Machine Learning methods for Surface Echo Tracking and Identification for Radar Level Gauges

Master's thesis in Complex Adaptive Systems

JOSEFIN NILSSON & WENJIN YUAN

DEPARTMENT OF ELECTRICAL ENGINEERING

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2022
www.chalmers.se

Master's thesis 2022

Machine Learning methods for Surface Echo Tracking and Identification for Radar Level Gauges

JOSEFIN NILSSON & WENJIN YUAN

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden 2022

Machine Learning methods for Surface Echo Tracking and Identification for Radar Level Gauges
JOSEFIN NILSSON & WENJIN YUAN

Supervisor: Christer Frövik, Emerson
Examiner: Jonas Sjöberg, Chalmers University of Technology

Master's Thesis 2022
Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2022

Machine Learning methods for Surface Echo Tracking and Identification for Radar Level Gauges

JOSEFIN NILSSON & WENJIN YUAN
Department of Electrical Engineering
Chalmers University of Technology

Abstract

This thesis presents two methods for liquid surface identification on radar level gauges. One method is a tracking algorithm that utilizes velocity differentiation between objects, and the other one is a supervised learning algorithm that learns the general solution for a specific tank including the relation between the liquid surface level and other reflecting objects in the tank. The main contribution behind both methods is the application of Range-Doppler Fast Fourier transform, in which velocity differentiation is obtained.

By adding velocity information to the tracking algorithm, it could easily distinguish the liquid surface even when it is close to another reflecting object in both amplitude and distance. This improved the performance on all generated test cases where a disturbance and liquid end up at the same level for some time. However, the tracking algorithm will always demand previous detections as its source of reference when classifying new detections, causing the algorithm to be totally useless when previous detections are unknown. The supervised learning algorithm on the other hand, is expected to find the liquid surface level when previous detections are unknown. Therefore, feed-forward neural networks were trained and evaluated on three different feature sets. The first feature set contains ratio relations between distance, amplitude and velocity for all detections made at one time, the second feature set contains amplitude information about one detection, and its surrounding amplitudes on a 2D grid, the third feature set is simply a combination of the two. The network with best performance was trained on both peak ratios and grid-based amplitudes, where it classified the liquid surface level correctly for all generated test cases. Concluding that for best results the training data needs to include both velocity information as well as relationships between the liquid surface and other reflecting objects.

It is noteworthy that all models in this thesis are evaluated on simulated data and can therefore only be used for demonstrative purposes.

Keywords: FMCW radar, Range-Doppler FFT, Target tracking, Neural network, Classification.

Acknowledgements

This thesis has been carried out at Emerson Automation Solutions in Mölnlycke, spring 2022. We want to thank our supervisor Christer Frövik at Emerson and our examiner Jonas Sjöberg at Chalmers University of Technology, for their guidance and useful viewpoints throughout the work.

Josefin Nilsson & Wenjin Yuan, Gothenburg, 2022

Nomenclature

Below is the nomenclature that have been used throughout this thesis.

B	Bandwidth of a chirp
T_s	Chirp time
T_r	Duration time between two chirps
f_t	Frequency of transmitted signal
f_r	Frequency of received signal
f_{if}	Intermediate frequency
τ	Time delay between transmitted and received signal
d	Distance from radar to detected object
c	Speed of light
ω	Phase shift
λ	Wavelength
v	Velocity of detected object
D_f	Doppler frame
Δv	Velocity resolution
N	Number of neurons
g	Activation function
θ	Threshold of a neuron
b	Argument of activation function
s	State of a neuron
w	Network weight
E	Error for weight updates
η	Learning rate
y	True labels
$\hat{y}$	Predicted labels
ε_r	Relative Dielectric Constant
I	Intensity

A	Amplitude
T	Total time of a measurement
P	Number of peaks detected at one time step
CMP	Closest moving peak feature
AR	Amplitude Ratio Feature
dR	Distance Ratio Feature
$\mathbf{x}$	Detection peak
p	Center point of a peak in range-Doppler map
M	Number of detections in memory
μ	Mean value of M previous surface detections (Tracking target)
D	Distance to target
$\mathcal{S}$	Set of liquid surface indices
ϕ	Score metric

Contents

List of Figures

List of Tables

1
Introduction

This chapter presents an introduction and motivation to why machine learning methods are of interest for radar level gauges. It also describes the scope of this thesis, defined issues, and how these are solved.

1.1 Background

Emerson provides FMCW radars for liquid surface level gauging in process tanks. Details about radar technique will be described in Chapter 2. Following subsections will describe why level measurement is of importance and when surface level gauging becomes difficult in existing applications.

1.1.1 Radar Level Gauging in Tanks

Figure 1.1: Process tank and its different constructions inside, which give rise to several peaks in a measured spectrum using level gauges.

Liquid level measurement in process tanks is essential for safety applications as well as optimization of process control and inventory management. Therefore, reliable estimates of the liquid surface level are of high importance. A tank can have one or more constructions inside, including beams, mixers, etc, which also reflects the radar signal. This makes it hard to distinguish the liquid surface level from other objects. Often the measurement work well with minimal configuration, but for more difficult situations, the user needs to manually input quite a lot information about the tank properties and its constructions inside. This is a problem Emerson deals

with today and would like to approach with new ideas, such as machine learning, in order to distinguish the liquid surface echo from other reflecting objects.

This thesis considers non-contacting radar gauges that are installed at the top of the tank providing a top-down measurement of distance. Figure 1.1 illustrates an example of a tank and its different constructions inside. Each construction will cause a reflection of the emitted signal from the radar, revealing its position. The distances to each object are obtained from a measured spectrum, read more about this in Chapter 2.

In tanks, liquid can be kept at a constant level, being drained, or being filled. This is illustrated in Figure 1.2, where an oil tank has first been filled and then drained. Positions of several objects causing detection in the level gauges are plotted over time.

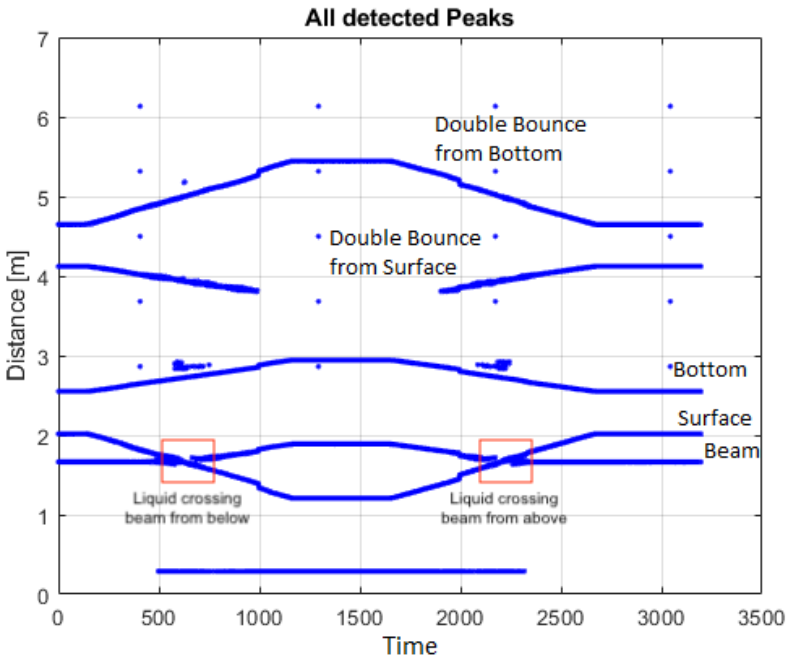


Figure 1.2: Detected positions by Emerson's level gauge of objects in an oil tank while being filled and drained.

During a typical filling and- or draining process, the liquid surface will pass the disturbances inside the tank if there is any. In Figure 1.2, one can observe that the liquid surface echo crosses a beam twice during the measured series. These are marked with red rectangles. Note also, the echo position from any disturbance under the liquid surface appears to be displaced due to slower propagation speed for radar wave through the liquid [1]. Consequently, if the liquid surface is moving above a disturbance, the disturbances will also appear to be moving due to refraction [2]. This can be observed in Figure 1.2 as the bottom echo appears to be moving, as well as the beam echo appears to be moving when it is below the liquid surface level and stationary otherwise. This knowledge is essential for the methods presented in

Chapter 4 since it is a beneficial characteristic for the liquid surface identification. By reasoning that if there are multiple detections at one time with velocity, the liquid surface is always the nearest of them. More about this characteristic will be explained in Subsection 6.1.1, Chapter 6. What is more, when the liquid is below a disturbance, the energy transmitted from the radar is partly blocked by the disturbance. Therefore, if the liquid is being filled and crosses the disturbance from below, the liquid surface will increase with a jump-discontinuity in amplitude level. Similarly, amplitude level will drop with a jump-discontinuity when the liquid is being drained and crosses the disturbance from above. What is more, the crossing of two echoes creates a "cloud" of detections due to interference between the two signals. Both the jump-discontinuity and interference cloud are shown in Figure 1.5, for a liquid surface passing a disturbance twice: first from below and then from above.

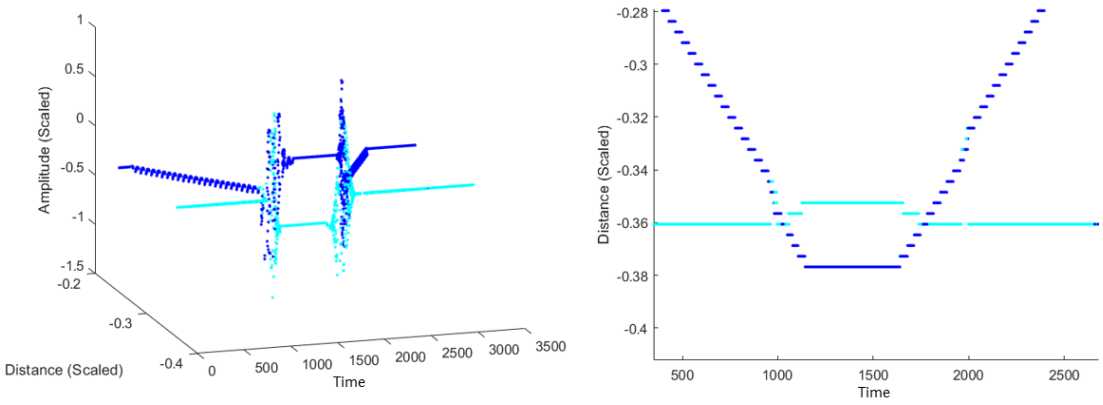


Figure 1.3: Representation of liquid (dark blue) passing a disturbance (light blue) in amplitude-distance-time domain to the left and distance-time domain to the right. The crossing causes interference clouds and jump discontinuity in amplitude, shown to the left.

Following two subsections (1.1.2, 1.1.3) describes two different issues for liquid surface classification.

1.1.2 Crossing Echoes Problem

In today's gauges new detections are considered to belong to the same object if it has the shortest Euclidean distance to previous detections. This method is based on a tracking algorithm and is simplified and illustrated in Figure 1.4. Note that different colors aim to demonstrate signals reflected from two separate objects, without any specific definition of object type.

Initially, the problem seems straightforward. However, complications occur when the liquid surface is crossing other objects, resulting in two objects being at the

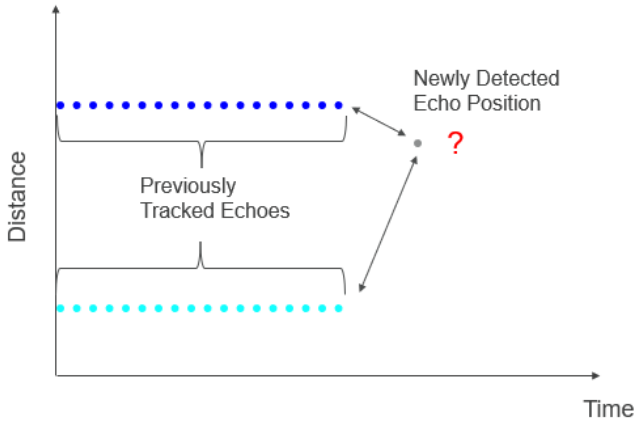


Figure 1.4: Tracking problem on a newly detected echo position. Since the newly detected echo position is closest to the dark blue echo it will be considered part of that track.

same distance from the gauge. Once the liquid surface is leaving that position and detections starts to divide into two distinct distances again, it is hard to determine which one to be the liquid surface. This also causes a problem for further detections since previous knowledge about the surface positions are lost. This is illustrated in Figure 1.5 and we refer to this as the crossing echoes problem.

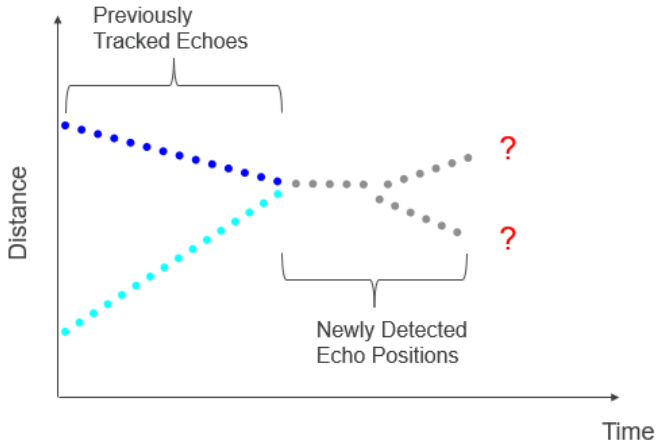


Figure 1.5: Crossing echoes problem when several echoes cross and separate.

1.1.3 Loss of Echoes Problem

The problem gets even more complicated when the liquid surface not only crosses another object but previous levels becomes unknown. This problem happens at for example a power outage, giving a time interval without any detections, and the liquid surface could have moved during that time. If this occur, the detections have

to be classified without any previous knowledge about the positions. This can be done using a supervised learning algorithm, instead of a tracking algorithm, which can learn structures in a specific tank. The problem is visually illustrated in Figure 1.6 and refered to as the loss of echoes problem.

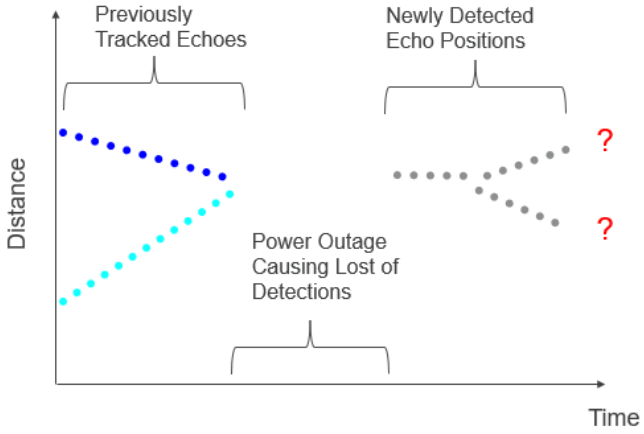


Figure 1.6: Tracking problem when there is a loss of detections.

1.2 Limitations and Scope

The main limitation in this thesis is the lack of recorded measured signals from real tanks, leading to difficulties simulating varying environments. The most easily interpretable recorded measurement is the oil tank illustrated in Figure 1.2. The simulations in this thesis are therefore based on the observations from this oil tank. Thus, neither foaming nor turbulent liquid are considered. Situations such as empty tank and liquid extremely close to the radar are also disregarded. Due to these calm conditions, the radar will always receive a detection of the liquid surface.

The simulation tool in its original form lacks the amplitude variations issued in Subsection 1.1.1, neither does it consider the shape of the tank or double bounces. However, the amplitude variation will be implemented in this thesis based on observations from the oil tank. This implementation will be further described in Chapter 4.

Further, the provided methods in this project only consider liquid surface identification. Everything besides the liquid surface is considered as disturbances, leading to a binary classification problem. Additionally, the supervised learning model that will be used in this project, described in Chapter 6, is trained on one type of tank circumstances and a liquid with slightly different amplitude. This is due to the knowledge that a tank can look very different, such as the number of mixers, beams inside and whether it has a rounded bottom or flat. More information about the

tank that the training data is based on is described in Section 4.2. However, the supervised model can be extended on more types of tanks and for more varied types of liquid amplitudes, but it is not covered in this thesis, but will be discussed in Future work, Section 8.1.

1.3 Contributions

The main contribution of this thesis is solving the two issues presented in Subsections 1.1.2, and 1.1.3. These are summarized and listed below:

1. Solving the crossing echoes problem (Subsection 1.1.2) by developing a tracking algorithm that utilizes velocity differentiation between objects. Method for this problem is presented in Chapter 5. Results are presented in Section 7.2.
2. Distinguish liquid surface in the loss of echoes problem (Subsection 1.1.3) by training a supervised algorithm. The feature set contains relations between the liquid surface and other reflecting objects, as well as its surrounding amplitudes on a 2D grid obtained by Range-Doppler FFT. Complete information about this method is covered in Chapter 6. Results are presented in Section 7.3.

Additionally, we have added distance-based amplitude variation to the original simulation tool (Section 4.1) and a custom-made evaluation metric for the problem (Section 7.1).

1.4 Outline

Chapter 2 provides theory about the radar gauges and signal processing for object detection. Chapter 3 presents basic ideas behind training a supervised learning algorithm. Details about signal simulations and methodology for processing the signals are given in Chapter 4, and so are test cases used for evaluation. The tracking algorithm with added velocity information is described in Chapter 5, and the supervised learning algorithm is presented in Chapter 6. Chapter 6 also include details of the feature sets used for training the algorithm. Results of this thesis are presented in Chapter 7, where Section 7.2 and 7.3 present the tracking and supervised model respectively. A following discussion is in Chapter 8. The final outcome is summarized in Chapter 9.

This thesis also includes Appendix A providing the test cases used for evaluation, Appendix B and C with illustrated respectively numerical results from the supervised model that are not included in Section 7.3.

2
Radar Level Measurement

In the following sections, fundamentals of radar signal processing are introduced. Starting with an introduction on how the raw signal is obtained by using FMCW radar, followed by information about how distance and velocity are estimated.

2.1 FMCW Radar

Emerson's latest non-contacting gauges offer Frequency-Modulated Continuous-Wave (FMCW) radar. FMCW radar modulates its operating frequency during the measurement by transmitting a signal whose frequency increases with time. After the transmitted signal hit a liquid surface, it is reflected to the gauge with a time-delay proportional to the distance to the liquid surface, [3], [4].

The transmitted signal is called a chirp and is characterized by a start frequency f_0 at time t_0 and an end frequency f_1 at time t_1, giving bandwidth $B = f_1 - f_0$ and chirp time $T_s = t_1 - t_0$. Figure 2.1 illustrates the transmitted signal with frequency f_t reflected from a single liquid surface in front of the gauge, resulting in one received signal with frequency f_r. Note that the received signal is time-delayed with $\tau = \frac{2d}{c}$ units, where d is the distance from the gauge to the detected liquid surface and c is the speed of light. The radar wave has speed c, assuming vacuum or air between the gauge and the liquid surface.

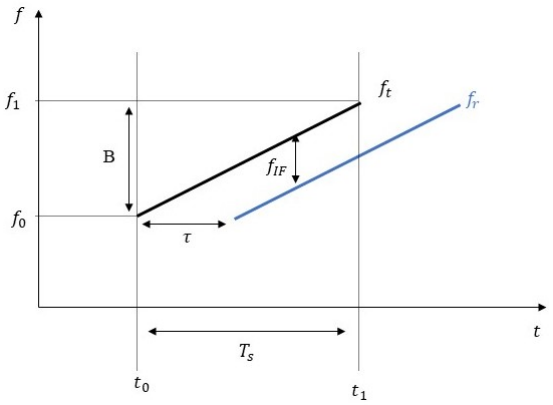


Figure 2.1: Transmitted and received signal, with frequencies f_t and f_r respectively, in an FMCW system with a single object in front of the radar. Bandwidth $B = f_1 - f_0$, chirp time $T_s = t_1 - t_0$, time-delay τ, and frequency shift f_{IF}.

The intermediate frequency f_{IF} is obtained by multiplication of the received signal and the transmitted signal. Assume the transmitted signal as $\cos(2 \cdot \pi f_t \cdot t)$ and the received signal $\cos(2 \cdot \pi f_r \cdot t)$. The multiplication rule of cosine gives

$$\begin{aligned} &\cos(2 \cdot \pi \cdot f_t \cdot t) \cdot \cos(2 \cdot \pi \cdot f_r \cdot t) = \\ &\frac{\cos(2 \cdot \pi \cdot (f_t - f_r) \cdot t)}{2} + \frac{\cos(2 \cdot \pi \cdot (f_t + f_r) \cdot t)}{2} \end{aligned} \tag{2.1}$$

When low-pass filtering (2.1) the addition part disappears due to its high frequency content. Taking the remaining frequency response from (2.1), we get the intermediate frequency which is the difference between f_t and f_r. The signal with the intermediate frequency will from now on be called the IF signal.

2.2 Signal Processing for Level Measurement

Since the intermediate frequency f_{IF} consists of multiple frequencies it must be processed using a Fourier transform to separate them to individual frequencies. Detection of distance is done by applying Fourier transform on the IF signal, and velocity can be detected from a second Fourier transform. More about these signal processing steps in following sections.

2.2.1 Distance Estimation

Distance is estimated using the fact that f_{IF} is also equal to τ multiplied by the slope of the ramp

$$f_t - f_r = f_{IF} = \tau \cdot \frac{B}{T_s}, \tag{2.2}$$

obtained from triangle similarities according to

$$\frac{\tau}{T_s} = \frac{f_{IF}}{B}. \tag{2.3}$$

The distance d is as mentioned proportional to the time shift $\tau = \frac{2d}{c}$, thus the problem of distance estimation becomes to find f_{IF} and its time shift. If there are multiple liquid surfaces in front of the radar, the antenna will pick up multiple received signals, each with a time delay proportional to the distance of one specific liquid surface.

To receive these individual frequencies, the IF signal needs some processing which is done in the radar software. First, the signal is sampled by a low pass filter (LPF) and an analog-to-digital (ADC) converter. Secondly, a Fast Fourier Transform (FFT) is performed on the ADC data creating a frequency spectrum. The peaks in the spectrum are directly corresponding to the distance of the liquid surfaces, [4].

2.2.2 Velocity Estimation

When the liquid in a tank is filled or drained, the surface may pass a disturbance echo. At some point the two echoes will be at the same distance from the gauge, resulting in just one peak in the FFT-spectrum. Here, velocity estimation is useful to resolve the objects detected within the same range.

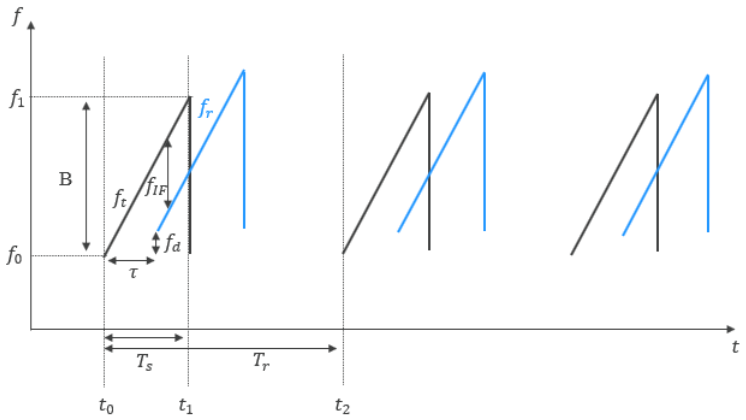


Figure 2.2: Illustration of the received signal shifting in frequency domain due to both distance (f_{IF}) and velocity (f_d). Bandwidth $B = f_1 - f_0$, chirp time $T_s = t_1 - t_0$, duration time $T_r = t_2 - t_0$, time-delay τ, and frequency shift f_{IF}. The blue chirps represent the received signals with frequency f_r and the black chirps the transmitted signals with frequency f_t.

The intermediate frequency caused by a moving target in front of the radar as illustrated in Figure 2.2 is described

$$f_{IF} = \frac{2d}{c} \cdot \frac{B}{T_s} + \frac{2vf_0}{c}. \tag{2.4}$$

However, the contribution from the second term which contains the velocity information in (2.4) is extremely small compared to the first term. Therefore, the term will be ignored and not be used for velocity estimation. The velocity estimation is rather carried out by studying the phase shift of the signal over multiple chirps. The phase shift between multiple chirps is directly proportional to the velocity of the target.

As the radar is transmitting signals with duration time T_r, if the liquid surface moves with a velocity v, it will have moved the distance vT_r during time T_r. The measured phase shift ω of the liquid surface alone, is therefore given by this distance multiplied by 4π divided by the wavelength λ,

$$\omega = \frac{4\pi v T_r}{\lambda}. \tag{2.5}$$

By rearranging (2.5), the velocity v can be calculated from the phase shift ω. The same technique is used for multiple objects in front of the radar. Suppose that the liquid surface moves with a velocity v_1 and the disturbance with v_2, resulting in frequency ω_1 respectively ω_2. The phases can be measured and the velocities can be back-calculated from

$$v_1 = \frac{\lambda\omega_1}{4\pi T_r}, v_2 = \frac{\lambda\omega_2}{4\pi T_r} \tag{2.6}$$

Thereby, an FFT performed on several time frames resolves the phase shift caused by moving targets, and therefor also the velocities. This is called Doppler-FFT, [4]. To help understand the connections among targets the signal is visualized in the Range-Doppler domain.

2.2.3 Range-Doppler FFT

The regular FFT is used to calculate the spectra from where the distance and amplitude of radar echoes can be detected. Additionally, the Range-Doppler FFT can be used to calculate a 2D map from where distance, amplitude, and velocity information can be read out. While a single chirp is used to calculate the regular FFT, a sequence of chirps is used as input to the Range-Doppler FFT. Mathematically, the Range-Doppler FFT is essentially just a 2D version of the FFT, [4]. Figure 2.3 presents the process, starting with performing FFT on the IF signals resolving amplitudes and distances. Next the second FFT is performed across multiple chirps resolving velocity. Resulting in one matrix per time, containing amplitude values, with column indices giving velocities and row indices giving distances.

To separate two equidistant objects with similar velocities the Range-Doppler velocity resolution must be sufficiently small. This resolution is dependent on the number of chirps on which the Doppler FFT is performed, called Doppler frame (D_f). For an FFT to separate two frequencies on a sequence of length D_f it is required that the phase shift $\Delta\omega > 2\pi/D_f$. The velocity resolution Δv can be derived as

$$\Delta v = \frac{\lambda}{2D_f T_s}. \tag{2.7}$$

Therefore, a larger D_f implies detecting decreasing velocity differences. However, a larger D_f also generates a greater Range-Doppler matrix leading to more required memory and computational resources.

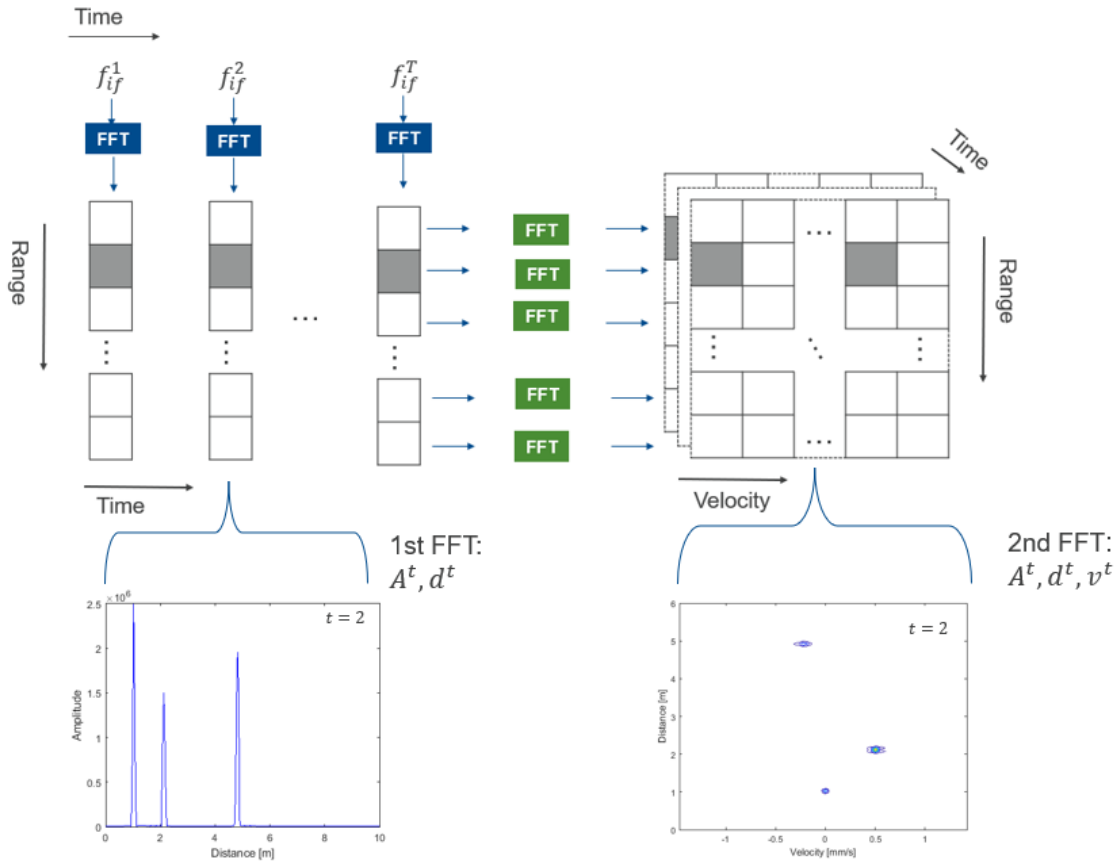


Figure 2.3: Range-Doppler signal processing and visualization of results from first and second FFT. Results from first FFT can be visualized as the lower left image with 2D peaks at detected distances, and results after second FFT can be visualized as a contour map shown in the lower right image.

3

Supervised Learning using Multilayer Perceptron

This chapter introduces some basics regarding supervised learning using Multilayer Perceptron. Concepts of artificial neurons, how these are connected into fully connected networks and how the learning process is carried out are described in each section.

3.1 Supervised Learning

Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. Given several training samples and associated outcomes, the algorithm iteratively fits an input-output function to the training samples. Supervised learning with deep networks is essentially regression analysis, often with many function fitting parameters. One main issue with complicated neural networks is the number of parameters hence difficult to optimize. Therefore, machine learning with neural network is not suitable for every problem, hence one needs understanding of both the algorithms and the actual problem to determine its suitability, [5].

For the network to learn the structures, a training set containing a list of input features and corresponding labels is provided. The data used for this thesis and extracted features are presented in Chapter 4 and Section 6.1 respectively. The network is trained by adjusting its model weights and applying these to unseen data and make predictions.

3.2 Artificial Neurons

Artificial neurons are at the heart of artificial neural networks. An artificial neuron is the computational unit of the network and a mathematical representation of a biological neuron. McCulloch and Pitts modeled the neurons as a binary threshold unit with only two possible outputs: active or inactive. To compute the output, the function sums up all weighted inputs and outputs values from a binary set (usually 0,1 or -1,1) based on a given threshold. If the sum exceeds that given threshold, the state of the neuron is said to be active, and inactive otherwise. Hence, the function which computes the state of each neuron is called an activation function. The function performs repeated calculations in discrete steps denoted as $t = 0, 1, 2, 3, ...$

Given N artificial neurons performing computations in parallel, the state of neuron i at step $t+1$ is described below,

$$s_i(t+1) = g(\sum_{j=1}^{N} w_{ij} s_j(t) - \theta_i), \tag{3.1}$$

where g is the activation function of the neuron and θ_i is the threshold. More information about activation functions is given in Subsection 3.4. What is more, j denotes the index of neuron from step t, neuron i at step $t+1$ is the one that does the computation of its state.[5] A schematic diagram of the mathematical representation of an artificial neuron is illustrated in Figure 3.1.

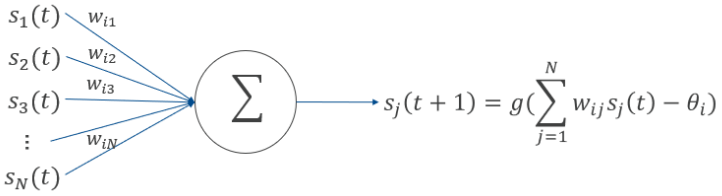


Figure 3.1: Schematic diagram of a McCulloch-Pitts neuron.

Further, we will denote the argument of the activation function for with b, called the local field. In the case of (3.1), we have

$$\begin{aligned} s_i(t+1) &= g(b_i), \\ b_i &= \sum_{j=1}^{N} w_{ij} s_j(t) - \theta_i. \end{aligned} \tag{3.2}$$

In short, the artificial neuron performs the weighted average of the previous states $s_1(t), ...s_N(t)$ in order to calculate a new state. The parameters w_{ij} are *weights*, representing a numeric relationship between the neurons. Weights can be positive and negative, and there is no connection when $w_{ij} = 0$.

3.3 Multilayer Perceptron

In 1958, Rosenblatt suggested to connect McCulloch-Pitts neurons into layered networks, which he referred as perceptrons (a single layer of an artificial neural network), [6]. A network of one layer can only solve linearly separable problems. By stacking layers in a network, the network will be able to solve linearly inseparable problems as well. The layout of a multilayer perceptron is illustrated in Figure 3.2. This is also commonly referred to as the feed-forward neural network. The leftmost layer consists of input terminals for the input data and therefore has as many neurons as the number of features of one sample. In Figure 3.2, the feature dimension of input data is denoted as ρ. In the middle, there are m hidden layers where each layer consists of n neurons. The hidden neurons are denoted as h_n^m. The rightmost layer consists of one output neuron, which outputs from a binary set consisting of either $\{0, 1\}$ or $\{-1, 1\}$ (depending on the activation function of choice) for binary classification tasks, [5].

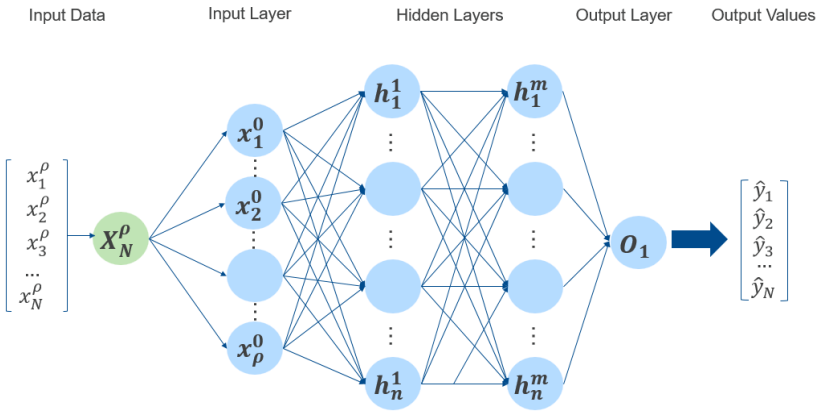


Figure 3.2: Illustration of a feed-forward neural network.

3.4 Activation Functions

As mentioned in Subsection 3.2, each neuron performs an operation on the weighted sum of every previous step. This operation is called an activation function and referred as g. If function g takes the form of a linear function, the neuron will perform linear regression or classification. However, in artificial neural networks, activation functions are in general non-linear functions which can solve classification tasks that are linearly inseparable. Common activation functions are: *Sigmoid*, *tanh*, *ReLU*, *Signum*, $Softmax$. They are either continuous or discrete functions; linear, non-linear, or piecewise linear functions, [5].

3.5 Backpropagation of Error

The numerical representation of connections between neurons are the weights, as mentioned in Subsection 3.2. The task for the algorithm is to find values for the weights that will optimize the network's classification performance. This is called *training* in the sense of artificial neural networks. At the beginning of a training process, the weights are initialized to small random values. After this, forward propagation will be performed for a training sample by the neurons and outputs a final value. This value is then compared with the true value. The discrepancy between the output value and the true value will then decide how the weights, or connection between neurons should be adjusted. The discrepancy is calculated by a *Loss Function* of choice. From here, the training objective is to minimize the discrepancy between the network output and the true targets of the samples. The training process, therefore, becomes an optimization problem and can be achieved by some optimization algorithm. Recall (3.1), given a loss function L, computing the error E_N between the true target y_N and computed output $\hat{y}_N$, its gradients with respect to weight w_{ki} can be found by partial differentiation

$$\frac{\partial E_N}{\partial w_{ki}} = \frac{\partial E_N}{\partial s_k} \frac{\partial s_k}{\partial b_k} \frac{\partial b_k}{\partial w_{ki}} \tag{3.3}$$

where k denotes the single output neuron for binary classification, hence $k = 1$, [5]. When the gradient is calculated, one can use an optimization algorithm such as gradient descent for updating the weights

$$w_{ki}^{new} = w_{ki}^{old} - \eta \frac{\partial E_N}{\partial w_{ki}}, \tag{3.4}$$

where η is the learning rate and usually a small number between 0 and 1, [7]. What is more, by applying chain rule, the error gradient can be computed for each weight in every previous layer in the network, hence the name "back propagation". For neurons in previous layer with respect to the layer where neuron i is located, the back propagation towards a neuron with index j proceeds as

$$\begin{aligned} \frac{\partial E_N}{\partial w_{ij}} &= \sum_k \frac{\partial E_N}{\partial b_k} \frac{\partial b_k}{\partial b_i} \frac{\partial b_i}{\partial w_{ij}} = \sum_k E_k \frac{\partial b_k}{\partial b_i} \frac{\partial b_i}{\partial w_{ij}}, \\ \frac{\partial b_k}{\partial b_j} &= \frac{\partial b_k}{\partial s_i} \frac{\partial s_i}{\partial b_i} = w_{ki} \frac{\partial s_i}{\partial b_i}. \end{aligned} \tag{3.5}$$

The same procedure is applied for updating bias where one takes the gradient of the error with respect to the bias from the previous neuron instead of the weight.
However, when the weights have been updated with respect to the first sample, the next sample will be presented to the network and so on. When the last training sample has been reached, and one *epoch* of training has been completed.

4

Signal Simulations and Processing

This chapter describes some of the important characteristics in the simulation tool provided by Emerson. General properties, simulations used for training the supervised network and simulations used for testing both the tracking and supervised learning algorithm are described in each section. Further, applied signal processing methods to extract important information are also described in this chapter.

4.1 Radar Level Gauge Simulation

The simulation tool provided by Emerson simulates IF signals mentioned in Section 2.1, representing different objects in a tank with a flat bottom. As mentioned in Subsection 1.1.1, objects positioned below a liquid surface will appear as displaced due to refraction [2]. The visual displacement is an effect due to the delay incurred by the reduced propagation speed of the radar energy through the liquid surface. This phenomenon is simulated as

$$d_{refraction} = d_a + d_u\sqrt{\varepsilon_r}, \tag{4.1}$$

where d_a is the distance from the radar to the liquid surface (a for *above*), and d_u is the distance from the liquid surface to the object that is under the liquid surface (u for *under*), and ε_r is the (Relative) Dielectric Constant or Relative Permittivity [1]. This also causes stationary objects below a moving liquid surface to appear as moving, due to its refracted distance caused by the liquid surface.

By its original form, the simulation tool does not consider any amplitude variation with respect to the distance to the radar or any jump discontinuity discussed in Subsection 1.1.1. However, amplitude variation with respect to distance and jump discontinuity has been implemented according to:

1. Amplitude varying according to the inverse square law $I \propto 1/d^2$ and $A^2 \propto I$ [12], making the relationship between amplitude and distance $A \propto 1/d$. Where I = intensity, A = amplitude and d = distance. In this thesis, $A \propto 1/d$ is carried out as $A = 1/d \cdot k$. Where k is a constant that differentiates different liquid types. Note that, if the distance to the measured object is below the liquid surface, the considered distance is the refracted distance.
2. Jump discontinuity caused by the assumption that a disturbance in front of the radar blocks 60% of its energy.

The parameters set for simulating the FMCW radar are found in Table 4.1 and is used for both training and test simulations described in following two subsections.

Table 4.1: Parameters for both training- and test simulations

Parameter	Value
Start Frequency	25.5 GHz
Chirp bandwidth	$2.9 * 10^9$ bps
Doppler frame	32
Maximum range	25.6m
Range resolution	0.0256m
Maximum velocity	2.9mm/time unit
Velocity resolution	0.0918mm/time unit

4.2 Simulation of Training Data for Supervised Learning

To solve the loss of echoes problem using supervised learning, mentioned in Section 1.3, simulation of filling a tank between the liquid level 1.5m to 2.5m were performed in a 4m deep tank. One disturbance is positioned 2m from the radar, which blocks 60% of the radar energy. The liquid surface velocity during this simulation was set to $8.5 \cdot 10^{-4}$m per time unit. The time unit used here is the chirp time mentioned in Section 2.1. Recall the distance-based amplitude variation mentioned in Section 4.1, the liquid specific constant k used for the training simulation is $3.3362 \cdot 10^4$.

The reason we chose to only train the model with signals extracted from the filling scenario is because there will be draining in test data that will be described in Section 4.3. We would like to investigate whether training on filling alone is enough for the model to perform the liquid surface identification task on both filling and draining scenarios. This is because we assume that draining is a mirrored version in time of the filling data and vice versa.

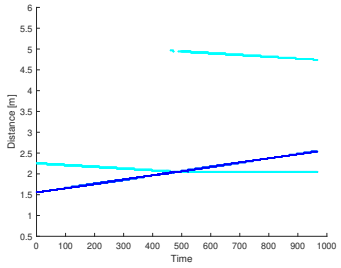
4.3 Test Cases for Performance Evaluation

Three types of test cases are presented for evaluating results for both the crossing echoes problem and loss of echoes problem. The test cases all represent situations where the liquid surface and a disturbance end up at the same level for some time, where the algorithm must choose between the two detections. We simulated three types of test cases in the same tank environment, i.e. a 4m deep tank with a disturbance at 2m, according to the scope in Section 1.2. The liquid specific constant k (Section 4.1) used for the test cases is $3.2027 \cdot 10^4$. One example of each case type

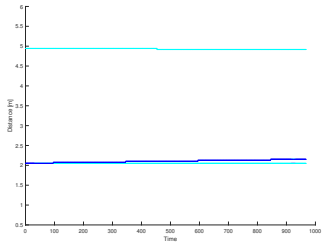
is illustrated in Figure 4.1 and are defined as follows:

1. Liquid surface moves and passes the disturbance once with velocity $\pm 1 \cdot 10^{-3}$m per time unit. (Figure 4.1a)
2. Liquid surface moves slowly with velocity $\pm 1 \cdot 10^{-4}$m per time unit. close to the disturbance. (Figure 4.1b)
3. Liquid surface moves to the disturbance position with velocity $\pm 2 \cdot 10^{-3}$m per time unit, stops, and then move again either in the same direction as before or the other. (Figure 4.1c)

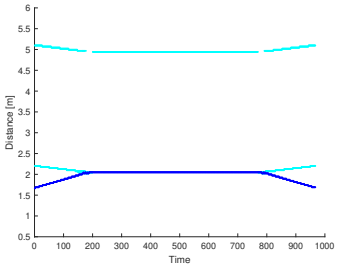
Because the liquid can be filled or drained, Case 1 and 2 has two different sub-cases each and Case 3 has four different sub-cases. We will refer the sub-cases with letters a to d. There are in total 8 test cases. The cases visualized in Figure 4.1 are 1a, 2a, and 3a. For visual representations other than the ones shown in Figure 4.1c, see Appendix A.



(a) Case 1 when liquid surface crosses the disturbance from above to below with some velocity $-v$.



(b) Case 2 when liquid surface crosses the disturbance from slightly above to below with some velocity $-v$.



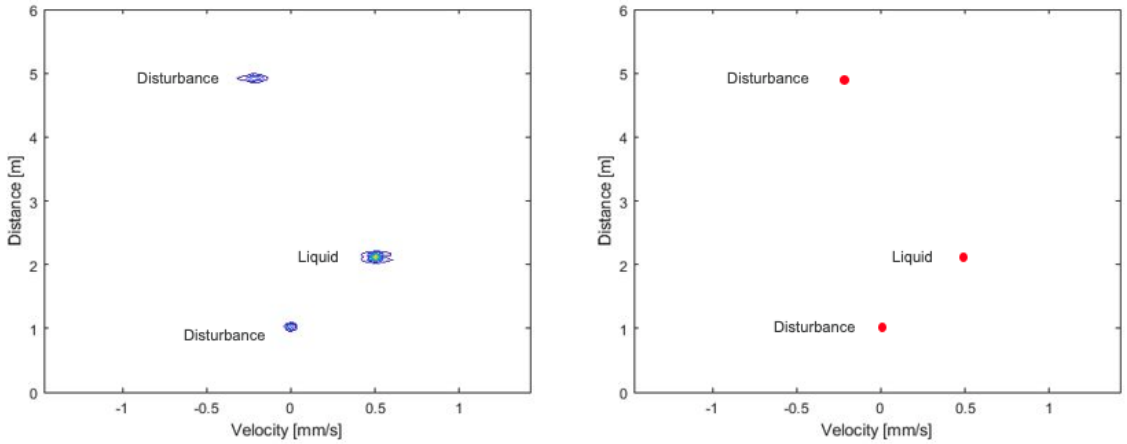
(c) Case 3 when liquid surface approaches the disturbance from above with some velocity $-v$, stays still and continues with velocity v.

Figure 4.1: Examples of the three test cases. liquid surface is marked dark blue and other disturbances as light blue.

As noted in Table 4.1, the Doppler frame is 32, i.e. the number of chirps which the Range-Doppler FFT is performed across (mentioned in section 2.2.3. However, for Case 2 the disturbance was not visible unless the Doppler frame was larger. Therefore, the Doppler frame was set to 128 in this case.

4.4 Preprocessing of Raw Measurements

Preprocessing is done to convert raw signal data into informative variables. The first steps include performing Range-Doppler FFT, resolving amplitude, distance, and velocity estimations for each object. This methodology is described in Section 2.2.3. However, the amplitude values are changed back to the values given from the first FFT to keep the same amplitude relations as described in Subsection 1.1.1 The Range-Doppler map has regions representing various objects, leading to several values in both range and velocity domain. By further applying a peak detection algorithm, the amount of data can be reduced since it gives one distinct value (peak) for each object. The Range-Doppler map before and after peak detection is visualized in Figure 4.2.



(a) Range-Doppler map

(b) Range-Doppler map after peak detection

Figure 4.2: (a) Range-Doppler map at one time step for a tank with two disturbances and a liquid surface, and (b) the same Range-Doppler map but after performing peak detection giving one distinct peak for each object. Each peak has one amplitude, distance, and velocity.

Considering, the Range-Doppler map as an image opens possibilities to use image processing techniques for peak detection. We use the function `imregionalmax` in MATLAB Image Processing Toolbox, [8]. It finds regional maxima which are defined as sets of connected pixels with same intensity surrounded by pixels having less intensity [9]. Applying this function to the Range-Doppler map results in a binary matrix with ones at distinct peaks and zeros elsewhere.

Multiplying this binary matrix with the original Range-Doppler map provides a new reduced map containing peak information: amplitude (A), distance (d) and velocity (v). Since we have multiple peaks at each time, we index the observations with time $t = 1, ..., T$ and peak $i = 1, ..., P(t)$. Where T is the total time of the measurement and $P(t)$ is the max number of peaks at time t. For the test cases and training data the number of objects giving echoes is 3, therefore $P(t)$ is set to be 3

at all times. This is done by only keeping the three peaks with highest amplitude. Figure 4.3 illustrates an example of the peak detection algorithm.

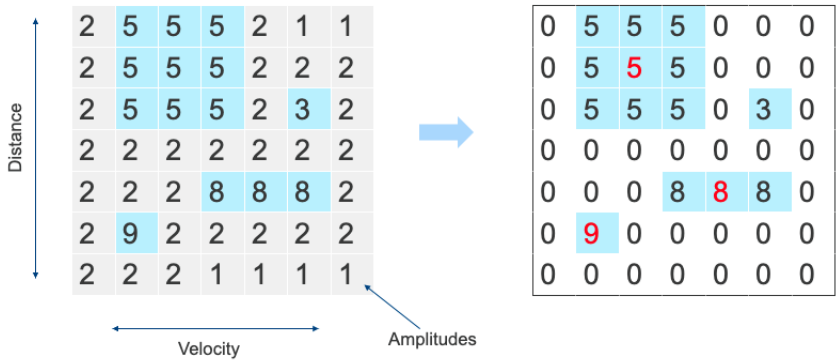


Figure 4.3: Illustration of peak detection algorithm. Regional maxima is colored blue in the left image, all values except these are set to zero in the right image. Next, the the center value of the three peaks with highest amplitudes are marked red. The red values represent the detections amplitudes and distance and velocity are found by the detections coordinates in the matrix.

The detection from one peak is denoted $\mathbf{x_i^t} = (A_i^t, d_i^t, v_i^t)$, and corresponding label is denoted y_i^t. Since the only detection of interest is the liquid surface, we use binary labels dividing each detection to either liquid surface ($y_i^t = 1$) or disturbance ($y_i^t = 0$), see Figure 4.4.

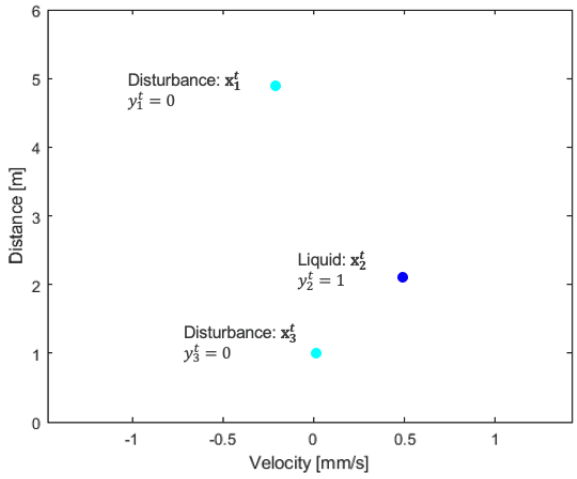


Figure 4.4: Detections at time $= t$, liquid surface ($\mathbf{x}_2^t$) is marked dark blue with label $y_2^t = 1$ and disturbances ($\mathbf{x}_1^t$, $\mathbf{x}_3^t$) light blue with labels $y_1^t = y_3^t = 0$. Each detection consists of one amplitude, one distance, and one velocity.

The values in each detection vary a lot in scale, the amplitude is for example often 10^6 times larger than the distance feature. Therefore, performing scaling is essential

for this problem, making all values equally weighed in. Another reason for scaling data is that it makes gradient descent in neural networks converge faster. The scaling was implemented using maximum absolute scaling, which divides each feature by its absolute maximum. Resulting in detections being in the interval $[0, 1]$.

5

Method I: Solution for Crossing Echoes

In this chapter, a tracking algorithm is used to solve the crossing echoes problem issued in Subsection 1.1.2. Details about the tracking algorithm are presented in sections below.

5.1 Echo Tracking

Liquid surface tracking is the process of predicting the liquid surface level based on its Euclidean distance to previous levels. To achieve this, previous liquid surface levels must be given to the model, starting with the initial levels notated at installation. If there is no information given about the liquid surface level we use the supervised model instead, described in Chapter 6.

The tracking algorithm that Emerson uses today has problems when the liquid surface is close to a disturbance, causing interference regions where the echoes are inseparable in distance and amplitude, as described in Subsection 1.1.2. The algorithm presented here solve this problem of crossing echoes by adding velocity information, which provides a higher dimension to the Euclidean space. As any continuous wave radar, velocity estimation can be obtained from the phase shift, described in Subsection 2.2.2.

5.1.1 Tracking Model using Velocity

The proposed tracking model is distance based where detections $\mathbf{x}_i^t = (A_i^t, d_i^t, v_i^t)$ are compared with the mean value of the latest detected liquid surface detections $\mu^t = (\bar{A}^t, \bar{v}^t, \bar{d}^t)$, we call this target. The values are found by performing the preprocessing steps described in Section 4.4. Comparing with mean value minimize the risk of tracking to an outlier. For each time the detection closest to the target is chosen as liquid surface, see Figure 5.1.

The distances (D_i^t) between each detection and target is stored in vector $\mathbf{D}^t = (D_i^t) = (D_1^t, .., D_{P(t)}^t)$, where $P(t)$ is the number of detections at time t. If an observation has the shortest D_i^t out of $\mathbf{D^t}$ it is classified as liquid surface. Labels $\hat{y}_i^t$ is set

according to

$$\hat{y}_i^t = \begin{cases} 1, & \text{if } D_i^t = \min(\mathbf{D}^t) \\ 0, & \text{if } D_i^t > \min(\mathbf{D}^t) \end{cases} \tag{5.1}$$

Each distance D_i^t is calculated using Euclidean distance metric, as in (5.2). The index $j = 1, 2, 3$ notates the variables in target and detection: amplitude, velocity, or distance. Distance is therefore the total difference in all dimensions and calculated as

$$D_i^t = \sum_{j=1}^{3} \sqrt{(x_{i,j}^t - \mu_j^t)^2}. \tag{5.2}$$

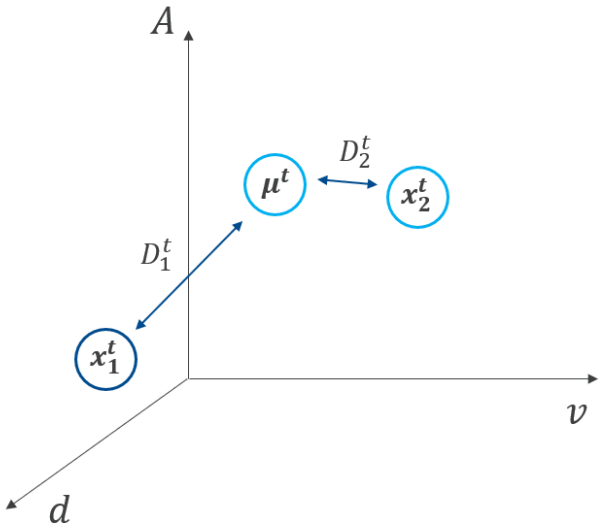


Figure 5.1: Visualization of tracking algorithm. Peak x_2^t will count as liquid surface since it is the closest peak detected at time t: $D_2^t = \min(D_1^t, D_2^t)$

The target μ^t is determined by first saving the time indices (t) and detection indices (i) where $\hat{y}_i^t = 1$ in the set

$$\mathcal{S} = \{(i, t) \mid \hat{y}_i^t = 1\}. \tag{5.3}$$

This set contains all previous liquid surface indices, however, since the echo track varies it is desired to only keep a few indices in memory. We set the number of liquid surface detections in memory to M and write the current $\mathcal{S}$ as

$$\mathcal{S}^t = \{\mathcal{S} \mid M \text{ largest values of } t\}. \tag{5.4}$$

For initial times $1 \leq t < M$, set $\mathcal{S}^{1:M}$ contains the indices where true labels $y_i^{1:M} = 1$. These labels are extracted from the liquid surface level notated at installation. Target μ^t is calculated by taking the mean value of the detections at the indices in $\mathcal{S}^t$,

$$\mu^t = \frac{1}{M} \sum_{(i,t) \in \mathcal{S}^t} \mathbf{x}_i^t. \tag{5.5}$$

Next target μ^{t+1} is then calculated by a new set $\mathcal{S}^{t+1}$, containing the indices found in previous time step. The new distances to detections $\mathbf{x}_i^{t+1}$ are calculated from (5.2)

and classified in (5.1). This will iterate until $t = T$ and the measurement stops.

So far, the model solves the problem when the liquid surface passes a disturbance, where the liquid surface velocity remains after intersection making a similar target. However, problem still occurs when liquid surface and disturbance are at the same level for a longer time, and then starts to move. Here, the algorithm continues the liquid surface track as the disturbance which remain having no velocity, making it closer to the previous liquid surface. Therefore, a condition must be added to the tracking algorithm that can track the liquid surface anyways.

Based on the aspect that if the liquid surface and disturbance are at the same location and one object starts to move, it will always be the liquid surface, due to slower radar propagation rate through the liquid surface mentioned in Section 4.1. Consider two detections at time t: $\mathbf{x}_i^t$ and $\mathbf{x}_{i+1}^t$, their respective velocities v_i^t and v_{i+1}^t, and distances d_i^t and d_{i+1}^t. From the original tracking algorithm, one of those detections is classified as liquid surface. If the two detections are located at the same level $d_i^t = d_{i+1}^t$ and one velocity is larger than the other $v_i^t > v_{i+1}^t$, detection $\mathbf{x}_i^t$ will be tracked to the liquid surface target. By this, the liquid surface track will contain velocities and when the two distances are separated again the algorithm continues as before.

6

Method II: Solution for Loss of Echoes

This chapter presents a method aiming to solve the loss of echoes problem issued in Subsection 1.1.3. Unlike Method I, described in previous Chapter 5, this method classifies only one detection at the time without knowledge of previous positions. This is achieved by implementing a neural network that can learn a general solution for a specific tank. To validate relevant input data, three feature set alternatives are tested, presented in following sections. The first alternative is referred to the "Model Trained on Peak Ratios" and the second as "Model Trained on Nearby Amplitudes". The third method is simply a combination of these two.

Feed-forward neural network (multilayer perceptron) is used with different numbers of layers and neurons in each layer. The main difference between the networks is their input training data rather than its structure. The training is performed on data described in Section 4.2 and tested on test cases presented in Section 4.3.

6.1 Feature Extraction for Supervised Learning

Unlikely to tracking, the detection data ($\mathbf{x}_i^t = (A_i^t, v_i^t, d_i^t)$) does not contain enough information for the supervised model. Because the data from previous times are unknown, the only input from $\mathbf{x}_i^t$ would be one amplitude, distance, and velocity which might as well be a disturbance as the liquid surface. Therefore, we extract features providing more information about each peak and the ratio be different peaks at one time. The two different feature sets are presented in Subsections 6.1.1 and 6.1.2 respectively. These feature sets are then used as input data for the network each, but they are also combined together as a third feature set.

6.1.1 Peak Ratio Features

Peak ratio features are extracted to utilize information existing between detections. The first features are the actual distance and amplitude of the current detection. Velocity is however disregarded since it does not contain any relevant information by itself, and it is undesired learning the network a specific velocity for the liquid surface. Instead, a feature is created to symbolize that the liquid surface almost always is the closest moving detection. This works due to slower propagation speed for radar through the liquid surface, making other disturbances appear to move

only if they are below the liquid surface (described in Section 4.1). This information is added to a feature called "closest moving peak" (CMP), which can have three values:

$$CMP_i^t = \begin{cases} 1, & \text{if } \mathbf{x}_i^t \text{ is closest with velocity} \\ 0, & \text{if } \mathbf{x}_i^t \text{ is not closest with velocity} \\ 2, & \text{if none of the detections at } t \text{ has velocity} \end{cases} \tag{6.1}$$

Next features are created from the ratios between the detection's own amplitude and distance, and the other detections at the same time. Dividing the current peak amplitude A_i^t with the other amplitudes, and dividing current distance d_i^t with the other distances, at time t separately. For this feature to work the number detections at one time has to be constant otherwise the feature vector will have different length for different cases. However, in this thesis the number of detections $P(t)$ found by the peak finder algorithm is always set to 3, as mentioned in Section 4.4. Giving amplitude ratio features: $AR1_i^t$, $AR2_i^t$, and distance ratio features: $dR1_i^t$, $dR2_i^t$. Resulting feature set is: $X_i^t = \{A_i^t, d_i^t, CMP_i^t, AR1_i^t, AR2_i^t, dR1_i^t, dR2_i^t\}$.

6.1.2 Nearby Amplitude Features

These features are inspired by object detections in image classification, but we do not have an image and that the resolution of our input is a lot smaller than an image. Therefore, no convolution layer is needed in this task. The extracted features are directly inputted to a fully connected feed-forward network. The training set in this method is obtained by creating a "bounding box" around each detection. After applying Range-Doppler FFT and peak detection, we receive the central location of each peak representing an object. The bounding box is then created by the nearest 3×3 grid around the peak center, as illustrated in Figure 6.1, where p_i^t denotes the center point of that peak. The nearby values are then aligned into a 1×9 vector.

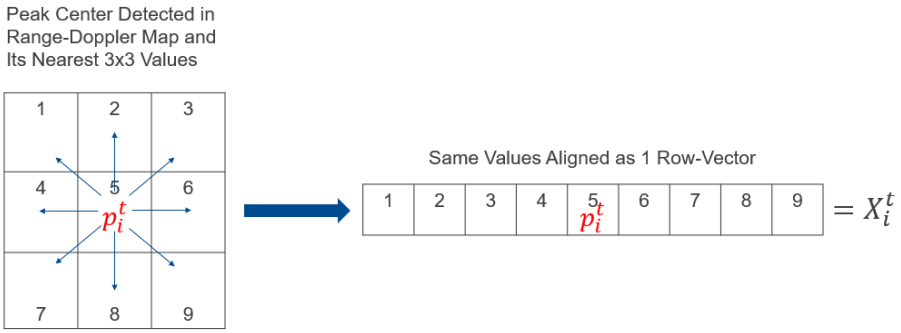


Figure 6.1: Graphical illustration of extracting Nearby Amplitude Features. The peak center (p_i^t) is aligned with its nearest 3x3 amplitude values in the Range-Doppler map forming feature set X_i^t.

This method assumes that there might be some micro-movements around the liquid surface that affects its amplitude and that these movements are specific only for the liquid surface. To model this phenomenon, small normally distributed noise

was added around the liquid surface simulation. The network is then trained on liquid surface simulation with- and without added noise to determine whether micro-movements improve liquid surface identification.

6.2 Network

The network is build using MATLAB Deep Learning Toolbox and the function `feedforwardnet()` in the category "Function Approximation and Nonlinear Regression", [10]. The network returns a feedforward neural network and has two adjustable parameters: `hiddenSizes` and `trainFcn`. The first parameter specifies the sizes of hidden layers as a row vector and the second parameter defines the optimization algorithm. For this implementation Levenberg-Marquardt [11] is used as the optimization algorithm, which is also the default function in `feedforwardnet()`.

Because different features are tested with different feature dimensions the input layer must match those dimensions. To observe the performance of various inputs three different nets are created with following features:

- Net1: Nearby Amplitudes of Liquid Surface without Noise
- Net1*: Nearby Amplitudes of Liquid Surface with Noise
- Net2: Peak Ratios of Liquid Surface without Noise
- Net3: Peak Ratios and Nearby Amplitudes of Liquid Surface without Noise

The hidden layers for each net were found by manual iterations choosing the structure with best results. The parameter `hiddenSizes` for the nets are set as following: Net1 & Net1*: $[8, 15, 8, 12]$, Net2: $[7, 7, 8]$, Net3: $[20, 11, 6]$. The output contains one node giving a binary classification of one feature vector. Each network was then trained for 1000 epochs each.

7
Result Evaluation

This chapter presents the result found by both Method I and Method II, introduced in Chapter 5 and 6 respectively. Recall, that Method I uses a tracking algorithm to solve the crossing echoes problem introduced in Subsection 1.1.2 by utilizing information about previous liquid surface levels and velocity information. Moreover, Method II uses supervised learning to solve the loss of echoes problem introduced in Subsection 1.1.3.

All results will be visually presented for the test cases given in Section 4.3 and evaluated using the evaluation metric presented in Section 7.1. The evaluation metric is used to give a more precise comparison between the cases and methods, since all results visually show that the methods classify the liquid surface correctly.

7.1 Evaluation Metric

For performance evaluation, we have created our own metric relevant for the test cases. This is because the data this project is dealing with is heavily imbalanced between the "liquid surface class" and "not liquid surface class", since there is only one liquid surface but can be multiple disturbances. However, there are also a lot of evaluation metrics that are applicable for imbalanced data, but in this case, there are more issues than the class imbalance. The interference of two signals crossing each other creates a "cloud" of detections, as shown in Figure 1.3, Section 1.1. In this area, it is very hard to compare the predicted class with the true class and it will not give a representative evaluation while using traditional classification metrics such as confusion matrix, accuracy, F1 score and so on.

This success rate metric uses the distance and class labels only since there is just one true distance for the liquid surface at one time t. The reason only distance is considered in the evaluation metric is because the "signal cloud" in the amplitude domain is the main reason regular evaluation metric does not work in our case. In this cloud, every point has the same value in the distance domain but has different amplitudes. The true liquid instances and disturbance instances are all mixed and randomly distributed within this cloud area, causing difficulty for evaluation.

Therefore, we consider the algorithm to have succeeded at time t, if it manages to classify the liquid surface instance correctly only with respect to its distance, and only considers one distance to be the true liquid surface distance at one time. For

a binary task with predicted label we have:

$$\hat{y}_i^t = \begin{cases} 1 & \text{, if liquid surface} \\ 0 & \text{, other} \end{cases} \tag{7.1}$$

for detected peaks $i = 1, 2, ..., P(t)$ at time t. Evaluation of whether the algorithm has classified the detection with correct class label with respect to the true distance of the liquid surface is stored in a score metric ϕ_i^t, described as

$$\phi_i^t = \begin{cases} 1, & \text{if } \hat{y}_i^t = 1 \text{ and } d_i^t = d_{liquidsurface}^t \\ 1, & \text{if } \hat{y}_i^t = 0 \text{ and } d_i^t \neq d_{liquidsurface}^t \\ 0, & \text{if } \hat{y}_i^t = 1 \text{ and } d_i^t \neq d_{liquidsurface}^t \\ 0, & \text{if } \hat{y}_i^t = 0 \text{ and } d_i^t = d_{liquidsurface}^t \end{cases} \tag{7.2}$$

where d_i^t is the distance to current detection and $d_{liquidsurface}^t$ is the liquid surface level at the same time. The score metric ϕ_i^t will obtain values $[0, P(t)]$. If the algorithm has identified the true label *only* for the true position, it receives a score of positive 1. Else, it will receive score 0. If the algorithm manages to correctly classify every peak i at time t, the sum of ϕ_i^t is equal to $P(t)$. This in turn leads to a further score metric:

$$score(t) = \begin{cases} 1, & \text{if } \sum_{i=1}^{P(t)} \phi_i^t = P(t) \\ 0, & \text{if } \sum_{i=1}^{P(t)} \phi_i^t < P(t). \end{cases} \tag{7.3}$$

This score can obtain values $[0, 1]$. This simply implies that, for each time t, the algorithm can only obtain score 0 for failing or 1 for success. The total $score(t)$ is then divided by the total measurement time for that specific test case, and we receive the final success rate of that test case according to

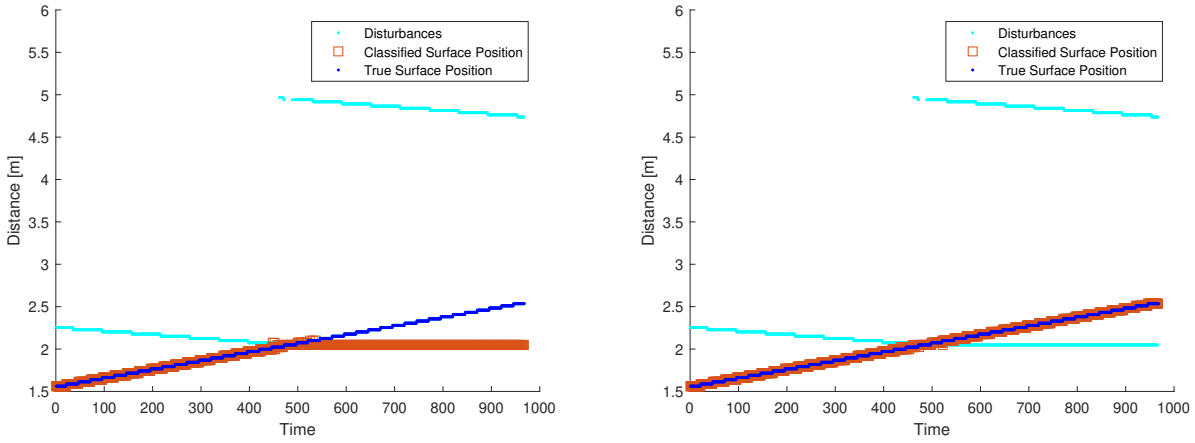
$$Success\ Rate = \frac{\sum_{t=1}^{T} score(t)}{T}. \tag{7.4}$$

The success rate can have values between 0 and 1, where 1 indicates that the liquid surface has been predicted correct for all times.

7.2 Performance: Method I

This section presents the results from the tracking algorithm (Method I) described in Chapter 5, aimed to solve the crossing echoes problem presented in Subsection 1.1.2. To verify the improvement of adding the velocity information, a model without velocity information is presented in this section as well. Worth mentioning is that the model without velocity is not the one used by Emerson today, it is simply our presented tracking method but with only two dimensions: amplitude and distance. In short, the model with velocity performed better than the model without velocity in all test cases with a mean success rate of 97% instead of 51%, calculated by the evaluation metric in Section 7.1. Further, one example of each test case is visually presented, with and without velocity.

Starting with Case 1a when the liquid surface passes a disturbance once by draining liquid. The resulting echo tracks are presented in Figure 7.1. The model with velocity tracks the liquid surface correctly (Figure 7.1b), while the model without velocity (Figure 7.1a) chooses the disturbance as liquid surface after intersection.

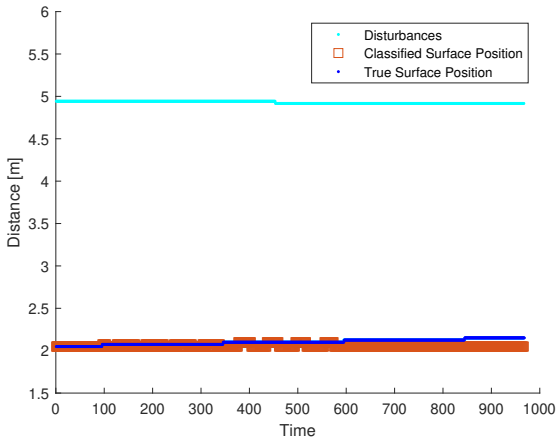


(a) Tracking without velocity information (b) Tracking with velocity information

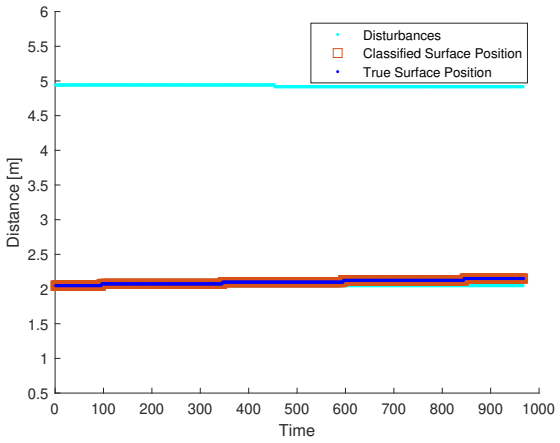
Figure 7.1: Results for Case 1a with and without velocity information, (b) contain velocity information and therefore it picks the right track after intersection.

For Case 2 the model with velocity also performs better than the model without velocity, see Figure 7.2 showing the results for tracking Case 2a. Note that the liquid surface is the echo with greater distance at the last time step.

When it comes to Case 3 (Figure 7.3), the model without velocity (Figure 7.3a) works equally good as the model with velocity (Figure 7.3b) in the first part of the measurements, however problems occur when the liquid surface starts to move again after the intersection part.

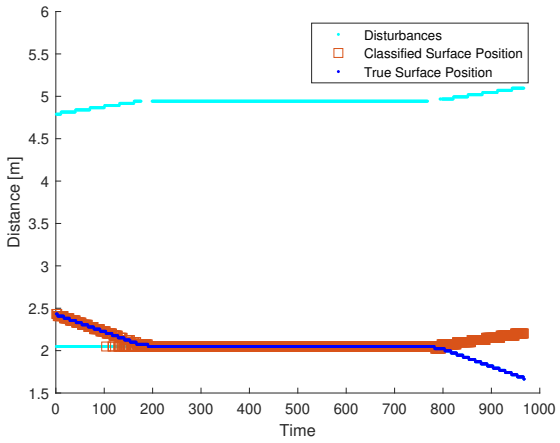


(a) Tracking without velocity information

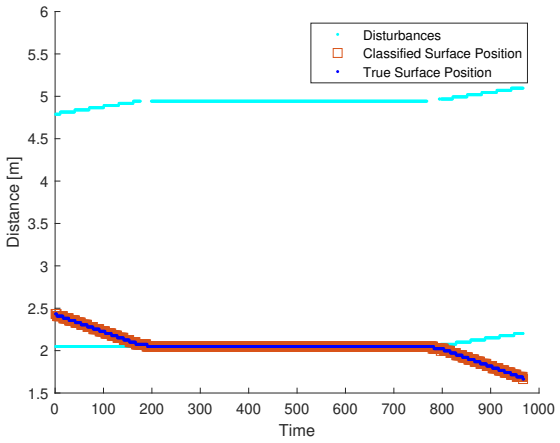


(b) Tracking with velocity information

Figure 7.2: Results for Case 2a with and without velocity information, (b) contain velocity information and therefore it picks the right track.



(a) Tracking without velocity information



(b) Tracking with velocity information

Figure 7.3: Results for case 3b with and without velocity, (b) contain velocity information and therefore it picks the right track after intersection.

For all test cases the liquid surface were classified correctly after adding velocity information. However, these test cases are simulated with similar amplitude values for disturbance and liquid surface. If the amplitude values would differ more the model without velocity information would perform better because the distance and amplitude information would have higher impact, especially for Case 2 where the velocity of the liquid surface is very small.

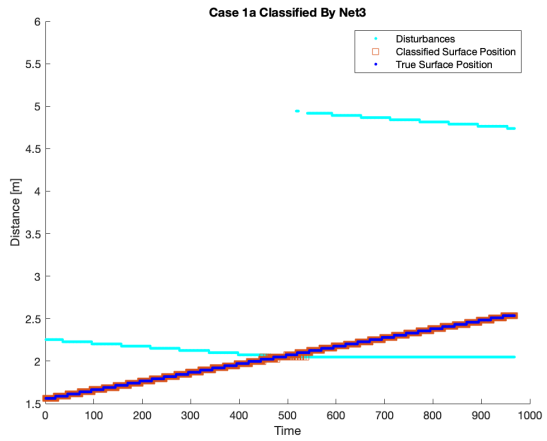
7.3 Performance: Method II

This section presents the results of Method II described in Chapter 6 where new detections is classified using neural networks. Unlike Method I (results in previous Section 7.2) new detections are classified without knowledge of previous times, solving the loss of detection problem introduced in Subsection 1.1.3.

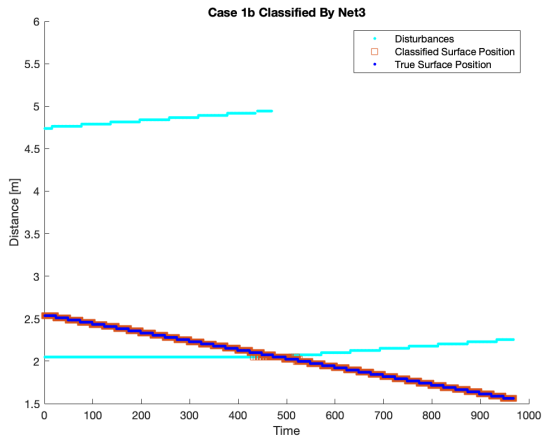
Net3 is by far the best performing network, therefore following subsection presents visualizations of classified liquid surface level by Net3. Classifications made by Net1, Net1*, Net2 are visualized in Appendix B. Information about the different networks is given in Chapter 6 and the test cases are presented in Section 4.3. A summarizing numeric evaluation for all networks are given in Section 7.3.2. Net3 is the network trained on both nearby amplitudes and peak ratios and we can therefore conclude that both Range-Doppler FFT and the information about other reflecting objects is of importance to the model. Since, the other networks, trained on the two different feature sets alone, performed worse than the combined.

7.3.1 Visual Performance of Net3

The classifications made by Net3 for Case 1 is presented in Figure 7.4. The network finds the liquid correctly for both the draining process (Figure 7.4a) and the filling process (Figure 7.4b).



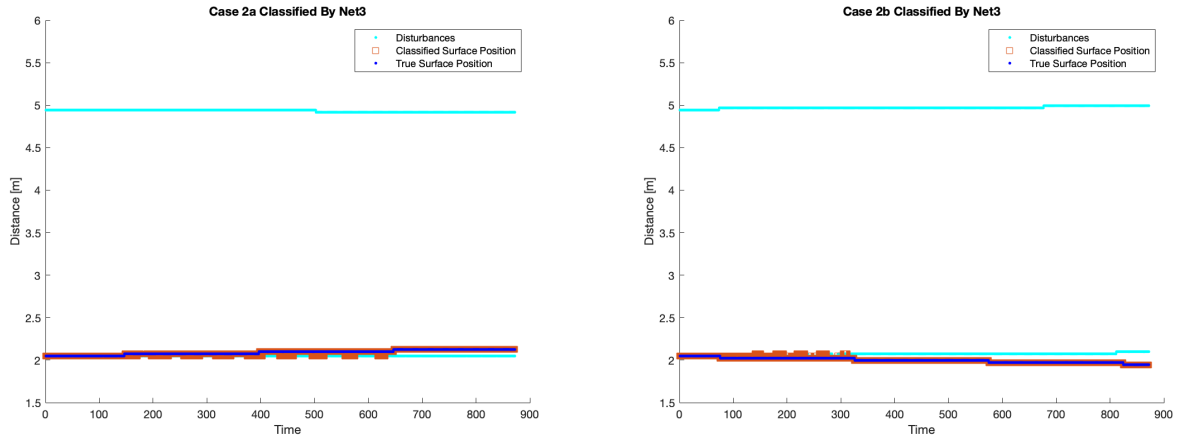
(a) Case 1a, classified by Net3.



(b) Case 1b, classified by Net3.

Figure 7.4: Results using Net3 for Case 1a and Case 1b. The model performed well in both cases.

For Case 2 (Figure 7.5) the liquid surface level is correctly classified in the end part of the measurement where the surface is more separated from the disturbance. In the beginning of the measurement the network classifies the disturbance as liquid surface too since the two positions are very close.

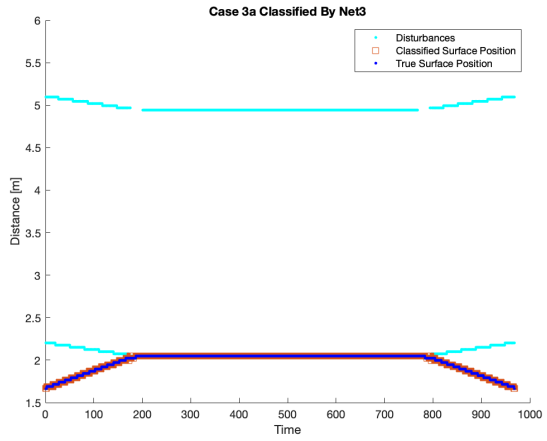


(a) Case 2a, classified by Net3.

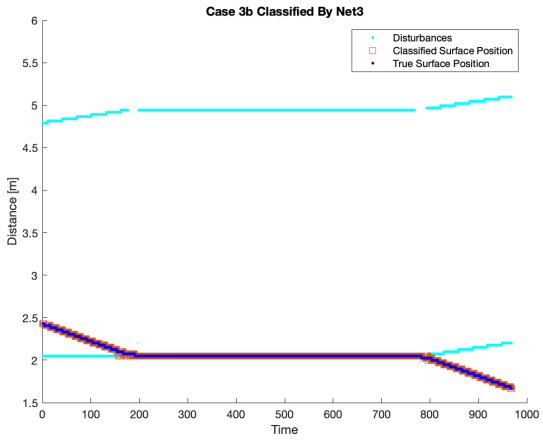
(b) Case 2b, classified by Net3.

Figure 7.5: Results using Net3 for Case 2a and Case 2b. The network classifies the liquid surface well in the end of the measurement but performs worse in the beginning where disturbance and liquid surface levels are close.

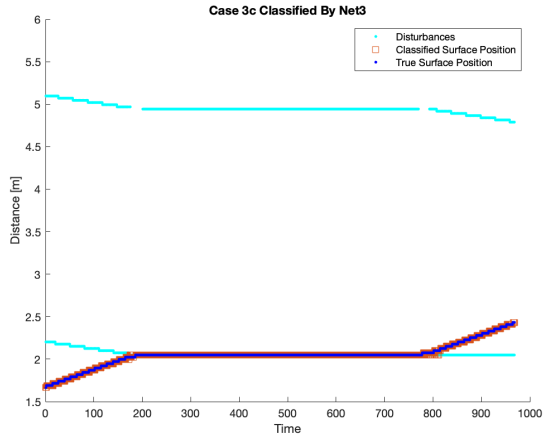
The classifications for Case 3 are visualized in Figure 7.6. The network classifies the liquid surface correctly for all sub-cases.



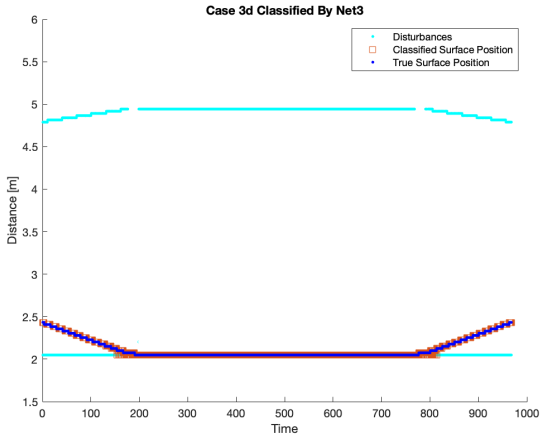
(a) Case 3a, classified by Net3.



(b) Case 3b, classified by Net3.



(c) Case 3c, classified by Net3.



(d) Case 3d, classified by Net3.

Figure 7.6: Classification results using Net3 for Case 3a, 3b, 3c, and 3d. The network classifies the liquid surface correctly for all sub-cases.

7.3.2 Numeric Evaluation

The numeric evaluation is based on success rates calculated as described in Section 7.1. As seen in the figures in previous section, Net3 appears to classify all liquid surface levels correctly. However, the mean success rate per case is calculated to 0.875. This is because the success rate considers miss-classifications near the liquid surface the same as those far away. Therefore although not 100% success rate, the maximum miss-classification made is only 5.12cm away from the true liquid surface level. The same maximum error goes for all cases classified by Net3. Recall, the range resolution for the simulated gauge is 2.56cm (from Table 4.1) and consequently 5.12cm is just the next level that could be detected. The success rates presented in this section is therefore just used to briefly compare the networks and the cases. For a visual comparison see Appendix B.

Table 7.1 show the mean success rate for each network. Net1 has slightly better performance than Net1* indicating that adding normally distributed noise around the liquid surface did not have any sufficient impact for these networks. Moreover, both Net1 and Net1* performed better than Net2 which means that the networks trained on nearby amplitudes had better results than the network trained on peak ratios. However, the best performing network was Net3 indicating that both feature sets is needed to maximize the success rate.

Table 7.1: Mean success rates for each network performed on all test cases, calculated by the evaluation metric from Section 7.1

Network	**Net1**	**Net1***	**Net2**	**Net3**
Mean	0.794	0.733	0.249	0.875

Mean success rate obtained on each case type performed by Net3 is shown in Table 7.2. Based on this the most difficult case is Case 2, where the liquid surface is very close to the disturbance. For full evaluation obtained by each network and case, see Appendix C.

Table 7.2: Mean success rates for each case type performed by Net3, calculated by the evaluation metric from Section 7.1

Case	**Case1**	**Case2**	**Case3**
Net3	0.927	0.730	0.921

8

Discussion

The results for Method I show that adding velocity information clearly improved the performance in the test cases, solving the crossing echoes problem issued in Subsection 1.1.2. Therefore, we suggest adding velocity information to the gauges, giving more precise results when the liquid surface interacts with another disturbance. A potential problem with the tracking algorithm is the jump discontinuities in amplitude domain, happening when the liquid passes a disturbance. If the jump is too large the target could become too far away making it choose the disturbance as the liquid surface. Here, some sort of condition could be implemented making the tracking algorithm recognize interference and jump discontinuities and only track according to the other two features: distance and velocity in these cases. Since this thesis only consider calm liquid conditions in the test cases the tracking algorithm works for them all. For other circumstances, such as when the liquid echo disappears for a while, the tracking will not work as it is executed now. Therefore this thesis also investigated a supervised learning approach (Method II) which works when tracking is insufficient.

Method II solves the loss of echoes problem issued in Section 1.1.3, as can be seen in the figures in Subsection 7.3.1, where the liquid surface level is correctly classified in all test cases. By comparing each network trained on different feature sets (Section 6.1), one can see the nearby amplitude features are of high importance compared to peak ratios. This is because network trained on nearby amplitudes alone (Net1) has shown a mean success rate of 0.794, while network trained on peak ratio features alone (Net2) has shown very little success, with a mean success rate of 0.249, but adds further helpful information above the nearby amplitude features. This is shown in the network trained on both these feature sets combined (Net3). Net3 obtained the best result of all networks with a mean success rate of 0.875. This is not surprising since higher feature dimensions always improves the network performance, but with longer training time in return. Although, the success rate do not reach 100%, Net3 can be considered successful since the farthest miss-classification is only 5.12cm from the true liquid surface level.

However, Net3 does not perform better than the tracking algorithm with added velocity (Mean success rate 0.973 vs. 0.875). It is though important to keep in mind that the network performance is carried out without any knowledge about previous detections. What is more, the networks used in this thesis are rather simple, as well as the simulation used for training the network. Therefore, there is a lot of room for improvements. Further improvements for the supervised learning model is discussed

in Section 8.1.

In this thesis, the test cases are simulated using a slightly different liquid property compared to the training data (read more about this in Section 4.2 and 4.3). In reality, liquids used in industrial tanks can be very different hence reflecting different amount of energy back to the radar. The result is therefore not sufficient enough to ensure that, training set based on one type of liquid can be used for any type of liquid. To increase the variety of training data, one could easily change the parameters in the simulation tool. However, this will rather be unpractical in real circumstances since it would require one to fill the tank with a new liquid. Considering the volume of industrial tanks, it is highly unnecessary to waste that amount of substances, which would also require permits from environmental authorities just to provide training data for the model. As there already lacks documented tank data from real measurements today, to provide enough training data from real measurements for a supervised model to be more general seems like a far goal.

Additionally, little has been done in model selection and parameter tuning for Method II (supervised learning), leading to somehow poor performances. The focus has been extracting and comparing different features extracted for the supervised model. By adding noise to the liquid surface has shown no effect on improving model performance compared to combine peak ratios and nearby amplitudes. However, we only use information from the first adjacent amplitudes, namely a 3×3 grid around the peak center for all cases. There can be potential in expanding the "bounding box" around the peak center. This is something that needs to be further investigated with respect to the Doppler frame size. 32 has been used as the Doppler frame size in this thesis, leading to 1000×32 Range-Doppler FFT matrices. However, if the "bounding box" is too big with respect to the Doppler frame size, it will simply imply large overlaps between peaks. Comparing bounding boxes in object detection problems, if one wants to train an algorithm to recognize dogs in a bounding box, the training certainly will be unsuccessful if every bounding box also contain one half cat. Increased Doppler frame size simply imply higher velocity resolution, but it also decreases the number of samples that could be extracted from a measurement. Conversely, if the "bounding box" is too small with respect to the Doppler frame size, it might also contain too little information about the surrounded object. This could be the reason why Case 2 is the one with poorest performance for all networks. In Case 2, the liquid surface moves very closely to the disturbance, which required a Doppler frame size of 128, compared to 32 in Case 1 and Case 3. When 3×3 grid around each peak center has been applied to all test cases, it might have been too small for a Doppler frame size 128, leading to the poor performance of Case 2, shown in Table 7.2. Therefore, while using the "bounding box" method, or "nearby amplitudes" on Range-Doppler matrices, one should always consider a balance between the "bounding box" size and the Doppler frame size and how many samples one wants to extract from a measured series. However, this is just a hypothetical reasoning and has not been verified.

8.1 Future Work

The supervised learning method (Method II, introduced in Chapter 6) primarily focused on the input data and signal processing to create useful conditions for machine learning. In future work, the network must be optimized to find the best architecture for this problem, creating a more robust model. The network structure `feedforwardnet()` from MATLAB is to some extent limited since it uses standard settings with only two adjustable parameters. As a result, there are great opportunities to change this method further. When implementing a new network structure some sort of hyperparameter tuning would also be essential for creating optimal results. In all tests, a binary classification of the liquid is provided giving no indication on if the model is uncertain in any decisions. An alternative output could be to implement a continuous activation function in the output layer, for example, the softmax function. By this, uncertain classifications can be removed and for instance be replaced by the last detected position times the velocity.

However, regardless of network optimization, the training data must include all relevant information for the network to learn difficult cases. In this thesis the training data comes from the same tank as the test data. Providing results for the concept where the customer must have the gauge installed for a while before it is fully trained. It would be interesting to also try a concept of a model where training data can come from many tanks providing a pre-trained model that can be installed in any tank. This would also imply that the customers do not have to fill up liquid from bottom to top as a calibration step, nor keep the same type of liquid.

When it comes to features, one could also add higher feature space by aligning features collected during several chirps. Since each chirp is carried out extremely fast (micro- to milliseconds), one could simply train- and classify several chirps at once without affecting the user experience negatively. What is more, the feature set containing nearby amplitudes only contain amplitude information obtained from the first FFT and not the second one. Besides adjusting grid-size with respect to the Doppler frame size, one could also utilize the same concept of nearby amplitudes but on amplitudes obtained from the second FFT. In this way, we receive "two layers" of amplitude information instead of only one that was used in this thesis. (Recall RGB layers in images, where 3 layers are combined and fed into a CNN.) However, higher feature space will cause longer training time for the model and sometimes causing the model being unable to converge. By further investigating beneficial features for this problem, analysis about feature importance should be carried out, followed by regularizations in order to minimize unnecessary features, or by applying convolution layers.

Perhaps the most important future work is to train the models on measured data from real tanks instead of simulated data. For this simulator, there is quite a lot differences between real measured signals compared to simulated, consequently the network will not work on measured data. Additionally, because of the limitations in this thesis, we only simulate calm liquid conditions, resulting in much simpler data

since this entails that the liquid surface always reflects the radar. Improving the simulator to make it produce turbulent and foamy liquid surfaces could contribute to generating more realistic simulations. If the simulator is not improved, more measured data must be collected to train and evaluate new models. However, if measured data is used it must be labeled before a network can learn anything from it. This would require a lot of manual work unless the data can be labeled directly by a tracking model.

If the final goal is to have pre-trained gauges, a suggestion could be to implement a transfer learning algorithm. This technique has potential to pre-train on a large amount of simulated data and then use it as a starting point for a new model trained on less measured data. Which could minimize the effort of manually labeling much measured data.

9
Conclusion

One main conclusion of this thesis is the importance of Range-Doppler FFT for both the tracking algorithm and for supervised learning features. This importance has specifically shown in the supervised learning task. According to results presented in Subsection 7.3.2 training on peak ratios alone performed worse than combining them with nearby amplitude features. It is therefore suggested to use Range-Doppler FFT in future gauges, together with information about the relations between different reflecting objects in the tank.

On the network tested for nearby amplitude features we tried imitating micro movements on the liquid by adding normally distributed noise around the liquid simulation. This did not give any performance improvement, however it could be interesting to use a larger grid of amplitudes to see if that make a difference. The best performance was achieved with Net3, combining nearby amplitude and peak ratio features, with mean success rate 87.5%, where the miss-classification farthest away from the true liquid surface was 5.12cm.

In future work one must look more into data from real measurements. Obviously for a final implementation but also just for improving the simulator, making it create more realistic training data. The networks tested in this thesis were also just for demonstrative purpose and to increase the performance a more robust network would have to be made. Finally, a model trained on multiple tank scenarios would be interesting since the proposed training data in this thesis only use one tank for training and testing.

Bibliography

[1] Brillouin, L. (1960). *Wave Propagation and Group Velocity* (Vol.8). Academic Press.

[2] Britannica, T. Editors of Encyclopaedia (2021, May 11). *refraction. Encyclopedia Britannica.* Retrieved June 3, 2022 from `https://www.britannica.com/science/refraction`

[3] Global Headquarters, Emerson Automation Solutions (2017) *Frequency Modulated Continuous Wave (FMCW) technology improves radar level measurement accuracy and reliability in challenging applications* [White paper]. Retrieved Mars 7, 2022 from `https://www.emerson.com/documents/automation/white-paper-frequency-modulated-continuous-wave-technology-rosemount-en-1265738.pdf`

[4] Iovescu, C. & Rao, S. (2021) *The fundamentals of millimeter wave radar sensors.* Texas Instruments.

[5] Mehlig, B. (2021). *Machine Learning with Neural Networks.* Cambridge University Press.

[6] Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*

[7] Kubat, M. (2017).*An Introduction to Machine Learning.* Springer Publishing.

[8] MATLAB, Image Processing Toolbox R2022a, Retrieved Mars 14, 2022 from `https://se.mathworks.com/help/images/index.html?s_tid=CRUX_lftnav`

[9] Vincent, L. (1993). Morphological grayscale reconstruction in image analysis: applications and efficient algorithms. *IEEE transactions on image processing.*

[10] MATLAB, Deep Learning Toolbox R2022a, feedforwardnet, Retrieved May 19, 2022 from `https://se.mathworks.com/help/deeplearning/ref/feedforwardnet.html`

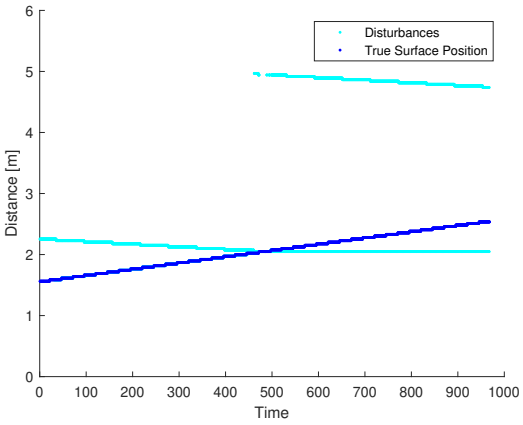
[11] Yu, H., & Wilamowski, B. M. (2011). Levenberg-Marquardt training. *Industrial electronics handbook, 5*(12),1

[12] Elert, G. (2021). *Intensity.* The Physics Hypertextbook. Retrieved June 3, 2022 from `https://physics.info/intensity/`

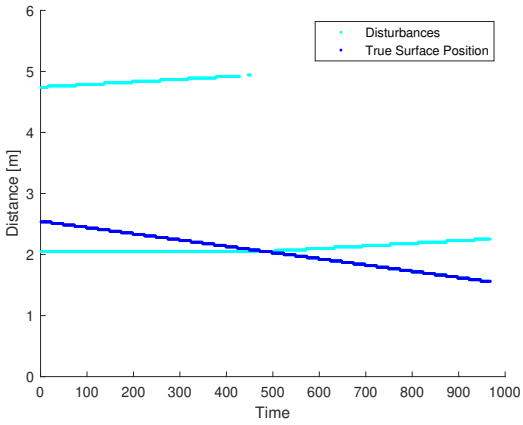
A

Appendix

The different scenarios of each test case is presented in Figure A.1, A.2, A.3.

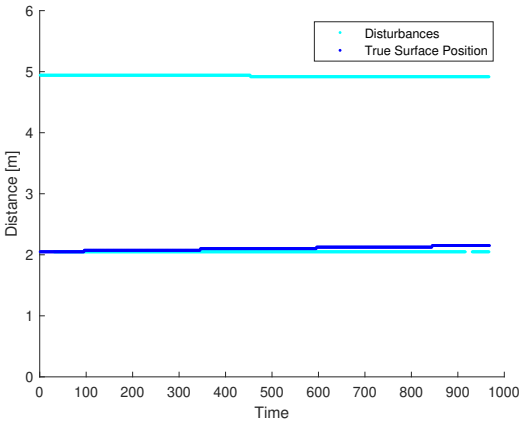


(a) Case 1a

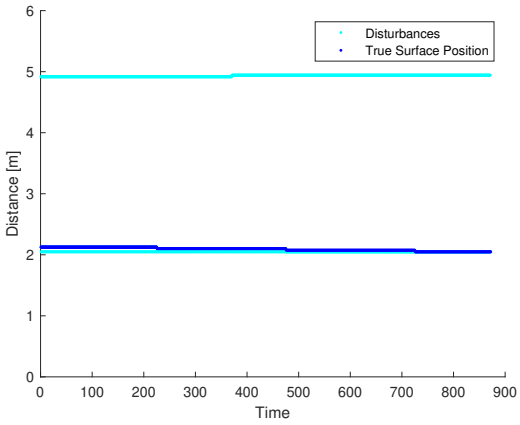


(b) Case 1b

Figure A.1: Case 1

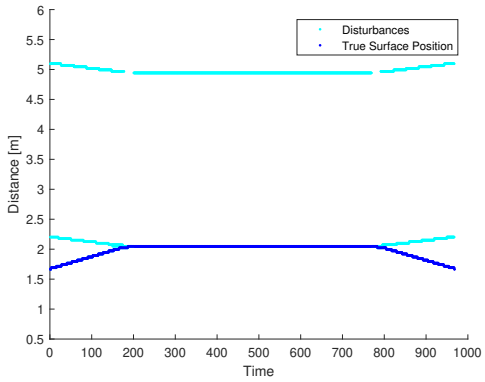


(a) Case 2a

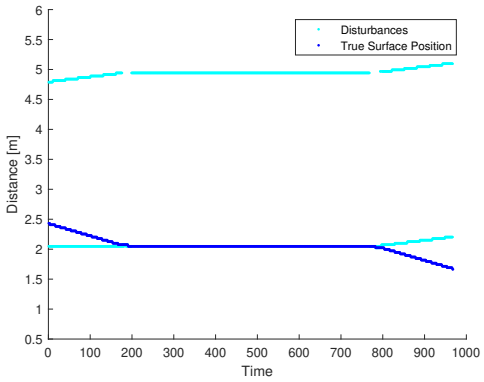


(b) Case 2b

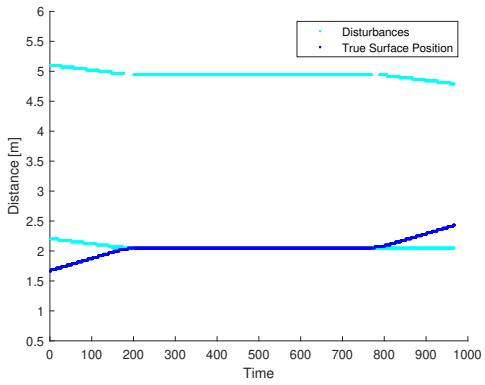
Figure A.2: Case 2



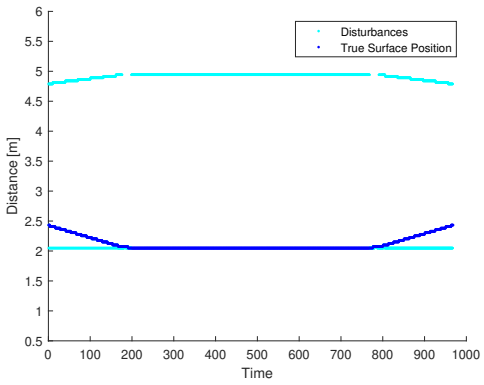
(a)



(b)



(c)



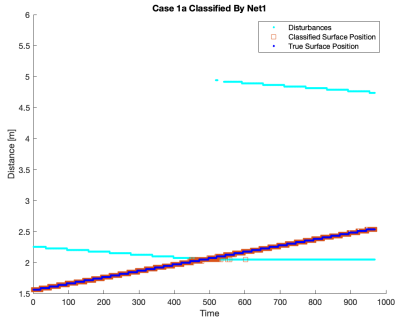
(d)

Figure A.3: Case 3

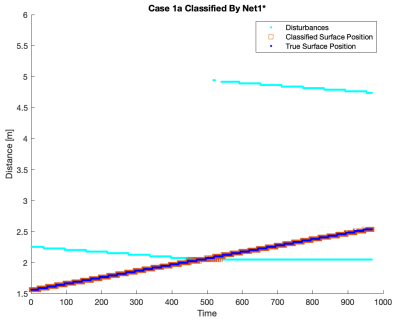
B

Appendix

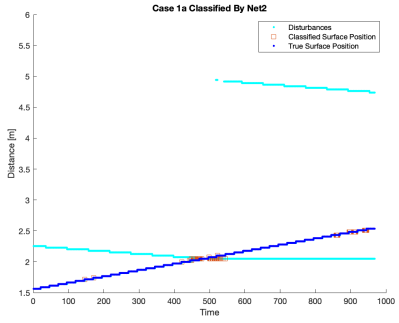
Results from the supervised models for all test cases. Subfigure (a) is results from Net1 using data without extra surface noise, and subfigure (b) is with extra surface noise (Net1*). The other two subfigures give the results for Net2 (c) and Net3 (d).



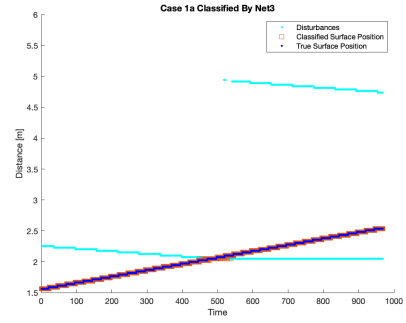
(a) Case 1a classified by Net1.



(b) Case 1a classified by Net1*.

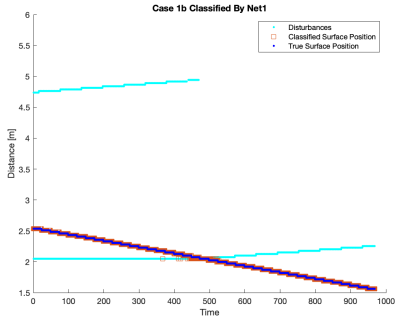


(c) Case 1a classified by Net2.

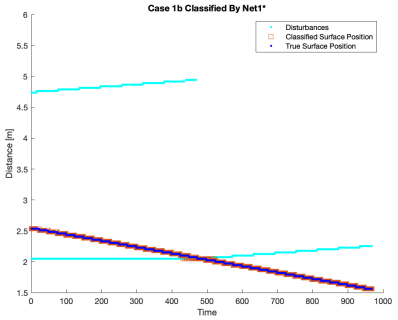


(d) Case 1a classified by Net3.

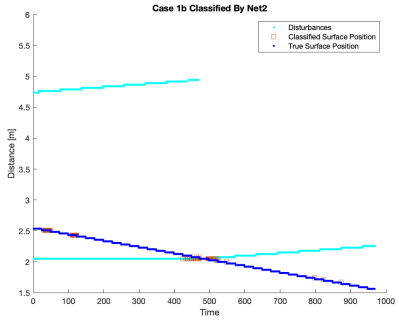
Figure B.1: Results for Case 1a using supervised learning



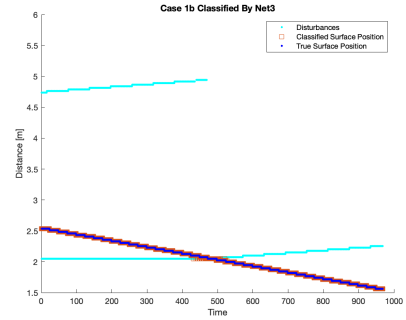
(a) Case 1b classified by Net1.



(b) Case 1b classified by Net1*.

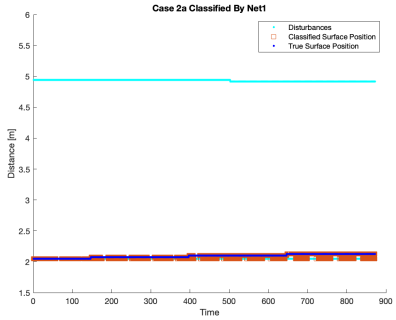


(c) Case 1b classified by Net2.

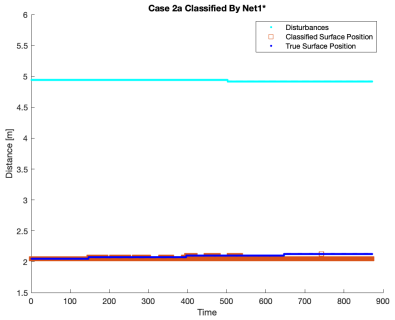


(d) Case 1b classified by Net3.

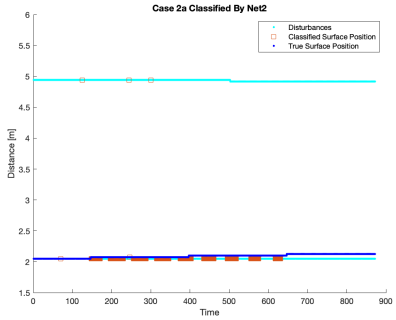
Figure B.2: Results for Case 1b using supervised learning



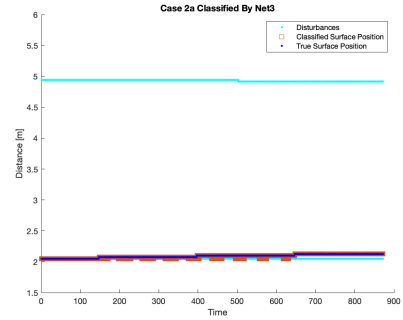
(a) Case 2a classified by Net1.



(b) Case 2a classified by Net1*.

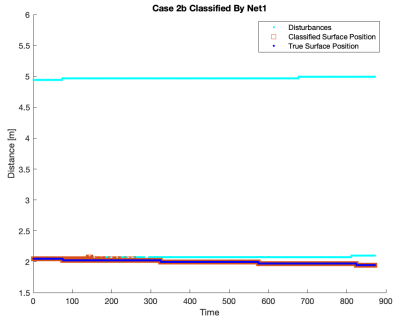


(c) Case 2a classified by Net2.

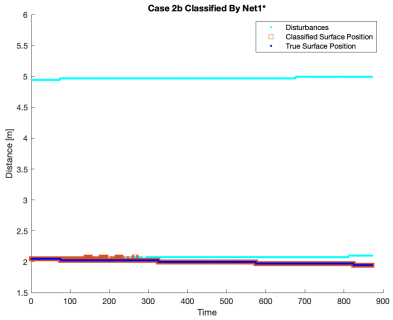


(d) Case 2a classified by Net3.

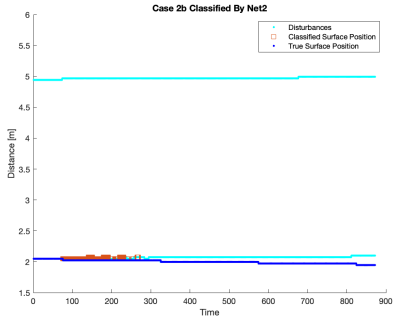
Figure B.3: Results for Case 2a using supervised learning



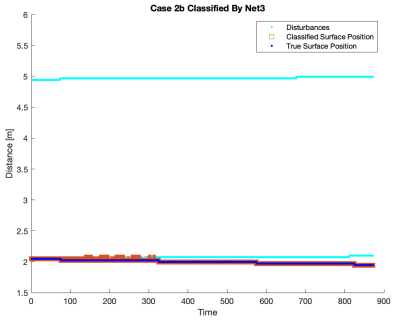
(a) Case 2b classified by Net1.



(b) Case 2b classified by Net1*.

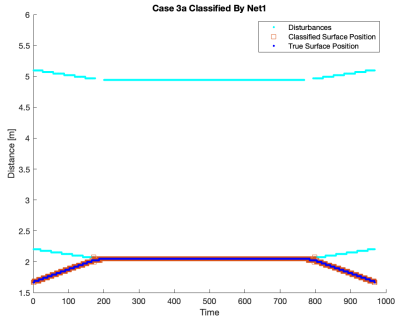


(c) Case 2b classified by Net2.

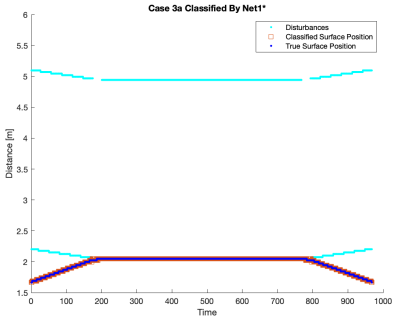


(d) Case 2b classified by Net3.

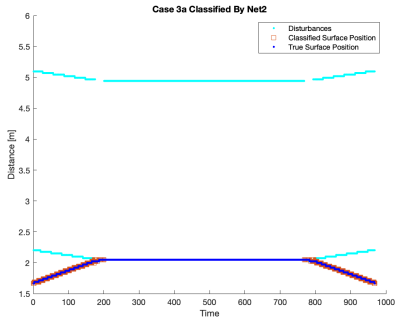
Figure B.4: Results for Case 2b using supervised learning



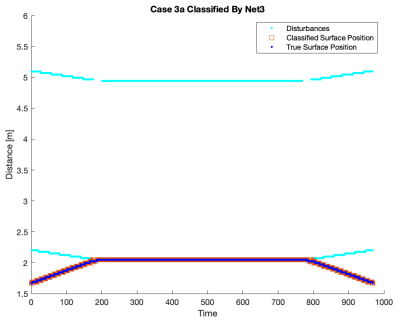
(a) Case 3a classified by Net1.



(b) Case 3a classified by Net1*.

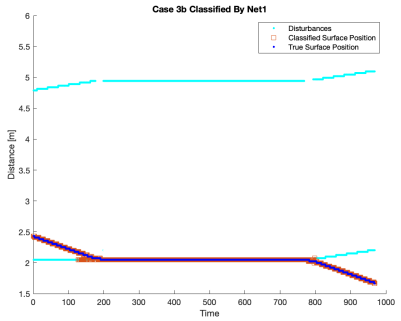


(c) Case 3a classified by Net2.

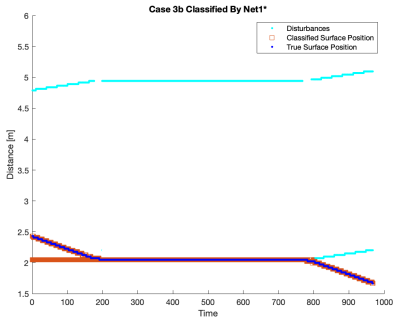


(d) Case 3a classified by Net3.

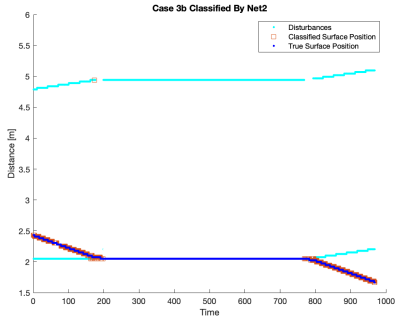
Figure B.5: Results for Case 3a using supervised learning



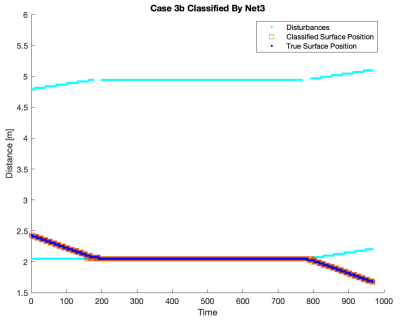
(a) Case 3b classified by Net1.



(b) Case 3b classified by Net1*.

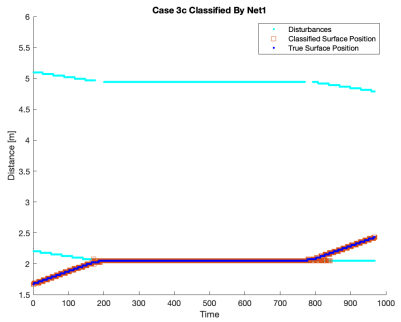


(c) Case 3b classified by Net2.

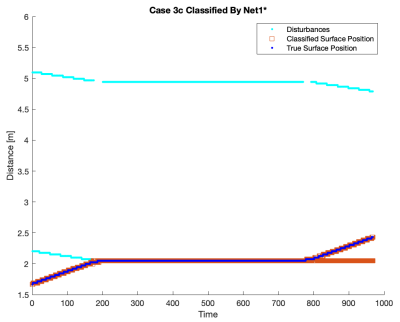


(d) Case 3b classified by Net3.

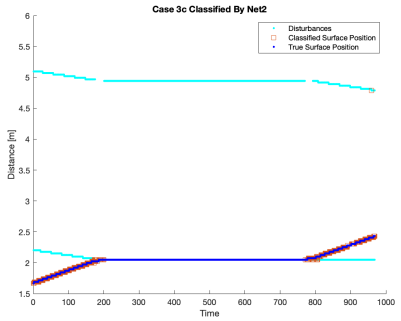
Figure B.6: Results for Case 3b using supervised learning



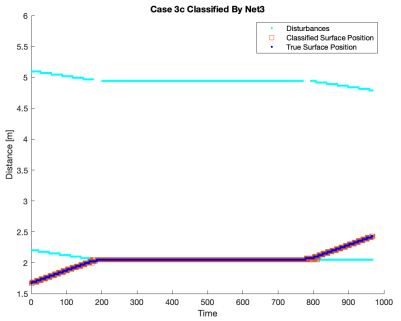
(a) Case 3c classified by Net1.



(b) Case 3c classified by Net1*.

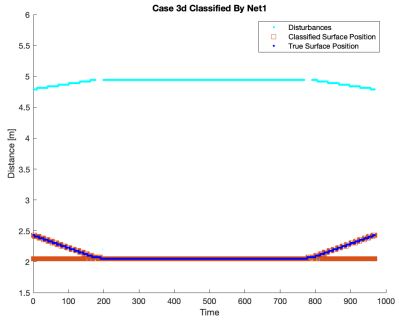


(c) Case 3c classified by Net2.

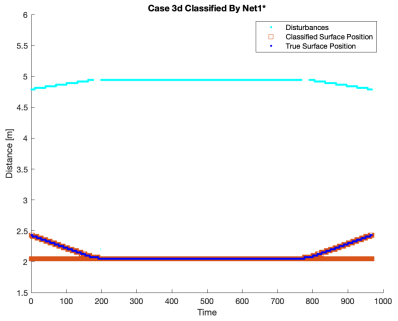


(d) Case 3c classified by Net3.

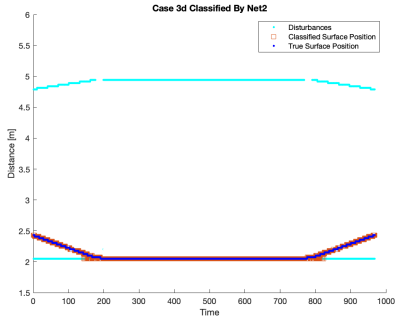
Figure B.7: Results for Case 3c using supervised learning



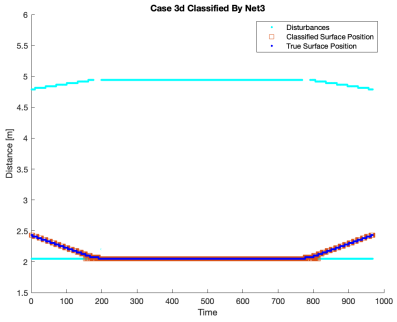
(a) Case 3d classified by Net1.



(b) Case 3d classified by Net1*.



(c) Case 3d classified by Net2.



(d) Case 3d classified by Net3.

Figure B.8: Results for Case 3d using supervised learning

C
Appendix

Table C.1: Success rates for all networks on all test cases, calculated by the evaluation metric from Section 7.1

Case	**Net1**	**Net1***	**Net2**	**Net3**	**Mean**	**Case Type Mean**
1a	0.861	0.852	0.065	0.899	0.669	
1b	0.912	0.901	0.039	0.956	0.702	
						0.686
2a	0.311	0.135	0.003	0.674	0.281	
2b	0.878	0.822	0.002	0.786	0.622	
						0.452
3a	0.940	0.957	0.340	0.946	0.796	
3b	0.924	0.800	0.333	0.958	0.753	
3c	0.899	0.785	0.309	0.931	0.731	
3d	0.624	0.616	0.898	0.851	0.747	
						0.757
Mean	0.794	0.733	0.249	0.875		

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY