NTNU

Norwegian University of
Science and Technology

Positioning of periodic acoustic emitters using an omnidirectional hydrophone on an AUV platform

Sverre Søbstad Løvskar

Master of Science in Cybernetics and Robotics
Submission date: February 2017
Supervisor: Jo Arve Alfredsen, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Problem Description

Acoustic fish telemetry constitutes a powerful scientific methodology for investigating the behavior and ecology of migrating fish and other aquatic animals in the marine environment. Experimental setups normally involve intraperitoneal tagging (using miniature acoustic transmitters) of a number of individuals that act as representatives of the fish population under investigation, while deploying an array of automatic acoustic listening receivers at fixed locations in the geographic area of interest. Signals from migrating tagged fish will then be picked up, time-stamped and recorded as the fish move within the detection range of the receivers, enabling subsequent reconstruction of the fish' migratory pattern. This approach is cost efficient compared to the alternative of manually tracking the fish, which demands extensive use or resources in terms of boats and skilled personnel over extended periods of time. However, the quality of data relies heavily on the actual selection of receiver locations, as well as the spatial resolution of the receiver array. In this project we propose a complementary approach by employing AUV as a mobile receiver platform for robotic fish localization and tracking, enabling observations of fish behavior on a finer scale than currently allowed by fixed receiver arrays, and without the high resource demand and perils associated with manual tracking. The focus of the project will be on developing techniques for estimating the transmitter position based on passive acoustic localization algorithms. The project consists of the following tasks:

- Literature survey of passive underwater localization and tracking techniques, with emphasis on single receiver, synthetic aperture array solutions
- Investigate the possibility and develop algorithm(s) for estimating transmitter position from AUV with the following assumptions (reflecting current limitations in transmitter/receiver technology):
  - Fish/transmitter location stationary, or moving slowly relative to AUV velocity
  - Transmitter characteristics: pulsed, single frequency, fixed or pseudo-random pulse repetition rate, pulse transmissions timed to whole second boundaries (relative to transmitter local clock), transmitter depth coded in pulse train, (uncertain) SNR vs. range relationship available
  - Receiver characteristics: omnidirectional, single frequency, pulses timestamped to a resolution of 1 ms, SNR calculated for each pulse reception AUV: known position (x,y,z) to a certain accuracy, moving freely
- Develop simulation scenarios to investigate the properties and limitations of the proposed algorithm(s) and discuss the results with respect to achievable localization and tracking accuracy, as well as the potential for improvements by enhancing AUV receiver configuration and capabilities (e.g. utilizing other measurements than time, etc.)

Contents

Abbreviations

AUV	=	Autonomous Underwater Vehicle
LAUV	=	Light Autonomous Underwater Vehicle
IMC	=	Inter-Module Communication Protocol
RT	=	Real-Time
CTD	=	Conductivity, Temperature and Depth
GPS	=	Global Positioning System
SNR	=	Signal-to-Noise Ratio
TDOA	=	Time Difference of Arrival
DUNE	=	Unified Navigation Environment

List of Figures

Summary

The origin position of an emitted signal is commonly determined using arrays of synchronized receivers or through the use of directional receivers. In this paper a single omni-directional hydrophone used to estimate the position of periodic acoustic emitters in three dimensions is proposed and studied. The theory behind the method and the alterations to the common range differencing problem to adjust for a single receiver is explained. An extended Kalman filter and a Particle filter are suggested to provide iterative estimators for the state estimation problem. The primary focus of this paper is on the performance and robustness of these filters using measurements available from a single observer. The system containing a moving receiver and periodic emitter was modeled and simulated to validate the theory and analyze the performance when put through different noisy scenarios.

Experiments in the field with commercially available hardware have been conducted. The first trial used a boat as platform for the moving receiver, and data was collected and post-processed. The second trial used an AUV as platform where data was processed on-board in real time. Results from these trials show the algorithms combined with the hardware setups produce satisfactory position estimates for the purpose of finer scale behavioral tracking.

Chapter 1

Introduction

A better understanding of the life and behaviors below the ocean surface can go a very long way. It can help understand the impacts of fishing, problems for aqua culture production and avoid inadvertently disrupting bio-marine habitats and ecosystems. Better behavioral knowledge can expand the foundation from which decisions are made, and could assist in forming better and more informed decisions.

Part of understanding behavior can be deduced from investigating how and where individuals of a species moves. For life above the surface GPS based trackers can be used, but below the surface these no longer work. For sub-surface applications acoustics is often used as the default means of communication. Small acoustic tags which function as beacons may be attached to a target. By using the precise time at which the emitted signal arrives at multiple known locations it can be worked out where the signal was sent from. Unfortunately acoustic emitters have a limited range at which they are reliably observable meaning a receiver must be within a range of only some hundreds of meters.

One commonly used approach for positioning tags is strategically placing stationary acoustic receivers in some grid pattern across some bounded area of interest. Here tag detections must be offloaded from each of the receivers and post-processed, or some sort of network with distributed processing is needed for real time results. This approach requires that the area of interest is decided ahead of time. If a target with high mobility is to be tracked over longer distances a common method is listening for the acoustic tags with a directional hydrophone and manually follow the signal with a boat. At any rate both of these methods can be prohibitively costly and labor intensive to be viable for researchers.

A more feasible solution could be an Autonomous Underwater Vehicle(AUV) which is deployed upon need and may operate continuously for several hours without extensive supervision. This should reduce the labor cost while also maintaining the mobility desired to track targets over longer distances. This approach requires fil-

tering algorithms and methods to automatically detect and position acoustic tags. Positioning using AUVs equipped with directional or stereo-hydrophone rigs is a subject which has already seen some research[11][4]. This paper will look into the potential of the more compact setup of an AUV equipped one single omni-directional hydrophone and its performance in positioning periodic acoustic tags.

The hydrophone used is a Thelma Biotel TBR700-RT omni-directional hydrophone along with Thelma Biotel acoustic tags. An OceanScan MST Light-AUV is the designated platform for the experiments. Both an extended Kalman filter and a Particle Filter are used and analyzed for the position estimation. The filters and their supporting code are implemented and analyzed in Matlab. After being verified they were implemented in C++ on an on-board embedded computer.

This paper will first dig into the theory behind positioning with periodic tags, then move onto how filters may be used to provide estimates from real world, noisy measurements. In Chapter 3 the AUV setup is simulated in Matlab. Results and lessons learned are presented here. Then a proof of concept sea water experiment is conducted in Chapter 4 and the results from that looked into. Lastly, in chapter 5 this is implemented on a Light-AUV and tested again.

Chapter 2

Theory

2.1 Means of Positioning

2.1.1 Hyperbolic Positioning

Time Difference of Arrival

Simply by measuring the time at which a signal arrives at multiple different locations it is possible to work out the position of the signal source. The only requirement here is a sufficient number of receivers with known locations and synchronized time between them.

Imagine an emitter at $[x_e, y_e]$ transmitting a signal to a receiver at $[x_1, y_1]$. The distance the signal needs to travel before reaching the receiver is

$$d_{e,s0} = \sqrt{(x_e - x_1)^2 + (y_e - y_1)^2}$$

If we assume the signal is traveling through some medium at a constant propagation velocity V_s then it will cover that distance in some time t.

$$d_{e,s0} = V_s t$$

Combining these leads to an expression for the time of flight, t_1, for a signal at position x_1 and y_1.

$$t_1 V_s = \sqrt{(x_e - x_1)^2 + (y_e - y_1)^2}$$

In this expression both the time of flight and position of emitter is unknown, and can therefore not be used directly for working out the position of the emitter. By adding an additional receiver which is synchronized with the first receiver it is possible to eliminate the time of flight variable and instead address the difference in arrival time at the receivers.

$$t_1 V_s = \sqrt{(x_e - x_1)^2 + (y_e - y_1)^2} \tag{2.1}$$

$$t_2 V_s = \sqrt{(x_e - x_2)^2 + (y_e - y_2)^2} \tag{2.2}$$

Subtracting equation 2.2 from equation 2.1 brings us to the equation for the Time Difference of Arrival(TDOA) which when multiplied with the propagation speed is commonly referred to as range difference.

$$V_s(T_1 - T_2) = \sqrt{(x_e - x_1)^2 + (y_e - y_1)^2} - \sqrt{(x_e - x_2)^2 + (y_e - y_2)^2} \tag{2.3}$$

where T_1 and T_2 are the absolute time of arrival as measured by the synchronized receivers. The TDOA expression when solved for $[x_e, y_e]$ yields a hyperbolic curve on which the emitter is located. This is illustrated in Figure 2.1a. Here two receivers Rx_0 and Rx_1 have received a signal from Tx, and the resulting TDOA Hyperbola is plotted. The emitter may be positioned at any point along the curve.

In order to derive a single solution for the emitter location multiple hyperbolas are needed to find a point of intersection. For each additional receiver that is added to the system another hyperbola can be made by comparing the difference to the first receiver as before. The set of equations for $N-1$ hyperbolas using N number receivers then becomes

$$V_s(T_m - T_0) = \sqrt{(x_e - x_m)^2 + (y_e - y_m)^2} - \sqrt{(x_e - x_0)^2 + (y_e - y_0)^2}$$

for $1 \leq m \leq N$.

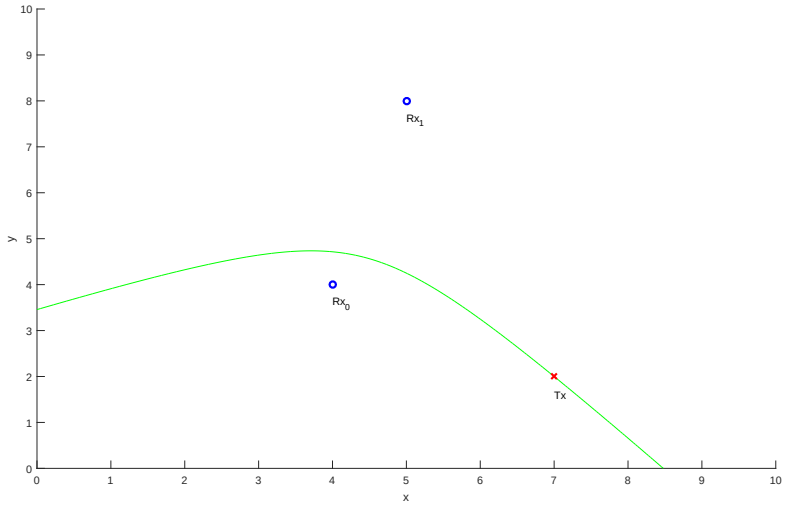
The plot in Figure 2.2 shows a set of TDOA Hyperbolas from which a single solution can be found.

Extending this to a three dimensional system is quite trivial, but it should be noted that doing so turns the 2D hyperbolas into 3D hyperboloids. The extra dimension means a minimum of three hyperboloids are required to locate a point in space and at least 4 receivers. The set of equations is still the same, but for brevity matrix notation will be used from here on.

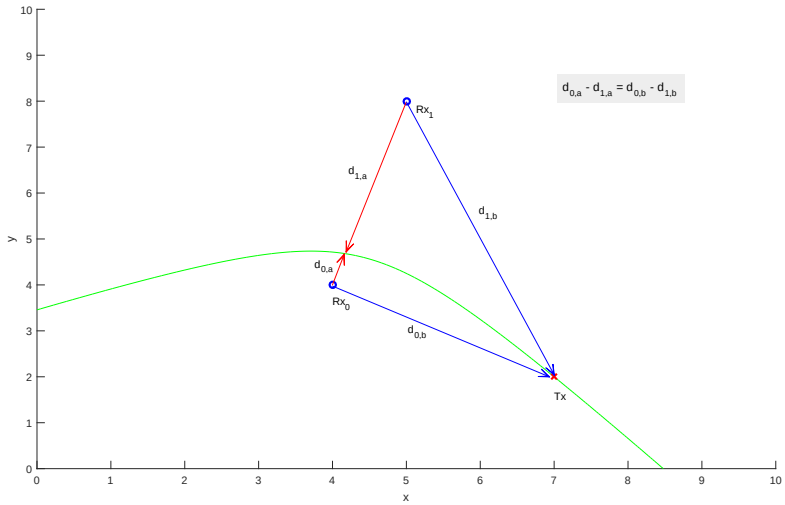
$$X_e = \begin{bmatrix} x_e \\ y_e \\ z_e \end{bmatrix} \; X_m = \begin{bmatrix} x_m \\ y_m \\ z_m \end{bmatrix}$$

$$V_s(T_m - T_0) = \left\|X_e - X_m\right\| - \left\|X_e - X_0\right\|$$

for $1 \leq m \leq N$.



(a) Emitter lies on the hyperbolic curve for the Time Difference of Arrival between Rx_0 and Rx_1.



(b) Distance difference is equal for all points along the hyperbole

Figure 2.1: Single hyperbola showing possible emitter locations

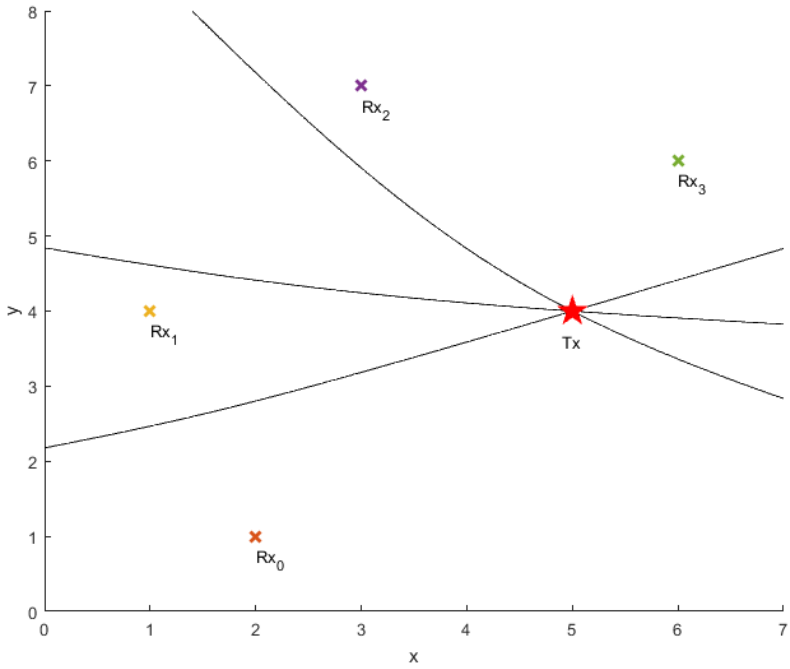


Figure 2.2: A single solution found at the intersection of multiple hyperbolas

Time Difference of Arrival using Periodic Emitters

Until now a setup with multiple synchronized receivers listening to a single emitter and recording the difference in time of arrival have been discussed. This paper proposes the use of a setup with a single moving receiver listening to emitters with known periodicity. This works in much the same way as explained above, but the time of arrival is synchronized on the single receiver by subtracting the known period of the emitter. The TDOA hyperbola equations deduced earlier are altered to include this.

$$V_s(T_1 - T_0 - P) = \|X_e - X_1\| - \|X_e - X_0\|$$

for $1 \leq m \leq N$ where P is the transmission interval of the emitter.

The setup with a single moving receiver is illustrated in Figure 2.3. After correcting for the emitter period such that the time of arrival appear synchronized the problem becomes quite similar to the common TDOA problem.

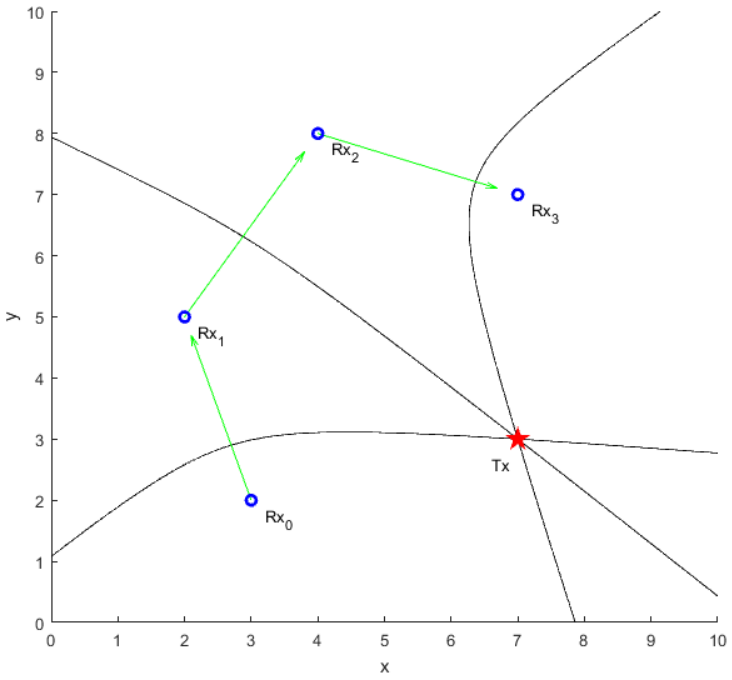


Figure 2.3: A moving receiver which receives a signal at multiple different locations and the resulting range difference hyperbolas.

Emitters can be produced with very precise timers while still preserving a low power consumption, and receivers with little power constraints may use external time synchronization such as GPS. Despite this some drift may be expected on both ends and to minimize the impact of temporal drift it is beneficial to compare the most recent measurement with the previous measurement. Temporal drift is normally a percentage of time, and keeping δt low should minimize the effect of drift. The emitter period can either be constant as used in this paper, or more follow a pseudo-random pattern to avoid collisions in an environment with multiple emitters.

Solving the hyperbole problem

For the TDOA hyperbolas to be useful in a positioning algorithm it is desirable to be able to computationally solve for the intersection of the quadratic hyperbolas. Using Bancroft's algorithm[1] or a later development by Bucher and Misra[2] it is possible to reduce the quadratic equation set to a inhomogenious linear equation. This is done through some clever rearranging and using one of the receivers to eliminate the square root terms. In turn that means at least 5 receivers are needed to solve the problem in 3 dimensions.

$$0 = Ax + By + Cz + D \tag{2.4}$$

where

$$A = \begin{bmatrix} \frac{2x_2}{V_s t_2} - \frac{2x_1}{V_s t_1} \\ \frac{2x_3}{V_s t_3} - \frac{2x_1}{V_s t_1} \\ \vdots \\ \frac{2x_m}{V_s t_m} - \frac{2x_1}{V_s t_1} \end{bmatrix}, B = \begin{bmatrix} \frac{2y_2}{V_s t_2} - \frac{2y_1}{V_s t_1} \\ \frac{2y_3}{V_s t_3} - \frac{2y_1}{V_s t_1} \\ \vdots \\ \frac{2y_m}{V_s t_m} - \frac{2y_1}{V_s t_1} \end{bmatrix}, C = \begin{bmatrix} \frac{2z_2}{V_s t_2} - \frac{2z_1}{V_s t_1} \\ \frac{2z_3}{V_s t_3} - \frac{2z_1}{V_s t_1} \\ \vdots \\ \frac{2z_m}{V_s t_m} - \frac{2z_1}{V_s t_1} \end{bmatrix}$$

$$D = \begin{bmatrix} V_s t_2 - V_s t_1 - \frac{x_2^2 + y_2^2 + z_2^2}{V_s t_2} + \frac{x_1^2 + y_1^2 + z_1^2}{V_s t_1} \\ V_s t_3 - V_s t_1 - \frac{x_3^2 + y_3^2 + z_3^2}{V_s t_3} + \frac{x_1^2 + y_1^2 + z_1^2}{V_s t_1} \\ \vdots \\ V_s t_m - V_s t_1 - \frac{x_m^2 + y_m^2 + z_m^2}{V_s t_m} + \frac{x_1^2 + y_1^2 + z_1^2}{V_s t_1} \end{bmatrix}$$

for $2 \leq m \leq N$

The linear equations may then be rearranged and solved for x, y, z.

$$\begin{bmatrix} A & B & C \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = -D \tag{2.5}$$

A common approach for solving this set, especially when over-determined, is using the Gauss-Newton algorithm for the least-squares problem. Care should be taken before solving that none of the time differences are zero values, in which case zero-division will cause the equations to break down.

Observability

Consider for moment the case where all the receivers are placed on a single line such that

$$y_{s,n} = ax_{s,n} + b$$

Since the receivers are all along a single line we can deduce the range to the emitter, but not the exact location of it as there is no single solution in this case. Attempting to solve this in the planar situation will allow two possible solutions for transmitter locations on either side of the line. This problem is illustrated in Figure 2.4 where the TDOA hyperbolas are plotted. Extending this into three dimensions yields an infinite set of solutions on a circle rotated around the line. Observing along a straight line should therefore be avoided.

Placing the observations on an ever so slightly non-linear curve eliminates this problem in ideal conditions as the hyperbolas on one side of the curve will no longer intersect and a single solution is available. However, once noise is introduced there might no longer be a simple solution on either side of the curve and the problem becomes one of finding the best match. In that case a close to linear path could potentially yield a result on the incorrect side of the curve. For the best results this should be taken into account when planning the path or by other means distributing the receivers.

If we imagine using a straight line trajectory from our starting position towards the emitter, then the emitter is also placed on the same line as the receivers. In which case the hyperbolas collapse and become a ray with a starting point at one of the sensors and stretching infinitely from there. The emitter direction is then known, but the distance to it is not. Using such a straight line trajectory towards the target it is not until the emitter is passed that it is possible to observe its location. This is another argument against using straight line trajectories.

For everything so far we have assumed the emitter being stationary, or at least relatively slow moving compared to the AUV and sampling rate. If that is not the case then TDOA positioning using a single, moving receiver may break down. If the emitter is moving towards the observer then the measured TDOA is shifted down and skewing the hyperbolas to lie closer to the most recent measurement position. The opposite is the case if the emitter is moving away from the observer. If the emitter velocity is sufficiently high then then hyperbolas might not overlap at all. This skewing effect can be canceled if one is able to know the emitter velocity.

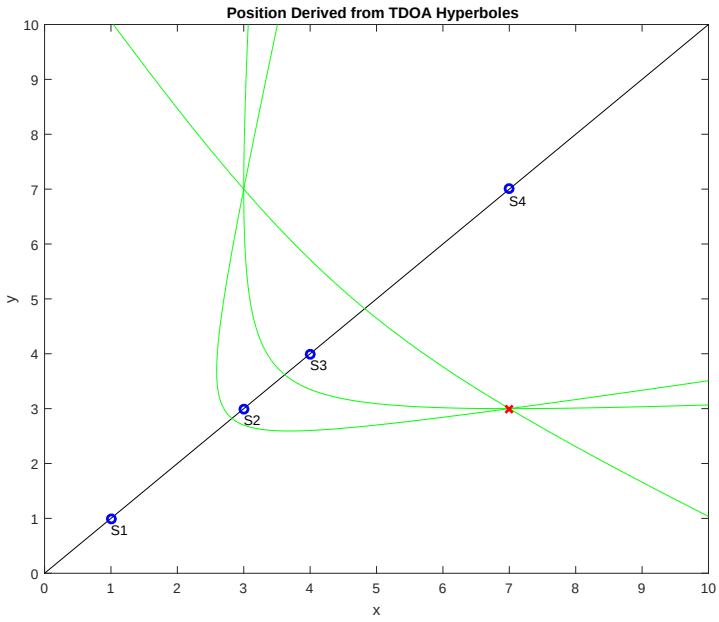


Figure 2.4: Hyperboles with all receivers on a line

An approach in correcting for this could be to measure the shift in the received frequency and comparing it the known emitted frequency and use that to calculate the relative velocity, which in turn can be used to cancel the effect of a moving emitter. This might however require equipment with capabilities outside the scope of this paper.

Temporal resolution

The range difference r_{tdoa}, calculated from TDOA and speed of sound is constrained by the following

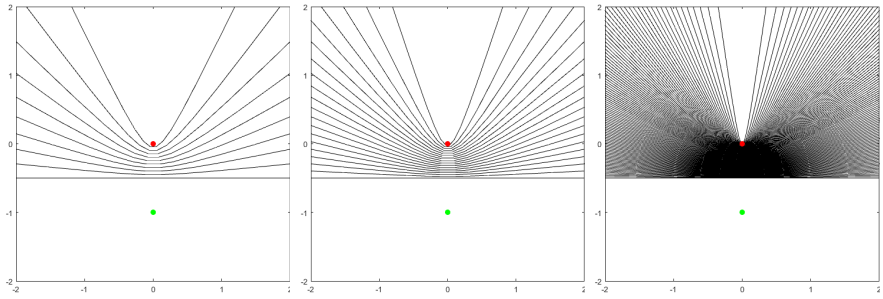
$$-\frac{\|x_2 - x_1\|}{2} \leq v\,t = r_{tdoa} \leq \frac{\|x_2 - x_1\|}{2}$$

Provided there are limitations in the temporal resolution then there is a finite set of values t may hold within this constrain. Full continuous time cannot be assumed as this is to be implemented on hardware operating with discrete time. This finite set of t in effect determines the precision of the range differencing. The preciseness of the differencing depend on two factors. The distance between the two receivers and the time resolution. Due to the geometry of the hyperbolas their spatial resolution is at its highest at $t = 0$ and decreases as the measured range difference approaches its bounds. This effect can clearly be seen in Figure 2.5 where all measurable

hyperbolas are plotted given different temporal resolutions.

$$0 \leq r_{\mathsf{tdoa}} < \frac{1}{2} \left\| x_2 - x_1 \right\|$$

Note the significant drop in spatial resolution for the hyperbolas closest to the observer (where $|r_{tdoa}| \to \frac{1}{2} \left\| x_2 - x_1 \right\|$).



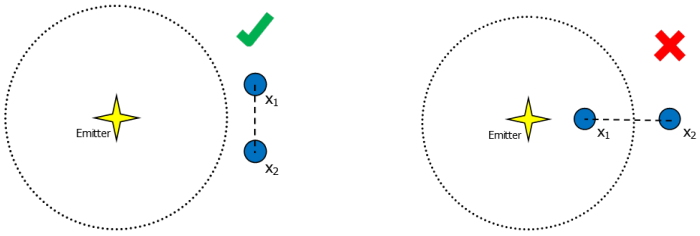
(a) 1x time resolution
10 possible hyperbolas

(b) 2x time resolution
20 possible hyperbolas

(c) 10x time resolution
100 possible hyperbola

Figure 2.5: Single hyperbole showing possible emitter locations

This effect should be accounted for when planning the layout of receivers or in this case the path of a moving receiver. Moving directly towards or away from the emitter should be avoided if possible. Instead it is beneficial to keep the observer layout close to tangent the signal wave front. Avoiding this of course implies already having some estimate of the emitter location which is not available in the initial stages.



(a) Most precise

(b) Least precise

Figure 2.6: Observer layout relative to emitter

2.1.2 Ranging using Signal Strength

Consider an emitter with some known, constant emitting power emitted from a source. The signal will dissipate as a function of distance traveled through the

medium from the source. By knowing the signal strength emitted and the rate at which it dissipates it should be possible to estimate the distance between the observer and the the emitter. Using this it is also possible to solve for the position

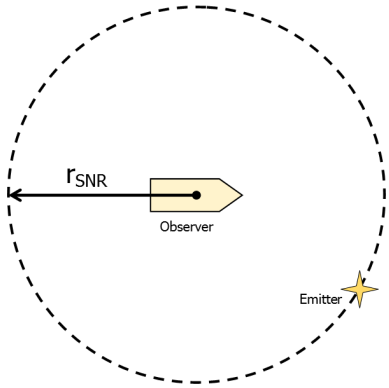


Figure 2.7: Range to emitter based on signal strength. The emitter may be found at some position along the perimeter of the circle.

of the emitter by taking several measurements from different positions. Provided perfect measurements the emitter will lie at the intersection of the circles. However, it cannot be assumed that the measurements are perfect as the measured signal travels through a medium with many non-observable variables. Sea water is not uniform and the observed signal strength may vary significantly due to path selection, variations in salinity, etc. Estimates of the distance may be made by assuming a dampening coefficient and treat variations as noise.

To convert signal strength to a range some function, $r = f(dB)$, for the signal dampening is needed. This function may be deduced by measuring the received signal strength at multiple different positions surrounding the emitter. The emitter is assumed to transmit at the same level every time. With a high number of measurements it should be possible to evaluate both the dampening effect over distance and also the variance in the measurements. If we assume noise to be Gaussian then we get a function for the range based on signal strength $r = f(dB) + w(t)$ where $w(t) \sim \mathcal{N}(0, q_{dB})$.

2.2 Filtering

For each new signal transmitted by the emitter another range difference(or TDOA) becomes available, and it is therefore desirable to develop an iterative filter based approach for position estimation which for each received signal may further improve the estimate. Two filters will be constructed and assessed in the following sections, the Extended Kalman Filter and the Particle Filter.

2.2.1 Extended Kalman Filter

The optimal Kalman filter requires a linear model and unfortunately the aforementioned TDOA hyperbolas provided by the measurements are quadratic, meaning the kalman filter cannot be used. The extended Kalman filter(EKF)[10][9][3] is an alternative for non-linear systems which works by linearising the model for each time step. This linearization of non-linear functions may cause less robustness in that the filter may not always converge correctly.

The model for the extended Kalman filter is

$$X_k = FX_{k-1} + \omega_k \tag{2.6}$$
$$z_k = h(X_k) + v_k \tag{2.7}$$

where the state is $X_k = (x_k, y_k, z_k)^T$ and the transition matrix for a stationary emitter becomes

$$F = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.8}$$

Assuming a stationary emitter may not be reasonable for fish as their movement might not be always negligible. In that case the model could be expanded to include velocity by setting $X_k = (x_k, \dot{x}_k, y_k, \dot{y}_k, z_k, \dot{z}_k)^T$, and modifying the update matrix to include a constant velocity based upon the sampling interval.

$$F = \begin{bmatrix} 1 & \Delta_x & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \Delta_y & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta_z \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.9}$$

After determining initial conditions the filter progresses through the following steps to produce an estimate.
Time update

$$\hat{X}_k^- = F\hat{x}_{k-1} \tag{2.10}$$
$$P_k^- = FP_{k-1}F^T + Q \tag{2.11}$$
$$\tag{2.12}$$

Measurement update

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \tag{2.13}$$
$$\hat{X}_k = \hat{X}_k^- + K_k(z_k - H\hat{X}_k^-) \tag{2.14}$$
$$P_k = P_k^- - K_k H P_k^- \tag{2.15}$$

Where H_k is the Jacobian of the measurement function $h(X_k)$ evaluated at step k.

$$h_k(X) = \left\|X_e - X_{rx1}\right\| - \left\|X_e - X_{rx0}\right\| \tag{2.16}$$

$$\begin{aligned}\hat{H}_k &= \frac{\partial h_k(X)}{\partial X} \\ &= \frac{(\hat{X}_k^- - X_{rx1})^T}{\left\|\hat{X}_k^- - X_{rx1}\right\|} - \frac{(\hat{X}_k^- - X_{rx0})^T}{\left\|\hat{X}_k^- - X_{rx0}\right\|}\end{aligned} \tag{2.17}$$

which when rewritten to matrix form is

$$\hat{H}_k = \begin{bmatrix} \frac{\hat{x}_k - x_{rx1}}{v} - \frac{\hat{x}_k - x_{rx0}}{u} \\ \frac{\hat{y}_k - y_{rx1}}{v} - \frac{\hat{y}_k - y_{rx0}}{u} \\ \frac{\hat{z}_k - z_{rx1}}{v} - \frac{\hat{z}_k - z_{rx0}}{u} \end{bmatrix} \tag{2.18}$$

with

$$\begin{aligned} v &= \sqrt{(\hat{x}_k - x_{rx1})^2} + \sqrt{(\hat{y}_k - y_{rx1})^2} + \sqrt{(\hat{z}_k - z_{rx1})^2} \\ u &= \sqrt{(\hat{x}_k - x_{rx0})^2} + \sqrt{(\hat{y}_k - y_{rx0})^2} + \sqrt{(\hat{z}_k - z_{rx0})^2} \end{aligned}$$

The extended Kalman filter can be augmented to include ranging estimates based on the signal-to-noise ratio. This introduces a second measurement function for the range between emitter and receiver, which is written

$$h_{2,k}(X) = \left\|X_e - X_{rx1}\right\| \tag{2.19}$$

The Jacobian for the new measurement equation is

$$\begin{aligned}\hat{H}_{2,k} &= \frac{\partial h_{2,k}(X)}{\partial X} \\ &= \frac{(\hat{X}_k^- - X_{rx1})^T}{\left\|\hat{X}_k^- - X_{rx1}\right\|}\end{aligned} \tag{2.20}$$

leading to the final Jacobian for the system with both range differencing and absolute ranging.

$$\hat{H}_k = \begin{bmatrix} \frac{\hat{x}_k - x_{rx1}}{v} - \frac{\hat{x}_k - x_{rx0}}{u} & \frac{\hat{x}_k - x_{rx1}}{v} \\ \frac{\hat{y}_k - y_{rx1}}{v} - \frac{\hat{y}_k - y_{rx0}}{u} & \frac{\hat{y}_k - y_{rx1}}{v} \\ \frac{\hat{z}_k - z_{rx1}}{v} - \frac{\hat{z}_k - z_{rx0}}{u} & \frac{\hat{z}_k - z_{rx1}}{v} \end{bmatrix} \tag{2.21}$$

$$\begin{aligned} v &= \sqrt{(\hat{x}_k - x_{rx1})^2} + \sqrt{(\hat{y}_k - y_{rx1})^2} + \sqrt{(\hat{z}_k - z_{rx1})^2} \\ u &= \sqrt{(\hat{x}_k - x_{rx0})^2} + \sqrt{(\hat{y}_k - y_{rx0})^2} + \sqrt{(\hat{z}_k - z_{rx0})^2} \end{aligned}$$

$$Q = \begin{bmatrix} q_{11} & 0 & 0 \\ 0 & q_{22} & 0 \\ 0 & 0 & q_{33} \end{bmatrix} \tag{2.22}$$

The process noise in the system should be assumed non-existing as the emitter location is stationary. Q cannot be a null matrix however as that would cause singularities. q_{33} should be set to some small positive value to avoid this.

$$R = \begin{bmatrix} r_{11} & 0 \\ 0 & r_{22} \end{bmatrix} \tag{2.23}$$

Measurement noise, R, must be determined from on the quality of the measurements from the hydrophone and GPS. If the filter does not benefit from absolute ranging measurements then a scalar may be used instead such that $R = r_{11}$. r_{11} states the variance of the measured range difference. Time precision, inconsistencies in propagation speed, positioning error all factor in here. r_{22}, if used, contains information about quality of the measured signal to noise ratio. Determining proper values for r_{11} and r_{22} is important for the behaviour of the filter and will be further discussed in later sections.

The Kalman filter must be initialized with an estimate which should be of reasonable accuracy to avoid improper convergence. The method used here to provide an initial estimate for the filter is to utilize the first 5 or 6 measurements to find an approximate solution using algorithms described in the theory section.

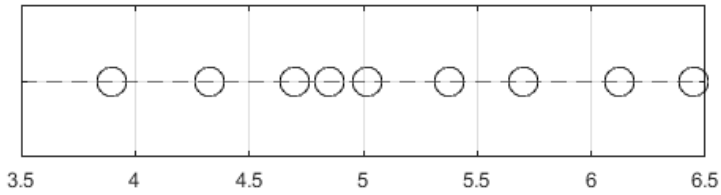
2.2.2 Particle Filter

The particle filter operates by assuming numerous different possible states spread throughout the state space. These different states are called particles. A measurement is then used to evaluate the probability of each of particle being the true state of the system. The particles are then assigned weights according to their probability. The less likely particles are then shifted, or re-sampled, towards the more likely particles. The mean value of the particles after re-sampling is the output estimated state. Once a new measurement becomes available the process is repeated. While one could simply discard the less likely particles and carry on with the most promising ones the shifting is important to avoid depletion.

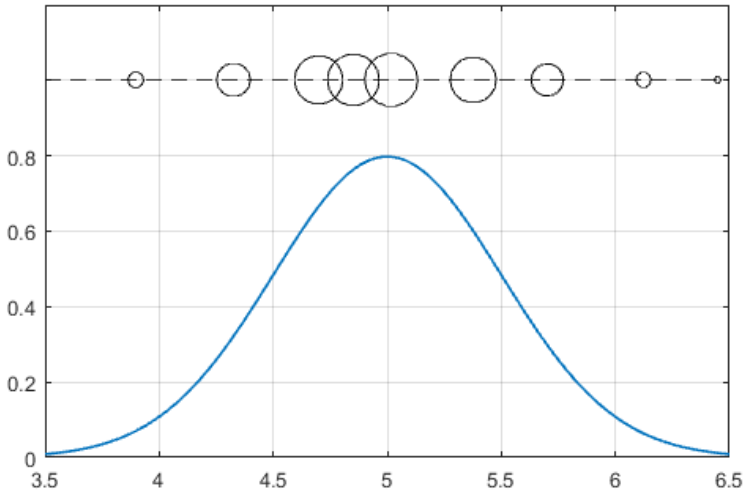
In short the particle filter can be summarized in the following steps. This is also illustrated in Figure 2.8.

1. Initiate a set of N particles spread across some space.
2. Calculate particle weights from the probability of each particle given some measurement.
3. Normalize weights
4. Re-sample particles based on their weights.

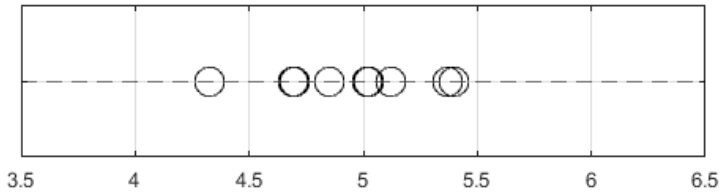
5. Extract estimate from the posteriori particles (e.g. mean of particle).
6. Reset weights and repeat from step 2.



(a) Particles initially distributed along some line



(b) Measurement becomes available and particles are assigned weight according to some probability distribution



(c) Particles are grouped through re-sampling. Note that the total number of particles remain constant and the weights are reset

Figure 2.8: Core steps of the particle filter

One of the strengths of the particle filter is not needing to specify an initial state. The particle needs only an initial state space region to search within and a probability distribution for the measurement. The state space region could simply be set

as some radius surrounding the AUV. If there are more than one likely state then the particles may cluster in multiple groups around those probable states. This will remain the case until one of the states distinctly proves to be more likely than the others. This feature of convergence towards multiple states lets an implementation work with several possible estimates rather than picking one which may be very helpful. The Kalman filter in a similar case would simply return a single state with an uncertainty.

The downside to the particle filter is that it is computationally heavy. For each time step several iterations over each particle is required. While the Kalman filter has constant execution time the particle filter $\mathcal{O}(n)$ or even $\mathcal{O}(n^2)$ for n particles depending on the re-sampling algorithm. It also requires sufficient memory to hold all the particles. This growth in complexity must be taken into consideration when choosing the number of particles.

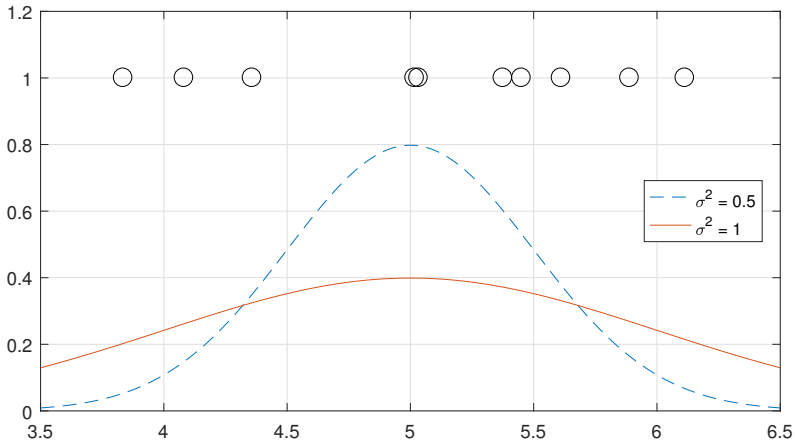


Figure 2.9: Two normal probability distributions of different variances demonstrating weight assignment to particles.

$$f(x|\mu) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.24}$$

where μ is the mean and σ^2 is the variance of x.

Just as for the Extended Kalman filter some information about the statistical characteristics of the measurement is needed for correct behaviour of the filter. The variance in a Gaussian white noise signal, σ^2, determines the shape of its distribution curve. For a measurement which is precise and has low variance a probability distribution with a sharp peak should be used for efficient convergence.

If the measurement on the other hand has a large amount of noise then considerations must be taken for the larger variance in the signal and a more flat probability distribution should be used. Essentially this implies a trade-off between rapid convergence and robustness of the filter. This is visualized in Figure 2.9 where the flatter distribution allows for less distinct weights and a less aggressive grouping in the re-sampling step.

$$h_{k,1} = \left\|Xp_i - X_{rx1}\right\| - \left\|Xp_i - X_{rx0}\right\| \tag{2.25}$$

$$h_{k,2} = \left\|Xp_i - X_{rx1}\right\| \tag{2.26}$$

And

$$p_{rd}(x_k|h_{k,1}) = \frac{1}{\sqrt{2\pi\sigma_{rd}^2}}\, e^{-\frac{(z_{rd}-h_{k,1})^2}{2\sigma_{rd}^2}} \tag{2.27}$$

$$p_r(x_k|h_{k,2}) = \frac{1}{\sqrt{2\pi\sigma_r^2}}\, e^{-\frac{(z_r-h_{k,2})^2}{2\sigma_r^2}} \tag{2.28}$$

The weight for each particle is calculated by inserting equation 2.25 into equation 2.27 and then normalized.

$$w_i = \frac{p(x_i|h_k)}{\sum_{j=1}^{N} p(x_j|h_k)} \tag{2.29}$$

Resampling

The resampling step in particle filters preform two important functions. First it avoids particle depletion. If particles were not redistributed then the less likely particles would be removed completely, and the number of particles would decrease. Resampling for particle filters is a whole topic of research in itself, however for this paper only one resampling method is used and studied. Methods of resampling is further discussed in [9] and [6].

For each particle x_i in the set $i = 1, 2, ..., N$, generate a random uniformly distributed value $u_i \sim U[0, 1)$. Create the cumulative sum of the particle weights such that

$$cw_i = \sum_{j=1}^{i} w_j$$

Reassign the particles in the set such that

$$x_i := x_j : \ cw_j < u_i$$

Less formally stated this means to go through each particle in the set from x_1 through x_N, and for each assign $x_i := x_j$ when the random generated value u_i is greater than the cumulative sum up until w_j. Meaning the particle x_i is made identical to some other particle x_j in the set which weight triggers the condition $cw_j < u_i$.

2.3 Discussion

As previously mentioned the precision of the range differencing depends on the angle at which the signal is received. An interesting idea which has not been further investigated in this paper is making the filters adaptive to the relation between estimate and the two observing positions. The measurement variance may be assumed higher when the current estimate is close to lining up with the observer trajectory and lower when it is orthogonal. It can be seen in some of the trajectories from the trial shown later that the estimator struggles when headed straight toward or straight away from the emitter. Making the filter more conservative during these portions of the trajectory could remedy this issue.

Chapter 3

Modeling and Simulation

It is a well established phenomenon that ideas which look good on paper might not always be successful when accounting for measurement noise. For that reason it was desirable to create a simulator not only to verify the theory, but also to check the robustness of the algorithms when subjected to noise and hardware constraints.

3.1 Method

In the setup there are two entities moving independently of each other. The first is the AUV, or rather the receiver, which is dynamically moving through the water at changing course and speeds. The other are the sound waves emitted from the acoustic tag which are propagating through the water. When these intersect the signal is considered detected and a time is noted.

A state space model, $\dot{x}(t) = Ax(t) + Bu(t)$, of a moving object in 3 dimensions was set up where $x = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^T$. A maximum speed limit was set meaning it could not accelerate beyond the AUVs maximum forward speed of 4 knots or roughly 2 m/s.

An emitter was then placed at some predetermined but arbitrary position and the state space model of the AUV given initial conditions placing such that it is in range of the emitter. The sound wave was set to begin propagating at t_0, and repeat at provided intervals. In order to find the intersection between signal and AUV the differential equation must be solved and compared with how far the signal has propagated. This was solved numerically using Matlab's ordinary differential equation solver, ODE45(), which is an implementation of explicit Runge-Kutta methods of 4th/5th order. This solver is also capable of detecting events while it is running which may be boundary conditions such as a ball hitting the ground and then changing direction. In our case the event trigger is the signal arriving at our moving receiver. This allowed a simulation of a dynamically moving vehicle as well as a signal propagating towards the moving vehicle from a stationary emitter.

Next came writing the filtering algorithms. Both filters were implemented as functions calls which took two observation positions and a range difference between them as input and returned a position estimate. For the Kalman filter one additional function had to be written to produce an initial estimate based on the first N measurements. The function attempts to find a solution to the over-determined linear equations discussed in the theory section.

For the first few runs no measurement noise was added so that correct behavior of the simulator and filter implementations could be verified. After removing a few bugs and confirming the algorithms were performing as expected then noise was introduced to the measurements to see how robust the filtering methods were. This simulation was intended as a test to see if the algorithms could feasibly be implemented using existing available hardware. Therefore it was most interesting to see if the filter algorithms could hold up when realistic amounts of measurement noise was introduced. Some of the measurement noise factors were known ahead of time, most central of which was time precision of the TDOA. Ideally this should be of unlimited resolution which would have yielded excellent position estimates every time. However the receiver available has finite precision where received messages are time-stamped with millisecond precision. One of the most interesting things to look at in simulations were the effect of limited precision time-stamps. In simulation this is dealt with by rounding the numerically solved time of arrival to the desired precision. Also added in was worst case clock drift on the emitter side. Another noise factor is the precision of the GPS position recorded at the time of signal arrival. Gaussian white noise was added to the position to simulate this measurement noise from the GPS.

3.2 Results

3.2.1 S-curve Trajectory

For this simulation the AUV starts off 200 m west and 100m south of the true emitter position. It travels northward following an s-curve trajectory with a maximum speed of 3 knots. Only the TDOA, or range difference, were utilized as measurement here. The transmitter emits every 8.00 seconds with a clock drift of 20ppm(0.002%). The time of arrival at the AUV is rounded down to the nearest millisecond to match the 1 ms precision of the hydrophone. GPS measurement noise is not enabled. This means the only noise introduced is caused by the time precision limit. Speed of sound is set to 1485 m/s. The simulation stops after 340 seconds.

Extended Kalman Filter

The Extended Kalman Filter is initialized after the first 6 measurements. At that point an initial estimate is calculated, and the Kalman filter attempts to improve that estimate for the remaining measurements. A map of the trajectory and the estimated positions are shown in Figure 3.1. As there is no noise introduced the

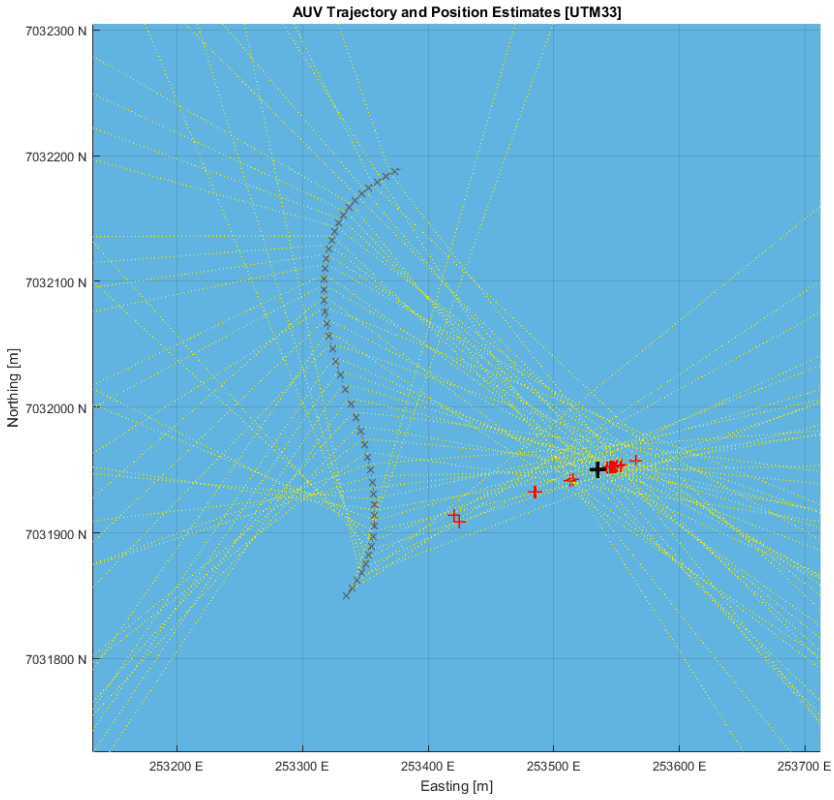


Figure 3.1: Simulated trajectory of the AUV and corresponding estimated emitter positions. The true position of the emitter marked with a black $+$, AUV position at the time of a received packet marked with grey x and estimated positions marked with red $+$. The dotted lines show the cross section of hyperboloids at the sea surface level.

Kalman filter is fully deterministic in this case and yields the same result every time.

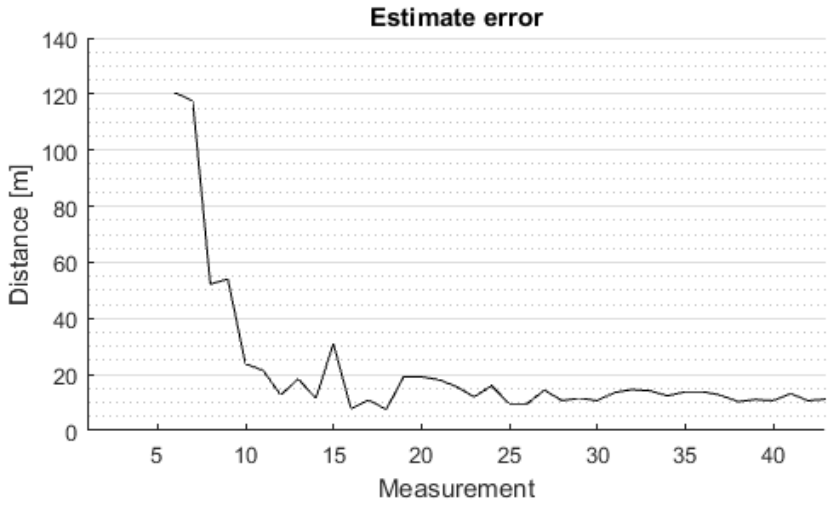
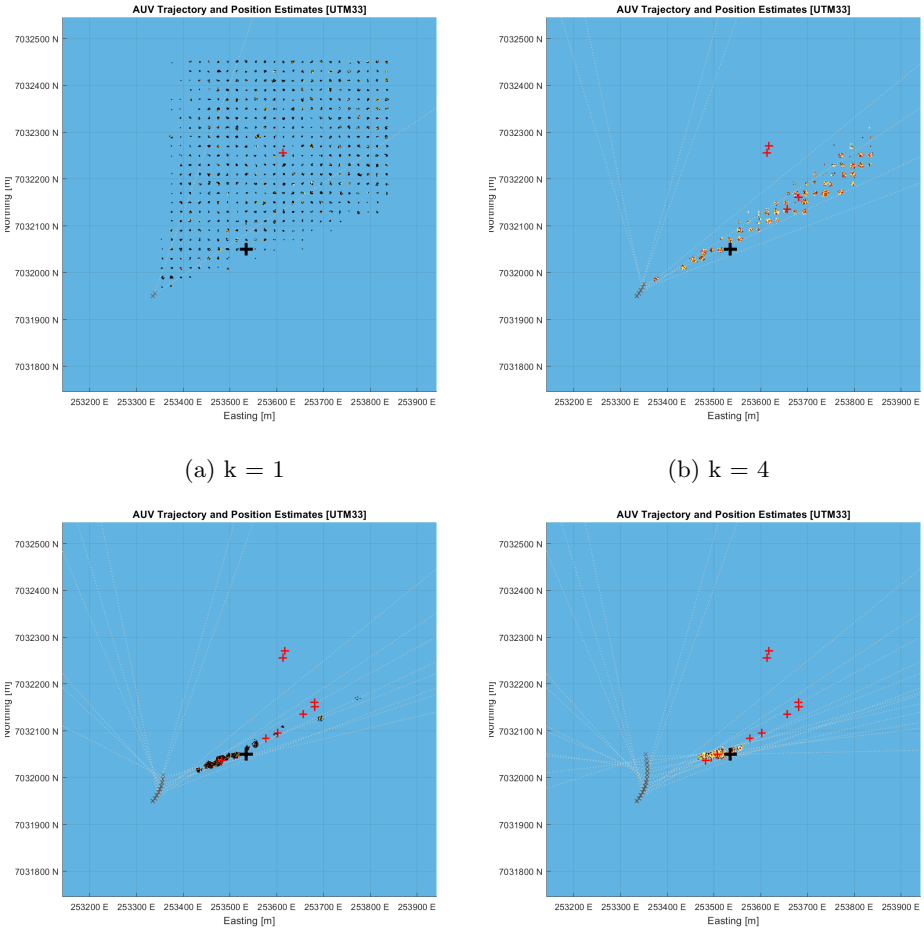


Figure 3.2: Positioning error for each filter update in terms of Euclidean norm. Kalman filter initialized after 6 measurements.

Particle Filter

The particle filter was here initialized with 5202 particles spread uniformly over a cubic bounding box measuring 1000m North and East and 20m high centered at the AUVs position. This means the particle filter was only looking for emitters within a 500m range north or east. Estimating starts when the first range difference is available after two received pings. The estimate deduced from the particles is here simply the mean of the particle positions. The progression of the particle filter with the estimate produced is shown at several time steps in Figure 3.3.



(a) k = 1

(b) k = 4

(c) k = 8

(d) k = 13

Figure 3.3: Particles shown from above after k iterations. The color of the particles indicate their current assigned weight ranging from black to yellow, where yellow is most significant.

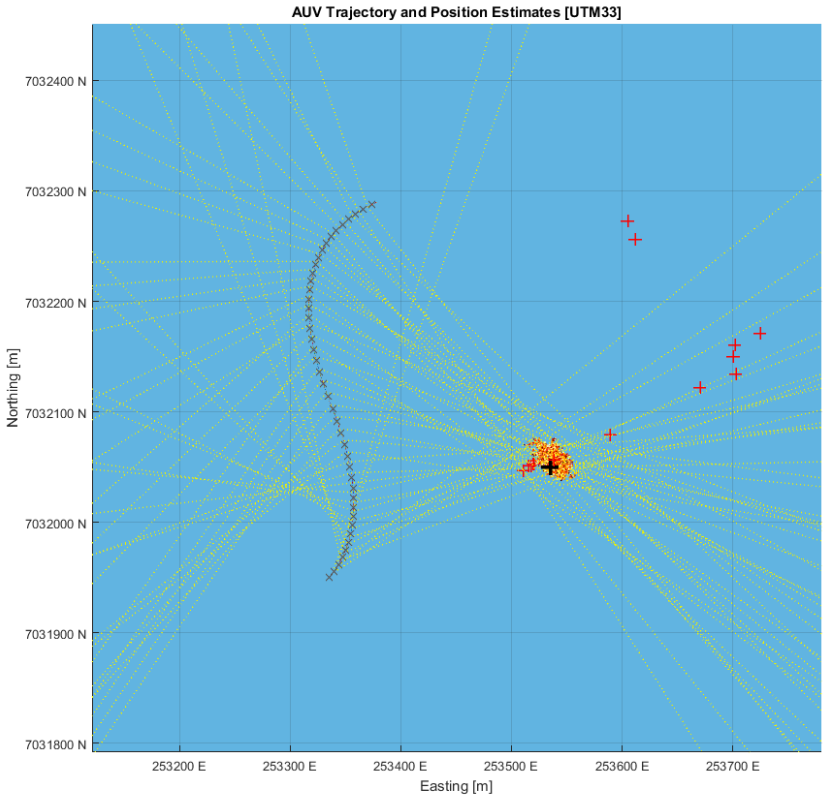


Figure 3.4: Same trajectory for the AUV and estimated positions. A cloud of particles can be seen surrounding the black $+$.

The particle filter utilizes random variables in the re-sampling stage and results will therefore vary to some degree even when there is no noise.

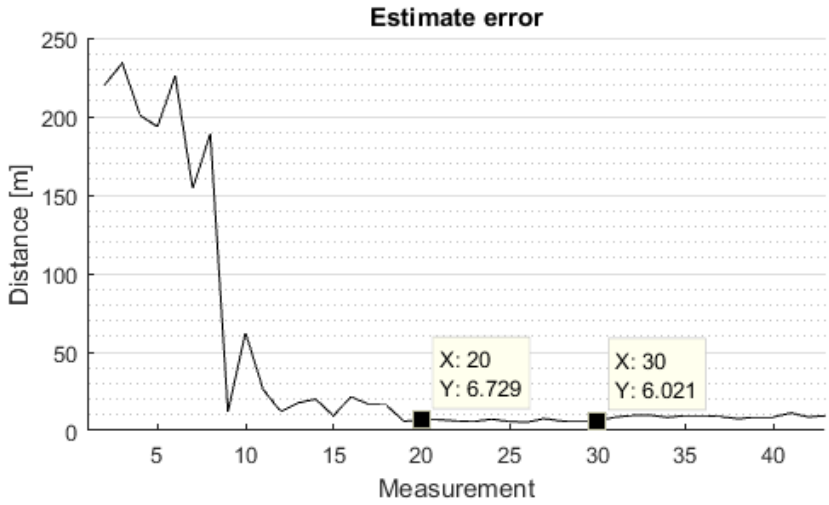


Figure 3.5: Paticle Filter estimation error for the simulation run shown in Figure 3.4.

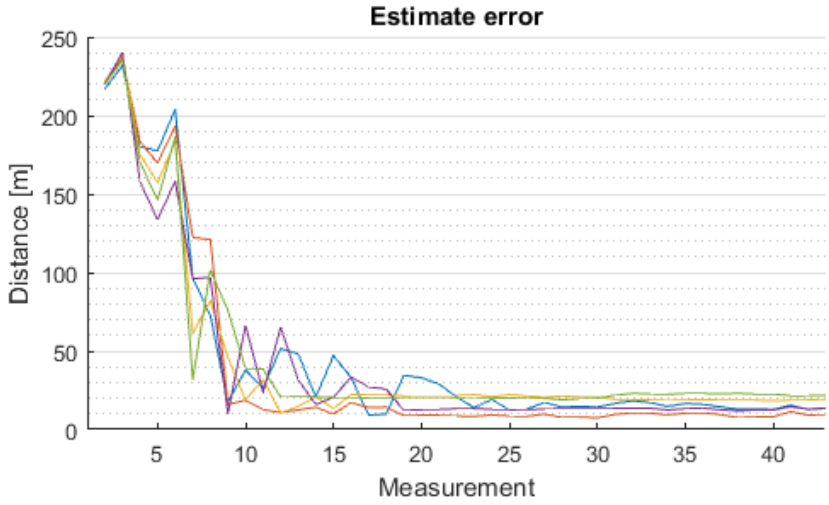


Figure 3.6: The non-deterministic results of 10 simulations with identical initial conditions.

3.2.2 Convex Trajectory

In these simulations the AUV has an initial position 100m east and 200m south of the emitter. Its heading is northbound and turning port, ending up westbound. GPS noise is enabled and set to a maximum deviance of 0.5m and then later to 2m. The measurement variance has been increased for both filters to compensate for this. All other variables remain unchanged from the previous simulation.

Extended Kalman Filter

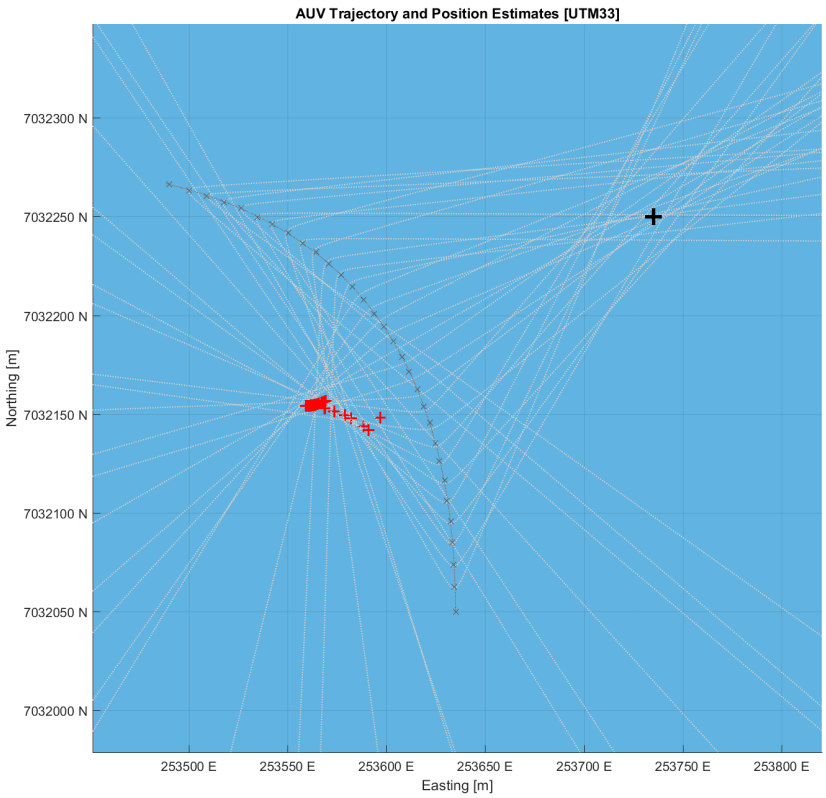
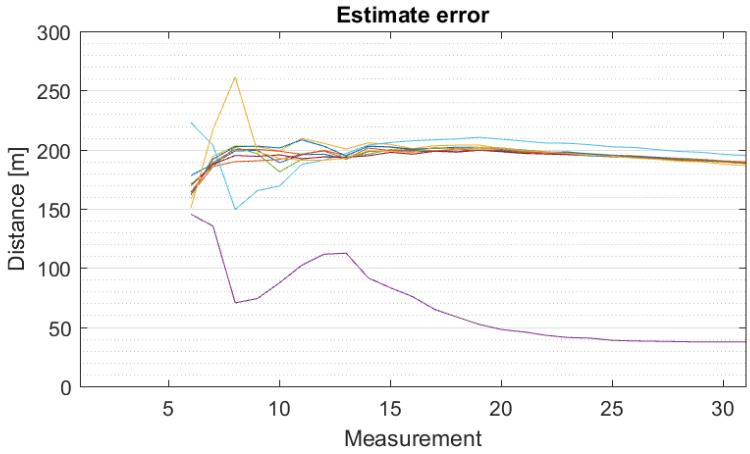
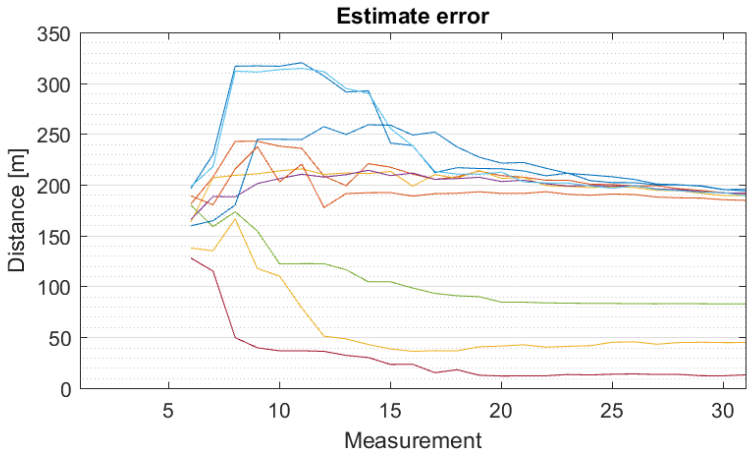


Figure 3.7



(a) AUV Position deviance: 0.5m.



(b) AUV Position deviance: 2m.

Figure 3.8: Estimation error for 10 runs using same conditions. Kalman filter initialized after 6 measurements.

Particle Filter

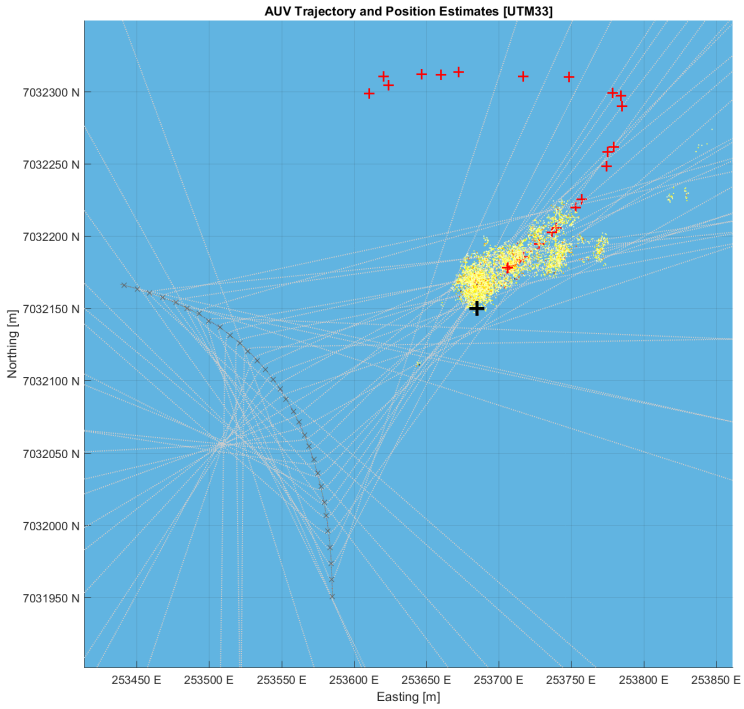


Figure 3.9

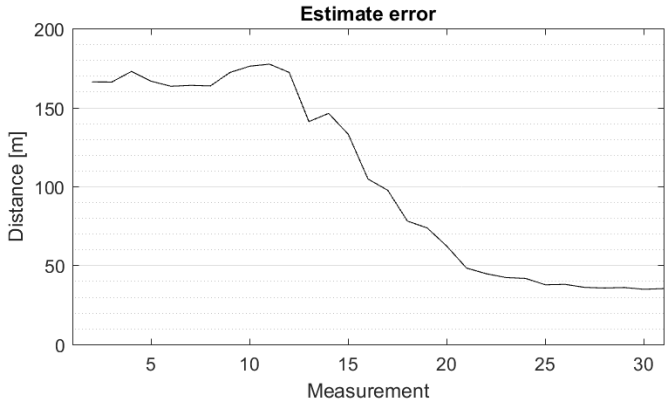
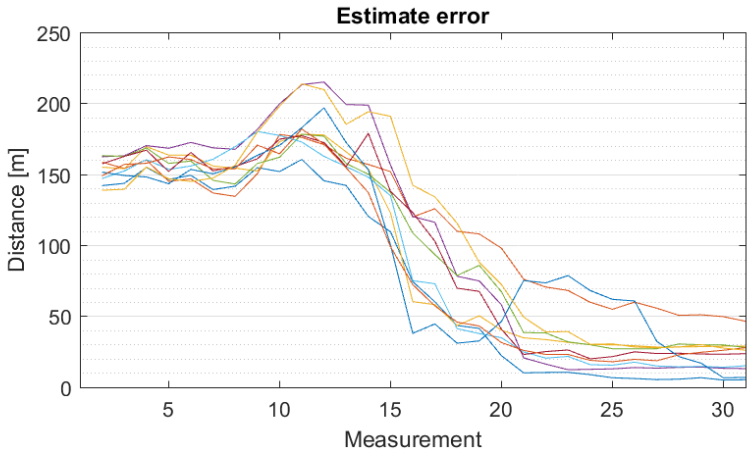
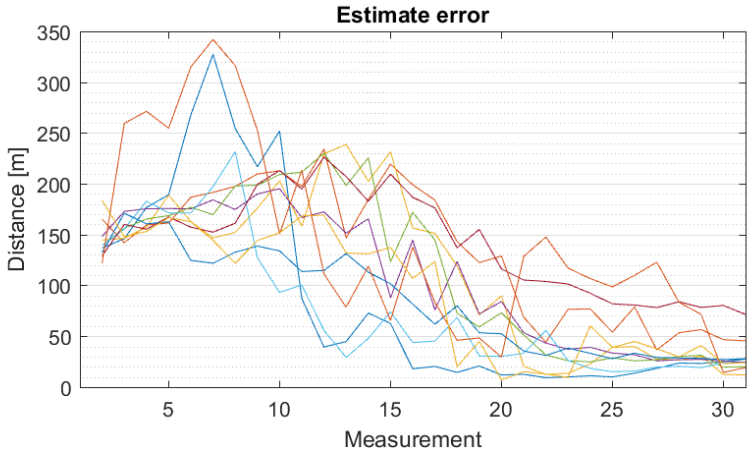


Figure 3.10



(a) AUV Position deviance: 0.5m



(b) AUV Position deviance: 2m

Figure 3.11: Estimation error for 10 runs using same conditions. Kalman filter initialized after 6 measurements.

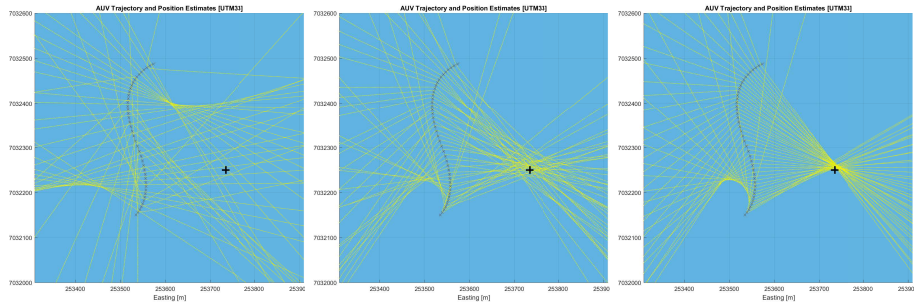
3.3 Discussion

The results from the simulations show that positioning of emitters with either the Kalman filter or particle filter may be entirely feasible with the constraints imposed by hardware. It was a concern that the millisecond precision time-stamps on the receiver would not be accurate enough for the algorithms. The results from the simulation merit further experiments using actual hardware.

The results also show that some trajectories, particularly the convex one, require more care than others. By looking at plot in Figure 3.7 one can see intersections of the hyperboloids on both sides of the trajectory, and that the Kalman filter often falsely assumed the intersection on the left was the correct one. The particle filter with its ability to maintain multiple clusters of particles over some time dealt better with this situation.

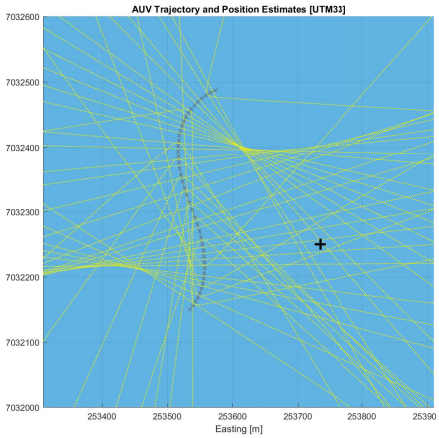
Some simplifications were done in the simulated model. For one it was assumed that the sound propagates at a constant speed in all directions from the emitter making the sound wave front a perfect circle with increasing radius over time. This is most likely not the case as sea water is inhomogeneous when it comes to sound speed. Another simplification is assuming all signals emitted are detected by the hydrophone. At no point is the signal considered too weak to be detected nor are any signals simulated as lost. The receiver always stayed within the known detectable range of the emitter during the simulations. Lost signals were for that reason considered a rare occurrence, but it should be noted that in the real world signals might not always be detected by the receiver even when in range.

An interesting observation during simulation was the importance of precision in time-stamping. Moving from 1 millisecond precision to 100 microseconds, i.e. improving precision by a factor of 10, drastically improved results. Talks were had with the manufacturer of the hydrophone in which they expressed higher precision being possible, but at the cost of higher

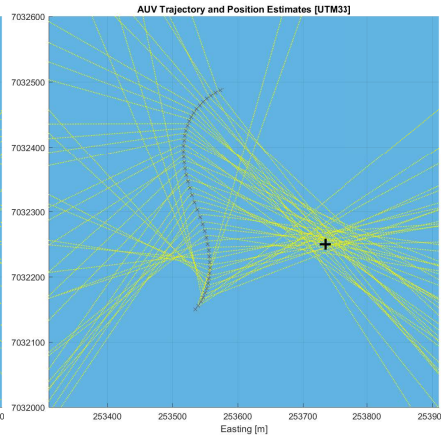


(a) 10 ms (b) 1 ms (c) 0.1 ms

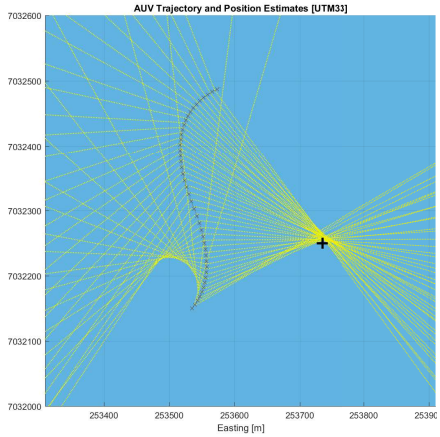
Figure 3.12: The effect of time precision on TDOA hyperboloids



(a) 10 ms



(b) 1 ms



(c) 0.1 ms

Figure 3.13: The effect of time precision on TDOA hyperboloids

Chapter 4

Sea Water Experiment

Results always get more interesting when moving away from assumptions and moving onto actual hardware implementations and true data. This sections presents an experiment conducted using a boat as platform and a hydrophone combined with a GPS-reciever, all feeding data into algorithms running in Matlab.

4.1 Preparations

(a) Thelma Biotel TBR700 with RT connector

(b) GPS Synchronization Module

Figure 4.1: Hydrophone and GPS Sync Module

The hydrophone used in the experiments is the TBR700-RT developed by Thelma Biotel in Trondheim. It is capable of time-stamping received messages from tags with 1 millisecond precision. The hydrophone can optionally be deliv-

ered with a real time interface over RS-485 as shown in Figure 4.1a. A GPS-synchronization module developed at NTNU for another project was also used. The GPS-synchronization module has two important functions. First being that it uses GPS time to synchronize the internal clock of the hydrophone which allows for distributed setup of hydrophones with synchronized clocks. The second function is appending a GPS fix to each message received by the TBR700. Using RS-232 the GPS-sync module transmits a text string containing time, latitude, longitude and some other support data as shown here.

$$\begin{aligned}&\$\mathrm{TBR},000025,1477645345,310,\mathrm{S256},40,1,20,10800,\ldots\\&\qquad 6325.12363,\mathrm{N},01024.07033,\mathrm{E},1,12,1.1,*3\mathrm{D}\end{aligned}$$

The TBR700 was hooked up to a computer through a GPS-sync module, and some dry runs were carried out to verify correct behavior from the Matlab code. Messages needed to be checked for validity using a checksum, correctly parsed and then passed onto the filtering algorithm.

Figure 4.2: The acoustic emitter used in the experiment. One of the largest emitters produced by Thelma Biotel customized with a static 7 second transmission interval.

Also tested before carrying out the field test was the consistency the interval between transmissions from the emitter tag. The algorithm relies heavily on the period being very accurate. Multiple sets of a few hundred consecutive samples were taken and the time interval between them checked. Unfortunately this was when it was discovered that the period was not as steady as first expected. A peculiar first discovery was that it turned out every 23rd transmission was delayed by about +60 ms. Not a major concern as it was deterministic bias that could be corrected for in the processing algorithm, but strange behavior none the less. A greater concern was that a deviation in period time of ± 2 ms was normal for periods of $\sim$8 seconds. Given that sound moves $\sim$1.5 meters each millisecond that added noise is not very welcome. It also seems like a lot considering that the transmitter

has a oscillator with 20 ppm accuracy, meaning it should deviate a maximum of $0.16\mu s$ over 8 seconds.

After checking with the manufacturer it became clear that the emitter used was

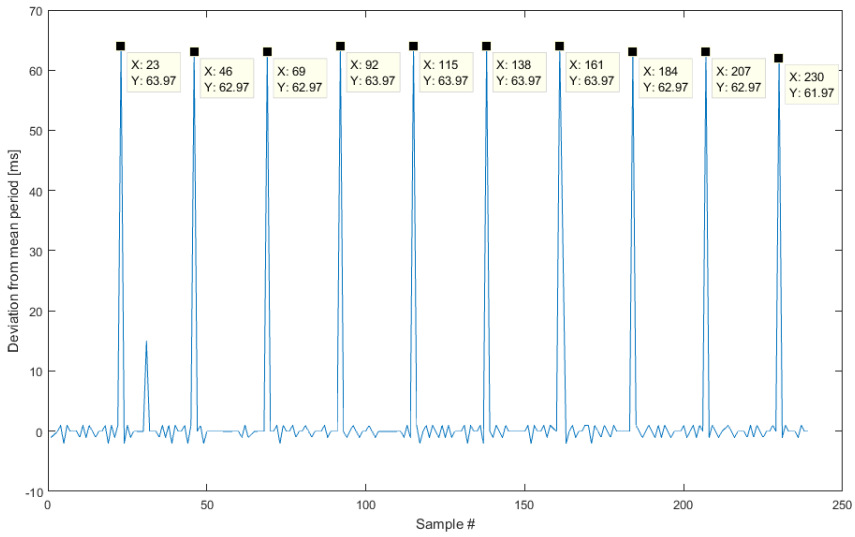


Figure 4.3: Deviations in periodicity for the first tag tested

running old firmware which utilized a more haphazardly time schedule. A new emitter with updated firmware with stricter timing constraints was then acquired and the tests re-run. Using this the odd +60 ms deviations were gone and the period time a neat 7.000 seconds, but unfortunately the smaller deviations of ± 2 ms were still present as seen in Figure 4.4. These measurements were taken using the TBR700 and it may be the case that the tag emitted its signal more precisely than indicated in the tests. Using frequency analysis it is up to the algorithm running on the hydrophone to determine the exact time of the sound wave. Variations in the propagation of energy and sampling on the hydrophone may be the cause the deviations seen in Figure 4.4. These tests were carried out in air with the emitter and hydrophone next to each other. When the distance to the emitter was increased the deviations also seemed to increase slightly.

One of the lessons learned from earlier simulations was that the system would benefit greatly from improved time resolution. New simulations were run with added noise reflecting the measured deviance in period. As expected the additional noise does skew the TDOA hyperbolas by a not insignificant amount, but it still seemed feasible to handle this additional noise.

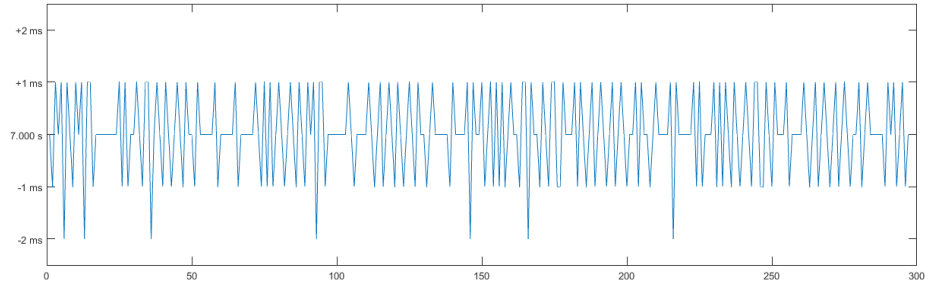


Figure 4.4: Tag with updated firmware with a more steady periodicity

4.2 Processing of measurement data

Data collected from the GPS-sync module over RS-232 could be extracted and processed with some minor alterations to the simulator code. Rather than simulating the input data to the filters it was now possible to feed the filter algorithms true measurement data after applying some sanity checks. Some additional functions were written for converting coordinates from WGS-84(World Geodetic System) to UTM(Universal Transverse Mercator) and vice versa within a few nanometers[7]. The GPS outputs data using WGS in which a position is represented by degrees of latitude and longitude. For the filter algorithms and map plotting it is more convenient to represent positions as meters north and east of a fixed point which is the case with UTM. Functions were also written to precisely calculate the speed of sound in sea water given salinity, temperature and pressure[12], but as the device used for measuring temperature and salinity also calculated sound speed it was not entirely necessary. To produce more visually pleasing plots vector data was downloaded from the Norwegian Mapping Authority(Kartverket) such that a crude map could be drawn showing coastal lines, docks and depth curves along with the path and position estimates.

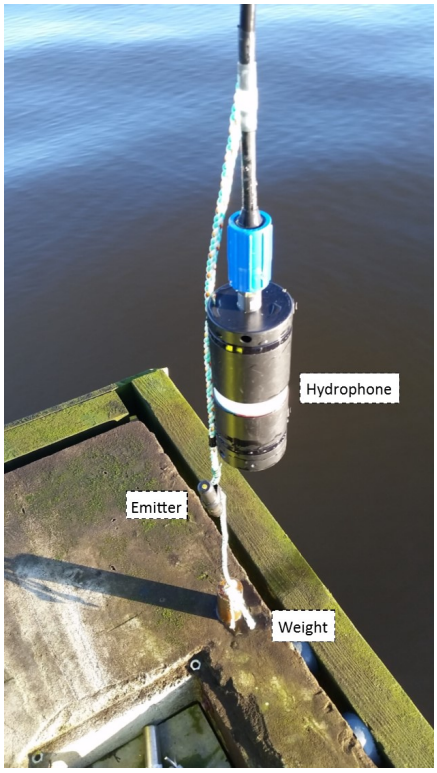
4.3 Testing in Sea Water

The point was reached at which simulations seemed promising and the logical next step was testing this using real data. A test was scheduled to take place in the fjord leading into Trondheim near Børsa, see Figure 4.5. As the purpose of this first test was simply to prove the acoustic positioning algorithm working the AUV was omitted and a motor boat used in its place. As far as weather goes conditions could hardly be better this day. Sunny, windstill and completely flat waters.

The experiment was set up in the following way. An emitter attached to a rope and suspended at a depth of 4 meters from the far end of a moored floating dock. See figure 4.6a. Also attached to the rope next to the emitter was a hydrophone connected to a GPS-synchronization module. This way the location of the emit-

Figure 4.5: Site of the sea water test. ©Kartverket

ter would remain stationary and its emitted signals would be recorded and time-stamped. Since the hydrophones are synchronized using GPS-time the clock in both hydrophones can be assumed to be synchronized to well within the precision used by time-stamps.



(a) Emitter setup as used in the trial

(b) Being set up on the floating dock

The boat used was a 6m aluminium motor boat with an outboard motor. Here a hydrophone was strapped to a rope, and suspended at a depth of 3 meters below the surface. Efforts were made to keep the rope hanging straight down in the water using some heavy weights. Despite that the rope was left at an angle due to drag force when the boat was moving, meaning the hydrophone was a little closer to the surface than 3 meters when moving. The hydrophone was connected to a GPS-synchronization module which was secured to the handrail at the bow of the boat such that there were no obstructions for the GPS antenna. This setup can be seen in Figure 4.7.

Multiple different data collecting trajectories for the observer were selected ahead of the experiment. These trajectories included different convex and concave curves with the emitter approximately placed in the foci, a straight line towards the emit-

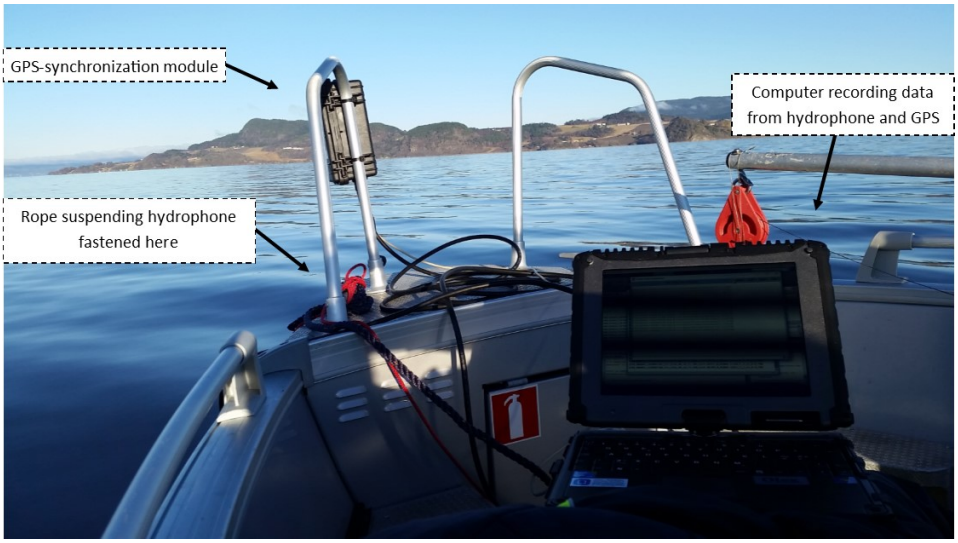


Figure 4.7: Setup in boat the used as observer

ter and a circle without any specific center. The complete recorded trajectory for the entire test can be seen in Figure 4.8. Code for data collection combined with real-time position estimation on-site had been prepared for the experiment. Unfortunately it was discovered that some needed function calls had been altered between the Matlab 2013 version installed on the computer used for the test and Matlab 2016 version used in development. Rather than attempting a perhaps time consuming fix on-site we quickly opted for a simpler pure data collecting approach instead. This was no major issue. It simply meant validation of the algorithm had to wait until we were back off-site. The code itself was written to handle most thinkable errors in faulty serial connection, data parsing, bad data and so on, but it was not tested on this new machine. This simply demonstrates the importance of having a fall back plan when using new equipment for the first time.

Data during the experiment was stored continuously as multiple different trajectories of interest were executed. The data was then later chopped into snippets and then feed through the filter. The variances in measurements learned were used here. First out was parts of an elliptical orbit around the emitter. This was assumed to be the most friendly path for position estimation as there is added information by circling the emitter and the hyperbolas even when erroneous will naturally intersect at the emitter location. The result is seen in figure [].

4.3.1 Preliminary Data Collection

A CTD(Conductivity-Temperature-Depth) device was lowered into the water at the location of the emitter. A CTD device takes frequent measurements of the

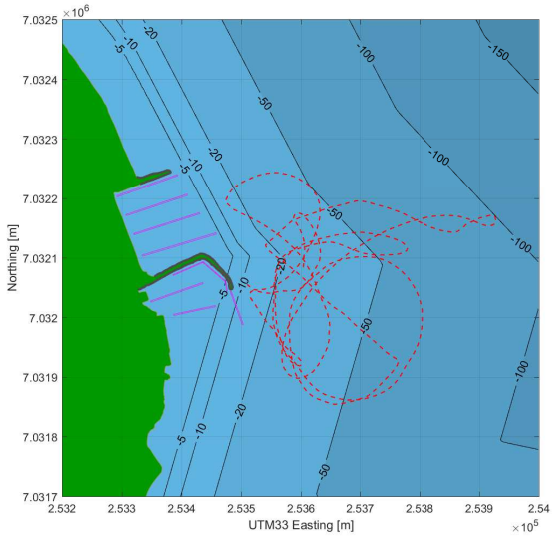


Figure 4.8: The complete trajectory of the observer while collecting data

temperature, differential pressure and conductivity as it is lowered down. From these measurements a detailed profile of the sea water is made. See 4.9. The conductivity is used to calculate the salinity, which as mentioned earlier, plays a role in increasing the sound velocity. The largest impact on the speed of sound is made by temperature however. Knowing the temperature, pressure and salinity the speed of sound can be estimated with high precision using a variation on the UNESCO formula[12].

The purpose of the trial was to see how well the emitter position could be determined. Knowing the true position of the emitter was therefore important. Using an assisted GPS unit a high precision measurement at the location of the emitter was taken. The measurement is shown in Figure 4.10.

4.4 Results

The data from the trial was logged and recorded to a file then later processed in Matlab after the experiment had concluded. As all the raw data measurements were kept this allowed for multiple aspects of the signal to be analyzed, some of which were not considered ahead of the experiment. For positioning the data was isolated into snippets containing a certain trajectory which were then fed through the filtering algorithms.

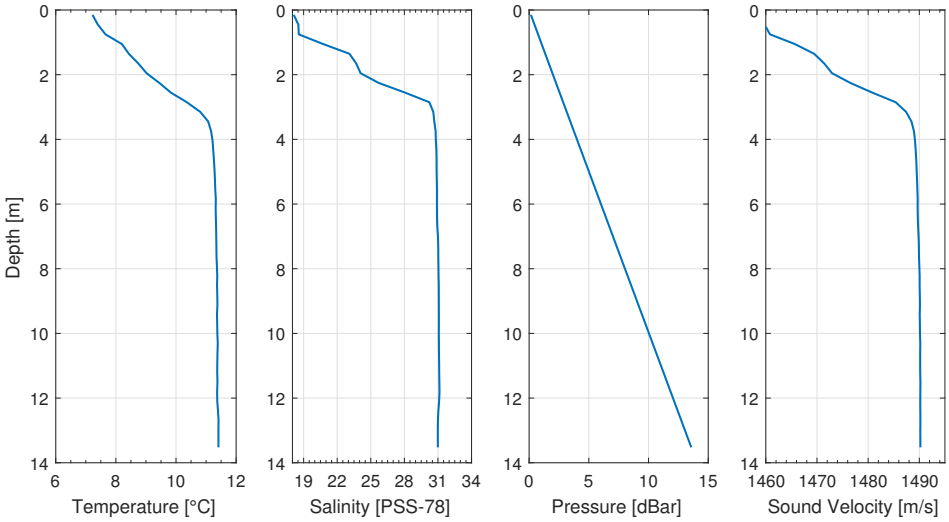


Figure 4.9: Characteristics of Sea Water as measured on the day of the trial

Target Position	
Date	2016-11-02
Time	12:13:55
ImageFilename	Target Position_G100670_20161102_121355.jpg
Distance	1.00m
TrueBearing	137.0°
TargetLatitude	63.332705276°
TargetLongitude	10.074226716°
TargetAltitude	3.90m

Figure 4.10: GPS Position of emitter

4.4.1 Measurement Analysis

Due to a hardware setup issue a minor correction needed to be done to the data. There was a known time delay between the timestamp set by the hydrophone and the recorded GPS position. This comes down to the low bitrate of acoustic transmissions. The message emitted takes multiple seconds from start to finish. The very start of the message is timestamped, however the message is not available until it has been fully received. The GPS unit used was originally intended for stationary receivers and simply assigns its current position when a message is fully received. A delay of 4.6 seconds was estimated using timing diagrams for the message protocol of the emitter tags. The solution used was to shift all GPS positions back 4.6 seconds using their velocity vectors. This approximately synchronizes the time and position.

One of the key areas of interest in analyzing the data is looking at the measurement variance. Figure 4.11 shows a comparison between the received time differences of arrival against what should be the expected TDOA based on actual distance from emitter and sound velocity. The actual distance here is calculated using the true position of the emitter and the GPS fixes for the timestamps. Noise is introduced by the GPS inaccuracies, inconsistent propagation speed in water and temporal resolution of the timestamps. The measured signal deviance from the expected true value is shown in Figure 4.11. The signal variance obtained from these comparisons were used to condition the filter for measurement noise.

Along with the time stamp for each packet a Signal-to-Noise ratio was also recorded. It was earlier postulated that SNR might be used to help improve the performance of the filters. However it was not clear how noisy the SNR would be. The main concern was that the signal strength could vary massively given small alterations in conditions such that it perhaps would be more sensitive to changes in receiver attitude than distance. It was therefore quite interesting to investigate whether SNR used for acoustic signals underwater could provide a reliable decay over distance. The distance at which each packet was observed was calculated using the GPS position associated with the packet and the measured true position of the emitter. Figure 4.12 shows the decay in SNR over distance.

Using linear regression the trend seen in the graph can be plotted as

$$r_{SNR} = -0.0845 \left\| X_{emitter} - X_{auv} \right\| + 50.8098$$

This linearised model for SNR can be used to roughly estimate the range to the emitter. Looking at the spread of the scatter plot it is obvious that this is a measurement with somewhat high variance, but will provide a ballpark estimate of the range to the emitter. This linear ranging model was used to provide the filters with a range input from SNR in addition to the range difference given by the time stamps. It was suspected that the orientation of the hydrophone would impact the received signal strength, but little data was available on the degree of impact

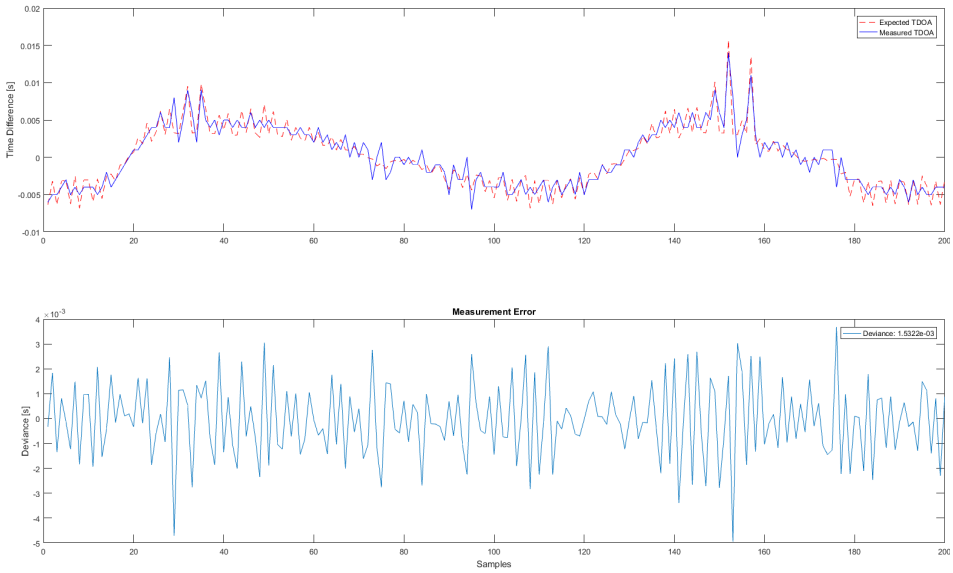


Figure 4.11: Comparison of measured TDOA and expected TDOA

available. Ideally the hydrophone would maintain the same orientation relative to the emitter for the entire experiment. One of the trajectories driven during the test was a straight line path as seen in Figure 4.13a. During this path the hydrophone should have held approximately the same orientation relative to the emitter. The SNR values were isolated and can be seen plotted in Figure 4.13b.

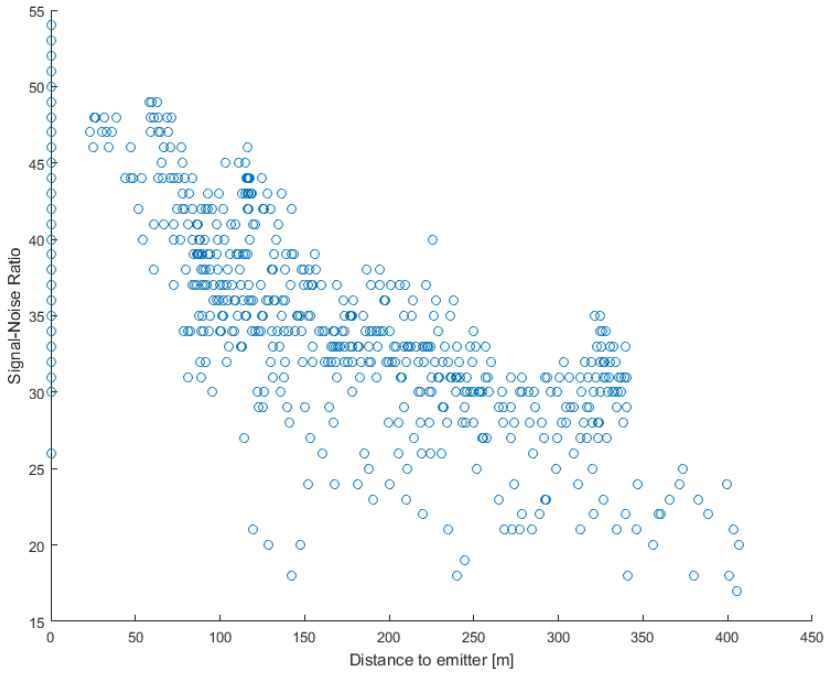
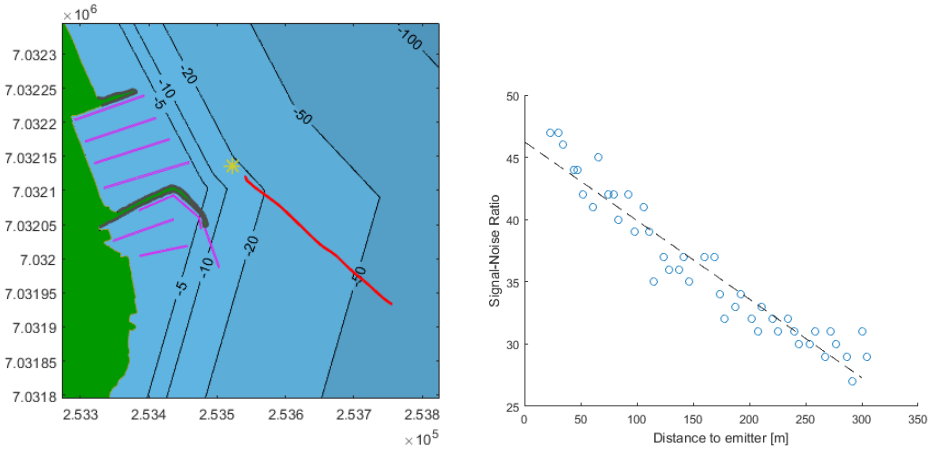


Figure 4.12: SNR of every received message and their true distance from emitter



(a) Emittor on hyperbolas

(b) Distance difference is equal along the hyperbole

Figure 4.13: Straight line path and its measured SNR

4.4.2 Positioning results

Now lets move onto the most interesting part and attempt some position estimation using the data recorded during the trials. Some trajectories snippets from the trial were extracted and feed into the filter giving different start conditions.

Orbital path

The first trajectory to be tested was an elliptical path with the emitter in the center. The start and end points were chosen such that the circle would be as closed as it could be without crashing into the pier or marina. This orbital path should be the most friendly path for position estimation as there is added information by circling the emitter and the hyperbolas even when substantially erroneous will naturally intersect at the emitter location. It is in a sense assisted and given more information by perfectly circling the target. It should be capable of locating a target without this additional information for this method to valuable in any way. It did however serve as a systems test to see the algorithm and data feeding working as expected.

Concave path

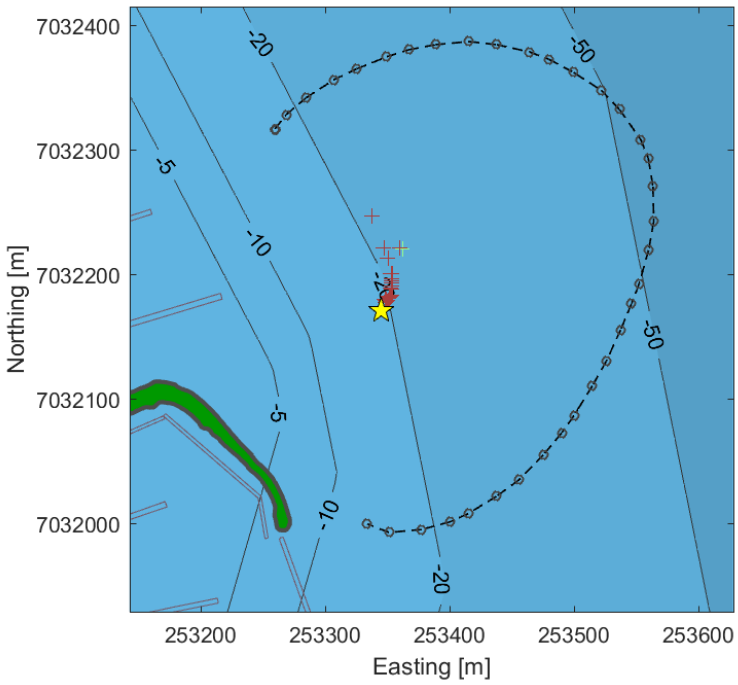
Here the boat traveled northward in a concave trajectory. In this case the particle filter outperformed the kalman filter. Both filers are struggling with estimations which may be due to the trajectory being close to a straight line for some time.

Elliptical path

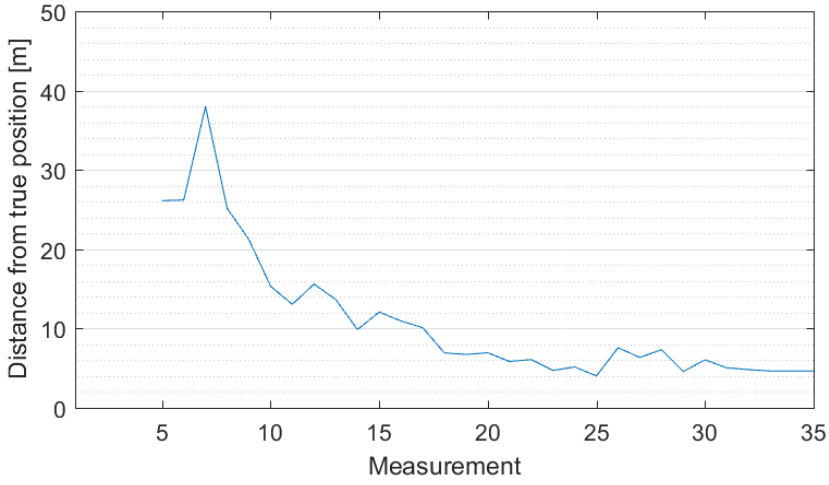
This is perhaps the most interesting of all the trajectories. If one is able to locate the emitter while circling an arbitrarily selected point in range of the emitter then there is some real value to the method. When doing this there is no apriori knowledge of the emitter in the trajectory. As seen in the figures both filters converge to within 30 meters of the target even when circling a point roughly 200m away from the emitter.

Convex path

For much of this trajectory the emitter is close to directly in front or behind the observer. The Kalman filter is not converging well in this situation and converges toward a point on the concave side of the path. The particle filter on the other hand has correctly determined the convex side but has uncertainties when it comes to the correct distance to the emitter. Even after 50 measurements it is still maintaining a few hotspots about 300m off.

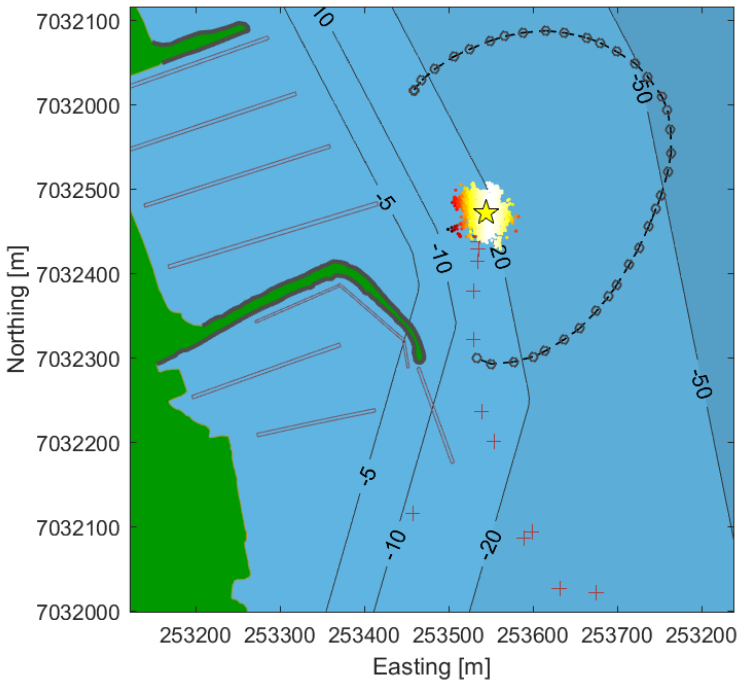


(a) Map plot of path and Kalman Estimates

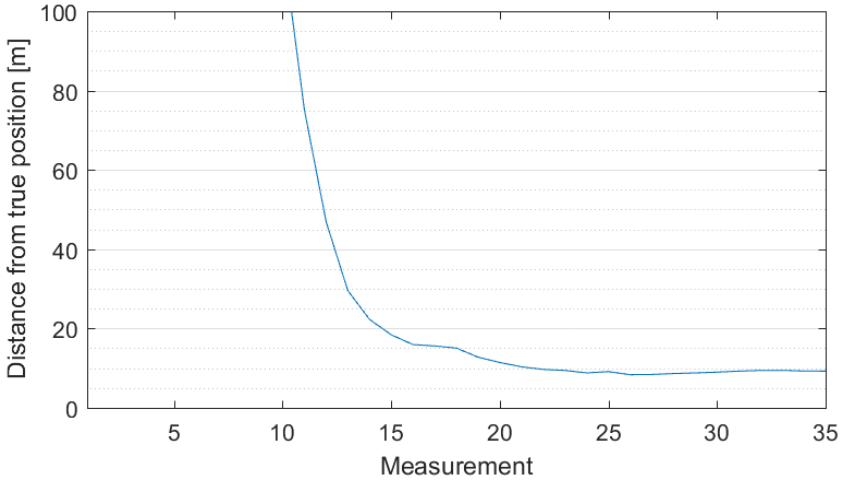


(b) Estimate error from true position

Figure 4.14: Kalman Filter Estimation
Orbital Path

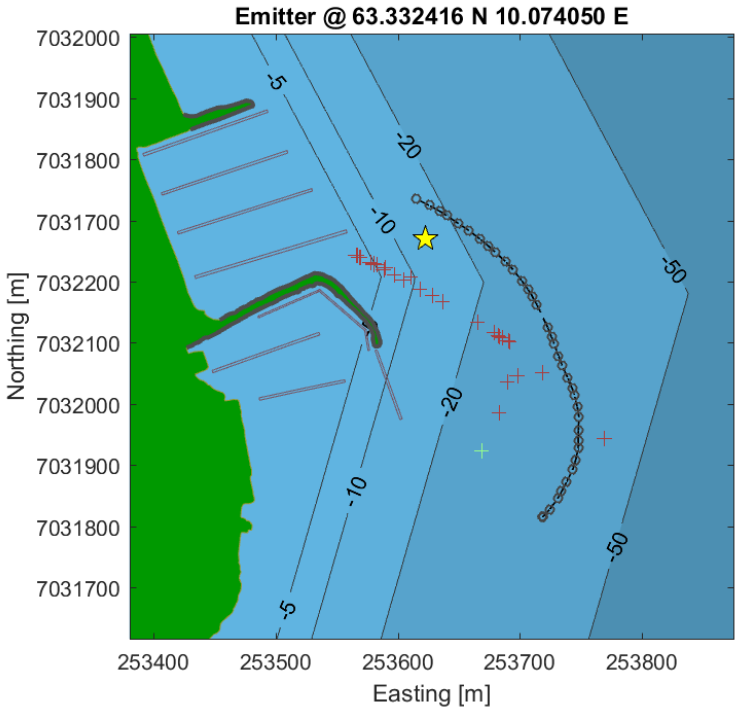


(a) Map plot of path and Particle Filter Estimates

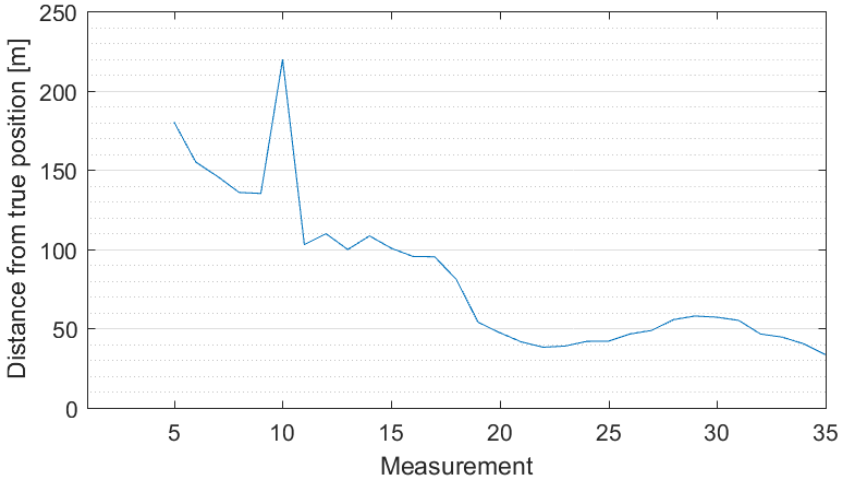


(b) Estimate error from true position

Figure 4.15: Particle Filter Estimation
Orbital Path

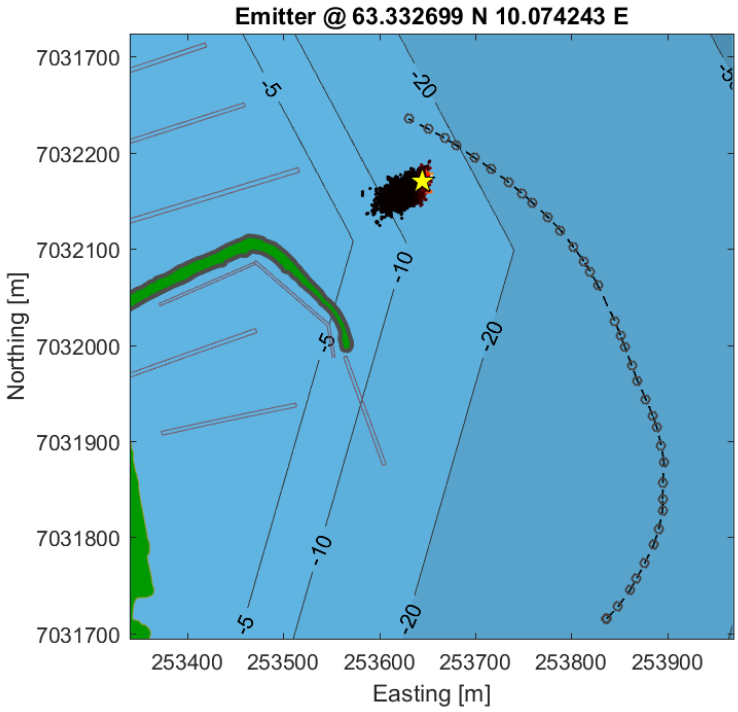


(a) Map plot of path and Kalman Estimates

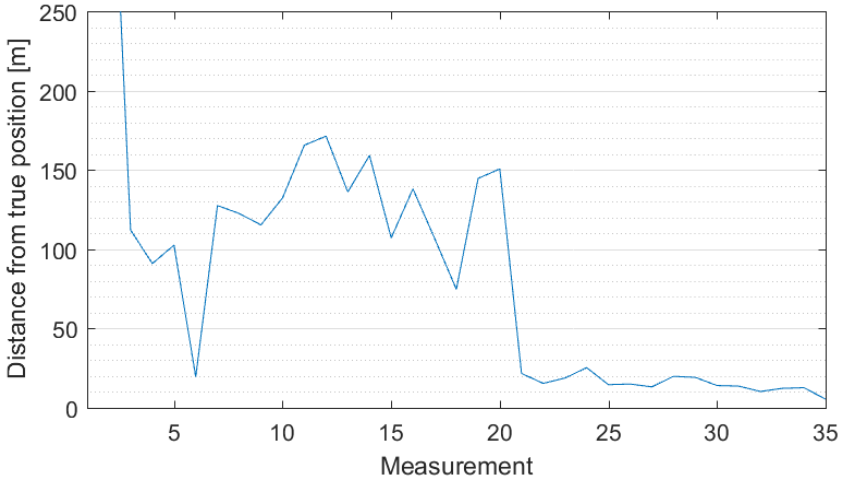


(b) Estimate error from true position

Figure 4.16: Kalman Filter Estimation
Concave Path



(a) Map plot of path and Particle Filter Estimates



(b) Estimate error from true position

Figure 4.17: Particle Filter Estimation
Concave Path

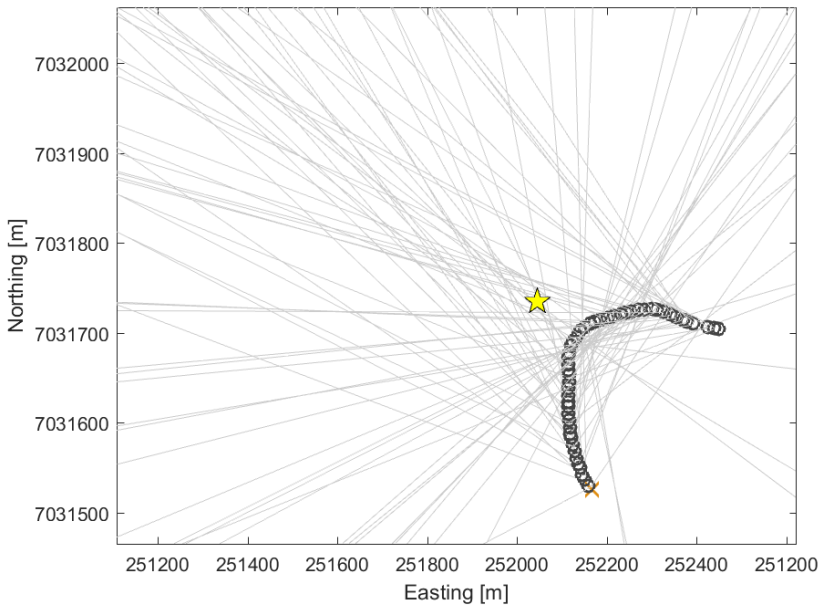
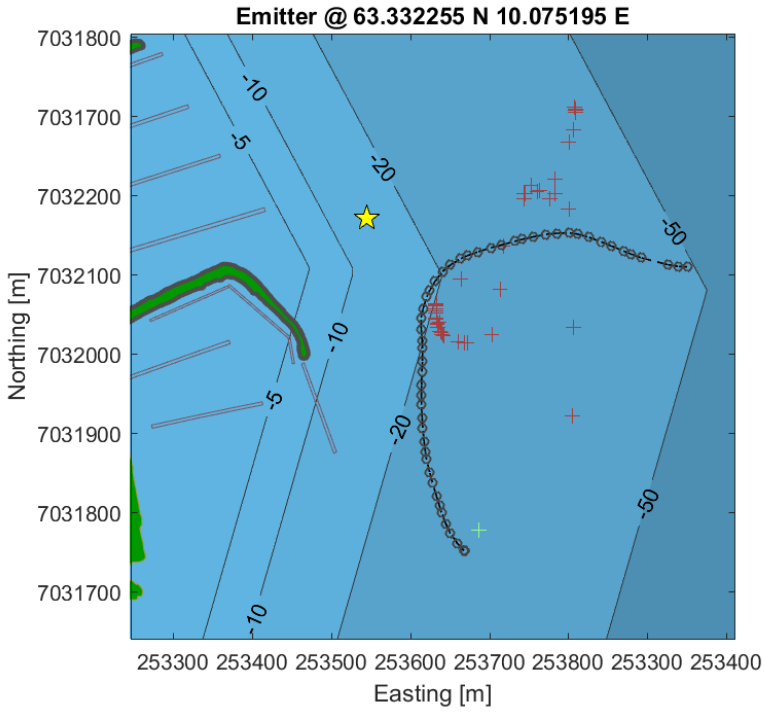
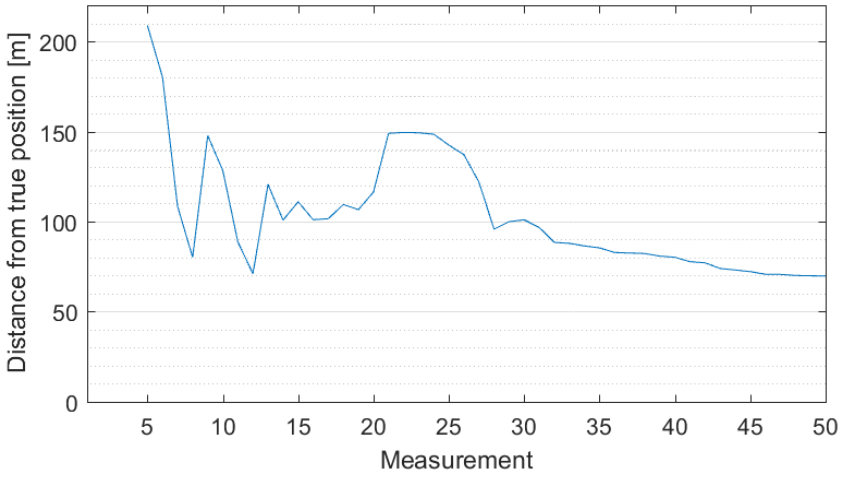


Figure 4.18: Range Difference Hyberbolas
Convex Path

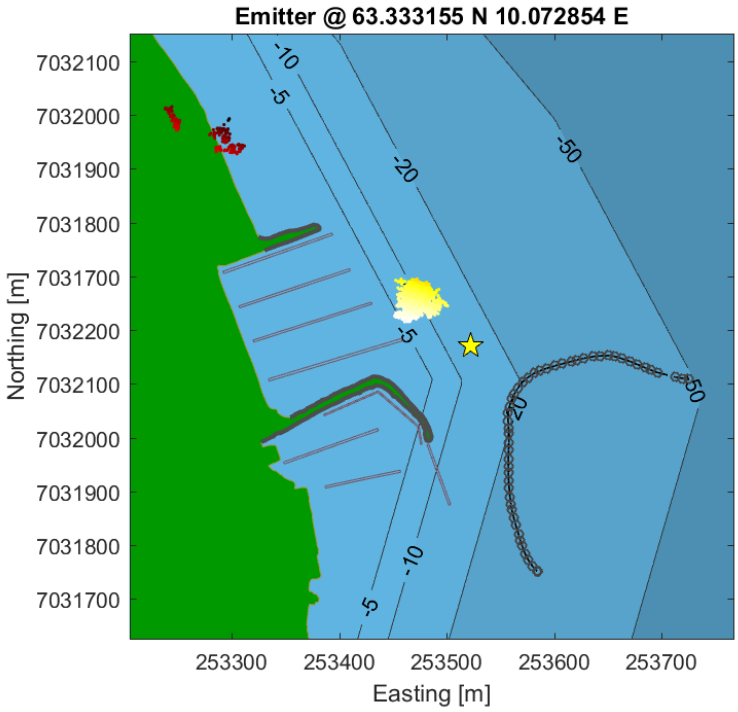


(a) Map plot of path and Kalman Estimates

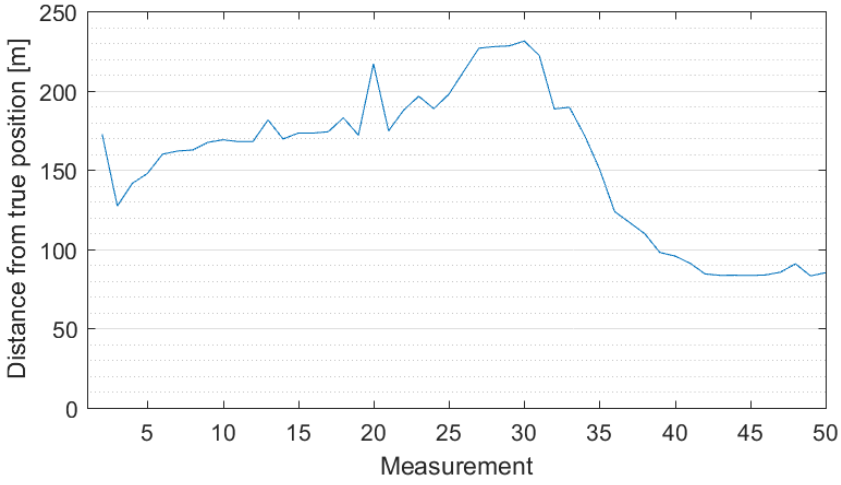


(b) Estimate error from true position

Figure 4.19: Kalman Filter Estimation
Convex Path

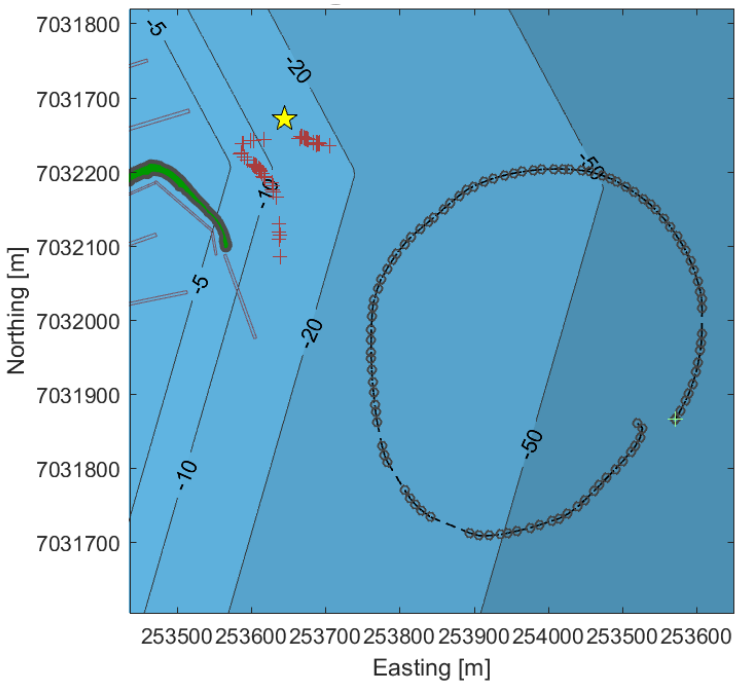


(a) Map plot of path and Particle Filter Estimates

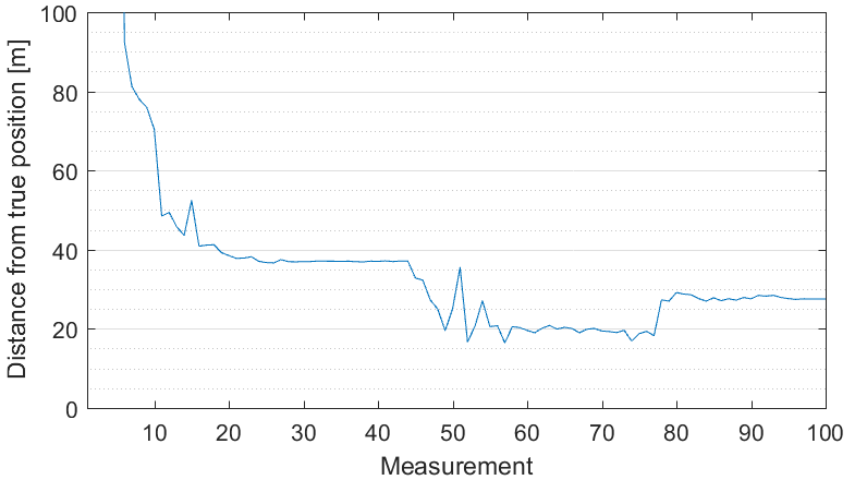


(b) Estimate error from true position

Figure 4.20: Particle Filter Estimation
Convex Path

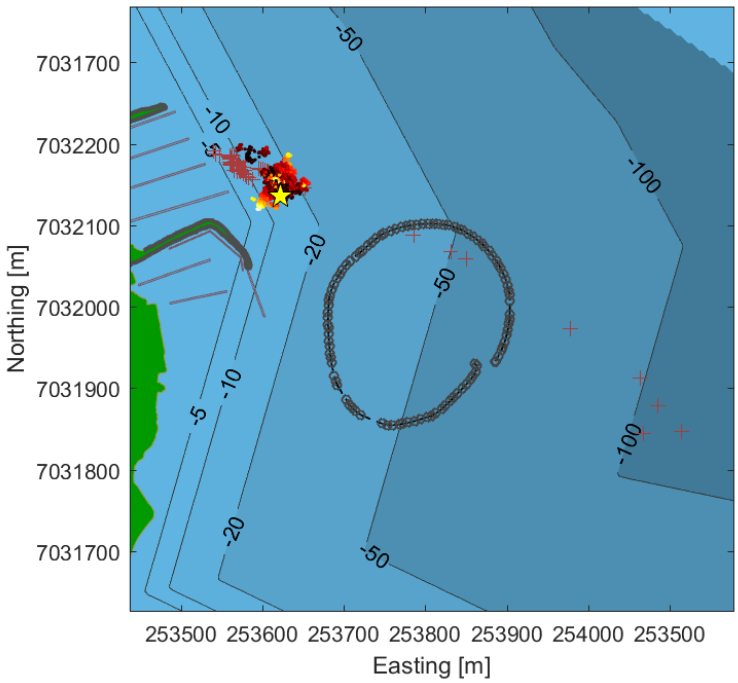


(a) Map plot of path and Kalman Estimates

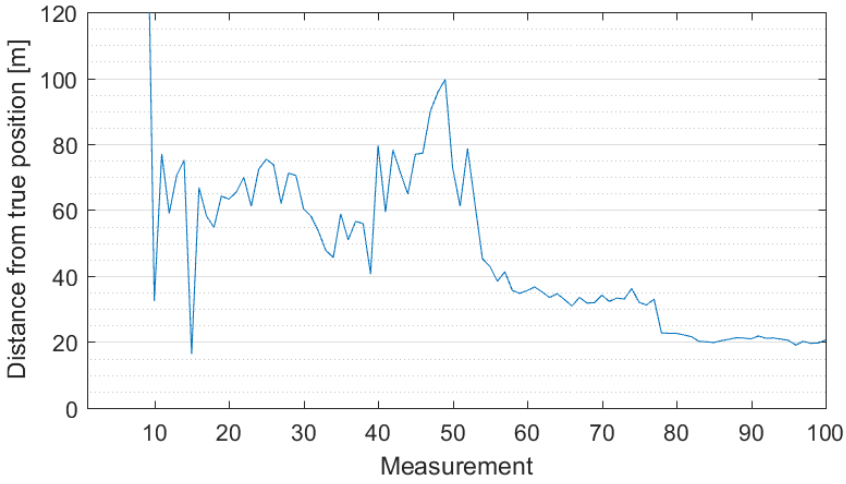


(b) Estimate error from true position

Figure 4.21: Kalman Filter Estimation
Elliptical Path



(a) Map plot of path and Particle Filter Estimates

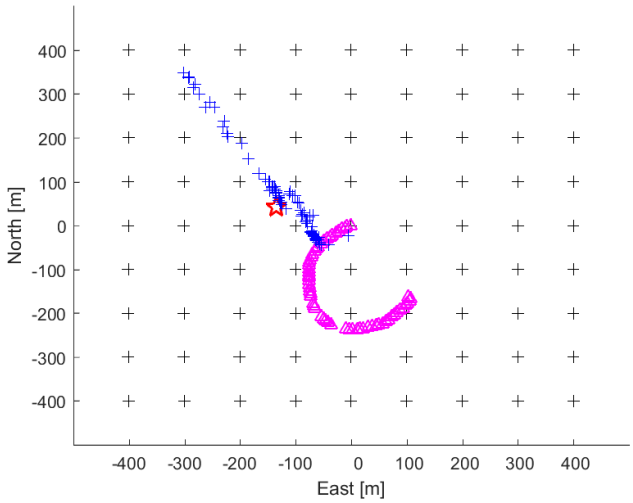


(b) Estimate error from true position

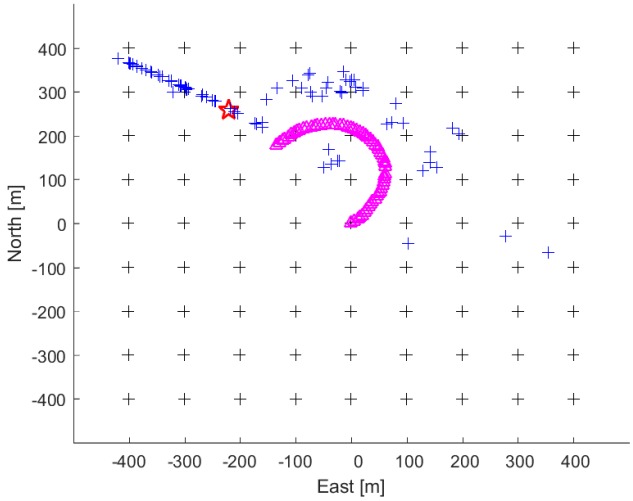
Figure 4.22: Particle Filter Estimation
Elliptical Path

4.4.3 Extended Kalman Filter Robustness

As a test of robustness of the Kalman filter when given poor initial estimates it was feed multiple times with the raw measurement data from the trial, but each time seeded with different starting positions for the filter. This should give an indication as to the filters capability to converge even in poor conditions. Figure 4.23 and 4.24 show the final estimate after 60 measurements have been received for each of the initial starting conditions. The start positions are spread out in a grid pattern being 100m apart as indicated by the black crosses. In Figure 4.24b the filter was only given poor initial estimates spread across a 50 meter grid pattern.

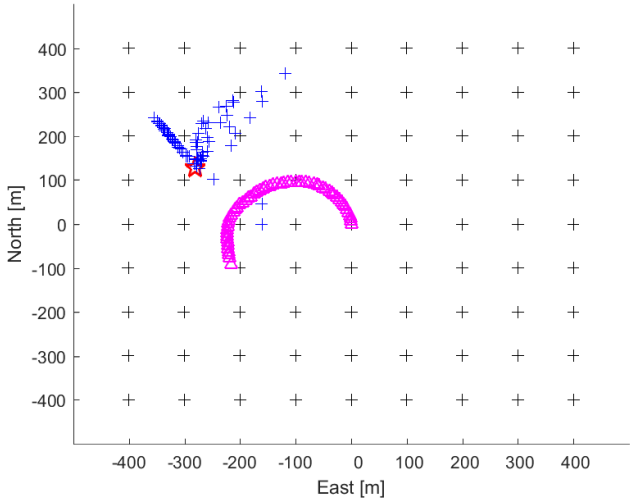


(a) Final estimate(blue) made after 60 measurements for each of the initial estimates(black) given the trajectory shown.

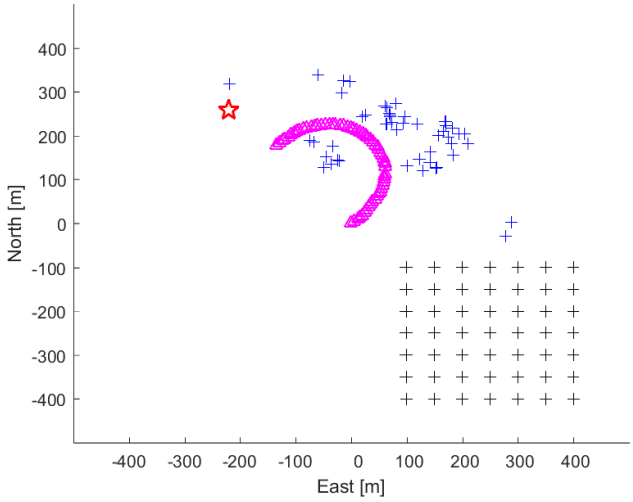


(b) Final estimate(blue) made after 60 measurements for each of the initial estimates(black) given the trajectory shown.

Figure 4.23: Kalman Filter Robustness: Results 1



(a) Final estimate(blue) made after 60 measurements for each of the initial estimates(black) given the trajectory shown.



(b) Final estimate(blue) made after 60 measurements for each of the very poor initial estimates(black) given the trajectory shown. Grid size of the initial estimates is here 50m.

Figure 4.24: Kalman Filter Robustness: Results 2

4.5 Discussion of the first trial

The results from the first trial were uplifting and demonstrate that positioning with the existing hardware is actually feasible. The results also verified that assumptions made in modeling and simulations were not unreasonable. The value of the data collected at this first trial was quite high moving forwards as they gave true data to analyze and help understand more about the actual signal variance and imperfections in transmissions than could be learned from simulations.

From the results one of the primary lessons learned is that a viable strategy for reliably acquiring a good position estimate seems to be circling an arbitrarily selected point until a rough estimate has been made. It is of course important to remain within the range of the emitter while preforming this circle maneuver. Results also indicate that the estimate gathered from the initial circling maneuver can be further improved if desired by circling the estimate location rather than the arbitrarily selected point. Doing so seems to push estimates to within 10 meters or less off target as can be seen in the results from the orbital trajectories.

The plots of the TDOA hyperbolas verify that spatial resolution is lost when the observer is traveling directly towards or away from the emitter. For the convex path this was the case for much of the trajectory. This minimizes the accuracy of the TDOA hyperbolas which in turn makes determining a good estimate harder for the filters. The variance could potentially be increased here to compensate for this, but this has not been looked into. Especially the Kalman filter often struggles in these situations and might begin converging toward a false position.

The Kalman filter needs to be seeded with an initial estimate. This first estimate needs to of sufficient quality for the Kalman filter to be able to correctly converge on the target location. Given a poor initial estimate the filter may diverge or converge on an incorrect location. Making this initial estimate is not always straightforward as there may not be a clear solution for the first few received measurements.

From the trials it has been observed that the particle filter is a more robust filter for the application. The nature of the particle filter allows it to maintain several hot spots for some time which are all evaluated at each time step. This upheld uncertainty is beneficial in this application as the TDOA hyperboloids often do not distinctly indicate a single solution for a set of multiple consecutive measurements. The particle filter copes with multiple solutions quite well by distributing the particles to multiple locations. The Kalman filter on the other hand must select a single point in space and iterate from there as is therefore sometimes converge to false TDOA hot spots.

Chapter 5

Positioning using AUV Platform

5.1 Preparation for second trial with AUV

5.1.1 AUV Overview

LAUV Harald is a Light AUV made by OceanScan MST. As the name implies it is a light weight AUV at about 50 kilograms designed to be man-portable. Instructions to the vehicle is given through a graphical user interface, Neptus. Here trajectories can be plotted on a map and uploaded to the vehicle. The progression can be also be monitored throughout the execution of tasks. It navigates using a compass and GPS when available, and comes with a built in IMU for inertial underwater navigation. In addition to this it is also fitted with a Doppler Velocity Log(DVL) which by scanning the sea bed can be used to as a reference frame to compensate for drift in the inertial navigation module. LAUV Harald is capable of speeds up to 4 knots and may operate on batteries for up to 8 hours at 3 knots.

Figure 5.1: LAUV Harald

Harald is fitted with two on-board computers. The main computer runs DUNE, which is a navigation and control environment developed by LSTS(Underwater Systems and Technology Laboratory in Porto, Portugal). It preforms a series of tasks among which is estimating the AUVs position, controlling actuators and carrying out the planned trajectory. The secondary computer is the user computer and may freely be used without concern of impeding with the vehicle's navigation system. This is the one which will be used to collect data and run the filtering algorithms in this experiment. The user computer utilized is a Nvidia TX1 Embed-

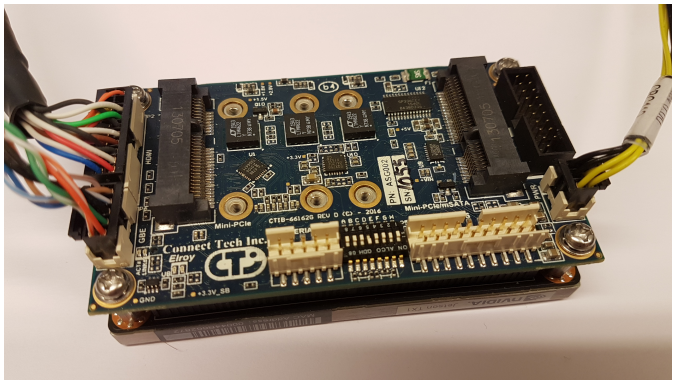


Figure 5.2: Nvidia Jetson TX1 on the Elroy Carrier

ded System Module, which is based on a ARM Cortex-A57 processor containing 4 cores. The main feature of the Nvidia TX1 is its 256 CUDA core GPU capable of intense parallell computation. It is unlikely that these will be utilized in this experiment as there is a limited degree of parallelism in the algorithms. The particle filter could be set up to benefit from this, but one must take into consideration that load/store operations from CPU to GPU is costly and may eliminate some of the benefit. Nvidia's System-on-module was combined with Connect-Tech's Elroy carrier board, which makes the peripherals of the computer more available. It has a header pin layout for a series of common peripherals such as high-speed ethernet, USB 3.0, SD-card, HDMI, and serial communication. It also properly power cycles the TX1 module when power is applied to the board and provides some input protection.

5.1.2 Connecting and interfacing the TBR700

The first task at hand was getting the hardware working with the Nvidia TX1. The TBR700 has a real-time interface based on half-duplex RS-485 serial protocol. This is a differential bus over a twisted pair of wires which makes it more robust in noisy environments than single-ended protocols. This is a reason it is commonly found as the underlying physical layer in industrial networks such as Modbus and Profibus, and in DMX for stage lighting. Only the physical layer is defined in the RS-485 specification and communication protocols on top of it is up to the

application. The hydrophone uses a proprietary protocol in which real time data is transmitted as comma separated values using ASCII characters.

The Connect-Tech Elroy carrier board came with a transceiver chip capable of converting UART from the Nvidia TX1 to RS-232 or RS-485, making it mostly ready to connect to the TBR700. A simple adapter from the Elroy's classic DB9 serial connector to the more sturdy industrial connector used on the hydrophone was soldered. After confirming the RS-485 bias resistors on the Elroy it was simply a matter of connecting everything together. A driver in C was then written to communicate and parse messages from the hydrophone. The driver also set error flags and logged events when communication was not successful.

TBR700 has a decent internal clock, but to avoid drift for real-time applications the clock may be kept synchronized by transmitting messages at predetermined times. The time synchronization is done by issuing a text sequence every time the UTC time in seconds is some multiple of 10, meaning this signal is transmitted at 10 seconds past the minute, 20 seconds past the minute and so on. Since the linux distribution running the code has no true real time guarantees. All tasks are best effort and decided by the scheduler. In this case the thread running the time syncronization was given very high priority in the scheduler, meaning it should always be first in line after triggering. There is no pre-emption, meaning other tasks could potentially delay the execution of this high priority task. This time synchronization is therefore only best effort and may be erroneous. The computer running these tasks is onthe other hand very fast and the undeterministic delay in this approach is likely to be so small that it is insignificant. Some measurements were done to confirm this and can be seen in Figure 5.3. Here a time synchronization message was set up with a software timer to transmit every 10 seconds and a logic analyser used to record outgoing messages on the bus. This logic analyser is fitted with a 50ppm clock, meaning a maximum potential measurement error of 0.005% or 500μs over 10 seconds[5]. As shown in the figure these messages were separated by approximately 9.99992 seconds with only a few microsecond variation in periodicity over multiple measurements. Based on this the software timer function as time synchronization was determined to be within adequate bounds for the precision needed.

Support for linear algebra is not part of the standard C or C++ libraries, and a linear algebra support would be highly beneficial for the implementations of both filtering algorithms. A survey for decent linear algebra libraries was conducted. At first the de-facto standard C-library called Basic Linear Algebra Subprograms(BLAS) seemed to be the obvious choice. These are highly optimized subroutines written in assembly. The downside to BLAS is that they do not utilize structs or classes, but instead require a long list of arguments and memory management. The following is a simple matrix by matrix multiplication, $A\,B = C$.

```
cblas_dgemm(CblasRowMajor, CblasNoTrans, CblasNoTrans,
            m, n, k, alpha, A, a_n, B, b_n, beta, C, c_n);
```

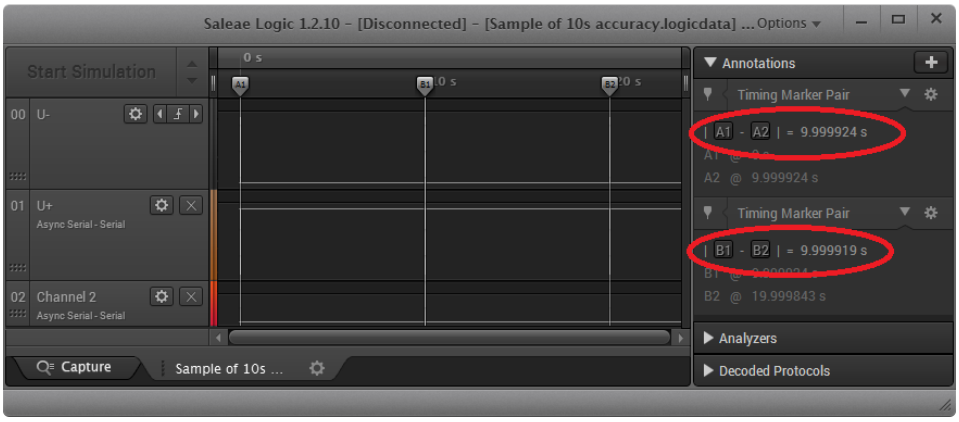


Figure 5.3: Logic Analyser

This level of control is excellent for application where high performance is crucial, however it is also prone to making subtle mistakes when chaining operations and may require large efforts in debugging a code with low maintainability. Another library candidate surveyed was Armadillo[see 8] is a linear algebra library for C++ which to a degree follows the syntax in Matlab. It too is based on underlying BLAS subroutines, however it defines matrices as objects with overloaded operators in conjunction with template functions making the syntax much more user friendly. While the switch of language was not made lightly the advantages of easy to use syntax and preserving code maintainability in this case outweighed the cost of converting existing C code into C++.

5.1.3 Communicating with the AUV main computer

The brain of the AUV is on the main computer running DUNE. This is an task based environment developed for control of any type of craft, and has been ported to UAVs, AUVs, ROVs and surface vehicles. The environment is task based where each task is responsible for some subsystem.

To communicate DUNE uses Inter-Module Communication Protocol(IMC). IMC is a set of predefined messages which may be transmitted using some transport layer. In the case of the AUV this transport layer is UDP/IP. The DUNE task responsible for estimating the current position of the AUV also dispatches an IMC message containing current state data. This Estimated State message contains a WRT84 latitude and longitude fix along with the current position offset in meters. By default the estimated state is updated 50 times per second. By tweaking the configuration file of DUNE on the main computer this message can be configured to be transmitted to the user computer using UDP. It is possible to send custom data using IMC by defining a custom message, but it is then necessary to recompile the IMC message parser for all systems receiving this message. If data from the

positioning algorithm is to be sent back to DUNE or sent to the graphical user interface then this must be done.

A custom IMC parser was written from scratch in C++. The program binds and listens to UDP messages on a specified port. All packets received on this port are expected to be IMC messages and will be rejected otherwise. Received packets are subjected to a number of checks for validity including checksum verification. The header of the message is checked for bit ordering(Little Endian or Big Endian) before it is cast to a specific subtype of message and the contents parsed. An IMC message handler was written to receive, parse and buffer data and make it available to the estimation algorithm. Since there was no need for full fledged IMC support on the user computer only a few IMC messages were implemented, most important of which is Estimated State. Adding support for additional messages is quite trivial.

Wireshark, a program which records network traffic, became a valuable debugging tool while coding the UDP/IMC parser. It was used to confirm UDP packets contained data which was structured correctly according to the IMC definitions. It was also useful for comparing the data parsed to the pure binary data stream and for removing pointer arithmetic errors from the code.

5.1.4 Integration with LAUV Harald

The AUV returned to the Applied Underwater Robotics Laboratory(AURLab) in Trondheim leaving us with three days before the absolute final deadline for the experiment. This meant we were racing the clock to get the system installed and running correctly in only two days so that a test could be conducted on the third day.

The first task at hand was physically installing the TBR700 hydrophone in the vehicle. A machined slot in the wet section at the front of the AUV was part of the product specification when it was ordered, but a hydrophone had not been attempted mounted into it before. It was therefore a relief to see the hydrophone fit without additional modifications. The front section was picked apart to gain access and install the hydrophone. A real-time cable for RS-485 was connected to the back of the hydrophone with a wet connector. From there the cable ran through a separation wall into the dry section of the vehicle. The finished install of the hydrophone is shown in Figure 5.5.

Next up was opening the sealed dry section so that the secondary computer could be wired to the TBR700. Before connecting the serial bus to the secondary computer it was confirmed with a logic analyzer that the signal from the TBR700 could be received on the other end. This check also let us identify the polarity of the bus line. The bus lines were then soldered onto a connector to the carrier board of the secondary computer.

The required software libraries for linear algebra was installed on the Nvidia

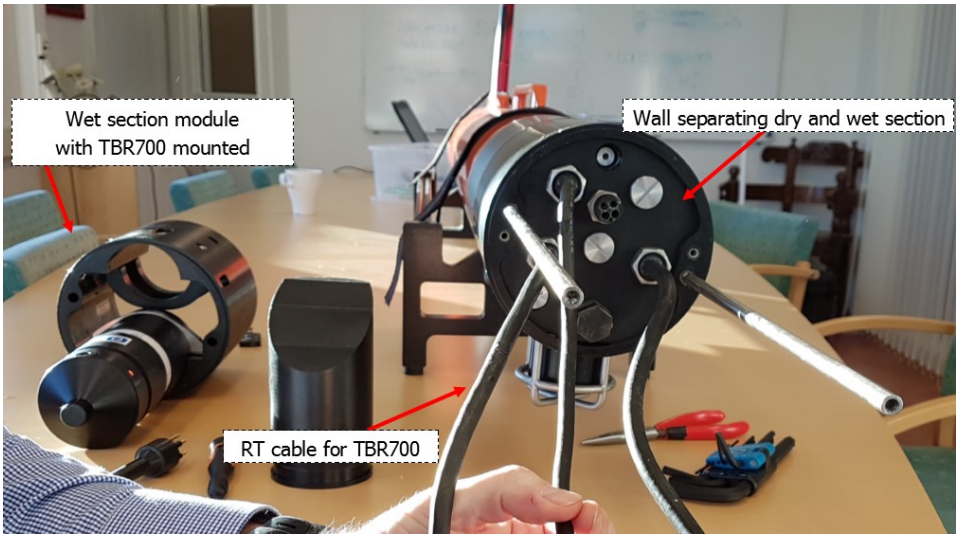


Figure 5.4: Front section of the AUV was taken apart to install and connect the hydrophone

Figure 5.5: Hydrophone successfully mounted and connected in the wet section

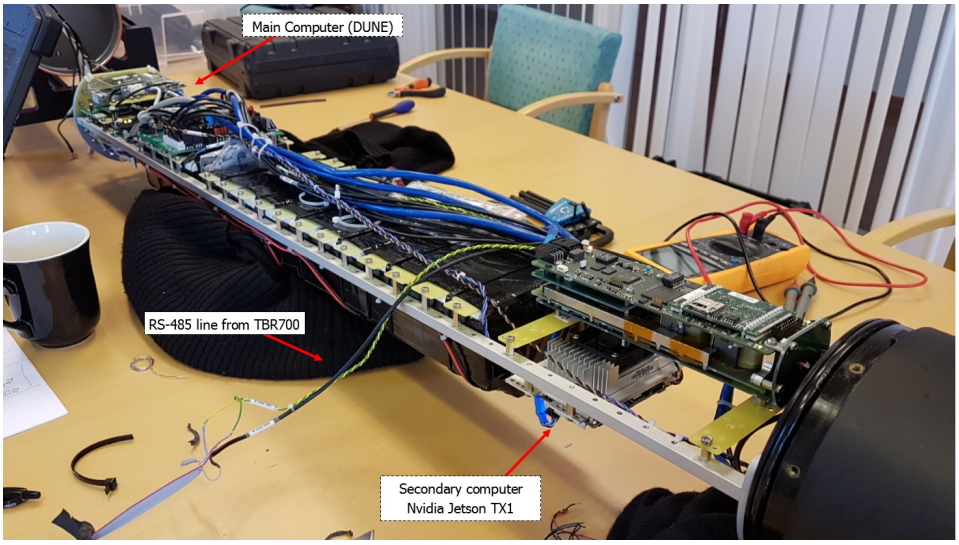


Figure 5.6: The dry section of the LAUV pulled out from the sealed compartment

Jetson TX1 secondary computer before the program for data merging and positioning was loaded onto it. Communication with the TBR700 was then set up, and an issue with significant noise on the RS-485 was discovered. As previously mentioned the TBR700 transmits an ascii-string containing comma separated data, and here many of the strings characters were replaced by other characters when received. It was first thought that the issue was caused by incorrect baud rate on the receiver or some other serial port configuration error. At this point time was becoming a critical factor and a decision was made to accept the noise and correct for it in post-processing. On the main computer DUNE was configured to transmit IMC messages for Estimated State of the vehicle and measured speed of sound, and some last tests were done on the user computer to verify that it was receiving all the messages and logging data. LAUV Harald was then put back together, sealed and vacuum tested so that it would be ready for experiments the next day.

The noise issue on RS-485 was still root for concern and after much digging late at night with the spare Nvidia TX1 the conclusion was drawn that the RS-485 errors were due to incompatible voltage levels. The Exar SP336E RS-485 transceiver on the secondary computer has a differential threshold between lines A (non-inverted) and B (inverted) in the range of 50 to 200mV. The logic analyzer was used to sample analog values of the bus lines. When the TBR was transmitting a binary zero it was able to pull line A down from 3 volts to around 0.8 volts, while only pulling B up to around 0.9 volts. This voltage difference of only $\sim$ 100mV caused undefined behavior of the transceiver when attempting to receiving the zeros and the sampled output value from it would fluctuate. Once the cause was identified it was quickly fixed using an additional bias resistor on the B line such that the

voltage difference was greater than 200mV.

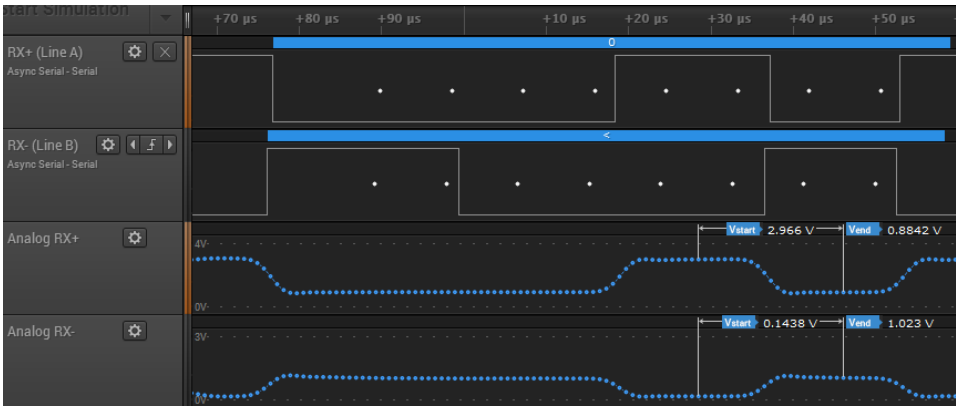


Figure 5.7: Capture of a '0' character (hex 30) sent from the hydrophone, which is interpreted as '<' (hex 60) due to the insufficient voltage difference. Note the Vend for both lines.

The next morning the problem was rapidly corrected inside the AUV as well with some last minute soldering, though it meant the dry section of the AUV had to be opened once again and a new vacuum test needed to be performed to ensure the vehicle was properly sealed. While this cost us some in-field time for experiments it allowed data to be collected in the way that as first intended.

5.1.5 On-site setup

The experiment was set up much in the same way as the previous trial with the boat. The tag with 7 second transmission interval was once again used and suspended at a depth of 3 meters at the exact same location as before. As in the previous trial the conditions of the water were measured using a CTD scanner. The temperature, salinity and sound speed can be seen in Figure 5.8.

Command of the vehicle was done through Neptus, the user interface and command program for DUNE. In Neptus a mission for the AUV is planned by setting waypoints for the AUV on a map and then uploaded to the vehicle. The floating pier was not marked on map available in the user interface. For that reason we needed to be careful with setting waypoints near the pier. LAUV Harald was launched from the floating dock and carried out a set of maneuvers to calibrate its compass. It was then ordered to circle a point further out in the fjord with a radius of 50 meters. It was also commanded to first preform a full circle maneuver before setting a course towards the emitter and arch around the emitter. The operating area of the AUV was visually monitored at all times and a boat was kept on standby in case a situation would arise with loss of control over the AUV or other traffic approaching the operating area.

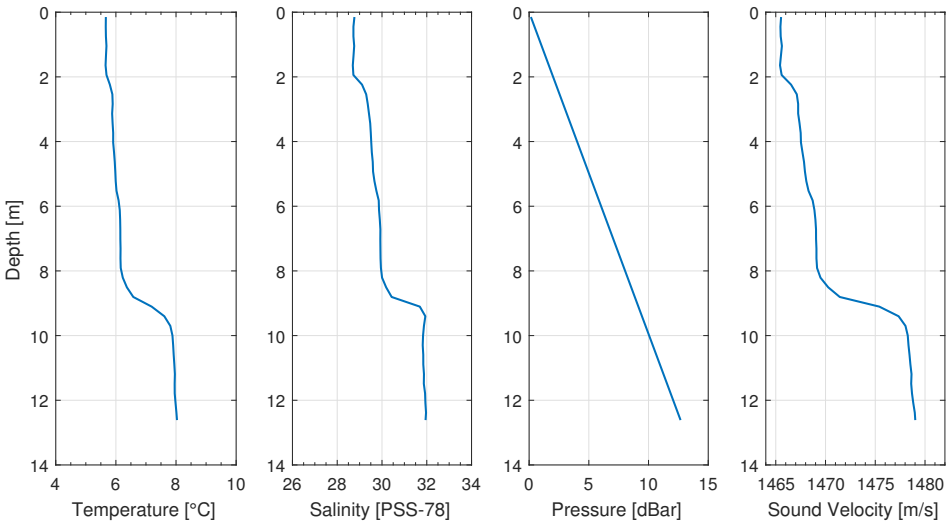


Figure 5.8: Characteristics of the sea water at the time of the trial

Figure 5.9: LAUV Harald launched and headed towards its first waypoint

The state of the estimator was monitored through a SSH connection to the secondary computer of the AUV, and in order to maintain communication with the AUV we refrained from diving in this experiment and stayed close to surface level. Most of the vehicle was below the surface with the hydrophone on the underside of the body fully submerged, and only the antenna on the top of the vehicle above water. A number of unintentional dives did still occur due to waves altering the pitch of the vehicle causing it to dive and then quickly resurface.

Figure 5.10: Telemetry with the AUV during mission execution

5.2 Positioning Results

5.2.1 Circling

From the previous trial with the boat it was observed that circular paths were effective for the positioning algorithm. A circular trajectory was therefore chosen for the experiment with the AUV as well. Here the vehicle is circling a arbitrary point roughly 200 meters away from the emitter with a circle radius of 50 meters.

Extended Kalman Filter

As before the first 6 measurements are used to calculate an initial estimate for the Kalman filter, and the estimation begins at the 7th measurement. As can be seen in Figure 5.12 the error drops rapidly below 20 meters after the first 17 measurements and eventually levels out at around 8 meters off target.

Particle Filter

Due to an incorrect model for ranging based on SNR values the Particle Filter overestimated the distance to the emitter. This is seen in Figure 5.14a, and the error plot shows the particle filter missing by >100 meters. The model used was $r = -0.0845z_{snr} + 50.8098$, while the least squares fit for the SNR measurements taken, see Figure 5.11, suggest it should have been $r = -0.0971z_{snr} + 41.6358$. For this reason the rest of the on-board particle estimates were considered uninteresting and were discarded. A particle filter estimate without the use of SNR was later made using the raw data from the experiment, and the result is seen in Figure 5.15. It was also done with the corrected SNR-range model which can be seen in Figure 5.16.

5.2.2 Circling and converging on the estimate

In this part the AUV followed much of the same circular trajectory used in the previous results before course was diverted and the AUV started to home in on the estimate to provide more accurate measurement data for the filtering algorithms.

Extended Kalman Filter

The figure 5.17 shows the real time on-board estimates made by on the AUV.

Particle Filter

The Figure 5.18 shows the results using the raw data collected during the experiment.

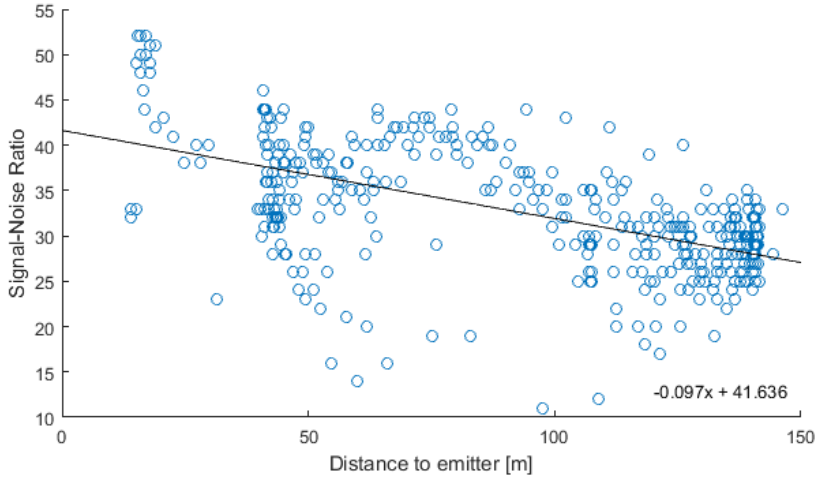
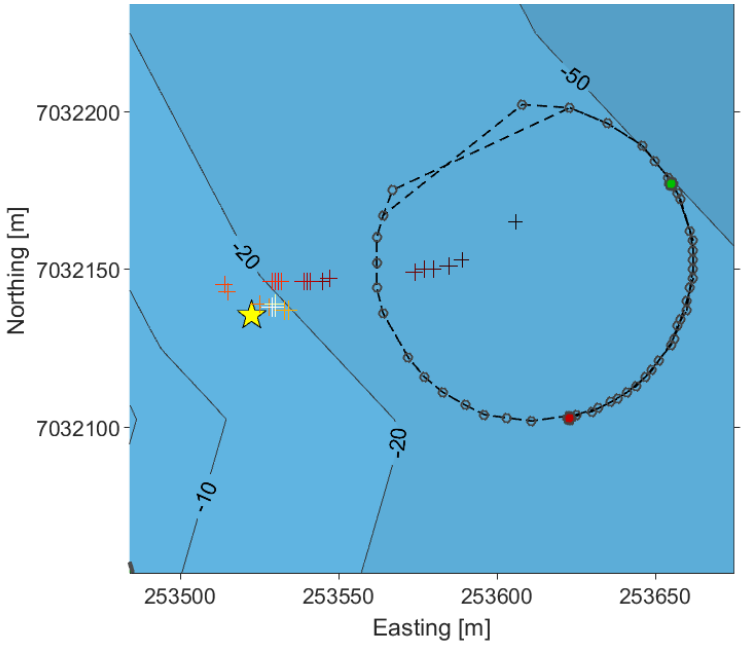
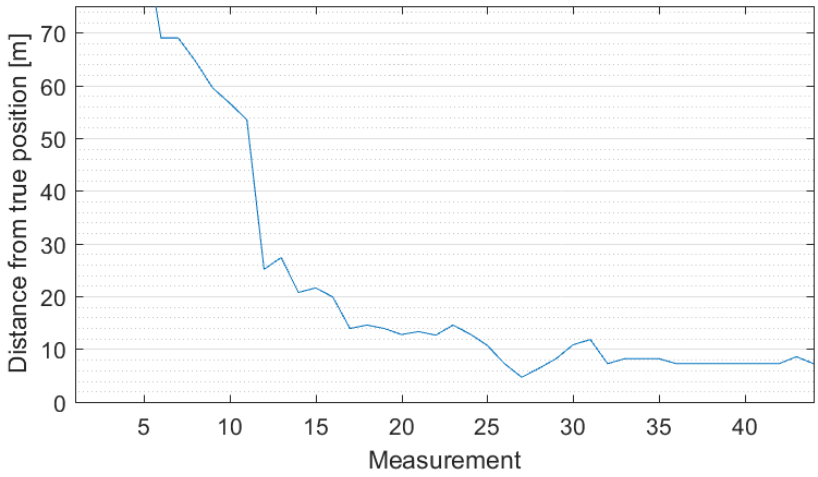


Figure 5.11: Measured SNR values for the entire data set collected using the AUV.

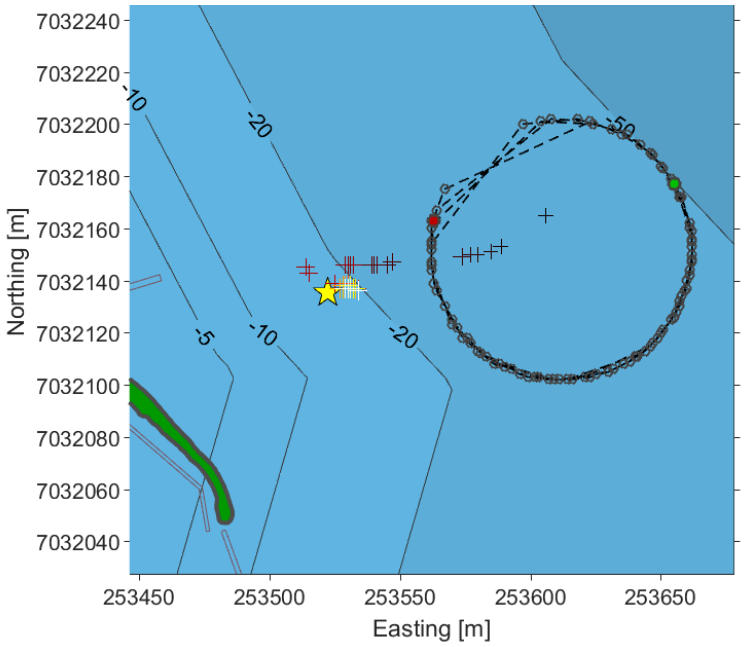


(a) Map plot of AUV trajectory and Kalman Filter Estimates. Estimates are shaded from black through white, where white is the most recent estimate. The green circle marks the start of the trajectory and red marks the end.

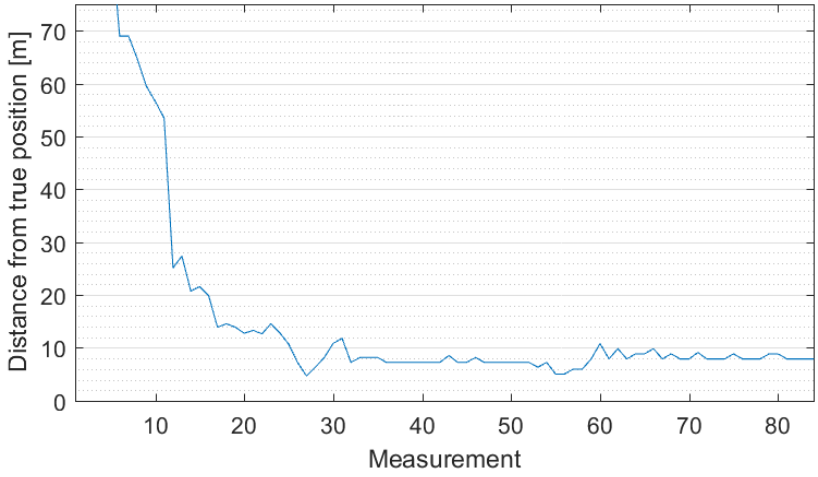


(b) Estimation error from true position

Figure 5.12: On-board Kalman Filter Estimation
Circular Trajectory with 1.5 revolutions

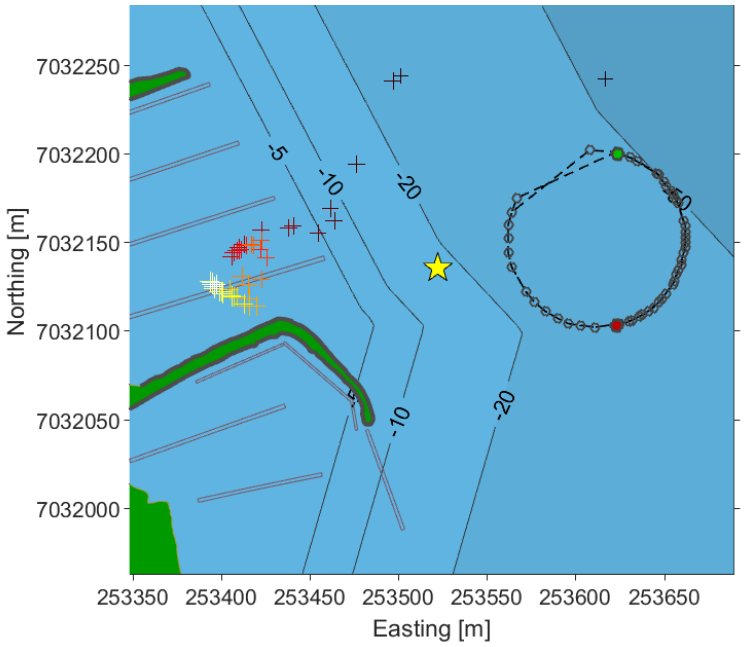


(a) Map plot of Kalman estimates. AUV trajectory with four rotations of a circle during which near 90 measurements were collected.

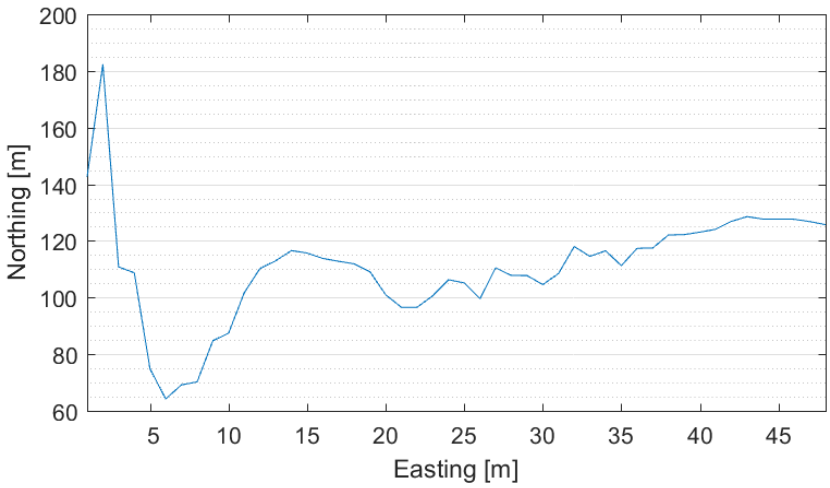


(b) Estimation error from true position

Figure 5.13: On-board Kalman Filter Estimation
Circular Trajectory with 3.5 revolutions

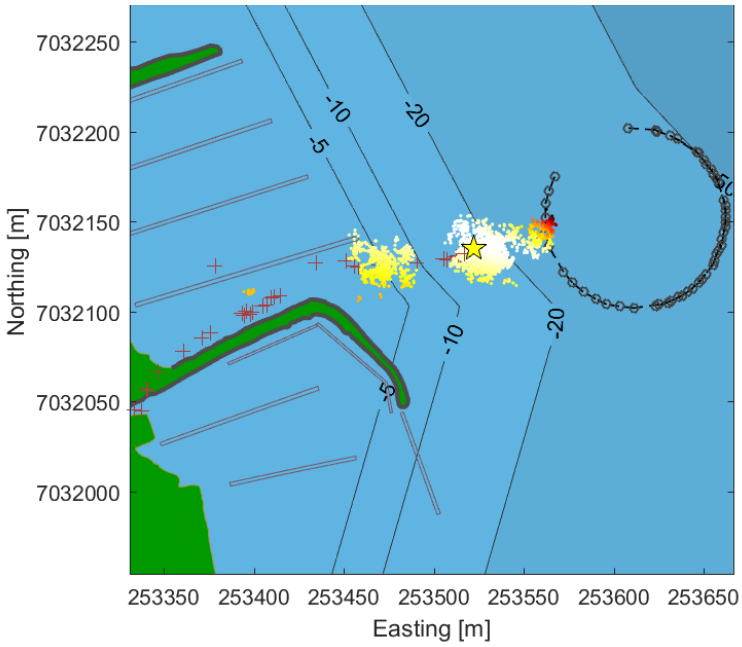


(a) Map plot Particle Filter Estimates and the AUV Trajectory. Estimate markers are colored from oldest(black) to most recent(white).

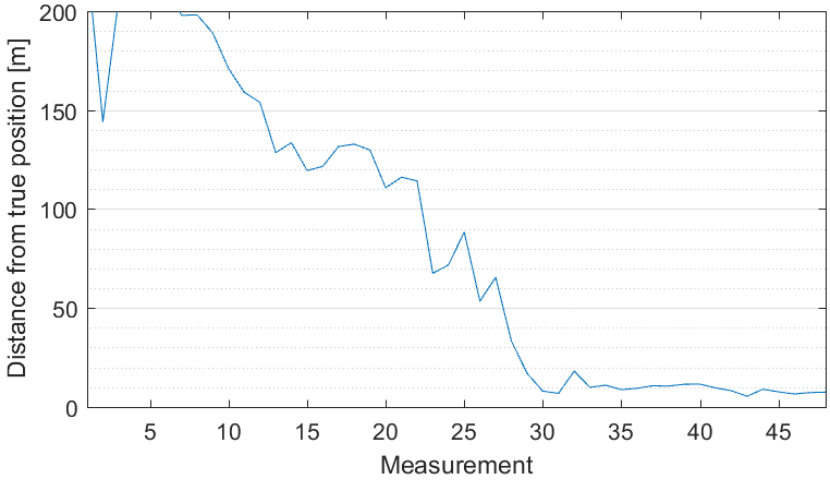


(b) Estimation error from true position

Figure 5.14: On-board Particle Filter Estimation
Circular Trajectory with 1.5 revolutions

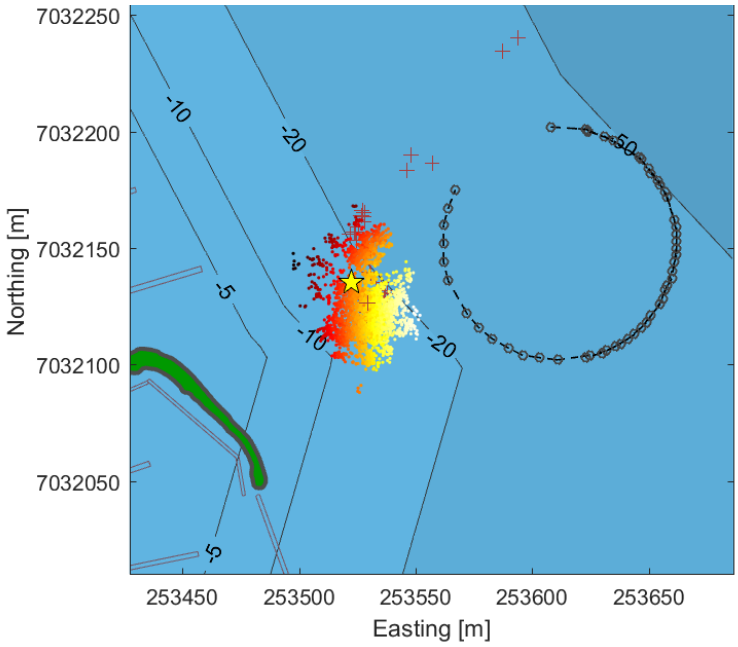


(a) Map plot of Particle Filter Estimates using only time difference of arrival.

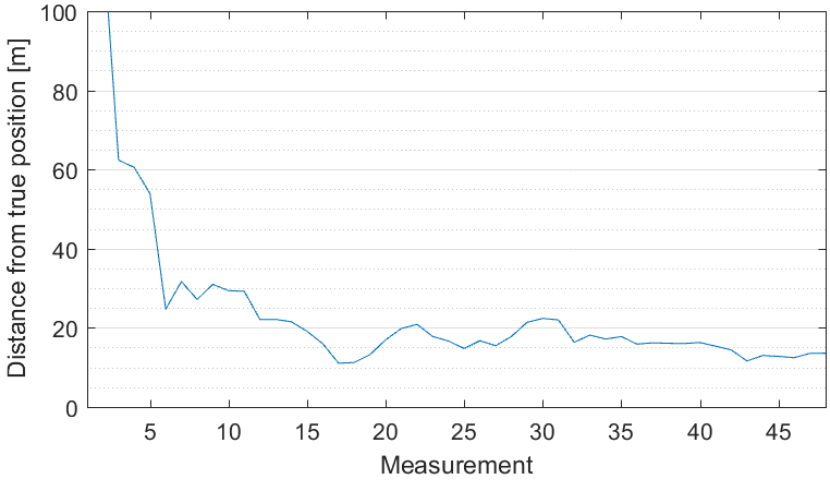


(b) Estimation error from true position

Figure 5.15: Off-board Particle Filter Estimation
Circular Trajectory with 1.5 revolutions

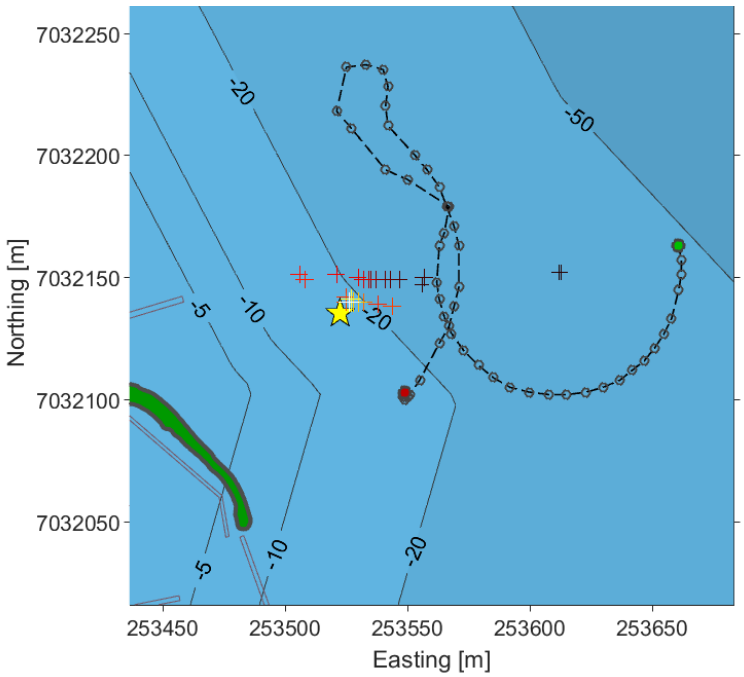


(a) Map plot of Particle Estimates with a corrected SNR-range model.

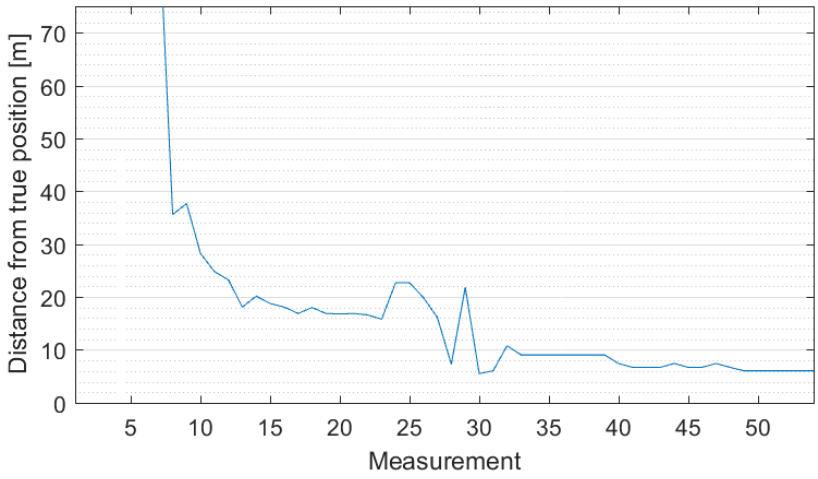


(b) Estimation error from true position

Figure 5.16: Off-board Particle Filter Estimation
Circular Trajectory with 1.5 revolutions

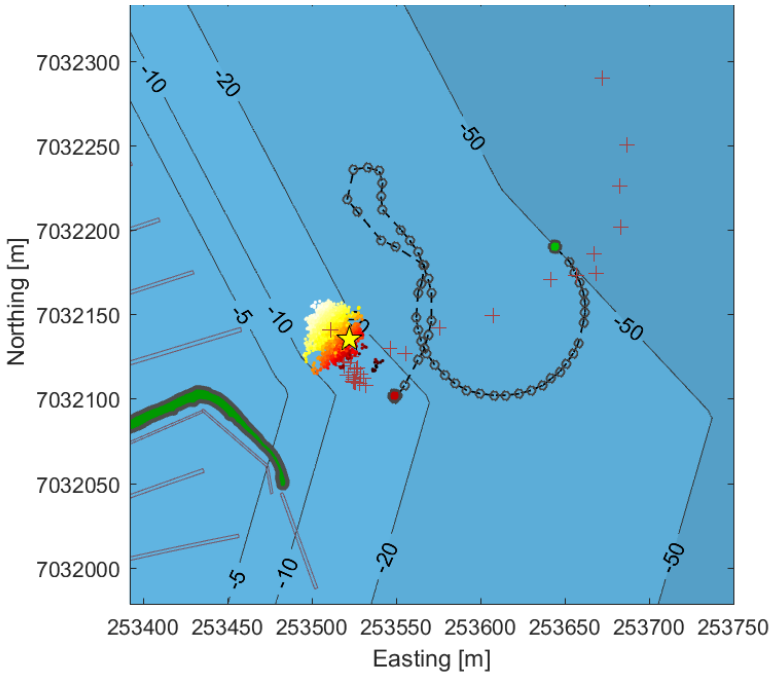


(a) Map plot of on-board Kalman Filter Estimates. White estimate markers are the most recent estimates. Green circle marks the start of the AUV trajectory and red marks the end.

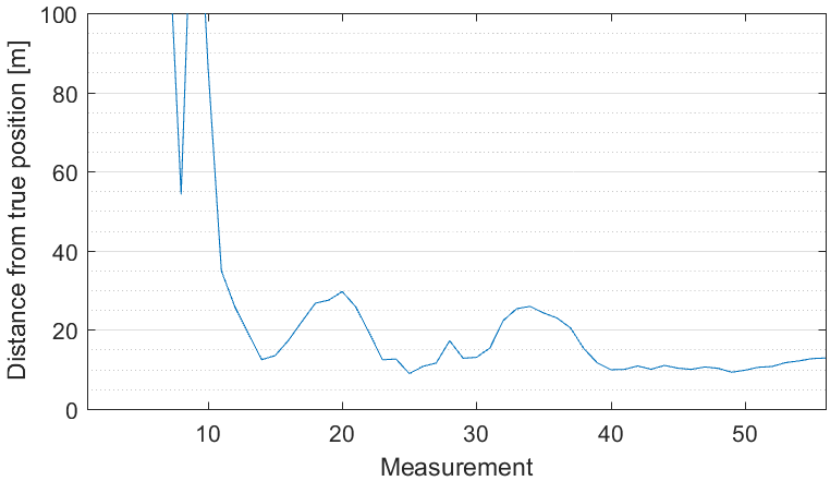


(b) Estimation error from true position

Figure 5.17: On-board Kalman Filter Estimation
Circular Trajectory with convergence on estimated position



(a) Map plot of The Particle Filter estimate and the particles.



(b) Estimation error from true position

Figure 5.18: Off-Board Particle Filter Estimation
Circular Trajectory with convergence on estimated position

5.3 Discussion of AUV Trial

The results show that using the algorithms and setup described one may expect position estimates less than 20 meters off target. If the path of the AUV is altered such that it circles the current estimate then the estimation accuracy may further improve to below 10 meters off target. These are considered quite acceptable results for the purpose of finer scale positioning of fish behavior.

For this first experiment with the AUV it was kept close to surface level such that the antenna of the vehicle for the most part was sticking out of the water. This was done so that we could maintain high speed communication with the AUV while it was operating, and let us monitor both its execution of the pre-planned paths and also the behavior of the positioning algorithm in real time. It also meant that GPS-positioning was available for the most part. While this was done in our case to maintain the connection there should be no issue operating the positioning algorithm below water with positioning data from the inertial navigation system combined with the doppler velocity log. As far as acoustics go it should in fact be beneficial to operate further below the surface ensuring that the hydrophone is properly submerged and has a proper reception of the signal.

It was noted that for all the circle trajectories there is a gap in the received measurements between 10-o-clock and 12-o-clock on the circle. The reason for this is assumed to be caused by the body of the AUV itself blocking the signal. The AUV body aligns tangentially on the circle and during this segment of the path the entire body of the AUV is between the emitter and receiver.

In this experiment it was assumed that SNR values would follow a similar linearised trend as was seen in the experiment with the boat. This was not the case however and caused the particle filter to incorrectly estimate that the emitter was farther away than it was. While it was unfortunate to discover the on-board particle filter estimates were off due to being fed exaggerated range estimates it was however not a major issue as the raw input data was kept. Rerunning the trial with a corrected particle filter therefore possible. It is also interesting to note that the the Kalman filter seemed to cope better with this and was able to preform well even with the erroneous input. The use of SNR for range estimation needs some further work, and a more robust model for SNR ranging which perhaps accounts for impacts by changes in salinity, temperature and other affecting factors should be developed before this can be fully taken advantage of. Using only the range difference hyperboloids have been shown to be fully possible, e.g. Figure 5.15, but it requires more iterations before a satisfactory convergence is reached.

Chapter 6

Conclusion

In this paper algorithms for acoustic emitter positioning using a single moving omni-directional hydrophone which utilizes the known periodicity of the emitters have been developed. The algorithms were first verified through modeling and simulations. They have also been field tested first using a boat as platform and data post-processed, then using an AUV and processed in real time.

It has been demonstrated that it is possible to perform on-board real time processing of measurement data on an AUV platform producing estimation results with a precision below 20 meters off target after 20 transmissions from the emitter. Often also converging to less than 10 meters off target when given additional time and measurements, especially when the AUV path is altered to home in on the current estimate.

Both the Kalman filter and particle filter have been tested with success. While both filtering algorithms produce very acceptable results in most cases it is clear that for this particular application that the particle filter offers additional robustness over the extended Kalman filter.

It would have been interesting to carry out further trials with the AUV if time had allowed, but sufficient data was collected to show that on-board processing of real time data provides satisfactory results. Most importantly this demonstrates that positioning of periodic emitter tags using a single receiver and an AUV platform is a viable approach for finer scale localization, and may serve as more cost efficient alternative to current methods.

Figure 6.1: LAUV Harald back on shore after completing its mission

Chapter 7

Further Work

7.1 Post-Detection Trajectory Planning

Now that it has been shown that the AUV is capable of positioning emitter tags using pre-planned paths the next logical step towards a fully autonomous tag detection and tracking is to automate the path planning.

The major undertaking in automating the path planning is in keeping the AUV within safe limits and avoiding obstacles. While the vehicle may be provided operation boundaries on a map there may be unforeseen obstacles which are not present on a map, such as other vessels or fishing equipment. A fully automated solution should take considerations toward other vessels and collect data to actively avoid collisions with any unforeseen objects.

Beyond that the planning itself should be quite straightforward and commanding the vessel with waypoints and other routines can be done through IMC messages. A viable method for detecting tags without any advance knowledge could be following some predetermined search path until a tag is detected. At that point the AUV could start circling some arbitrary point within range of the emitter and collect an initial estimate of its position. Results from both trials done in this paper show that this is a good approach in determining an initial estimate. When sufficient confidence in the estimate is achieved the AUV could start circling the estimate location to further improve it. If the AUV is tracking a living target then it may be beneficial to maintain a sufficient range as not to disturb the being.

A tag should likely not be considered detected until more than one message has been received. This is due to the occurrence of false messages which is when background noise follows the characteristics of an actual signal making it indistinguishable from an actual signal. There are checksums in place to reduce the likelihood of noise being interpreted as a signal, but given hours of service this still should be accounted for.

7.2 Increasing Temporal Resolution

Results from simulations show that increasing the time stamp resolution by a factor of 10, from 1 ms to 100 μs, would greatly increase the precision and convergence rate of the filters. The hydrophone used in the experiment is intended as an ultra low-power device and designed to operate for months on a single battery. Increasing the time resolution would affect the battery life time. An AUV however does not have the same constraints on power consumption and it would be beneficial to take advantage of a hydrophone with an even greater temporal resolution.

Bibliography

[1] S. Bancroft. "An Algebraic Solution of the GPS Equations". In: *IEEE Transactions on Aerospace and Electronic Systems* AES-21.1 (Jan. 1985), pp. 56–59. ISSN: 0018-9251. DOI: 10.1109/TAES.1985.310538.

[2] Ralph Bucher and D. Misra. "A Synthesizable VHDL Model of the Exact Solution for Three-dimensional Hyperbolic Positioning System". In: *VLSI Design* 15.2 (2002), pp. 507–520. DOI: 10.1080/1065514021000012129.

[3] Fiona Fletcher, Branko Ristic and Darko Musicki. "Recursive estimation of emitter location using TDOA measurements from two UAVs". In: *2007 10th International Conference on Information Fusion*. July 2007, pp. 1–8. DOI: 10.1109/ICIF.2007.4408174.

[4] C. Forney et al. "Tracking of a tagged leopard shark with an AUV: Sensor calibration and state estimation". In: *2012 IEEE International Conference on Robotics and Automation*. May 2012, pp. 5315–5321. DOI: 10.1109/ICRA.2012.6224991.

[5] Mark Garrison. *Measurement Error & Logic*. Saleae. Sept. 2016. URL: http://support.saleae.com/hc/en-us/articles/208667166-Measurement-Error-Logic-timing-digital-pulse-width- (visited on 26/01/2017).

[6] Jeroen D. Hol, Thomas B. Schon and Fredrik Gustafsson. *On resampling algorithms for particle filters*. Publication. Division of Automatic Control, Department of Electrical Engineering, Linkoping University, 2006.

[7] Charles F. F. Karney. "Transverse Mercator with an accuracy of a few nanometers". In: *Journal of Geodesy* 85.8 (2011), pp. 475–485. ISSN: 1432-1394. DOI: 10.1007/s00190-011-0445-3. URL: http://dx.doi.org/10.1007/s00190-011-0445-3.

[8] Conrad Sanderson and Ryan Curtin. "Armadillo: a template-based C++ library for linear algebra." In: *Journal of Open Source Software* 1 (2016), p. 26. URL: http://arma.sourceforge.net/armadillo_joss_2016.pdf (visited on 19/01/2017).

[9] Dan Simon. *Optimal State Estimation*. John Wiley & Sons, Inc., 2006. ISBN: 978-0-471-70858-2.

[10] Karl Spingarn. “Passive Position Location Estimation Using the Extended Kalman Filter”. In: *IEEE Transactions on Areospace and Electronci Systems* AES-23.4 (July 1987).

[11] Connor F. White et al. “Human vs robot: Comparing the viability and utility of autonomous underwater vehicles for the acoustic telemetry tracking of marine organisms”. In: *Journal of Experimental Marine Biology and Ecology* 485 (2016), pp. 112–118. ISSN: 0022-0981. DOI: http://dx.doi.org/10.1016/j.jembe.2016.08.010. URL: http://www.sciencedirect.com/science/article/pii/S0022098116301502.

[12] George S. K. Wong and Shi-ming Zhu. “Speed of sound in seawater as a function of salinity, temperature, and pressure”. In: *The Journal of the Acoustical Society of America* 97.3 (1995), pp. 1732–1736. DOI: 10.1121/1.413048. eprint: http://dx.doi.org/10.1121/1.413048. URL: http://dx.doi.org/10.1121/1.413048.