in prices. Because of the multitude of uses for these kinds of vehicles and the decreased price for computational power, Linköping University started the development of a remotely operated vehicle, henceforth referred to as an ROV, in 2010. The aim of the project was to develop a platform on which a relatively cheap autonomous underwater vehicle or AUV could be based. The development has since its start been continuous, with contributions from different courses as well as a few master theses.

1.2 Purpose and objective

To understand the purpose of this thesis, one can start by asking:

"Why is it necessary for the ROV to be able to locate itself?"
Remotely operated systems might not need a way of locating themselves if the operator is close enough to see the vehicle and its surroundings. But if the line of sight between the two is obstructed, be it due to long distance or other obstacles, the vehicle needs to be able to tell the operator where it is. Otherwise, depending on the purpose and environment of the vehicle, the consequences could be severe as the operator would be as blind as the vehicle.

If the vehicle in some way could sense or take measurements of its surroundings, then the operator could estimate the position of the vehicle and control it in a better informed way. Accurate readings help in to analyze the dynamic motion of the vehicle as well and are necessary for system identification.

It is however a slow process and sometimes the operator would need other instruments to make sense of the information sent back by the vehicle. A faster approach would be to give the vehicle itself the tools to autonomously estimate its own position. This way the operator can focus on controlling the vehicle according to its purpose. If the vehicle independently could locate itself, it could then be given the tools to independently control itself. This would be the step from an ROV to an AUV, which is the overlaying goal of the project at LiU. For this step to be taken, the tools for localization has to be implemented first.

The objective of this thesis is to locate an ROV using side scan sonars, together with an inertial measurement unit and a pressure-sensor.

1.3 Related work

As this is a joint project between two different departments at LIU, a number of different reports have been written with this ROV as focus. IEI has produced reports such as TMPM01 [2013] and masters thesis Ericsson [2012] with focus on the mechanical side, while reports under ISY have had a focus on the control side of the project, Bernhard and Johansson [2012] and TSRT10 [2012].

Localization by ranging, i.e. measuring the distance from a sensor to a landmark, is a common practice that is used in many other fields apart from nautical applications. Many projects on land use ranging as means of localization. As the density of air is smaller than that of water, lasers or radars are the preferred tool of measurement. Spinning lasers and radars can give accurate 360 degree measurements of the area, which is a great way to map and localize in 2 dimensions, see Callmer et al. [2011]. In 3 dimensions, ranging measurements can be trickier and are often done along an axis where relevant information is most likely to exist. Other than submerged vehicles, flying ones are another category that have to deal with problems of localization in 3 dimensional space. Many of the techniques used for localization, such as dead-reckoning and terrain-matching, are common to both fields, see Nordlund and Gustafsson [2009]. Flying vehicles does however have the possibility to use GPS navigation which is not available to submerged crafts, they also have the ability to cover a larger area in a smaller amount of time.
Dead-reckoning algorithms, i.e estimating change in position by integrating velocity readings, are viable in a short time-frame. The problem with this approach is that a bias in the velocity readings will cause a cumulative error due to the nature of integration. If a way to measure velocities is not available but measurements of acceleration is, a position estimate can be achieved by integrating the acceleration readings twice. This will cause a cumulative error of a cumulative error. To get a more robust dead-reckoning algorithm the use of a doppler velocity log, which measures shifts in frequency from sound waves directed at the sea-floor to get velocity readings. This would reduce the computations of displacement to one integration, still subject to drift but not as severe as with acceleration measurements only. Snyder [2010] describes how this is a viable option when integrated with other techniques to get a fixed position. This thesis is done under the assumption that a doppler velocity log is not present in the ROV.

Terrain-matching techniques often gives rise to a complex probability distribution, and can give uni-modal filters problems, see Huang and Dissanayake [2007]. Because of this there have been a rise in the use of multi-modal filters such as the Particle Filter (PF) as seen in Nakatani et al. [2009], Fairfield and Wettergreen [2008]. These studies does not focus on the particle filter alone but instead as a complement to other sensors. The filter does not handle high dimensions well, and therefore marginalization techniques have to be used to make it a feasible option.

To use terrain-matching techniques, a map of the operating area is necessary. When doing bathymetric surveys, grid-based topological maps is often the preferred method of storing data, GEBCO [2013]. The resolution of the grids varies greatly and tends to be better near coastlines and islands with much water traffic. A good map could also help if one wants to save time on physical testing or if such testing is not viable at the time. Simulated sonar readings are not an uncommon way to analyze techniques for localization or vehicle position estimation, Reed et al. [2006].

1.4 Limitations

Unfortunately, the sonars that were going to be used to collect real data have not been available during the time allocated to write this thesis. Instead a simple model of how the sonars work by using ray-casting methods have been implemented and described in Chapter 4. Because the sonars have been modeled and data from actual sonars have not been used, the need for an artificial environment to handle queries from the modeled sonars is clear.
This chapter will present the physical systems that are used to, and is supposed to be localized. How the ROV looks and some of its specifications are presented to give the reader a better picture of what is being discussed later in the report. There is plenty of electrical components inside the ROV, but these will not be described as they do not have an essential impact on the work that have been done. A more detailed description of the ROV is available in Ericsson [2012].

\textbf{Figure 2.1: The ROV in its current appearance}

\section{2.1 ROV}

The physical ROV has been developed by the Department of Management and Engineering (IEI) at Linköping University and most of the individual parts are also manufactured at the university.

The ROV has a torpedo-like design aimed at reducing drag in the surge direction. It has a total of five thrusters, two in its horizontal plane to control sway as well as the yaw-rate. Two more in the vertical plane to control heave as well as pitch, and the main thruster is located at the back in the intersection of the vertical and
horizontal planes of the ROV to control the surge. See figure 2.2. The thrusters are in the vertical and horizontal planes are tunnel-thruster with specially designed propellers which allows it to generate approximately the same amount of force in both directions.

Unfortunately the single thruster in the back of the ROV also causes an undesired, and currently uncontrollable, roll-motion. The roll is not severe enough to make the ROV tip over and it will stabilize eventually. This problem is well known and is currently being addressed at IEI. As such, an assumption will be made in Chapter 4 that the undesired roll is not present.

![Figure 2.2: A visualization of the ROV from a) above and b) from the side. Image taken from Ericsson [2012].](image)

### 2.2 Sensors

The ROV carries several sensors, and is being developed with a plan to carry several more. These sensors will play a vital role in the transition from a ROV to an AUV. At present time 3 sensors are available for the localization problem. One IMU that consists of accelerometers, gyroscopes and magnometers, a pressure sensor for depth measurements and two side mounted sonars.

#### 2.2.1 IMU

An Inertial Measurement Unit from Xsens is carried on-board the ROV. It gives readings from three different sensor-types and calculates acceleration, turn rate and angular deviation from a magnetic north in 3 dimensions. Xsens calls the system a gyro-enhanced Attitude and heading reference system (AHRS) and claims that it is relatively free from drift unless there are a lot of ferrous metals present in the vicinity of the sensor.

Orientational output from the IMU can be presented in 3 different ways:
Unit quaternions: Further described in appendix B.

Euler angles: Roll, pitch and yaw in the fixed local coordinate system

Rotation Matrix: A complete representation of rotation with unit-vectors for the sensor coordinate system represented in the local system described in each column.

The measurements from the IMU have an angular resolution with standard deviation of 0.05 degrees or $\pi/3600$ radians. Static accuracy is less than $1/2$ a degree for roll/pitch and 1 degree for yaw. Under motion the dynamic accuracy is about 2 degrees but that depends on the type of motion. The maximum output frequency from the unit is 120 Hz.

The sensor have its own coordinate system factory-set to the body of the physical sensor, according to figure 2.3. This can be changed to fit a specific coordinate system if desirable.

![Figure 2.3: The IMU from Xsens on board the ROV](image)

The IMU does come with an integrated sensor-fusion algorithm which is optimized for the unit. All information about the unit are acquired from Xsens [2007].

2.2.2 Sonars

A Sound Navigation And Ranging (SONAR) system is usually divided into the classifications active and passive.

Passive sonars works by listening for sounds that are not produced by the sonar itself but by other sources. This is usually used in military applications where the goal is to detect without being detected.

Active sonars works by emitting a sound-pulse or ping, and then measuring the time it takes for the sound to return as well as the amplitude of the returned signal. The sonars on the ROV are classified as a active sonar system and carries
two units of transducers and receivers that emit respectively records sound on each side of the ROV. This is further classified as a Side-Scan Sonar system.

The sonars that will eventually be used on the ROV are part of the PC-scan system from Marine Sonic Technology, Ltd. In addition to the transducers, the system also comes with a pc-104 acquisition board as well as a controller board for the transducers, see figure 2.4

![Picture of the sonars. Image acquired from Marine Sonic Technology, Ltd](image)

**Figure 2.4:** Picture of the sonars. Image acquired from Marine Sonic Technology, Ltd

### 2.2.3 Sonar resolution

Side-scan sonars have a very narrow swath and takes measurements perpendicular to the traveling path of the vehicle. These measurements are then represented as recorded amplitude over time and presented according to figure 2.6. Resolutions in these measurements varies due to a number of different variables such as the vehicle’s Speed Over Ground (SOG) which impacts the resolution along the traveled path of the vehicle. The resolution is also affected by the distance to the reflective object as the sound wave emitted by the transducers widens the further it travels. Because sonars use sound for measurements, they are dependent on the speed of the sound in water, which unfortunately is not constant. The speed of sound in water is subject to salinity, temperature and pressure. A simple equation for calculating the speed of sound in water was presented by Mackenzie [1981]:

\[
c = 1448.96 + 45.9 \times T - 5.304 \times 10^{-2} \times T^2 + 2.374 \times 10^{-4} \times T^3 \\
+1.340 \times (S - 35) + 1.630 \times 10^{-2} \times D + 1.675 \times 10^{-7} \times D^2 \\
-1.025 \times 10^{-2} \times T \times (S - 35) - 7.139 \times 10^{-13} \times T \times D^3
\]  

(2.1)
2.3 The environment

The ROV is expected to operate within a few different environments. These are a large pool, either at SAAB underwater systems or at municipal pool in Linköping. The pool is a natural environment to test and evaluate the vehicle and its equipment as well as its dynamical models.

The ROV will eventually operate in a multitude of environments, such as canals, harbors or out at sea. This thesis will focus on the testing environment for the ROV, i.e the pool, as well as an open sea premise.

\[ c = \text{Speed of sound in water} \]
\[ T = \text{Temperature in degrees Celsius} \]
\[ S = \text{Salinity in parts per thousand} \]
\[ D = \text{Depth in meters} \]

This is a model that is simple to use in simulation calculations. More complex models do exist, such as the UNESCO model by Wong and Zhu [1995]. For the sake of keeping the presentation of equations as neat as possible and because the performance is enough for the scope of this thesis, only the Mackenzie model will be presented.

Figure 2.5: Image illustrating the spread of a sonar sound wave. Image acquired from Marine Sonic Technology, Ltd
Figure 2.6: Visual representation of measurement data from both sonars. Image acquired from Marine Sonic Technology, Ltd
There are quite a few techniques and methods to navigate in an unknown environment and there are a number of studies done every year on the subject. In Section 1.3 a number of studies and techniques for navigation have been mentioned. This chapter expands on the techniques and theories used in underwater navigation.

### 3.1 AUVs and localization

A distinction between two types of studies seems to exist. One type appears to be based around *exploration* and have a focus where the vehicle should be able to perform mapping as well as localization, these algorithms are usually referred to as *Concurrent mapping and localization* (CML) or *Simultaneous localization and mapping* (SLAM). Such studies include Tena Ruiz et al. [2004] which makes use of a stochastic map where new landmarks are stored in a state vector, and re-observed ones contribute to a measurement update. This re-observation poses a problem for side scan sonars as they need to make several passes over the same area to get a measurement update in the stochastic map. If a pass never is made over the same area twice, then the vehicle would not be able to localize itself. A data-association problem also exits in which measurement are to be associated with which landmark. Landmarks have to be distinguishable independently of the vehicle orientation, which is not a trivial problem.

The use of a stochastic map to store features seems to be a common practice when trying to localize vehicles in an underwater environment. Studies such as Folkesson et al. [2007] also use stochastic maps for landmark storage. An overview of current navigation techniques for underwater navigation can be found
3.2 Extended Kalman filter

First developed by Kalman [1960] as an optimal filter for linear systems, an extension of this filter was later developed by Smith et al. [1962] and Schmidt [1966] to the domain of nonlinear dynamics and was intuitively named the Extended Kalman Filter or EKF. This extension does not give an optimal estimate, as the Kalman filter usually provides. Instead it approximates an optimal estimate by linearizing the dynamics around the most recent state estimate. This gives rise to the problem that unlike the regular Kalman filter, the extended version may diverge due to loss of information in the linearization step. The linearization is usually performed by a first or second order Taylor-expansion according to:

$$f(x) = f(\hat{x}) + f'(\hat{x})*(x - \hat{x}) + \frac{f''(\hat{x})}{2}*(x - \hat{x})^2$$ (3.1)

In this thesis the first order EKF will be used, and the algorithm for the first order EKF is derived from Gustafsson [2012] as follows:

Assume a nonlinear state-space system:

$$x_{k+1} = f(x_k, u_k, v_k), \quad (3.2)$$
$$y_k = h(x_k, u_k, e_k) \quad (3.3)$$

where $v_k$ and $e_k$ are multivariate gaussian noise with $Q_k$ and $R_k$ respectively. To make the notation more compact $f(x_k, u_k, v_k)$ and $h(x_k, u_k, e_k)$ has been shortened to $f(x)$ and $h(x)$ in the following steps.

The measurement update:

$$S_k = R_k + h'(x_{k|k-1}) * P_{k|k-1} * h'(x_{k|k-1})^T \quad (3.4)$$
$$K_k = P_{k|k-1} * h'(x_{k|k-1})^T * S_k^{-1} \quad (3.5)$$
$$\epsilon_k = y_k - h(x_{k|k-1}) \quad (3.6)$$
$$x_{k|k} = x_{k|k-1} + K_k * \epsilon_k \quad (3.7)$$
$$P_{k|k} = P_{k|k-1} - P_{k|k-1} * h'(x_{k|k-1}) * S_k^{-1} * P_{k|k-1} \quad (3.8)$$

The prediction step:

$$x_{k+1|k} = f(x_{k|k}) \quad (3.9)$$
$$P_{k+1|k} = Q_k + f'(x_{k|k}) * P_{k|k} * f'(x_{k|k})^T \quad (3.10)$$

3.3 Particle filter

A particle filter is also known as a sequential Monte Carlo method. It can handle a continuous state-space representation while being able to sustain a multi-modal
posterior distribution. The particle filter became popular in applications after a paper by Gordon et al. [1993]. Since the publication of the paper the method have become more popular since computational power have become cheaper and more available. The main selling point of the method is its ability to handle nonlinear systems relatively well at the cost of computational power. The computational cost for the filter grows exponentially to the number of dimensions it is expected to handle. This makes it difficult to implement in real-time for higher dimensional problems.

The filter works by placing a set of particles around a map and comparing actual sensor measurements to expected measurements for the respective particles. The particles are then weighted according to how likely the the expected measurements are and then re-sampled. The higher the weight, the more likely a particle is to be re-sampled. There are a number of ways to choose how the particles should be re-sampled but regardless of which re-sampling strategy that is chosen, the process could be described as survival of the fittest. Only the particles that have a good fit to the real measurements will survive. A framework for the particle-filter is described in Gustafsson [2012] as:

**Step 1:** Initialize the filter: Generate \(N\) particles according to a chosen proposal distribution, \(x_1^i \sim p(x_0)\), where \(i = 1, \ldots, N\) and distribute the weight of the particles equally \(w_{i|0}^i = 1/N\).

Step 2 to 5 are iterated as many times as the filter is meant to run, \(k\) indexes which iteration.

**Step 2:** Do a measurement update and update the weights for all the particles according to the how well the particle matches the measurement, \(p(y_k|x_k^i)\): for \(i = 1, \ldots, N\) do

\[
    w_{k|k}^i = \frac{1}{c_k} w_{k|k-1}^i p(y_k|x_k^i) \tag{3.11}
\]

where \(c_k\) is the normalization weight:

\[
    c_k = \sum_{i=1}^N w_{k|k-1}^i p(y_k|x_k^i) \tag{3.12}
\]

**Step 3:** Is the estimation step where the density of the filtering is approximated as

\[
    \hat{p}(x_{1:k}|y_{1:k}) = \sum_{i=1}^N w_{k|k}^i \delta(x_{1:k} - x_k^i) \tag{3.13}
\]

**Step 4:** Re-sampling. This step doesn’t necessarily need to be done for each step \(k\). For a suitable \(k\), choose \(N\) samples from the set \(\{x_k^i\}_i^N\). The \(i:th\) sample should be chosen with the corresponding probability \(w_{k|k}^i\). After the re-sampling is done, set all weights equal again \(w_{k|k}^i = 1/N\).
Step 5: The time update. Predict the next state of all samples according to the proposal distribution

$$x_{k+1}^i \sim q(x_{k+1}^i|x_k^i, y_{k+1})$$  \hspace{1cm} (3.14)

as well as update the weights for all particles according to

$$w_{k+1|k}^i = w_{k|k}^i \frac{p(x_{k+1}^i|x_k^i)}{q(x_{k+1}^i|x_k^i, y_{k+1})}$$  \hspace{1cm} (3.15)

### 3.3.1 Marginalization

As the ROV has six degrees of freedom which would mean that a particle filter would have to compute particles in six dimensions: position as well as the attitude of the vehicle. This would in practice mean one have to generate enough particles to cover every combination of all the angles at every plausible position in the map. The computational cost for this approach would be very high and in most cases unfeasible for localization in areas larger than a few meters. So instead of allowing the particle filter to assume that all the states are unknown, one can use other techniques for estimating the more linear states and leave the non-linear states for the particle filter. A smaller number of particles will be required as the number of dimensions the particle filter will operate on have been reduced. This technique is referred to as marginalization or Rao-Blackwellization. For more in-depth information on how this is done see Gustafsson et al. [2002].

### 3.4 Usage of filters

As the ROV carries an IMU – unit as well as a pressure sensor, the attitude and depth of the ROV will be computed using the EKF and the particle filter will be used to compute the position of the ROV.
This section will explain how the different parts described in the previous chapters are modeled for simulation purposes.

4.1 Dynamics

The dynamics for the ROV have been derived from TSRT10 [2012] as well as Bernhard and Johansson [2012] with a few small changes.

To help represent the orientation and position of the ROV, two coordinate systems will be used. One is fixed to the ROV with x,y and z-axes as illustrated in Figure 4.1. The other is used to represent the local area in which the ROV are supposed to operate and have east, north and down -axes (E,N,D). The attitude of the ROV in the local coordinate system is obtained as follows. First a temporary coordinate system with X”,Y”,Z” axes are created by rotating the local coordinate system by \( \phi \) radians around the E-axis. Next the temporary coordinate system is updated to a system with X’,Y’,Z’ axes by rotating the X”,Y”,Z” system around the Y”-axis by \( \theta \) radians. The last operation to get the ROV X,Y,Z axes is achieved by rotating the X’,Y’,Z’ system around the Z’-axis by \( \psi \) radians. This is illustrated in Figure 4.2. All rotations are done in a right hand system.
4.1 Modeling

Figure 4.1: ROV coordinate system.

Figure 4.2: Illustration on how the ROV coordinate system integrates with the local E,N,D system.

4.1.1 Model of the ROV

It is important to point out that in this thesis any control or model of the roll-variable is neglected. An assumption has been made that the roll exhibited due to acceleration in the surge direction is stable and is driven by a small gaussian process noise.

The following equations have been used to model the ROV. The model and the equations have been simplified. For a more complete explanation of the model see TSRT10 [2012].
4.1 Dynamics

<table>
<thead>
<tr>
<th>Variable</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_{M,zr,zf,yr,yf}$</td>
<td>Control signals to the thrusters, Main, rear- and front- vertical as well as rear- and front-horizontal.</td>
</tr>
<tr>
<td>$\phi, \theta, \psi$</td>
<td>Roll, pitch and yaw angles respectively [radians]</td>
</tr>
<tr>
<td>$p, q, r$</td>
<td>Roll, pitch and yaw velocities respectively [radians/s]</td>
</tr>
<tr>
<td>$e, n, d$</td>
<td>East, north and down position in world [m]</td>
</tr>
<tr>
<td>$u, v, w$</td>
<td>Surge, sway and heave velocities [m/s]</td>
</tr>
</tbody>
</table>

**Table 4.1: Model variables**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_{T_{M,zr,zf,yr,yf}}$</td>
<td>Translation coefficients for thrusters</td>
</tr>
<tr>
<td>$C_{R_{M,zr,zf,yr,yf}}$</td>
<td>Rotation coefficients for thrusters</td>
</tr>
<tr>
<td>$D_{\phi, \theta, \psi}$</td>
<td>Linear dampening coefficients for roll, pitch and yaw velocities</td>
</tr>
<tr>
<td>$D_{Q_{\phi, \theta, \psi}}$</td>
<td>Quadratic dampening coefficients for roll, pitch and yaw velocities</td>
</tr>
<tr>
<td>$D_{u,v,w}$</td>
<td>Dampening coefficients for translation in surge, sway and heave directions</td>
</tr>
<tr>
<td>$\rho$</td>
<td>Water density</td>
</tr>
<tr>
<td>$B_{\theta}$</td>
<td>Buoyancy constant for pitch</td>
</tr>
<tr>
<td>$V$</td>
<td>ROV volume</td>
</tr>
<tr>
<td>$m$</td>
<td>ROV mass</td>
</tr>
<tr>
<td>$M_{\phi, \theta, \psi}$</td>
<td>ROV total mass during rotation around roll, pitch and yaw axis respectively</td>
</tr>
<tr>
<td>$Ts$</td>
<td>Sample-time</td>
</tr>
</tbody>
</table>

**Table 4.2: Model parameters**

\[
\begin{align*}
\dot{u} &= C_{T_M} u_M |u_M| - D_u u |u| - \frac{g(m - \rho V)}{M} \sin(\theta) \\
\dot{v} &= C_{T_y} u_y |u_y| + C_{T_y} u_y |u_y| - D_v v |v| + \frac{g(m - \rho V)}{M} \sin(\phi) \cos(\theta) \\
\dot{w} &= C_{T_z} u_z |u_z| + C_{T_z} u_z |u_z| - D_w w |w| + \frac{g(m - \rho V)}{M} \cos(\phi) \cos(\theta) \\
\dot{q} &= C_{R_{\phi}} u_{\phi} |u_{\phi}| - C_{R_{\phi}} u_{\phi} |u_{\phi}| - D_{\theta} q - D_{Q_{\phi}} q |q| - B_{\theta} \sin(\theta) \\
\dot{r} &= C_{R_{\psi}} u_{\psi} |u_{\psi}| - C_{R_{\psi}} u_{\psi} |u_{\psi}| - D_{\psi} r - D_{Q_{\psi}} r |r|
\end{align*}
\]
The variable for angular roll acceleration $\dot{\phi}$ is stochastic with a normal distribution.

$$\dot{\phi} \sim N(0,1) \quad (4.6)$$

The rotation from the ROVs local XYZ coordinates to the global E,N,D frame is done according to:

$$R(\phi, \theta, \psi) = \begin{pmatrix}
\cos\phi\cos\theta & -\sin\psi\cos\phi + \cos\psi\sin\theta\sin\phi & \sin\psi\sin\phi + \cos\psi\cos\phi\sin\theta \\
\sin\psi\cos\theta & \cos\psi\sin\theta\sin\phi - \cos\psi\cos\phi\sin\theta & -\cos\sin\phi + \sin\psi\cos\phi\sin\theta \\
-\sin\theta & \cos\sin\phi & \cos\theta\cos\phi
\end{pmatrix} \quad (4.7)$$

This is used to compute the displacement of the ROV in the E,N,D coordinate system dependent on the velocities $u, v$ and $w$.

Because angular velocities are measured in the local coordinate system they need to be rotated into the global coordinate system before they can be used to compute new angles. This is done using the following rotational matrix:

$$T(\phi, \theta, \psi) = \begin{pmatrix}
1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\
0 & \cos\phi & -\sin\phi \\
0 & \sin\phi & \cos\theta
\end{pmatrix} \quad (4.8)$$

Change in position and angles is computed as:

$$\dot{\eta} = R(\phi, \theta, \psi) \begin{pmatrix} 0_{3x3} \\ T(\phi, \theta, \psi) \end{pmatrix} \nu \quad (4.9)$$

$$\dot{\nu} = f_b(\eta, \nu, u) \quad (4.10)$$

where $\eta = (e, n, d, \phi, \theta, \psi)^T$, $\nu = (u, v, w, p, q, r)^T$ and $f_b(\eta, \nu, u)$ is the compact notation of Equations (4.1) to (4.5). The complete model can then be described as:

$$\dot{x} = f(x, u) = \begin{pmatrix} f\nu \\ f_b(x, u) \end{pmatrix} \quad (4.11)$$

where $x = (\eta^T \nu^T)^T$ and $u = (u_M, u_zr, u_zf, u_yr, u_yf)^T$.

This model is discretized using euler-forward:

$$\dot{x} \approx \frac{x_{k+1} - x_k}{T_s} \quad (4.12)$$
4.2 The map

A proper representation of the environment is crucial, as it is very hard to locate something in an environment you do not know. It is possible to map and locate at the same time, but that is outside the scope of this thesis. When most of the work is done in a simulated environment the need for an accurate map becomes even clearer. There are a number of ways to represent a map, and in this thesis a Digital Terrain Model (DTM) composed of polygons have been chosen as this is a common representation for simulations.

To build these map representations, there are two options. Either model the map from real bathymetric surveys or create them from scratch. Much of the work presented has been done with a simple map of a rectangular pool as a premise. However it have also been done in a way that allows for a transition to a much more complex environment. How this is done will be the topic of the next sections.

4.2.1 The pool

To begin with a simple representation of a pool is easy to model as one can imagine it as a box, where the walls are the boundaries. The pool is assumed to line up with the E,N,D coordinate system thus making simulations of sonar responses easy as a measurement will just increment the distance from the ROV or particle in a set direction until it hits one of the bounding conditions:

\[-20 \geq E \leq 20 \] (4.13)
\[-20 \geq N \leq 20 \] (4.14)
\[-20 \geq D \leq 0 \] (4.15)

A simple starting point but nevertheless relevant as the ROV are supposed to be tested in a pool. The problem with this representation is that it’s very static and uses numerical constants as boundaries. If a map which boundaries cannot be represented mathematically is desirable, another method is needed to model the map.

4.2.2 Digital terrain model

There exists a number of different ways of representing height information data as a digitized map, common abbreviations of such data representation models are DEM (Digital Elevation Model), DTM (Digital Terrain Model) and DSM (Digital Surface Model). There seem to be no de facto standard on when which abbreviation should be used. Some argue that a DEM includes DTM and DSM, others seems to use DTM to refer to both DTM and DSM. One clear distinction between DTM and DSM is that the latter contains more information as it includes all surfaces, including roofs and roads, while the former only represents the geographical terrain. As there seldom is anything else than pure terrain below sea-level, the abbreviation used to refer to the digitized model of the environment will be DTM.
Data to build the DTM can be acquired from a number of different sources such as GEBCO [2013] or BSH [2013], this can then be represented as a surface structure of polygons, see Figure 4.3. The downside with large databases like these are that the resolution seldom are very good. GEBCO uses 30-arc second resolution between the grids in the sets that can be downloaded. An arc second is 1/60 of an arc-minute which is an angular measurement defined as 1/60 of a degree. This measurement unit is used in areas where representations of small angles is necessary, such as navigation and astronomy. A resolution of 30 arc seconds corresponds roughly to 1 km. The small resolution can be mitigated in simulations as larger areas can be shrunken to represent a smaller fictional area with higher resolution.

![Seafloor DTM](image)

**Figure 4.3:** A DTM created from data acquired from GEBCO

Alternatively one can create approximations of mountainous underwater or coastal regions by using Fractal Brownian motion (FBm) to generate terrain. FBm is described mathematically by the Weierstrass- Mandelbrot function:

\[
V(t) = \sum_{f=-\infty}^{\infty} A_f r^H f \sin(2\pi r^{-f} t + \theta_f) \tag{4.16}
\]

Where \(A_f\) is gaussian, \(\theta\) contributes a random phase, \(r\) defines the resolution of the landscape and \(H\) defines the dimension.

FBm has the disadvantage that it is not able to model natural dynamics as it is a stationary process. This means that a map created with FBm methods will lack features caused by tectonic movement or erosion such as cave systems or water streams. Cave systems is not something that bathymetric surveys are able to measure either. So as an approximation of underwater terrain FBm methods are good enough for the scope of this thesis, compare Figures 4.3 and 4.4. For a more
4.2 The map

A comprehensive reading on the merits and demerits of FBm methods please see Musgrave [1993].

The map is modeled by transforming a grid into polygon surfaces, the DTM to be used is given a list where each entry consists of three points in E,N,D coordinates which together represents a polygon-surface. This makes it possible to see if a line from an origin point to an endpoint passes through any of the polygons.

Even though the problem with a complex map representation have been solved by the use of a DTM, it quickly becomes infeasible from a computational standpoint if all the polygons in the DTM have to be queried. Another component is needed to make a reasonable representation of the environment, a way of sorting the polygons according to their location. This is achieved by using a so-called octree.

4.2.3 Octree

When queries of a large data set are necessary, the need for an efficient way of storing the data is apparent. One such way of dividing the data is a so called octree. An octree is a type of data-structure which receives its name from the fact that it is a tree where every node has eight children, see Figure 4.5, unless it has hit some sort of conditions declaring the node to be an end-node. The conditions for a node to become an end-node are usually that it has hit a maximum number of objects in the node, or that a maximum depth for the tree has been reached.

Octrees are popular in 3D graphics but can also be useful in state-estimation in non-linear problems, see Eberhardt et al. [2010]. Another advantage is that it allows for fast queries on where in a large set, an object or surface is located. This also makes it useful when collision detection is a required feature.
or Ray-tracing algorithms often are often used in combination with an octree, as collision checking is easy and fast to do with axis aligned bounding boxes as described in Williams et al. [2005].

The octree is made by enclosing the DTM in a box much like the initial representation of a pool. It is then expanded by creating 8 smaller boxes inside the original box and then 8 more inside each one of them and so on. This is done until 1 of 2 conditions have been met:

1: The number of polygons in the newly created box is less or equal to the maximum number of polygons allowed in one box.
2: A maximum number of box-divisions have been reached.

An octree will create a large amount of small boxes around E,N,D coordinates containing polygons and a few large boxes in areas without polygons, see Figure 4.6.

### 4.3 Sonars

Sound-wave propagation can be hard to model in a discrete 3-dimensional environment. However as sonars measure the returning amplitude and return time from a ping we can estimate a sound-wave by ray-casting. By giving the rays frequency and signal strength properties, a sound-wave can be simulated with an accuracy depending on the number of rays used for each ping. The strength of a reflected ray is subject to the angle of incidence of the surface it is reflected from as well as the distance to the surface. The frequency property of the ray governs how far the ray can travel before dying out. The sonar model can be seen as a very rudimentary implementation of the Bell and Linnett [1997] sonar model.
To initialize a sonar ping, the number of rays to be used in a sonar ping are generated on a normalized distance on both sides of the ROV. In other words, the rays are generated in the ROV pitch-, yaw-plane defined by the local $Y$, $Z$-axes as displayed in Figure 4.7. It is also scrambled around the $Y$-axis to properly simulate the dispersion of a sound-wave.

Each ray is then propagated through the octree until no more nodes are available or the maximum length of the sonar has been reached. References to the polygons in the final node it traverses are returned and a check if the ray hits any of the referenced polygons is performed. If a hit is detected, return time and amplitude are calculated.

Distance is calculated by taking the square root of the dot product of the ray vector. So if $P = [E, N, D]$ represents the position of the ROV and $I = [E, N, D]$ represents the ray-polygon intersection, the ray vector $V = P - I$ is used to calculate the distance $D$ as:

$$D = \sqrt{V \cdot V^T}$$

\[ (4.17) \]
The return time is then calculated as:

\[ T_{\text{return}} = 2 \times D \times c \]  \hspace{1cm} (4.18)

where \( c \) is the speed of sound in water according to the Mackenzie model (2.1).

Amplitude \( A \) of the returned ray is approximated as:

\[ A = \frac{\cos(\beta)}{2 \times D} \]  \hspace{1cm} (4.19)

where \( \beta \) is the angle between the ray-vector \( V \) and the normal of the polygon it intersects.

Each ray receives a time-stamp as well as an amplitude, this way each sonar measurement can be simulated as the returned amplitude over time. Which is the proper output of a physical sonar.

### 4.4 IMU

The **Inertial measurement unit** supplied by Xsens returns the angular velocity of the sensor in all three dimensions with the accuracy described in 2.2.1. The assumption here is that raw data from gyroscope and accelerometer is available, angular velocities are then measured as:
\[ p_k = \bar{p}_k + \varepsilon_p \tag{4.20} \]
\[ q_k = \bar{q}_k + \varepsilon_q \tag{4.21} \]
\[ r_k = \bar{r}_k + \varepsilon_r \tag{4.22} \]
\[ \varepsilon_{p,q,r} \sim N(bias_{p,q,r}, \pi/90) \tag{4.23} \]

The three variables \( \bar{p}_k, \bar{q}_k, \bar{r}_k \) are the true values of the ROV angular velocities at sample \( k \) and \( \varepsilon \) represents the measurement noise and bias of the sensors.

The accelerometers read accelerations in surge-, sway- and heave-directions.
\[ \dot{u}_k = \dot{\bar{u}}_k + \varepsilon_{\dot{u}} \tag{4.24} \]
\[ \dot{v}_k = \dot{\bar{v}}_k + \varepsilon_{\dot{v}} \tag{4.25} \]
\[ \dot{w}_k = \dot{\bar{w}}_k + \varepsilon_{\dot{w}} \tag{4.26} \]
\[ \varepsilon_{\dot{u},\dot{v},\dot{w}} \sim N(bias_{\dot{u},\dot{v},\dot{w}}, 1) \tag{4.27} \]

It will also have a constant acceleration downwards in the local E,N,D coordinate system due to gravitational pull. This gravitational acceleration vector can be used to approximate the pitch and roll of the ROV by calculating the angle between the gravitational acceleration vector \( G \) and the ROV-fix roll-axis \( X \) and pitch-axis \( Y \) respectively. The measured values from the IMU then becomes:
\[ \phi_k = \cos^{-1}\left(\frac{G \cdot X}{|G||X|}\right) + \varepsilon_{\phi} \tag{4.28} \]
\[ \theta_k = \cos^{-1}\left(\frac{G \cdot Y}{|G||Y|}\right) + \varepsilon_{\theta} \tag{4.29} \]
\[ \varepsilon_{\phi,\theta} \sim N(0, \pi/360) \tag{4.30} \]

The last part of the IMU is the magnetometer that measures the difference between the local magnetic north \( M \) and the roll-axis \( X \) of the ROV. This is done in the same way as (4.28) and (4.29).
\[ \psi_k = \cos^{-1}\left(\frac{M \cdot X}{|M||X|}\right) + \varepsilon_{\psi} \tag{4.31} \]
\[ \varepsilon_{\psi} \sim N(0, \pi/180) \tag{4.32} \]

All measurements are in the ROV local coordinate system. Note that angles given by the IMU are the same in the local X,Y,Z frame as they are in the global E,N,D frame. This is because the IMU measures the angles as the difference between its local axes and the gravitational vector as well as the magnetic north vector. Angular velocity and accelerometer readings however, are expressed in the local X,Y,Z frame and needs to be rotated into the global E,N,D frame.

If physical testing had been done there would be no need to model the IMU as real data would have been available. In this way, the model of the IMU is only valid if the model of the ROV is valid.
5

Localization

What variables are estimated by which filter/technique is the question that this chapter consider. In earlier chapters the description of the ROV and its on-board sensors have been explained, as well as how the different components have been modeled. How this will be used to actually locate the ROV will begin with the association of dimensions and variables to filters.

5.1 Attitude

To estimate the attitude of the vehicle, the IMU readings are used. The challenge is how to use the readings to extract the desired information.

According to section 4.4, the IMU can give the attitude of the ROV using its accelerometers. The problem is that these measurements are noisy as illustrated in Figure 5.1.

Instead of just using direct angle measurements, the angles could be derived from the angular velocity by means of integration. If \( \alpha_k = (\phi_k, \theta_k, \psi_k)^T \), \( \omega_k = (p_k, q_k, r_k)^T \) and \( bias_k = (bias_{\phi_k}, bias_{\theta_k}, bias_{\psi_k})^T \) then:

\[
\begin{align*}
\dot{x}_{k+1} &= \begin{pmatrix}
\alpha_{k+1} \\
bias_{k+1}
\end{pmatrix} = \begin{pmatrix}
\alpha_k + T_s(bias_k + T(\alpha_k^T)\omega_k)
\end{pmatrix} \\
&= \begin{pmatrix}
\alpha_k + T_s(bias_k + T(\alpha_k^T)\omega_k)
\end{pmatrix}
\end{align*}
\]

(5.1)

Where \( T_s \) is the time difference between two samples, or \( T_s = 1/f \) where \( f \) is the frequency at which the sensors are running. \( T(\alpha_k^T) \) is the rotational matrix described in 4.8.

Making use of integration will help to smooth out the noise that is present in readings from the accelerometers but it will also cause the approximation to drift
A combination of the readings are desirable and for that the Kalman filter will be used to fuse the readings together.

**The measurement update:** Get the last predictions:

\[
h(x_{k|k-1}) = \begin{pmatrix} \phi_{k|k-1} \\ \theta_{k|k-1} \\ \psi_{k|k-1} \end{pmatrix}
\]  

(5.2)

And use them to compute innovation covariance \( S_k \) and kalman gain \( K_k \) according to (3.4) and (3.5) respectively. The measurement covariance matrix \( R_k \) is set to the variances that the IMU sensor should have according to Section 2.2.1. The \( h(x_{k|k-1}) \) term is linear as it only takes the earlier predictions from the time update described in (5.1), so \( h'(x_{k|k-1}) \) becomes easy to obtain as it is an identity matrix.

Get measurements from the IMU:

\[
y_k = \begin{pmatrix} \phi_k \\ \theta_k \\ \psi_k \end{pmatrix}
\]

(5.3)

and get the difference between the earlier prediction and current measurement:

\[
\varepsilon_k = y_k - h(x_{k|k-1})
\]

(5.4)

Update state estimations by fusing earlier state predictions with \( \varepsilon_k \) by using the kalman gain according to 3.7. Compute covariance matrix \( P_{k|k} \).
5.1 Attitude

The prediction step: This step predicts the next states $x_{k+1|k} = f(x_{k|k})$ according to (5.1). The predicted covariance matrix is computed as described in (3.10). Here the term $f(x_{k|k})$ is not linear and the jacobian $f'(x_{k|k})$ needs to be computed.

This will produce results as shown in Figure 5.3. A "drift-free" and smooth attitude is attained.

Figure 5.2: Approximations of angle using integration of gyro readings subject to drift.

Figure 5.3: Fused sensor data
5.2 Estimation of position

Estimation of position can be done by the same principle as described in the previous section. As the IMU gives linear accelerations a dead-reckoning algorithm could be used to get an approximate trajectory. However this is subject to the same problem of drift due to integration. In this case the drift will be substantially larger as the approximation is obtained by means of double integration as a method of measuring translational speed is not available. What is available are the sonars as well as a depth sensor, so if an algorithm to estimate e,n coordinates from sonar readings were developed, then an EKF could be used to estimate the position as well. Unfortunately such an algorithm is not available since the physical sonars have not been used. Instead a method of simply comparing sets of expected measurements with the actual one is desirable. Here lies the main reason for the choice of a particle filter over an EKF.

If the terrain is similar in several places then the data-association algorithm have to be robust enough that can differentiate between them. Because the EKF is uni-modal it can only accept one set of measurement as plausible. The particle filter however is multi-modal and will therefore accept several sets of measurements as plausible until it has enough information to rule all but one out, which might never happen.

5.3 Dimension of filters

Chapter 4.4 describes how the IMU allows for measurements of the variables turn-rate \((p, q, r)\) and translational acceleration \((\dot{u}, \dot{v}, \dot{w})\) as well as estimations of the angles \((\phi, \theta, \psi)\). This in combination with the depth-measurement from the pressure sensor leaves the estimation of east and north coordinates.

In traditional localization methods this can be achieved using the EKF to estimate translational movement, by complementing integrals of the linear acceleration from the IMU with modeling information of the ROV. The IMU might give relatively drift-free outputs regarding orientation, but the small errors that it contributes will be enhanced over time due to the nature of integration. This will make the dead reckoning algorithm accumulate a bigger and bigger error. This error will be dampened somewhat by the modeling information and the EKF but will still be significant. The EKF could perform better if it had more sensors, such as a doppler velocity log.

This is under the assumption that the IMU actually performs like advertised, which it doesn't seem to have done according to TSRT10 [2012]. The yaw-angle IMU - output did not coincide with the actual yaw angle of the ROV, if this is due to faulty settings or other disturbances are not clear. One might argue that this warrants the expansion of the particle filter to 3 dimensions, thus incorporating the estimation of the yaw-angle as well as the e- and n-coordinates.

Two versions of the particle can be tested. One which only estimates the e and n
coordinates and one estimating the yaw as well. The state vector for the particle filter then becomes:

\[
x_1 = (e, n, \psi)^T \quad \text{(5.5)}
\]
\[
x_2 = (e, n) \quad \text{(5.6)}
\]

5.3.1 The reduced particle filter

Because an EKF have been used to compute attitude and depth, the number of dimensions left for the particle filter to compute have been reduced. The proper way to reduce the number of dimensions is to use a marginalized particle filter as described in Section 3.3.1 where every particle has a Kalman filter. In order to keep computations as low as possible and under the assumption that every particle would have received similar filters, the approach in this thesis only uses one extended Kalman filter. In every time step the particles will be updated with the attitude attained from the EKF with an extra process noise attained from the EKF variance, see Figure 5.4. This approach will in this thesis be referred to as the reduced particle filter.

The reduced particle filter algorithm can now be applied as follows:

**Figure 5.4:** Variance for angles from the EKF

**Step 1:** Initialize the filter: The initialization step is much like the generalized method described in Section 3.3 with the exception that the generation is done in a E,N - plane at a specified depth. The particles are generated accordingly to the probability density function which is a uniform distribution of particles in the E,N - plane, visualized in Figure 5.5.
Depending on the size of the environment and if the starting position is unknown, another approach could have been chosen. The probability density function would then be a uniform distribution of particles in a circle around the starting position of the ROV.

![Initial distribution of particles](image)

**Figure 5.5: Distribution of particles**

**Step 2:** Two cases exist here as well, depending on if the yaw angle is to be estimated by the particle filter or not. If the yaw angle is estimated by the magnetometer and the Kalman filter then the measurement updates are done in every time step for the attitude and the depth (D - coordinate) of the particles. The measurement update for the position (E,N - coordinates) is done with a frequency of 1 Hz. This means that a measurement update will only be performed when a new sonar scan is available.

If the yaw-angle is to be estimated by the particle filter then the measurement update frequency of the yaw-angle will drop to the same as the E,N-coordinates.

**Step 3:** The estimation of the states are done by multiplying the weights for each particle with the values of the states that are subject to the particle filter and the summarizing them. Just as described in (3.13).

**Step 4:** The re-sampling step is set to the same time steps as the measurements from the sonars are available. If there are no new measurements from the sonars there is no point in re-sampling as the weights have not been updated.

Re-sampling is done with the Re-sampling wheel. Imagine if a wheel or a circle where weights with a higher value takes up a larger portion of the figure. If a random portion of the wheel is selected then the parts with a
larger portion are more likely to be selected. This approach have a linear $O(N)$ complexity. See Figure 5.6

![Diagram of re-sampling particles](http://www.mrpt.org/wp-content/uploads/2013/10/Resampling_system.png)

**Figure 5.6:** Method of re-sampling the particles. Image acquired from [http://www.mrpt.org/wp-content/uploads/2013/10/Resampling_system.png](http://www.mrpt.org/wp-content/uploads/2013/10/Resampling_system.png)

**Step 5:** The time update. Propagate all particles according to the ROV model described in Chapter 4.

\[ x^{i}_{k+1} \sim q(x_{k+1}|x^{i}_{k}, y_{k+1}) \quad \text{(5.7)} \]

as well as update the weights for all particles according to

\[ w^{i}_{k+1|k} = \frac{w^{i}_{k|k} \cdot p(x^{i}_{k+1}|x^{i}_{k})}{q(x_{k+1}|x^{i}_{k}, y_{k+1})} \quad \text{(5.8)} \]

where $q(x_{k+1}|x^{i}_{k}, y_{k+1})$ is the although this step would only be useful if the particle leaves the boundary of the map, accidentally ends up below or inside the DTM.
Results

The simulations give results that differ greatly depending on the environment and number of states to be estimated.

6.1 2-dimensional particle filter

The two dimensional particle filter works under the assumption that the IMU can give accurate enough readings concerning the yaw-angle of the vehicle. If this is the case then only the E,N - coordinates are left for the particle filter.

6.1.1 The pool

For two dimensions in the pool the initial distribution of the particles are shown in Figure 6.1. Because the yaw of the particles is handled by the EKF, the localization becomes quite robust. The first scan will reduce the particle distribution to a subspace according to the likelihood of the measurement, see Figure 6.2. If the position or attitude of the ROV have moved out of the subspace created by the first scan, then the next scan will reduce the subspace even further according to Figure 6.3. If the ROV is moving in a subspace where the measurements from sonars are expected to be similar, no new information is gained and the probability density function (PDF) of the position will widen. See Figure 6.4. When the ROV changes direction, new information is attained from the sonars, causing the PDF to consolidate as shown in 6.5.

The localization becomes difficult however when the ROV is close enough to the bottom of the pool, causing the sonars to not reach a wall. The only measurements the filter will get are then from the pool floor. This will have the effect that the expected readings for all particles except those close enough to pick up on the
Figure 6.1: Distribution of particles as well as the PDFs of E,N coordinates and yaw.

Figure 6.2: Distribution of particles as after the first scan. Particles expecting a wall close to their right side will have higher weight.

wall will receive the same weights and the filter will not be able to estimate the ROV position.
Figure 6.3: A scan giving readings that a wall is close to its left side, eliminates almost all particles except for the ones surrounding the ROV.

Figure 6.4: The ROV moves along the left wall, causing the PDF for the \( n \)-coordinate to become wider as the subspace along the wall will give the same sonar readings for all particles.

If the ROV is on a depth where it can take measurements that give new infor-
Figure 6.5: When the ROV makes another turn, new information about the subspace along the n-axis is received causing the PDF of the n coordinate to become smaller.

Figure 6.6: The particle filter estimation of e and n - coordinates.
6.1.2 Generated terrain

The pool environment is very simple and the simulation of sonar readings is much quicker compared to when ray casting techniques are necessary, as it is in this case. This had the implication that the number of particles used in the pool scenario have been substantially greater than it have been here. Due to this restriction, the global localization problem have not been simulated. Instead the initial distribution in this case have been around a known starting position. This assumption that the starting area is known means that the problem of a multimodal distribution have been substantially reduced.

The variance of the terrain coupled with the yaw estimation from the EKF does make it easy for the filter to estimate the position of the ROV. However, as the number of particles are smaller than for the pool case, the estimation of the position will not be as robust.

![Figure 6.7: Initial distribution of particles and yaw-angle in a generated environment.](image)

It is unfortunate that the global localization problem could not be tested on an environment resembling a seafloor. But assuming a known starting position is not an unreasonable notion, and the results from the particle filter position estimation are promising nonetheless. See Figure 6.8.

6.2 3-dimensional particle filter

Two major problems arise when the particle filter tries to handle estimation of the yaw angle as well as the E,N - coordinates. One is that the number of particles
that are needed to get an estimate on par with the 2-dimensional one, will have to be exponentially larger. If the number of particles are too few then localization may not be achieved due to divergence of the filter.

The second problem is that the chance for a multi-modal distribution of the particles will increase. Causing the position estimate from the filter according to (3.13) to end up as an average between the modals.

### 6.2.1 The pool

For 3 dimensions the distribution of particles in the pool is the same as for the 2-dimensional case. The exception being that in this case the yaw-angle is also uniformly distributed between 0 and $2\pi$ radians. This is illustrated in Figure 6.9.

When the yaw-angle is unknown the number of particle clouds become greater as particles with mirroring yaw-angles will receive similar measurements, see Figure 6.10.

In the simulations performed in this thesis the number of particles used are not enough to cover all possible positions and attitudes. Which means that simulations will have different results dependent on the initial random distribution of the particles. If the distribution is favorable then it will give the same results as the 2-dimensional filter as illustrated in Figures 6.11 and 6.12. If the distribution is unfavorable, then the particle filter will diverge and estimate an erroneous position or get stuck at a multi-modal distribution, as seen in Figure 6.13 and 6.14.

The only reason the position is estimated correctly in some simulations is because
6.2 3-dimensional particle filter

**Figure 6.9:** Distribution of particles as well as the PDFs of E,N coordinates and yaw. Note the difference in yaw PDF compared to Figure 6.1.

**Figure 6.10:** Distribution of particles as after the first scan. All corners are a plausible position for the ROV.

there are not enough particles to cover all possibilities. If enough particles could have been used in the simulations there would be no way for the particle filter
Figure 6.11: Distribution of particles after the first scan. Favorable distribution of particles leads to a high PDF around the actual location.

Figure 6.12: Particle filter position estimation in 3 dimension for a favorable initial distribution of particles. No real difference from the 2 dimensional scenario.

to differentiate the true position from its mirror positions. This means that a
Figure 6.13: Distribution of particles after 10 seconds. Unfavorable initial distribution causes multi-modal PDF at two positions, neither of which are the true position of the ROV.

Figure 6.14: Particle filter position estimation in 3 dimension for an unfavorable initial distribution of particles. The filter never converges.

faster filter, able to handle more particles would never be able to estimate the
true position of the ROV in a symmetric environment.
6.2.2 Generated terrain

Again, the number of particles used have been decreased compared to the pool, mitigating the problem of a multi-modal distribution. This is unfortunate as it would have been interesting to see how well the filter could estimate the yaw angle in a varied environment.

It is still interesting to see if the yaw-angle could be estimated along with the E,N-coordinates if the starting position is known. The filter does perform well in this case and it can quickly disregard particles moving in a direction not having the estimated measurements from the sonars. The performance for this approach is similar to the 2-dimensional case, but with a slightly larger error in the position estimation, see Figure 6.15.

![Particle filter position estimation](image)

**Figure 6.15:** Particle filter position estimation in 3 dimensions for generated terrain and known starting position.
7.1 Conclusions

The study is unfortunately severely lacking in physical testing, as such it cannot be confirmed that the methods used in the thesis are robust enough to be applicable to the real-world problem. It has shown promise being able to locate and track the ROV in a symmetrical environment such as a pool, if the magnetometer is working and the ROV is operating on a depth such that it can register more than just the pool floor. More complex environment actually gives the particle filter better performance as measurements are more varied. If this would be implemented in a physical system it is very important the map representation is accurate enough, otherwise the simulated sonar responses from the particles would not be of any use. The simulations are slow, especially the measurement step as every particle makes a sonar simulation at its position to get estimated measurements. A better algorithm or faster code is preferred if the filter is to run in real time on the ROV hardware.

Even though the number of particles that have been used is relatively small, and the initial localization for the generated terrain case have not been carried out, the work done has shown that it is possible to theoretically locate the ROV using a particle filter comparing distance and amplitude measurements from side scan sonars.
7.2 Future work

Every area of the ROV is in a very early stage. As such there is a lot of room for improvements on every aspect discussed in this thesis.

7.2.1 Further work with ROV sensors

Further studies should include fusing the side-mounted sonars with a forward looking one. This can help overcome the problems with a stochastic map representation of landmarks, as a forward looking sonar can make several measurements of the same object in rapid succession. It would be a logical step if the ROV is to be able to feature collision detection and avoidance.

7.2.2 Speeding up the filter

If further work in the simulation of the ROV is desirable, optimization of code is necessary to make it plausible to use the techniques described in this thesis. Ideally all the classes and functions should be rewritten in C or C++ to speed up performance.

7.2.3 Improving on the models

The map representation could be improved with horizontal layers with different water-pressure and temperature as well as salinity-levels to better make use of the Mackenzie equation (2.1). This would make a full implementation of Bell and Linnett [1997] plausible.

When physical testing has been carried out and a proper model verification has been made, the next step would be to implement SLAM techniques. There are a number of SLAM techniques that are compatible with the particle filter, such as the FastSLAM algorithm.

7.2.4 Higher resolution seafloor grids

This thesis has tried a few plausible cases for the environment. When the models and speed of the filter have been improved, simulations on real environments where the ROV actually could be tested in should be used. Such high resolution seafloor grids can be ordered from Sjöfartsverket [2013].
Appendix
This thesis has had an focus on the sonars supplied by SAAB underwater systems and the IMU-sensor as hardware.

### A.1 Sonars

The sonars are side scan sonar system from Marine sonic technology ltd, which was originally designed to to be used as a towed system i.e. the sonars would be mounted on a vehicile that is towed behind a boat, see figure A.1. However the system has successfully been used in other ROV/UAV applications.

![Figure A.1: Original usage of the sonar system](image-url)
Purpose:
The intent of this document is to answer many of the common questions regarding the typical installation of Sea Scan PC in an un-manned configuration (e.g. AUV, UUV, etc.). This document discusses the selection of support hardware/software and common installation problems.

This document is not intended to be an instruction manual for the installation of Sea Scan PC. Most systems of this type are custom installations. As such, we can not provide a single exacting set of installation guidelines.

Hardware:

**Figure A.2:** A block diagram depicting how the different modules of the SideScan sonar system works together
Quaternions are and can be used for a number of reasons, in this thesis however quaternions are used as a way to avoid the gimbal-lock problem and to provide a more numerically stable calculations to rotation of the ROV than euler-angles would.

Quaternions are most often represented in 4 dimensions as $q' = (q_s, q_i, q_j, q_k)$ or $q' = (q_s, q)$ where $q = (q_i, q_j, q_k)$. There are a number of operations one can do with quaternions, the foremost ones used in this thesis being:

**Multiplication:**

$$q' \odot p' = (q_s * p_s + q * p^T_s, q_s * p + p_s * q + q \times p) \quad (B.1)$$

**Conjugation:**

$$q'^c = (q_s, -q) \quad (B.2)$$

When used for rotation a quaternion is created for the vector being rotated about as follows:

$$q''_r = (\cos(\theta/2), \sin(\theta/2) * r) \quad (B.3)$$

where $r = (x_r, y_r, z_r)$ is the vector being rotated about and $\theta$ is the angle of rotation.

The created quaternion can now be used to rotate other vectors around vector $r$ by expanding the vector one wishes to rotate into 4 dimensions. If the vector $v = (x_v, y_v, z_v)$ is to be rotated, simply create $v' = (0, v)$ and then get the rotated vector by quaternion multiplication:

$$p' = q' \odot v' \odot q'^c \quad (B.4)$$
\( p' = (p_s, p) \) where \( p = (x_p, y_p, z_p) \) is the rotated vector.

If \( q' = (q_s, q) \) such that \( q_s = \cos(\theta/2) \) and \( q = \sin(\theta/2) \ast r = (q_i, q_j, q_k) \). Then we can express the rotation as in matrix form:

\[
P = \begin{pmatrix}
q_s^2 + q_i^2 - q_j^2 - q_k^2 & 2q_i q_j + 2q_s q_j & 2q_i q_k - 2q_s q_j \\
2q_i q_j - 2q_s q_k & q_s^2 - q_i^2 + q_j^2 - q_k^2 & 2q_j q_k + 2q_s q_i \\
2q_i q_k + 2q_s q_j & 2q_j q_k - 2q_s q_i & q_s^2 - q_i^2 - q_j^2 + q_k^2
\end{pmatrix}
\] (B.5)

So that the rotation of a vector \( v \) about \( r \) is done by:

\[
p = P v
\] (B.6)

Resulting in the rotated vector \( p \).
Bibliography


Upphovsrätt

Detta dokument hålls tillgängligt på Internet — eller dess framtida ersättare — under 25 år från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för icke-kommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/

Copyright

The publishers will keep this document online on the Internet — or its possible replacement — for a period of 25 years from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for his/her own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its www home page: http://www.ep.liu.se/

© Erik Ferm