T.R.

GEBZE TECHNICAL UNIVERSITY

GRADUATE SCHOOL

NODE CLUSTERING AND FUSION FOR MOVING TARGET LOCALIZATION IN DISTRIBUTED SEISMIC SENSOR NETWORKS

ERDEM KÖSE

A THESIS OF DOCTORATE

DEPARTMENT OF ELECTRONICS ENGINEERING

ADVISOR: ASSIST. PROF. DR. ALİ KÖKSAL HOCAOĞLU

JULY 2024

T.R.

GEBZE TECHNICAL UNIVERSITY

GRADUATE SCHOOL

NODE CLUSTERING AND FUSION FOR MOVING TARGET LOCALIZATION IN DISTRIBUTED SEISMIC SENSOR NETWORKS

ERDEM KÖSE

A THESIS OF DOCTORATE

DEPARTMENT OF ELECTRONICS ENGINEERING

ADVISOR: ASSIST. PROF. DR. ALİ KÖKSAL HOCAOĞLU

JULY 2024

T.C.

GEBZE TEKNİK ÜNİVERSİTESİ

LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

DAĞITIK SİSMİK SENSÖR AĞLARINDA HAREKETLİ HEDEF KONUMLANDIRMA İÇİN SENSÖR KÜMELEMESİ VE FÜZYONU

ERDEM KÖSE

DOKTORA TEZİ

ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMAN: DR. ÖĞR. ÜYESİ ALİ KÖKSAL HOCAOĞLU

TEMMUZ 2024

GEBZE
TECHNICAL UNIVERSITY

DOCTORATE JURY APPROVAL FORM

A thesis submitted by Erdem Köse defended on 04/07/2024 before the jury formed with the 24/06/2024 date and 2024/32 numbered decision of the GTU Graduate Administration Board, has been accepted as a DOCTOR of PHILOSOPHY thesis in the Department of Electronics Engineering.

JURY

MEMBER (THESIS ADVISOR)	: Assist. Prof. Dr. Ali Köksal HOCAOĞLU
MEMBER	: Assoc. Prof. Dr. Serdar Süer ERDEM
MEMBER	: Prof. Dr. Koray KAYABOL
MEMBER	: Prof. Dr. Müştak Erhan YALÇIN
MEMBER	: Assoc. Prof. Dr. Sultan ALDIRMAZ ÇOLAK

APPROVAL

Gebze Technical University Graduate Administration Board
…/…/…… date and ……/… numbered decision.

SIGNATURE/SEAL

ABSTRACT

Localization is of paramount importance in various domains including navigation, military defense, transportation, earthquake monitoring, healthcare, and sports. While conventional positioning technologies such as the Global Positioning System (GPS) and Global Navigation Satellite System (GLONASS) are widely employed, communication technologies like Bluetooth and Long-Term Evolution (LTE) signals also contribute to location estimation through multilateration after distance assessment. Additionally, Radio Detection and Ranging (RADAR) technologies are commonly utilized for positioning by relying on the reflection of transmitted signals. However, alternative signal types such as sound and ground vibration, naturally emitted by targets, can also be exploited for localization.

Seismic sensors offer the advantage of passively capturing signals without detection by the target, via requiring relatively low energy consumption. Despite the advantages of operating at low frequencies, reduced power consumption and heat dissipation, seismic sensors impose limitations on their range. While their most common applications include traffic monitoring, railway health monitoring, earthquake early warning, vehicle detection, and classification, vehicle localization with seismic sensors remains a less explored area in the literature. As a result, our research endeavors to showcase the potential of seismic sensor networks in determining the positions and routes of military vehicles passing through critical zones, whether tracked or wheeled.

Although numerous positioning methods have been proposed, seismic sensors have predominantly been utilized for pinpointing the epicenter of intense ground movements, particularly in earthquake monitoring. However, adapting these methods for vehicle positioning encounters challenges stemming from factors such as terrain type, weather conditions, vehicle speed, and axle count, all of which influence the type of vibrations detected. This study critically examines the constraints of existing methods in extracting Raw Signal Strength (RSS) from seismic signals generated by different vehicles. To address these challenges, we propose a novel approach leveraging statistical signal processing and Long-Short Term Memory (LSTM)-based Convolutional Neural Networks (CNN). This innovative methodology aims to overcome the limitations associated with vehicle localization using seismic sensor.

Keywords: Seismic Sensor Networks, Localization, Machine Learning, Convolutional Neural Networks, Long-Short Term Memory, Statistical Signal Processing.

ÖZET

Konum belirleme, navigasyon, askeri savunma, ulaşım, deprem izleme, sağlık ve spor gibi çeşitli alanlarda büyük önem taşımaktadır. Küresel Konumlandırma Sistemi (GPS) ve Küresel Navigasyon Uydu Sistemi (GLONASS) gibi geleneksel konumlandırma teknolojileri yaygın olarak kullanılırken, Bluetooth ve 5G haberleşme sinyalleri gibi iletişim teknolojileri de mesafe değerlendirmesinden sonra çoklu yansıtma yoluyla konum tahminine katkıda bulunur. Ayrıca, Radyo Algılama ve Uzaklık Ölçme (RADAR) teknolojileri, iletilen sinyallerin yansımasına dayanarak konumlandırma için yaygın olarak kullanılmaktadır. Bununla birlikte, hedefler tarafından doğal olarak yayılan ses ve yer titreşimi gibi alternatif sinyal türlerinden de konum belirleme için yararlanılabilir.

Sismik sensörler, nispeten düşük enerji tüketimi gerektirerek hedef tarafından algılanmadan sinyalleri pasif olarak yakalama avantajı sunar. Sismik sensörlerin düşük frekanslarda çalışması, düşük güç tüketimi ve ısı yayılımı avantajlarına rağmen, menzillerine sınırlamalar getirmektedir. En yaygın uygulamaları arasında trafik izleme, demiryolu sağlık izleme, deprem erken uyarı, araç algılama ve sınıflandırma yer alırken, sismik sensörlerle araç konumlandırma literatürde daha az araştırılmış bir alan olmaya devam etmektedir. Bizim araştırmamız ister paletli ister tekerlekli olsun, kritik bölgelerden geçen askeri araçların konumlarının belirlenmesinde sismik sensör ağlarının potansiyelini ortaya koymaya çalışmaktadır.

Çok sayıda konumlandırma yöntemi önerilmiş olmasına rağmen, sismik sensörler ağırlıklı olarak özellikle deprem izlemede yoğun yer hareketlerinin merkez üssünü belirlemek için kullanılmıştır. Ancak, bu yöntemlerin araç konumlandırma için uyarlanması, arazi tipi, hava koşulları, araç hızı ve aks sayısı gibi faktörlerden kaynaklanan zorluklarla karşılaşır ve bunların tümü tespit edilen titreşim türünü etkiler. Bu çalışma, farklı araçlar tarafından üretilen sismik sinyallerden RSS elde etmede mevcut yöntemlerin kısıtlamalarını incelemektedir. Bu zorlukların üstesinden gelmek için, istatistiksel sinyal işleme ve Uzun-Kısa Süreli Bellek tabanlı Evrişimsel Sinir Ağlarından yararlanan yeni bir yaklaşım öneriyoruz. Bu yenilikçi metodoloji, sismik sensörler kullanarak araç lokalizasyonu ile ilgili sınırlamaların üstesinden gelmeyi amaçlamaktadır.

Anahtar Kelimeler: Sismik Sensör Ağları, Konum Tespiti, Makine Öğrenmesi, Evrişimsel Sinir Ağları, Uzun-Kısa Süreli Bellek, İstatistiksel İşaret İşleme.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my family—Hasibe Köse, Şirin Köse, and Sema Köse—for their unwavering support throughout this journey. A special acknowledgment goes to Hatice Köse for her diverse forms of assistance, especially during my time abroad.

I extend my heartfelt appreciation to Dr. Ali Köksal Hocaoğlu, my PhD supervisor, and Dr. Müştak Erhan Yalçın, for their invaluable guidance and enlightenment during this critical PhD journey. Their support has played a pivotal role in my success, and I am truly grateful for their mentorship.

TABLE OF CONTENTS

LIST of ABBREVIATIONS and ACRONYMS

1-D	:	One Dimensional
2-D	:	Two Dimensional
3-D	:	Three Dimensional
$\alpha - \beta(-\gamma)$ Filter	:	Alpha-Beta(-Gamma) Filter
AAV	:	Assault Amphibious Vehicle
ACF	:	Autocorrelation Function
AOA	:	Angle of Arrival
CART	:	Classification and Regression Tree
CA-CFAR	:	Cell-Averaging Constant False-Alarm Rate
CLEDM	:	Classification and Localization using Estimated Dynamics and Multimodal Data
CFAR	:	Constant False-Alarm Rate
CFT	:	Continuous Fourier Transform
CNN	:	Convolutional Neural Networks
ConvGRU	:	Convolutional Gated Recurrent Units
ConvLSTM	:	Convolutional Long-Short Term Memory
CPA	:	Closest Point of Approach
CS-ECNN	:	Compressed Sensing - Edge Convolutional Neural Network
DARPA	:	The Defense Advanced Research Projects Agency
DCT-II	:	Discrete Cosine Transform II
DFT	:	Discrete Fourier Transform
DTFT	:	Discrete Time Fourier Transform
DW	:	Dragon Wagon
DWT	:	Discrete Wavelet Transform
ELU	:	Exponential Linear Unit
ESN	:	Echo State Network
EWT	:	Empirical Wavelet Transform
FC-LSTM	:	Fully Connected Long-Short Term Memory
FD-SVM	:	Fractal Dimension-based Support Vector Machine
FDDL	:	Fisher Discrimination Dictionary Learning
FFT	:	Fast Fourier Transform
FNCC	:	Fast Normalized Cross-Correlation
FPR	:	False Positive Ratio
GMM	:	Gaussian Mixture Models
GO-CFAR	:	Greatest-of Constant False-Alarm Rate
GPS	:	Global Positioning System
GRU	:	Gated Recurrent Units
HDL	:	Hybrid Dictionary Learning
HMM	:	Hidden Markov Model
IIR	:	Infinite Impulse Response
IMM	:	Interacting Multiple Model
IMU	:	Inertial Measurement Unit
kNN	:	k-Nearest Neighbors
LReLU	:	Leaky Rectified Linear Units
LFCC	:	Linear Frequency Cepstral Coefficients

LO-CFAR	:	Least-of Constant False-Alarm Rate
LTE	:	Long-Term Evolution
LSFCC	:	Log-Sigmoid Frequency Cepstral Coefficients
LSTM	:	Long-Short Term Memory
MAE	:	Mean Absolute Error
MFCC	:	Mel-Frequency Cepstral Coefficients
MLP	:	Multilayer Perceptron
MACs	:	Multiply-Accumulate operations
MSE	:	Mean Squared Error
MSLE	:	Mean Squared Logarithmic Error
MUSIC	:	MUltiple SIgnal Classification
NCA	:	Neighborhood Component Analysis
PCA	:	Principal Component Analysis
PIR	:	Passive Infrared Sensor
POC	:	Phase-only Correlation
PRF	:	Passive Radio Frequency
PSD	:	Power Spectral Density
RADAR	:	Radio Detection and Ranging
RMSE	:	Root Mean Squared Error
RN	:	Reservoir Network
RNN	:	Recurrent Neural Networks
RSS	:	Raw Signal Strength
SensIT	:	Sensor Information Technology
SGD	:	Stochastic Gradient Descent
SITEX02	:	SensIT Situational Experiment 02
SNR	:	Signal to Noise Ratio
SPWVD	:	Smooth Pseudo Wigner-Ville Distribution
SSWCC	:	Spectral Statistics and Wavelet Coefficients Characterization
STA/LTA	:	Short/Long Time Average Ratio
STFT	:	Short Time Fourier Transform
TOA	:	Time of Arrival
TDOA	:	Time Difference of Arrival
TPR	:	True Positive Ratio
WIMF	:	Weighted Intrinsic Mode Functions
YOLOv3	:	You Only Look Once version 3

LIST OF FIGURES

Page

LIST OF TABLES

1. INTRODUCTION

In our technology-driven world, sensing and statistical analysis have become pivotal across multiple facets of society. Sensors, indispensable in everyday routines and military endeavors, serve the purpose of identifying, categorizing, locating, and tracking targets. These strides stem from extensive research into natural phenomena, culminating in the development of specialized devices. Perimeter security, a pivotal element in military operations, relies on sensing and machine learning for its effectiveness, which are outcomes of this extensive research. The need for highly reliable security systems in military or highly secured buildings has shifted focus towards intruder detection and classification systems, reducing reliance on security guard observations. Accurate identification and categorization of military vehicles within battlegrounds or other vehicles and trespassers within high-security compounds are in high importance for remote sensing missions.

Conventional multimedia surveillance, particularly reliant on cameras, has long been the primary approach [1], [2], but it has its drawbacks. Advanced infrared camera systems, while dependable in darkness, come at a steep individual cost [3]. Additionally, camera-centric security setups frequently suffer from blind spots due to limited angles, leading to a demand for more cameras [4]. Moreover, the computational demands for processing camera images are considerable. Although mobile computers can manage image processing, their battery life constraints hinder continuous, prolonged image processing [5].

Affordable options like acoustic, seismic, and magnetic sensors have greatly broadened the range of perimeter security [6]–[11]. These sensors provide better signal quality, accuracy, and cost effectiveness, transforming the technologies used for detection, classifying, and pinpointing locations in wireless sensor networks.

Acoustic sensors convert physical air waves into electrical signals, presenting a varied price range that accommodates diverse needs. They are widely used in detection [12]–[14], classification [13], [15]–[19] and localization [20]–[27]. However, their susceptibility to multiple surrounding noise sources poses challenges in noise filtration, emphasizing the importance of a high Signal-to-Noise Ratio (SNR) in acquired data.

On the other hand, seismic sensors detect the physical acceleration of fixed surfaces and translate it into electrical signals. They share cost advantages with acoustic sensors but encounter fewer noise sources. Seismic sensor networks are now commonly used for tasks like early earthquake warnings, distinguishing between human and animal behavior, recognizing human footsteps, and detecting, classifying and tracking vehicles. Yet, their limitation lies in range, necessitating increased sensor deployment. Within wireless sensor networks, optimizing resources like battery usage and bandwidth becomes crucial. Transmitting raw data or feature vectors escalates bandwidth usage and becomes impractical. A prevalent strategy involves local decision-making at sensor nodes, with these decisions relayed to a central fusion center for the final determination.

1.1. Motivation and Scope

Perimeter security holds a vital role in military operations, as previously outlined in Section 1. Our approach, highlighted by its cost-effectiveness and resilience, integrates seismic sensors. Our research centers on the detection and localization of tracked and wheeled military vehicles within seismic sensor networks, an area that has garnered increasing attention over the past two decades [28], [29]. This surge in interest underscores the growing focus on addressing challenges related to vehicle detection, classification, and localization within distributed networks. Key steps in our study include target detection [29]–[32] and localization [29], [33]–[38]. Additionally, fusion techniques [39]–[44], simulation studies on seismic behavior during vehicle passage [45]–[48], and thesis research [49]–[55] are available in the existing literature.

The detection and classification of military vehicles depend on carefully selected datasets [56]–[58]. The ease of accessing and collecting such specific datasets for military vehicles varies, posing difficulties in obtaining complete datasets. Additionally, Gouda et al. [59] have used Inertial Measurement Unit (IMU) data together with CNN to determine robot positions in uniformly distributed sensor networks. Also, Yuan et al. [60] introduced a method for 3-D indoor target localization using Passive Radio Frequency (PRF) distribution.

The only publicly available dataset for this study is the third Sensor Information Technology (SensIT) Situational Experiment conducted by The Defense Advanced Research

Projects Agency (DARPA), yielding the noteworthy SensIT Situational Experiment 02 (SITEX02) dataset [57]. SITEX02 encompasses seismic, acoustic and passive infrared signals from two distinct types of military vehicles: The fully tracked Assault Amphibious Vehicle (AAV) and the fully wheeled Dragon Wagon (DW). The DARPA SensIT program emphasizes target detection and identification at the sensor field level to augment remote situational awareness capabilities. Widely utilized in literature for classification and detection purposes, this dataset includes ground truth trajectory data recorded through Global Positioning System (GPS) transmitters, rendering it an exemplary benchmark for localization algorithms. Importantly, it stands out as the only freely available dataset among comparable datasets.

Given the extensive research on classification and detection, exploring seismic signatures for vehicle localization presents a unique challenge, especially in military contexts. Our approach draws upon recent advancements in military vehicle localization methods to demonstrate the potential of modeling seismic signatures for this purpose. While Li et al. [61] briefly mentioned studies involving the SITEX02 dataset, localization specifics remained unclear. Similarly, Kumar [62] investigated tracking using the SITEX01 dataset, but lacked detailed explanations. Boettcher et al. focused on localization using the SITEX00 dataset [63]–[65], employing Time Difference of Arrival from an acoustic standpoint. Brooks et al. [66] explored location-centric localization through acoustic signatures, while Ashraf et al. employed Passive Infrared Sensors (PIR) for region-based localization. Sheng et al. [67] investigated a combination of acoustic, seismic, and PIR modalities for localization. Duarte et al. [68] examined the impact of distance on detection, while Rahman [69] applied Optimized Maximum Likelihood with acoustic signatures for localization. Le Borgne et al. [70] delved into distributed regression using seismic signatures for target localization, and Sindhu [71] explored sensor node-wise tracking, albeit without direct location measurement. Inspired by these studies, we aim to leverage spatio-temporal seismic features and cutting-edge machine learning techniques for regression to estimate target localizations within sensor clusters. Seismic signatures offer robustness against noise, while modern machine learning algorithms excel at extracting models from data.

1.2. Contributions

In this thesis study, our contributions to military vehicle detection and localization using the SITEX02 dataset include:

- Examining the frequency domain response of seismic waves to determine the most efficient frequency range for detection and localization tasks.
- Proposing a feature extraction algorithm for classification based on these analyses and evaluating this algorithm on different classifiers.
- Investigating how seismic signals behave differently for various vehicles, such as AAV and DW, on different types of roads. The road information is generated from satellite images using GPS data from the SITEX02 dataset.
- Grouping closely located nodes for better detection based on distance. It is known that three sensors are needed for localization with signal strength, so to improve target localization, we combine data from these groups using spatio-temporal domain seismic features.
- Employing a Convolutional Long-Short Term Memory (ConvLSTM) architecture for better data interpretation with the spatio-temporal seismic features we extracted.
- Examining how the length of time frames affects the accuracy of detection and localization, since ConvLSTM uses both time and seismic features.
- Evaluating the detection and localization performance across different distances between targets and sensors, ranging from 10 to 100 meters in 10-meter steps, using GPS data. Comparing the results with previous studies.

1.3. Thesis Plan

The plan for the thesis is as follows:

In Chapter 2, we presented an overview of studies and state-of-the-art techniques, elucidating them both visually and mathematically. We began with an introduction to seismic sensors, followed by spectral estimation, a fundamental component of most feature extraction methods. Subsequently, we delved into signal detection, emphasizing its significance in reducing operational costs and regression noise. Next, we discussed classification, as regression with neural networks shares common elements with classification. Finally, we reviewed literature related to localization within this chapter.

In Chapter 3, we introduced the seismic dataset and conducted an investigation into the frequency domain. We elucidated the rationale behind our architecture and presented our feature extraction scheme, along with the detection-regression architecture employing ConvLSTM for seismic node clusters.

In Chapter 4, we performed various tests on the data itself. However, the primary focus of this chapter is to showcase the results of our proposed method. We conducted different tests across various distance ranges and temporal lengths, comparing the noise and accuracy of our proposed method.

Finally, in Chapter 5, we draw conclusions based on the studies conducted. Additionally, we suggest potential future research directions to further advance the topics explored in this dissertation.

2. LITERATURE REVIEW

The groundwork for signal processing in ground sensor networks involves several essential steps: preprocessing, feature extraction, detection and classification and/or localization. While classification may not always be mandatory, we will delve into key techniques as they have a connection with the localization process.

To initiate the process, a comprehensive understanding of sensor characteristics is imperative. Factors such as potential frequency attenuation below the sampling frequency and around 0 Hz, as well as the possibility of integer sampling rather than floating point, need consideration. Preprocessing measures, including downsampling and bitwidth compression, may be necessary prior to advancing through subsequent stages. This aspect is detailed in Section 2.1.

Subsequently, the focus shifts to feature extraction. Many methods rely on the conversion from the temporal to the frequency domain. Temporal domain feature extraction methods tend to be largely statistical, while Wavelet decomposition integrates both temporal and frequency domains to derive features. Further elaboration on this topic can be found in Section 2.2.

The third step involves detection, which is crucial for distinguishing between background signals and the actual signal relevant to the use case. Detection methods can leverage those outlined in Section 2.2, but specific techniques related to detection are expounded upon in Section 2.3.

Moving forward, the fourth step revolves around localization. Following the aforementioned steps, a timeframe with detected events emerges, applicable for classification (refer to methods in Section 2.4) and, predominantly, localization. Here, localization pertains to estimating the relative position of the target in relation to sensor nodes, detailed in Section 2.5.

2.1. Seismic Sensor Types

Since our study is in seismic sensor networks, this section will introduce the types of seismic sensor used in literature [56]–[58]. We will consider three examples from seismic sensor types: SM24 geophone, MMA7361 accelerometer and MiniSense 100 piezosensor. All of three sensors has their specific characteristics, advantages and disadvantages. But the reason we mention them, to show there are different physical methods to measure acceleration.

2.1.1. Geophone

A geophone serves as a device designed to transform ground movement, specifically velocity, into voltage, which can then be recorded at a designated recording station. The difference between the measured voltage and the baseline is termed the seismic response. This seismic response is systematically analyzed to gain insights into the structural characteristics of the Earth.

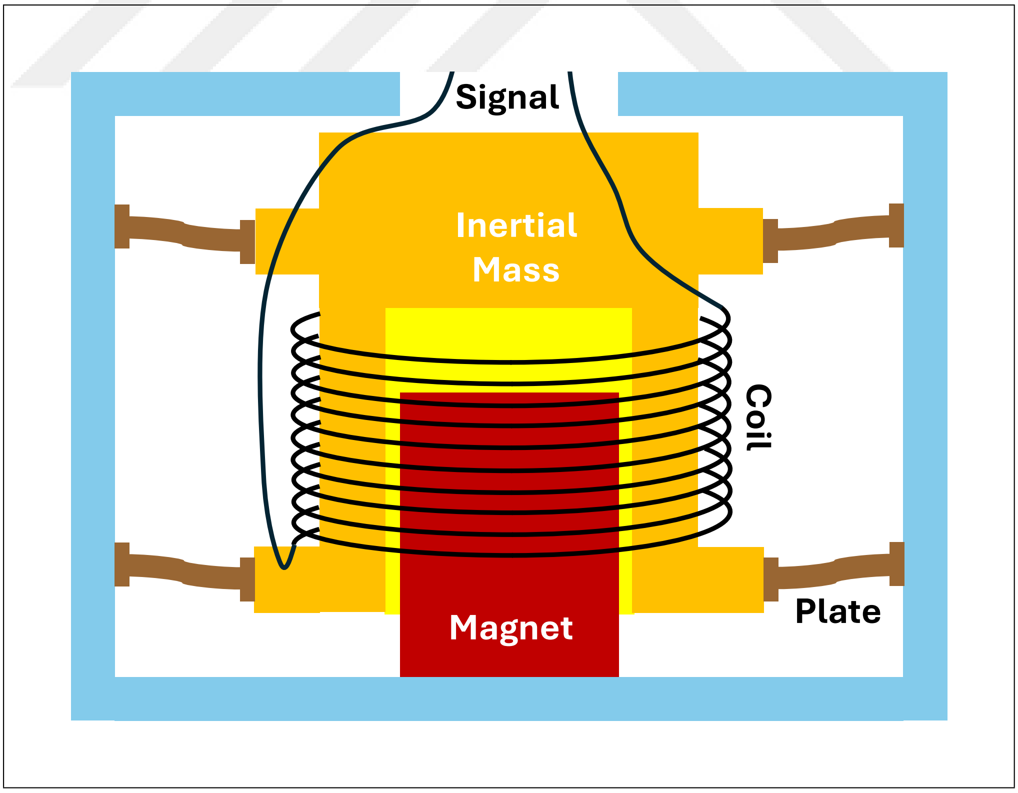


Figure 2.1: Geophone cross section.

The SM24 geophone [72], crafted with precision, leverages I/O Sensor technology for optimal seismic exploration. Prioritizing low distortion and high specifications, it guarantees high-fidelity data for 2-D and 3-D surveys. With an extended bandwidth crucial for 2 ms/24-bit recording systems, Figure 2.1 presents the geophone's chassis and internal structure, offering insights into its design for seismic research.

$$\mathrm{H}\,(s) = \frac{K s^2}{s^2 + 2\omega_{\text{natural}}\zeta s + \omega_{\text{natural}}^2} \tag{2.1}$$

The SM24 geophone is described by its transfer function as given in (2.1), in Laplace domain. This function involves parameters such as the gain K=28.8, natural frequency ω_{natural}=$2\pi 10$ rad/s, and damping ratio ζ=0.69, designed for a 1000 Ω shunt resistance. The aim is to achieve critical damping in the system, with a maximum correct frequency ω_{max}=$2\pi 200$ rad/s. Figure 2.2 displays the frequency response of the SM24 under various shunt resistance configurations.

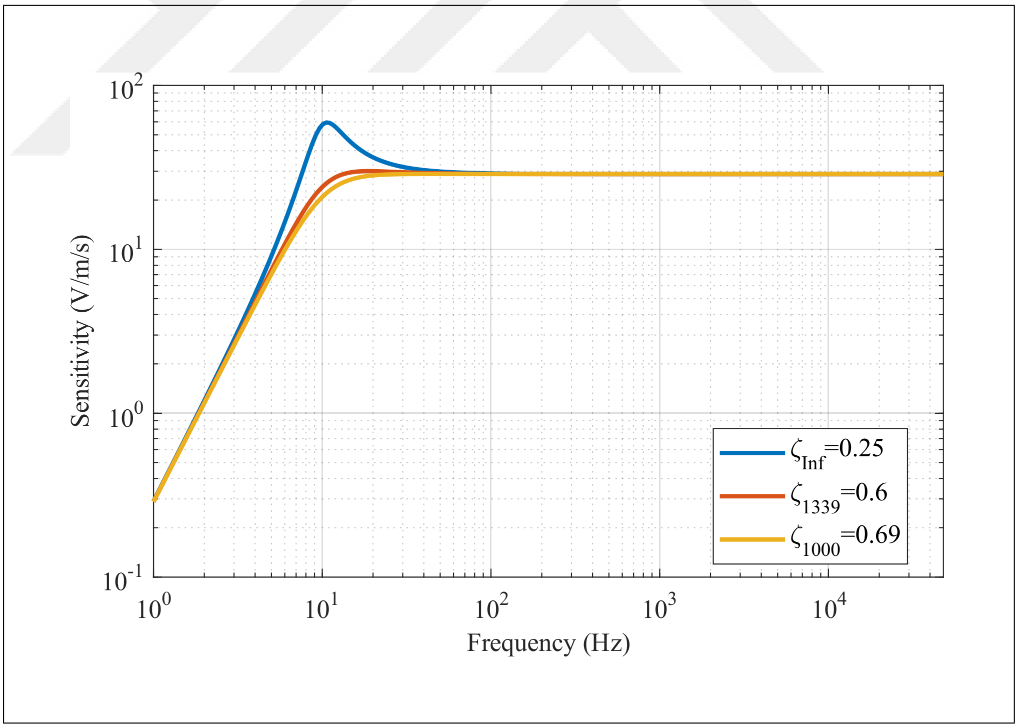


Figure 2.2: SM24 geophone frequency response.

2.1.2. Inertial Measurement Unit

An accelerometer is a tool designed to quantify the proper acceleration of an object. Proper acceleration denotes the acceleration, i.e., the rate of change of velocity, of the object concerning an observer in free fall, specifically with respect to an inertial frame of reference. It's important to distinguish proper acceleration from coordinate acceleration, which represents acceleration concerning a specific coordinate system, and this system may or may not be undergoing acceleration itself.

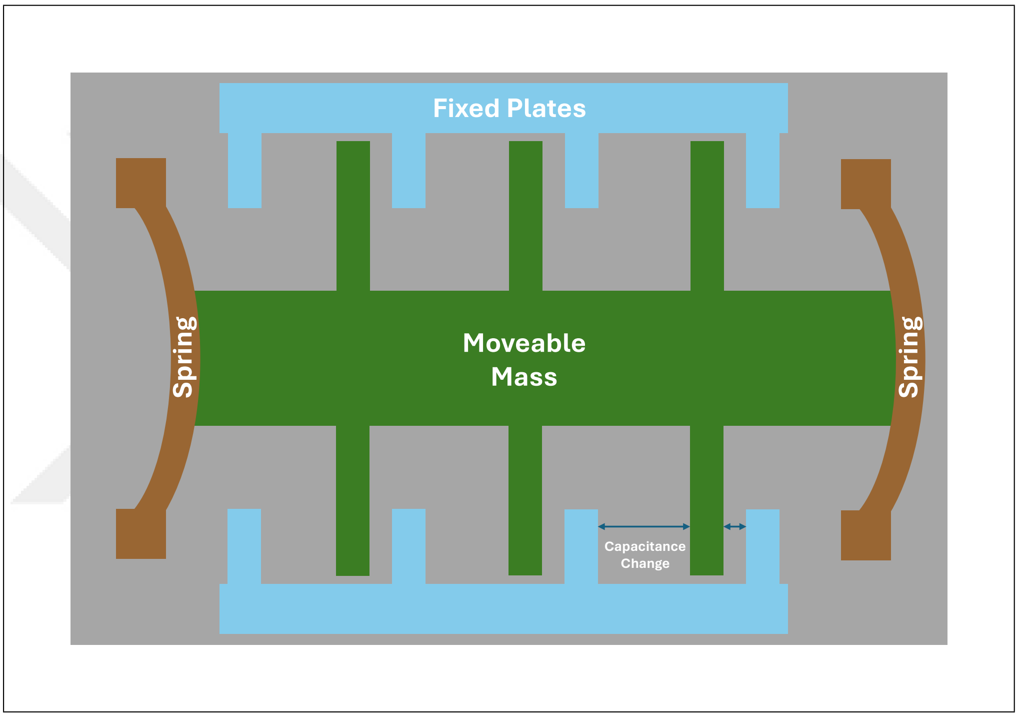


Figure 2.3: Accelerometer integrated circuit structure.

The MMA7361 IMU [73] distinguishes itself as a capacitive micromachine accelerometer, valued for its efficiency in power consumption and streamlined design. Equipped with signal conditioning, self-test capabilities, and a g-Select feature, it offers the flexibility to choose between two sensitivities: ±1.5g and ±6g. For an in-depth understanding, Figure 2.3 provides insight into the internal structure, where capacitance change is the measurement for acceleration.

$$\mathrm{H}\left(s\right) = \frac{K\omega_{\text{natural}}}{s + \omega_{\text{natural}}} \tag{2.2}$$

The MMA7361 IMU accelerometer is characterized by its transfer function presented in (2.2) in the Laplace domain. It exhibits a gain K=0.8, a natural frequency ω_{natural}=$2\pi300$ rad/s, a damping ratio ζ=0.01786, and a settling time T_{stl}=0.5 ms. The frequency response of the this accelerometer is depicted in Figure 2.2.

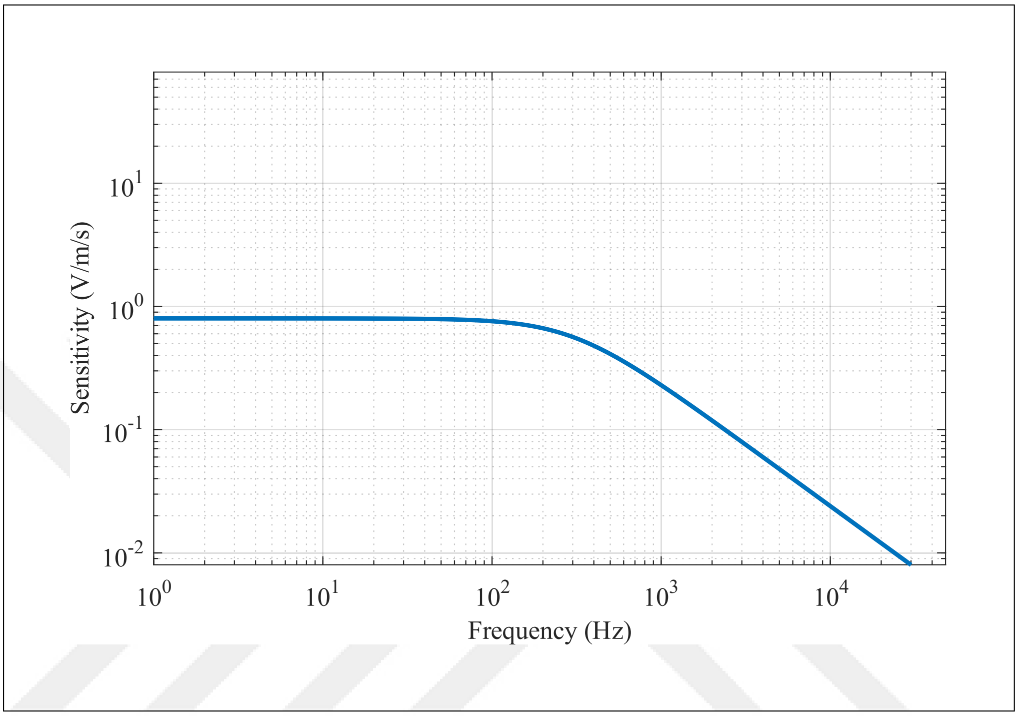


Figure 2.4: MMA7361 IMU frequency response.

2.1.3. Piezosensor

A piezoelectric sensor is a mechanism that employs the piezoelectric effect to gauge variations in pressure, acceleration, temperature, strain, or force. It accomplishes this by converting these physical changes into an electrical charge. For a detailed exploration, refer to Figure 2.5, offering a glimpse into the internal structure of the piezosensor.

The Minisense 100 [74] is an economical cantilever-type vibration sensor that employs a mass loading mechanism, enhancing sensitivity at lower frequencies. Featuring user-friendly pins for easy installation and soldering, it offers both horizontal and vertical mounting options, including a variant with reduced height.

The MiniSense 100 piezosensor generates a signal based on the force acting on strain gauges due to mass vibration, as depicted in Figure 2.5. This phenomenon is charac-

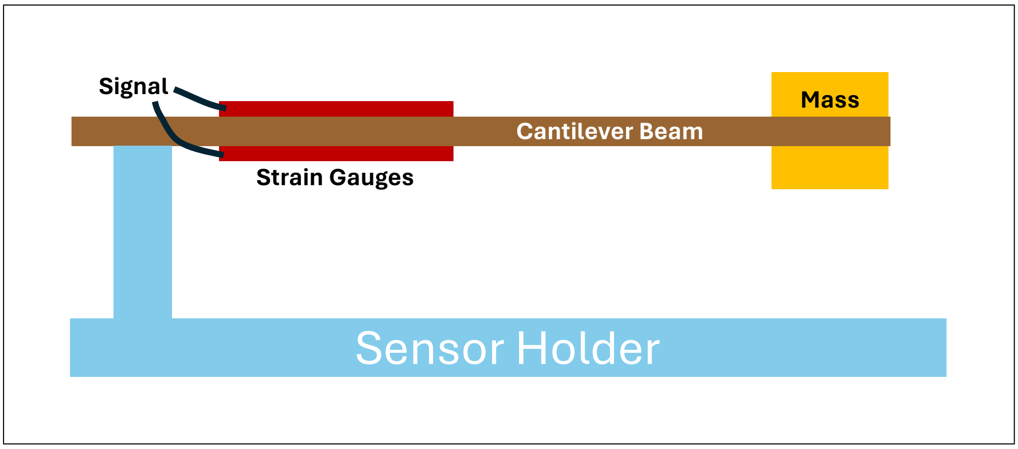


Figure 2.5: Piezosensor internal structure.

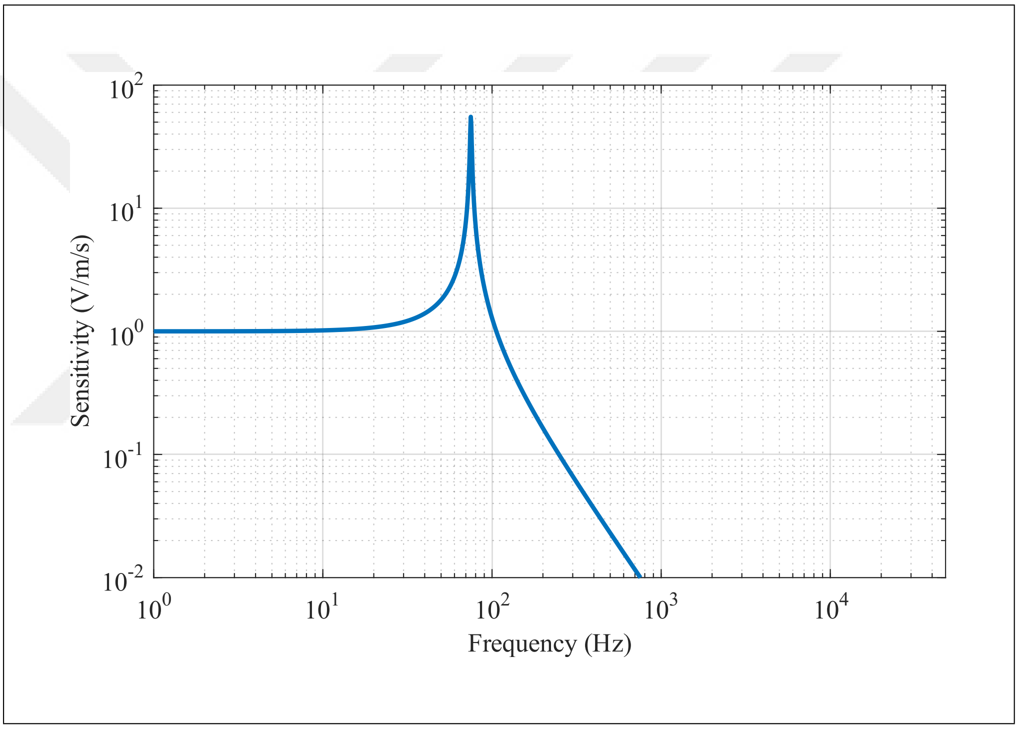


Figure 2.6: MiniSense 100 piezosensor frequency response.

terized by its transfer function, expressed in the Laplace domain and given by (2.3). The transfer function features a gain of K=1, a natural frequency ω_{natural}=$2\pi 75$ rad/s, a damping ratio of ζ=0.01786, and a settling time of T_{stl}=10 ms. The frequency response of this accelerometer is illustrated in Figure 2.2.

$$\mathrm{H}\left(s\right) = \frac{K\omega_{\text{natural}}^2}{s^2 + 2\omega_{\text{natural}}s + \omega_{\text{natural}}^2} \tag{2.3}$$

2.2. Spectral Estimation

Spectral estimation serves as the cornerstone of feature extraction in time series detection and classification. It facilitates the transformation of time series signals into spatial or spatio-temporal formats, with each spatial output bin being considered orthogonal under the assumption of no spectral leakage. This not only enables classifiers to efficiently model input data but also reduces the required training time.

For the SITEX02 dataset, researchers have applied various feature extraction methods that include both frequency domain and time domain approaches [57], [75], with a notable emphasis on frequency domain features within the literature. The introduction of Fast Fourier Transform (FFT) and Power Spectral Density (PSD) estimates via an Autoregressive (AR) model by Li et al. [61] marked an early advancement for the SITEX00 dataset. Following this, Duarte et al. employed coefficients from the Discrete Fourier Transform (DFT) of acoustic and seismic signals for the SITEX02 dataset. In another development, Kornaropoulos et al. implemented the 1-D Discrete Wavelet Transform (DWT) [76]. Additionally, Wang et al. demonstrated the efficacy of sparse representation, calculated through L1 minimization on MFCC data, exhibiting superior performance [77]. Taheri et al. assessed the acoustic dataset using a time-varying autoregressive stochastic model, providing qualitative performance evaluations [78]. In a subsequent study, Wang et al. explored the use of Mel-Frequency Cepstral Coefficients (MFCC) to derive multi-dimensional frequency spectrum features from acoustic sensors [79]. Ghosh et al. extracted 43 features using multi-domain transformations, which they optimized with Neighborhood Component Analysis (NCA) and Principal Component Analysis (PCA), achieving enhanced classification results compared to utilizing all features [80]. Finally, Kalra et al. introduced the Empirical Wavelet Transform (EWT) as a novel feature extraction method [81].

In this section, the first method employed is Fourier transform, renowned for its simplicity and widespread usage in spectral transformation. Building upon Fourier transform, Welch's method is then introduced, which involves segmenting time series into smaller windows and averaging them.

The third approach, Wigner distribution, statistically averages small windows, yielding smoother spectral responses. Unlike the Short Time Fourier Transform (STFT), it offers a spatio-temporal transformation.

Next, the autoregressive method is discussed. This method estimates a determined amount of sinusoidal components from the time series without relying on Fourier transform, characterizing its parametric nature.

Subsequently, the Multiple Signal Classification (MUSIC) method, another parametric technique, is presented. Designed to extract desired sinusoidal components from the noise subspace, it surpasses the autoregressive method in frequency accuracy, but lagging behind in amplitude accuracy.

Finally, cepstral coefficients are explored. This method relies on predetermined filter banks to extract specific frequency bands based on prior knowledge of the time series signal. However, it operates in the cepstral domain rather than the spectral domain.

2.2.1. Discrete Fourier Transform

The Fourier Transform, or Continuous Fourier Transform (CFT), is a linear transformation represents a function in a manner describing its frequency components. This transformation is derived from Fourier Series, specifically applicable to periodic signals. For Fourier Series coefficients a_n where $n \in \mathbb{Z}$, calculations are performed using (2.4) for an input function $\mathrm{u}_\mathrm{a}\left(t\right) \in [-P/2, P/2]$, where $t \in \mathbb{R}$ and $P \in \mathbb{N}^+$. The synthesis formula is provided in (2.5).

$$a_n = \frac{1}{P}\int_{-P/2}^{P/2} \mathrm{u}_\mathrm{a}\left(t\right) e^{-j2\pi\frac{n}{P}t} dt \tag{2.4}$$

$$\mathrm{u}_\mathrm{a}\left(t\right) = \sum_{n=0}^{N-1} a_n e^{j2\pi\frac{n}{P}t} \tag{2.5}$$

CFT as defined in (2.6) can be formally derived from the analysis formula (2.4) by taking the limit as $P \to \infty$, concurrently with $\frac{n}{P} \to f$ [82], where f represents frequency with Hz as the unit. Inverse Continuous Fourier Transform is provided in (2.7).

$$\mathcal{F}\{\mathrm{u}_\mathrm{a}\left(t\right)\} = \mathrm{U}_\mathrm{a}\left(f\right) = \int_{-\infty}^{\infty} \mathrm{u}_\mathrm{a}\left(t\right) e^{-j2\pi f t} dt \tag{2.6}$$

$$\mathcal{F}^{-1}\{\mathrm{U}_\mathrm{a}\left(f\right)\} = \mathrm{u}_\mathrm{a}\left(t\right) = \int_{-\infty}^{\infty} \mathrm{U}_\mathrm{a}\left(f\right) e^{j2\pi f t} df \tag{2.7}$$

We can sample the continuous function $\mathrm{u_a}(t)$ at a rate of f_s using $\mathrm{u}[n] = \mathrm{u_a}(n/f_s)$, where the sample index is denoted by $n \in \mathbb{Z}$, and the sampling frequency is $f_s \in \mathbb{R}^+$. Subsequently, the Discrete Time Fourier Transform (DTFT) and its inverse can be obtained using (2.8) and (2.9), respectively.

$$\mathrm{U}(f) = \sum_{n=-\infty}^{\infty} \mathrm{u}[n]\, e^{-j2\pi \frac{f}{f_s} n} \tag{2.8}$$

$$\mathrm{u}[n] = \frac{1}{f_s} \int_{-fs/2}^{fs/2} \mathrm{U}(f)\, e^{j2\pi \frac{f}{f_s} n} df \tag{2.9}$$

We can express the N-Point DFT and its inverse using (2.10) and (2.11), respectively, by discretizing the frequency variable f in the DTFT equations into N intervals. Here, $k \in \mathbb{N}$.

$$\mathrm{U}[k] = \sum_{n=0}^{N-1} \mathrm{u}[n]\, e^{-j2\pi \frac{n}{N} k} \tag{2.10}$$

$$\mathrm{u}[n] = \frac{1}{N} \sum_{n=0}^{N-1} \mathrm{U}[k]\, e^{j2\pi \frac{n}{N} k} \tag{2.11}$$

FFT denotes the efficient computation of the DFT using computers, a concept emphasized in various studies. Different approaches, such as the Cooley-Tukey algorithm [83] designed for power-of-2 lengths and the Bluestein algorithm [84] applicable to arbitrary lengths, along with hardware acceleration schemes [85]–[88], have been developed. The FFT stands out as the most practical and efficient method for transforming signals from the temporal domain to the frequency domain.

2.2.2. Welch’s Method

Welch’s method for PSD estimation is assuming that signal is m weakly stationary and weakly ergodic, and calculating the mean PSD of overlapping segments of input signal.

1. First, we need to get rid of mean of input signal as in (2.12).

$$\tilde{\mathrm{u}}[n] = \mathrm{u}[n] - \frac{1}{N} \sum_{n=0}^{N-1} \mathrm{u}[n] \tag{2.12}$$

2. Then creating segments of $\tilde{\mathrm{u}}\left[n\right]$ with $N_{ovl} < N_{FFT} < N$ and $N_{ovl}, N_{FFT} \in \mathbb{N}$ in (2.13).

$$\tilde{\mathrm{u}}_i\left[n\right] = \tilde{\mathrm{u}}\left[\left(i-1\right)\left(N_{FFT} - N_{ovl}\right) + n\right]\mathrm{w}\left[n\right] \tag{2.13}$$

   where $\mathrm{w}\left[n\right]$ is a window function while $n < N_{FFT}$ and $n \in \mathbb{N}$.

   Total number of segments will be N_{seg} as in (2.14), where $\lfloor\cdot\rfloor$ is rounding to floor.

$$N_{seg} = \left\lfloor \frac{N - N_{ovl}}{N_{FFT} - N_{ovl}} \right\rfloor \tag{2.14}$$

3. Last step is taking DFT of each segment using (2.8) and getting average of them as in (2.15), where $\tilde{\mathrm{U}}_i^*\left[k\right]$ is complex conjugate of $\tilde{\mathrm{U}}_i\left[k\right]$. And PSD will be (2.15).

$$\mathrm{S_{UU}}\left[k\right] = \frac{1}{N_{seg}} \sum_{i=0}^{N_{seg}-1} \tilde{\mathrm{U}}_i^*\left[k\right]\tilde{\mathrm{U}}_i\left[k\right] \tag{2.15}$$

2.2.3. Wigner Distribution

The Wigner Distribution, originally proposed by Wigner [89] for quantum corrections to statistical mechanics. Its importance is underscored by its dual capability: it transitions seamlessly into the spectral density function for stationary processes at any time, while also maintaining full equivalence to the non-stationary autocorrelation function. This dual functionality of the Wigner Distribution provides valuable insights into the temporal evolution of spectral density.

We omit the mention of STFT, as it primarily involves windowing the input signal and applying Fourier Transform. Instead, we focus on the Wigner Distribution, which yields results with reduced noise.

For zero-mean input signal $\mathrm{u_a}\left(t\right)$, Wigner distribution given in (2.16), where $\mathrm{u_a^*}\left(t\right)$ represents the complex conjugate of input signal.

$$\mathrm{S_{UU}}\left(t, f\right) = \int_{-\infty}^{\infty} \mathrm{u_a}\left(t + \frac{\tau}{2}\right)\mathrm{u_a^*}\left(t - \frac{\tau}{2}\right)e^{-j2\pi f\tau}d\tau \tag{2.16}$$

2.2.4. Autoregressive Model

The Autoregressive Model (AR) employs a regression approach on lag series derived from the initial time series. Unlike multiple linear regression, which predicts the outcome as a linear combination of various independent input variables, the autoregressive model predicts future data points using only past observations. Specifically, the output is formulated as a linear combination of the previous p data points, where p indicates the size of the lag window. This model is fundamentally grounded in the Wold decomposition theorem [90], which provides the theoretical underpinning for AR models.

The AR model incorporates feedback coefficients a_k in (2.17) for z-domain representation of an Infinite Impulse Response (IIR) filter, with the feedforward coefficient $b_k = \sigma^2 \delta [k]$. Here, $p + 1$ represents the count of AR model parameters, and σ^2 denotes the white-noise variance.

$$\mathrm{H}[z] = \frac{\mathrm{H_B}[z]}{\mathrm{H_A}[z]} = \frac{\sigma^2}{\sum_{k=0}^{p} a_k z^{-k}} \tag{2.17}$$

Autocorrelation Function (ACF) is a crucial element in estimating the AR parameters of a wide-sense stationary process, described in (2.18). Here, $\mathrm{u}^* [n]$ denotes the complex conjugate of the input sequence $\mathrm{u}[n]$, and $\mathrm{E}[\cdot]$ represents the expected value operator.

$$\mathrm{r_{uu}}[k] = \mathrm{E}[\mathrm{u}^* [n] \, \mathrm{u}[n+k]] \tag{2.18}$$

The relations for AR and ACF are given in (2.19).

$$\mathrm{r_{uu}}[k] = \begin{cases} -\sum_{l=1}^{p} a_l \mathrm{r_{uu}}[k-l] & \text{if } k \geq 1 \\ \sigma^2 - \sum_{l=1}^{p} a_l \mathrm{r_{uu}}[-l] & \text{if } k = 0 \end{cases} \tag{2.19}$$

The Yule-Walker equations, proposed by Yule and Walker [91], [92], are considered a highly effective method for determining filter coefficients a_k in an AR model with low noise variance [93]. Hence, (2.19) has been termed Yule-Walker equations for $k \geq 1$ in (2.20) and $k = 0$ in (2.21), where $\mathrm{r_{uu}}^* [k] = \mathrm{r_{uu}}[-k]$. This approach is particularly effective when dealing with low noise variance scenarios.

$$\begin{bmatrix} \mathrm{r_{uu}}[0] & \mathrm{r_{uu}}^{*}[1] & \ldots & \mathrm{r_{uu}}^{*}[p-1] \\ \mathrm{r_{uu}}[1] & \mathrm{r_{uu}}[0] & \ldots & \mathrm{r_{uu}}^{*}[p-2] \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{r_{uu}}[p-1] & \mathrm{r_{uu}}[p-2] & \ldots & \mathrm{r_{uu}}[0] \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} = - \begin{bmatrix} \mathrm{r_{uu}}[1] \\ \mathrm{r_{uu}}[2] \\ \vdots \\ \mathrm{r_{uu}}[p] \end{bmatrix} \tag{2.20}$$

$$\begin{bmatrix} \mathrm{r_{uu}}[0] & \mathrm{r_{uu}}^{*}[1] & \ldots & \mathrm{r_{uu}}^{*}[p] \\ \mathrm{r_{uu}}[1] & \mathrm{r_{uu}}[0] & \ldots & \mathrm{r_{uu}}^{*}[p-1] \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{r_{uu}}[p] & \mathrm{r_{uu}}[p-1] & \ldots & \mathrm{r_{uu}}[0] \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} \sigma^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{2.21}$$

Determining the process parameter p and the autocorrelation function $\mathrm{r_{uu}}[k]$ for the input signal allows us to obtain AR model parameters σ^2 and a_k. This is achieved by solving equations (2.20) and (2.21), with the condition $a_0 = 1$ based on the model definition.

With these parameters in hand, we can then estimate the PSD $\mathrm{S_{UU}}$ using equation (2.22).

$$\mathrm{S_{UU}}(f) = \frac{\sigma^2}{\left|1 + a[1]\,e^{-j2\pi f} + a[2]\,e^{-2(j2\pi f)} + \ldots + a[p]\,e^{-p(j2\pi f)}\right|^2} \tag{2.22}$$

2.2.5. Multiple Signal Classification

The differences between the MUSIC algorithm and the AR model stem from their distinct analytical approaches. The AR model utilizes autocorrelation with variable lag amounts to forecast future data points. In contrast, MUSIC applies spectral analysis and matrix algebra. This technique capitalizes on the orthogonal properties of signals within noise subspaces. Consequently, MUSIC achieves greater accuracy in frequency location estimation but falls short in amplitude estimation.

The signal model is described in (2.23), where $\mathrm{u}[n]$ denotes the complex-valued sinusoidal. Here, A_i indicates amplitudes and f_i indicates frequencies, respectively. The term $\mathrm{n}[n]$ represents additive observation noise. Assuming that $\mathrm{n}[n]$ is white noise with a variance of σ^2, as is typically done, the covariance of the signal x is presented in

(2.24).

$$\mathrm{u}\left[n\right] = \mathrm{n}\left[n\right] + \sum_{i=0}^{K-1} A_i e^{j2\pi f_i n} \tag{2.23}$$

$$\mathbf{R}_{\mathrm{UU}} = \sigma^2 + \sum_{i=0}^{K-1} |A_i|^2 \hat{\mathbf{e}}_i \hat{\mathbf{e}}_i^H \tag{2.24}$$

where

$$\hat{\mathbf{e}}_i = \begin{bmatrix} 1, & e^{j2\pi f_i}, & \dots & e^{j2\pi f_i (M-1)} \end{bmatrix}^T \tag{2.25}$$

and $M > K$, $\mathbf{e}_i^H$ is Hermitian i.e. complex conjugate and transpose of $\hat{\mathbf{e}}_i$.

The MUSIC algorithm applies eigen-decomposition to the covariance matrix $\mathbf{R}_{\mathrm{UU}}$ to estimate frequencies. These frequencies are identified at the K highest peaks of the PSD, detailed in (2.26). This PSD is known as the pseudo-spectrum because it aids in detecting sinusoidal components within the analyzed time series, although it is not an exact representation of a true PSD.

$$\mathrm{S}_{\mathrm{UU}}\left(f\right) = \frac{1}{\mathbf{e}^H (\sum_{k=K+1}^{M} \mathbf{v}_k \mathbf{v}_k^H) \mathbf{e}} \tag{2.26}$$

where

$$\mathbf{e} = \begin{bmatrix} 1, & e^{j2\pi f}, & \dots & e^{j2\pi f (M-1)} \end{bmatrix}^T \tag{2.27}$$

In (2.27), $\mathbf{v}_k$ represent the eigenvectors of $\mathbf{R}_{\mathrm{UU}}$, which correspond to its eigenvalues arranged in descending order.

2.2.6. Cepstral Coefficients

Cepstral coefficients phenomenon has a wide application field in audio based recognition studies. MFCC [94], [95] is the most used method for audio recognition [77], [79], [96].

Mel-Frequency is not suitable for low frequency signals like seismic ones because they are specialized for audio recognition so. Thus, Linear Frequency Cepstral Coefficients (LFCC) [97] and Inverse Mel-Frequency Cepstral Coefficients (IMFCC) [98] are more suitable for seismic researches. Also, Log-Sigmoid Frequency Cepstral Coefficients (LSFCC) is another and new approach for seismic recognition too [99]. Flow of cepstal coefficient extraction is given below.

1. Apply pre-emphasis filter

$$\tilde{\mathrm{u}}\left[n\right] = \mathrm{u}\left[n\right] - 0.95u\left[n-1\right] \tag{2.28}$$

2. Create segments of $\tilde{\mathrm{u}}\left[n\right]$ with $N_{ovl} < N_{FFT} < N$ and $N_{ovl}, N_{FFT} \in \mathbb{N}$ in (2.29).

$$\tilde{\mathrm{u}}_i\left[n\right] = \tilde{\mathrm{u}}\left[\left(i-1\right)\left(N_{FFT} - N_{ovl}\right) + n\right] w[n] \tag{2.29}$$

   where $w\left[n\right]$ is a window function while $n < N_{FFT}$ and $n \in \mathbb{N}$.

   Total number of segments will be N_{seg} as in (2.14), where $\lfloor \cdot \rfloor$ is rounding to floor.

$$N_{seg} = \left\lfloor \frac{N - N_{ovl}}{N_{FFT} - N_{ovl}} \right\rfloor \tag{2.30}$$

3. Get PSD of the segments

$$\mathrm{S}_{\mathrm{U_iU_i}}\left[k\right] = \tilde{\mathrm{U}}_i^*\left[k\right]\tilde{\mathrm{U}}_i\left[k\right] \tag{2.31}$$

4. Followings are mapping and inverse mapping functions for Mel space(2.32) (2.33), inverse Mel space (2.34) (2.35) or linear space (2.36), where ω_s is radial sampling frequency.

$$\omega_{\mathrm{MFCC}} = \mathrm{f}^{\mathrm{MFCC}}_{\mathrm{map}}\left(\omega_{\mathrm{linear}}\right) = 2\pi 1125 \log_{10}\left(1 + \frac{\omega_{\mathrm{linear}}}{2\pi 700}\right) \tag{2.32}$$

$$\omega_{\mathrm{linear}} = \mathrm{f}^{\mathrm{MFCC}}_{\mathrm{invmap}}\left(\omega_{\mathrm{MFCC}}\right) = 2\pi 700\left(10^{\frac{\omega_{\mathrm{MFCC}}}{2\pi 1125}} - 1\right) \tag{2.33}$$

$$\omega_{\text{IMFCC}} = \mathrm{f}_{\text{map}}^{\text{IMFCC}}\left(\omega_{\text{linear}}\right) = \mathrm{f}_{\text{map}}^{\text{MFCC}}\left(\frac{\omega_s}{2}\right) - \mathrm{f}_{\text{map}}^{\text{MFCC}}\left(\frac{\omega_s}{2} - \omega_{\text{linear}}\right) \tag{2.34}$$

$$\omega_{\text{linear}} = \mathrm{f}_{\text{invmap}}^{\text{IMFCC}}\left(\omega_{\text{IMFCC}}\right) = \frac{\omega_s}{2} - \mathrm{f}_{\text{invmap}}^{\text{MFCC}}\left(\mathrm{f}_{\text{map}}^{\text{MFCC}}\left(\frac{\omega_s}{2}\right) - \omega_{\text{IMFCC}}\right) \tag{2.35}$$

$$\omega_{\text{LFCC}} = \mathrm{f}_{\text{map}}^{\text{LFCC}}\left(\omega_{\text{linear}}\right) = \omega_{\text{lineear}} \tag{2.36}$$

$$\omega_{\text{linear}} = \mathrm{f}_{\text{invmap}}^{\text{LFCC}}\left(\omega_{\text{LFCC}}\right) = \omega_{\text{LFCC}} \tag{2.37}$$

5. To construct the filter bank, it is necessary to generate an array of $L+2$ equidistant frequencies denoted as $\boldsymbol{w}^{\text{new}}$ as in (2.38). These frequencies span between the transformed minimum and maximum frequencies, $\mathrm{f}_{\text{map}}\left(\omega_{\text{min}}\right)$ and $\mathrm{f}_{\text{map}}\left(\omega_{\text{max}}\right)$, respectively. Here, L represents the size of filter bank, ω_{min} is 0 rad/s, and ω_{max} is defined as $\frac{\omega_s}{2}$. For discrete frequencies, we will use (2.39).

$$\boldsymbol{w}^{\text{new}} = \mathrm{f}_{\text{map}}\left(\left[0, \frac{1}{L+1}, \frac{2}{L+1}, \frac{3}{L+1}, \ldots, \frac{L}{L+1}, 1\right] \frac{\omega_s}{2}\right) \tag{2.38}$$

$$\boldsymbol{k}^{\text{new}} = N_{FFT} \frac{\boldsymbol{w}^{\text{new}}}{\omega_s} \tag{2.39}$$

6. Obtain each filter in the bank of range $0 < l < L$ using (2.40), where $k_{l-1}^{\text{new}} \in \boldsymbol{k}^{\text{new}}$. Filter bank examples for LFCC, MFCC and IMFCC are given in Figure 2.7, Figure 2.8 and Figure 2.9 respectively.

$$\mathrm{H}_l\left[k\right] = \begin{cases} 0 & \text{, if } \quad k < k_{l-1}^{\text{new}} \\ \frac{k - k_{l-1}^{\text{new}}}{k_l^{\text{new}} - k_{l-1}^{\text{new}}} & \text{, if } k_{l-1}^{\text{new}} \leq k < k_l^{\text{new}} \\ \frac{k_{l+1}^{\text{new}} - k}{k_{l+1}^{\text{new}} - k_l^{\text{new}}} & \text{, if } k_l^{\text{new}} \leq k < k_{l+1}^{\text{new}} \\ 0 & \text{, if } k_{l+1}^{\text{new}} \leq k \end{cases} \tag{2.40}$$

7. Apply the filter bank to the PSD, sum the energy in each filter. Then take the natural logarithm of all filter bank energies.

$$\mathrm{S}_{\mathrm{U_iU_i}}^{\text{filt}}[l] = \ln\left(\sum_{k=0}^{N_{fft}-1} \left|\mathrm{H}_l\left[k\right] \mathrm{S}_{\mathrm{U_iU_i}}\left[k\right]\right|\right) \tag{2.41}$$

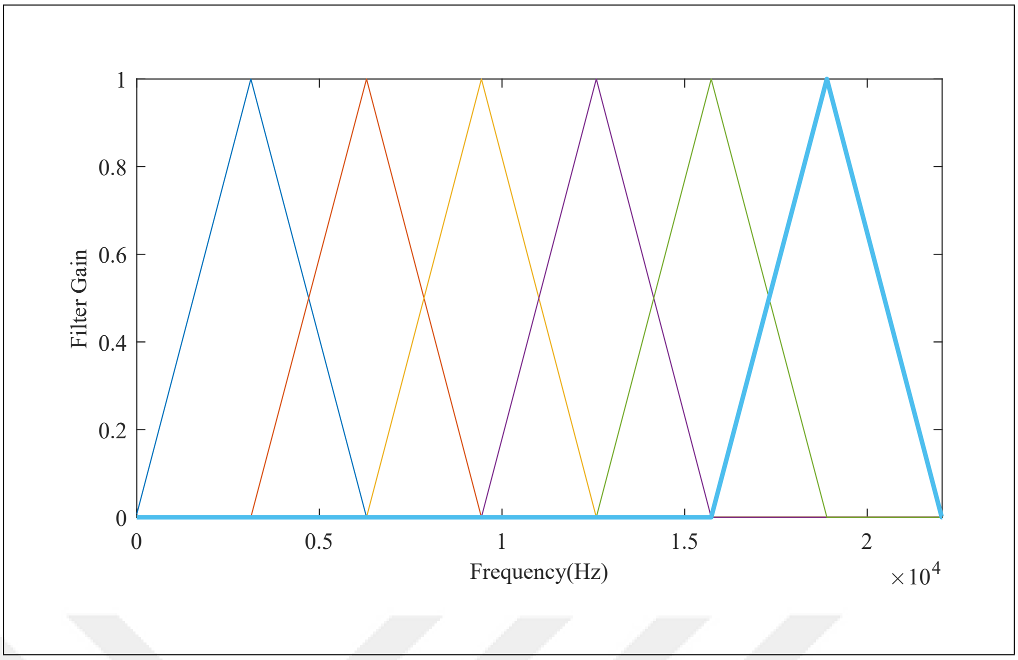


Figure 2.7: Linear filter bank for $f_s = 44100$ Hz with 6 filters.

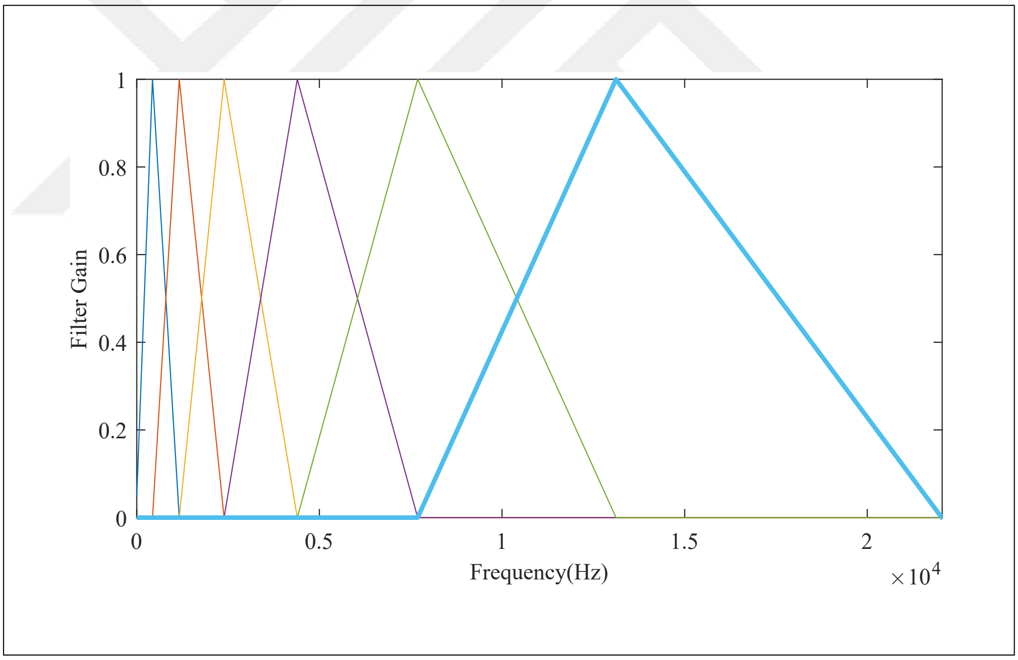


Figure 2.8: Mel filter bank for $f_s = 44100$ Hz with 6 filters.

8. Calculate the Discrete Cosine Transform II (DCT-II) of the logarithmic energies.

$$\mathrm{DCT_{II}}\left\{\mathrm{S}_{\mathrm{U_iU_i}}^{\mathrm{filt}}\left[l\right]\right\}\left[\tilde{l}\right] = \sum_{l=0}^{L-1} \mathrm{S}_{\mathrm{U_iU_i}}^{\mathrm{filt}}\left[l\right] \cos\left[\frac{\pi}{L}\left(l+\frac{1}{2}\right)\tilde{l}\right] \tag{2.42}$$

where $0 \leq l, \tilde{l} \leq L-1$, and $l, \tilde{l} \in \mathbb{N}$.

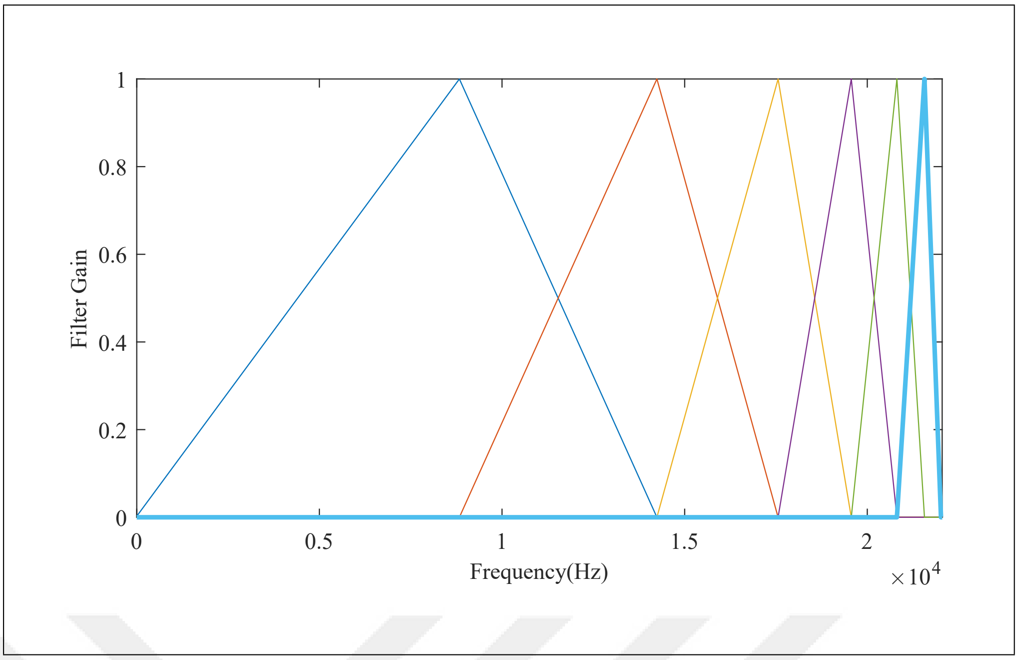


Figure 2.9: Inverse Mel filter bank for $f_s = 44100$ Hz with 6 filters.

Outputs of (2.42) are named cepstral coefficients. Unlike PSD, they do not directly represent the energies of frequency bands, but they are closely related.

2.3. Signal Detection

Signal detection is imperative for discerning the task-related component within a signal for subsequent processing. Detection indicates the occurrence of an event within a signal. Ideally, signal detection should consume less power than classification or regression.

Numerous studies in the literature have investigated the detection of various events for security purposes. For instance, there are studies on vehicle detection [100]–[104], pedestrian detection [47], [105]–[112], pedestrian-vehicle detection [39], [113]–[120], and earthquake and microseismic event detection [121]–[127].

In the literature specific to the SITEX02 dataset, various studies have utilized different techniques. Li et al. applied the Constant False-Alarm Rate (CFAR) technique alongside the Closest Point of Approach (CPA) within a virtually gridded sensor network, though their study was actually on the SITEX00 dataset [61]. Tian et al. introduced the use of wavelets for detection through Spectral Statistics and Wavelet Coefficients Characterization (SSWCC) [128]. Duarte et al. utilized the k-Nearest Neighbors (kNN) method for event detection in military settings [57]. Keally et al. developed Watchdog, a modality-agnostic event detection framework that clusters sensors using Hidden Markov Models (HMM) to meet specific detection accuracy requirements while reducing energy use significantly [129]. Kalra et al. examined the effectiveness of the Smooth Pseudo Wigner-Ville Distribution (SPWVD) in detecting moving military vehicles [130], [131]. Other studies employing SPWVD were conducted by Abidi et al. [132] and Nathani et al. [133]. Additionally, Bin et al. compared Seismic Fractal Features with the Short/Long Time Average Ratio (STA/LTA) and a Fractal Dimension-based Support Vector Machine (FD-SVM) for ground moving target detection [134].

In this section, we will first introduce CFAR, one of the most popular detection algorithms. Subsequently, Phase-only Correlation (POC) and Fast Normalized Cross-Correlation (FNCC) will be introduced. These algorithms, along with the other studies mentioned in the previous paragraphs, contribute diverse approaches to vehicle detection. Detection, from a machine learning perspective, involves two-class classification, which will be discussed in Section 2.4.

2.3.1. Constant False-Alarm Rate

CFAR detectors are predominantly utilized in radar signal processing. The most recognized CFAR methods include Cell-Averaging CFAR (CA-CFAR), Least-of CFAR (LO-CFAR), and Greatest-of CFAR (GO-CFAR). Among these, the CA-CFAR detector [135], [136] is likely the most widely used.

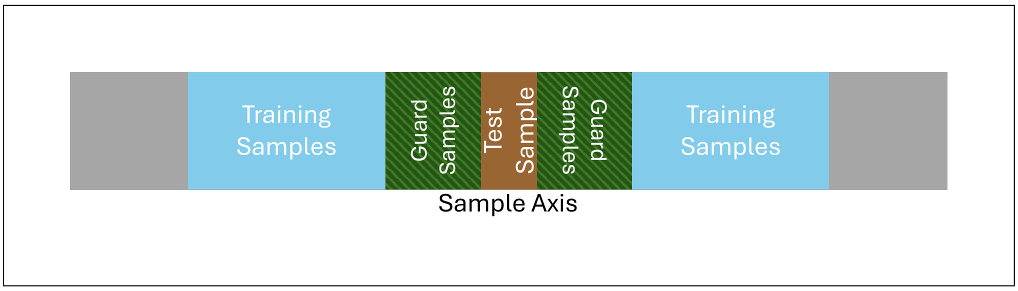


Figure 2.10: One dimensional CFAR representation.

In a CA-CFAR detector, both the leading and lagging samples, known as training cells, which around the test sample, used for extracting the noise samples. The estimated noise power, P_{noise}, is computed as outlined in (2.43), where N_{train} represents the number of training cells, and u $[n_{\text{train}}]$ denotes the sample in the training cells as shown in

Figure 2.10.

$$P_{noise} = \frac{1}{N_{train}} \sum_{n_{train} \in \text{Training Cells}} \mathrm{u}\left[n_{train}\right] \tag{2.43}$$

The threshold is given by (2.44), where P_{fa} represents the desired false alarm rate.

$$T_{CFAR} = N_{train} \left(P_{fa}^{-\frac{1}{N_{train}}} - 1 \right) P_{noise} \tag{2.44}$$

For test sample $\mathrm{u}\left[n_{test}\right]$ in Figure 2.10, CFAR detector function can be written as (2.45).

$$\text{CA-CFAR for } \mathrm{u}\left[n_{test}\right] = \begin{cases} \text{Non-Detect} & \text{, if } \mathrm{u}\left[n_{test}\right] \leq T_{CFAR} \\ \text{Detect} & \text{, if } \mathrm{u}\left[n_{test}\right] > T_{CFAR} \end{cases} \tag{2.45}$$

2.3.2. Phase-Only Correlation

POC [137], [138] phenomenon is a method which is being widely used in image registration [139], [140] and micro-seismic event detection [121], [124], [125]. POC is based on phase change of images which has the same magnitude spectrum. One dimensional POC $\mathrm{r}\left(t\right)$ is given by (2.46) for input Fourier spectrums $\mathrm{U}_1\left(f\right)$ and $\mathrm{U}_2\left(f\right)$ corresponding to stationary input waveforms $\mathrm{u}_1\left(t\right)$ and $\mathrm{u}_2\left(t\right)$, which are same but with different spatial-shift amounts τ_1 and τ_2, respectively. $\mathrm{U}_2{}^*\left(f\right)$ is the complex conjugate of $\mathrm{U}_2\left(f\right)$.

$$r\left(t\right) = \mathcal{F}^{-1}\left\{ \frac{\mathrm{U}_1\left(f\right)\mathrm{U}_2{}^*\left(f\right)}{\left|\mathrm{U}_1\left(f\right)\mathrm{U}_2{}^*\left(f\right)\right|} \right\} = \mathcal{F}^{-1}\left\{ e^{j2\pi f\left(\tau_1 - \tau_2\right)} \right\} = \delta\left(t + \left(\tau_1 - \tau_2\right)\right) \tag{2.46}$$

If we think (2.46) in discrete form for stationary input waveforms $\mathrm{u}_1\left[n\right]$ and $\mathrm{u}_2\left[n\right]$, which are same but with different sample-shift amounts $\tilde{n}_1$ and $\tilde{n}_2$, respectively, we obtain (2.47).

$$r\left[n\right] = \mathcal{F}^{-1}\left\{\frac{\mathrm{U}_1\left[k\right]\mathrm{U}_2{}^*\left[k\right]}{\left|\mathrm{U}_1\left[k\right]\mathrm{U}_2{}^*\left[k\right]\right|}\right\} = \mathcal{F}^{-1}\left\{e^{j\frac{2\pi k}{N}\left(\tilde{n}_1-\tilde{n}_2\right)}\right\} = \delta\left[n+\left(\tilde{n}_1-\tilde{n}_2\right)\right] \tag{2.47}$$

2.3.3. Fast Normalized Cross-Correlation

FNCC [141] is a temporal method used for image registration. We denote stationary input waveforms as $\mathrm{u}_1\left[n\right]$ and $\mathrm{u}_2\left[n\right]$, where $\mathrm{u}_1\left[n\right]$ is considered a patch with size N_1, and $\mathrm{u}_2\left[n\right]$ has size N_2. It is assumed that $N_1 \leq N_2$. Normalized Cross-Correlation (NCC) is computed as shown in (2.48), where $\mu_{\mathrm{u}_1\left[\tilde{n}\right]}$ represents the mean of $\mathrm{u}_1\left[n\right]$ around the lag $\tilde{n}$.

$$\gamma\left[\tilde{n}\right] = \frac{\sum_0^{N_1-1}\left[\left(\mathrm{u}_1\left[n-\tilde{n}\right]-\mu_{\mathrm{u}_1\left[\tilde{n}\right]}\right)\left(\mathrm{u}_2\left[n\right]-\mu_{\mathrm{u}_2}\right)\right]}{\sqrt{\sum_0^{N_1-1}\left(\mathrm{u}_1\left[n-\tilde{n}\right]-\mu_{\mathrm{u}_1\left[\tilde{n}\right]}\right)^2\sum_0^{N_1-1}\left(\mathrm{u}_2\left[n\right]-\mu_{\mathrm{u}_2}\right)^2}} \tag{2.48}$$

From the output of (2.48), we anticipate observing a Dirac delta function at the shift between two input images if these images are identical or similar enough. Otherwise, we would likely observe an approximately uniform distribution along the sample axis of $r\left[n\right]$. The vector $\gamma\left[\tilde{n}\right]$ has a length of $2N_1-1$, and its center represents the no-lag point. If $\mathrm{u}_1\left[n\right]$ and $\mathrm{u}_2\left[n\right]$ are identical at lag point $\tilde{n}$, we expect to see a Dirac delta function.

2.4. Classification

Classification serves as a decision-making mechanism through a learned function derived directly from the data. Detection can be seen as a form of two-class classification simultaneously. We predominantly employ classification to establish distinguishing mechanisms directly from the data, rather than analytically formulating it. For a classifier, we have an input tensor composed of features and an output tensor of classes.

Similarly, numerous studies in the literature have explored the detection of various events. Some target groups for classification include pedestrians [106], [108], [112], [142], [143], vehicles [18], [40], [99], [104], [133], [144]–[147], pedestrian-vehicle interactions [113], [115]–[117], [148]–[150], and earthquake and microseismic events [151], [152].

In the SITEX02 dataset-specific literature, Li et al. introduced kNN, Bayesian Classifier with Gaussian Mixture Models (GMM) and Support Vector Machine (SVM) for SITEX00 [61]. Similarly, Duarte et al. utilized the same methods with only FFT for the SITEX02 dataset [57]. Kim et al. proposed a decision tree of GMMs, where the outputs are normalized by measurement length, generated by the Classification and Regression Tree (CART) algorithm [153], utilizing tree outputs for sensor fusion. On a different approach, Guo et al. introduced a Hybrid Dictionary Learning (HDL) approach [96], which utilizes both analysis and synthesis dictionaries for generating discriminative codes and achieving class-specific reconstruction, showcasing improved performance in terms of time consumption and classification. Wang et al. introduced Fisher Discrimination Dictionary Learning (FDDL) in [79], leveraging the Fisher discrimination criterion, demonstrating its superiority, particularly with small amount of training samples, outperforming the widely used SVM. Moreover, Ntalampiras introduced an Echo State Network (ESN)-based classifier [154], rooted in Reservoir Network (RN), demonstrating its superiority over a HMM-based classifier. Jin et al. proposed using LFCC with Convolutional Neural Networks (CNN) and showed it outperforms FFT with CNN for classification [155]. Zhang et al. proposed wavelet-based feature extraction, demonstrating its outperformance over FFT [156]. Jin proposed Weighted Intrinsic Mode Functions (WIMF) for improving MFCC performance [157]. Bin et al. introduced an edge intelligence-based approach called Compressed Sensing - Edge Convolutional Neural Network (CS-ECNN). In this method, seismic signals are initially transformed into a compressed domain via Compressed Sensing (CS) to lower data dimensions while preserving most of the valuable seismic features. Subsequently, a CNN is used to directly extract implicit features from these compressed seismic measurements and classify, which reduces computation time [158].

This section begins with an exploration of the Multilayer Perceptron, a fundamental component of neural networks. We then proceed to discuss Convolutional Neural Networks, which can diminish time-source consumption and enhance the generalization of input tensors, often assumed to be orthogonal, such as photos.

Following this, we delve into Recurrent Neural Networks, which incorporate temporal dynamics into neural networks. Additionally, we introduce the convolutional variants of these Recurrent Neural Networks.

2.4.1. Multilayer Perceptron

Multilayer Perceptron (MLP) is multilayered feedforward neural network topology as can be seen in Figure 2.11. It is the widely used structure for machine learning tasks like regression and classification. It is the main idea behind most of the state-of-art neural network topologies like CNN [159], Recurrent Neural Networks (RNN) [160] and Attention Mechanism [161].

Most basic perceptron node equation can be defined as in (2.49) and (2.50),

$$\begin{aligned}\hat{\mathrm{o}}_{(m)}\left[n_{(m)}\right] =& \mathrm{w}^{\mathrm{bias}}_{(m-1)}\left[n_{(m)}\right] + \\ & \sum_{n_{(m-1)}=0}^{N_{(m-1)}-1} \mathrm{o}_{(m-1)}\left[n_{(m-1)}\right] \mathrm{w}_{(m-1)}\left[n_{(m-1)}, n_{(m)}\right]\end{aligned} \tag{2.49}$$

$$\mathrm{o}_{(m)}\left[n_{(m)}\right] = \mathrm{f}_{\mathrm{act}}\left(\hat{\mathrm{o}}_{(m)}\left[n_{(m)}\right]\right) \tag{2.50}$$

where m is layer number, $n_{(m)}$, $N_{(m)}$ node index and count of nodes in layer m respectively. $\mathrm{w}_{(m-1)}\left[n_{(m-1)}, n_{(m)}\right]$ is weight vector of layer m and $\mathrm{w}^{\mathrm{bias}}_{(m-1)}\left[n_{(m)}\right]$ is bias weight. $\mathrm{f}_{\mathrm{act}}\left(\cdot\right)$ is activation function to provide nonlinearity to the network.

Only for layer 0, the input layer, there are no weights, else there is weight relation between layers. $\mathrm{o}_{(0)}\left[n_{(0)}\right]$ is the input vector, and $\mathrm{o}_{(M)}\left[n_{(M)}\right]$ is the output vector of the network.

Every MLP architecture requires training from well-labeled dataset. Weights can be calculated from output layer to input layer with following equations which is also called Stochastic Gradient Descent [162], where λ called as learning rate. Let's assume $M-1$ as the last layer. We can define Q_m error variable from (2.51)

$$\mathrm{Q}_{(M)} = \frac{1}{2}\sum_{n_{(M)}=0}^{N_{(M)}}\left(\mathrm{o}^{\mathrm{GT}}_{(M)}\left[n_{(M)}\right] - \mathrm{o}_{(M)}\left[n_{(M)}\right]\right)^2 \tag{2.51}$$

where $\mathrm{o}^{\mathrm{GT}}_{(M)}$ is ground truth. The last layers error propagation $\delta_{(M)}\left[n_{(M)}\right]$ can be expressed as in (2.52), where $\mathrm{f}_{\mathrm{act}}{}'$ is first order derivative respecting to $\hat{\mathrm{o}}_{(M)}$.

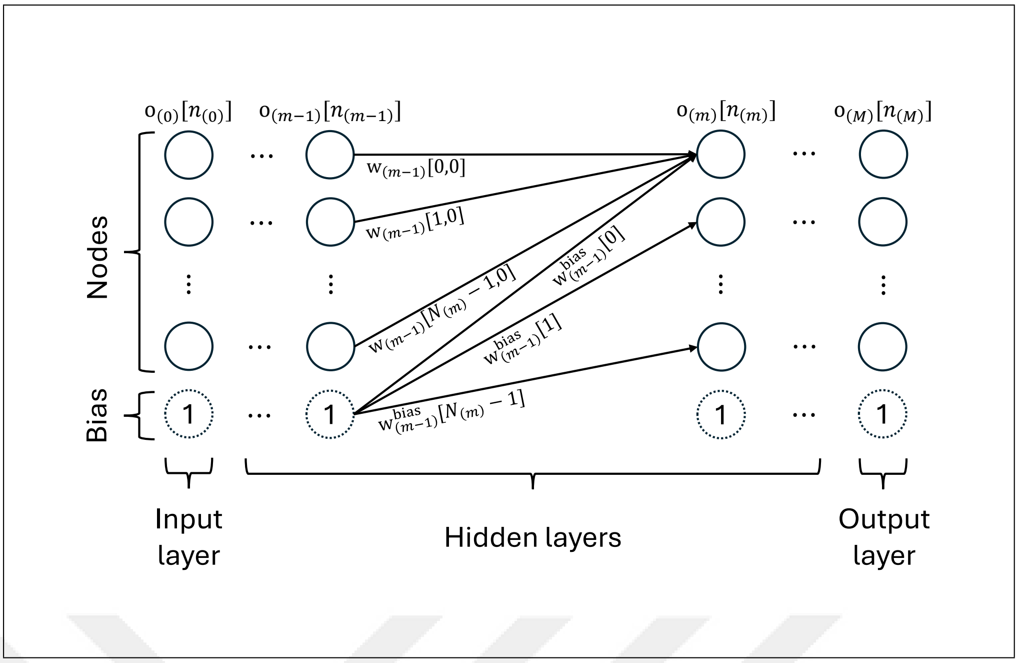

$\mathrm{o}_{(0)}[n_{(0)}]$
$\mathrm{o}_{(m-1)}[n_{(m-1)}]$
$\mathrm{o}_{(m)}[n_{(m)}]$
$\mathrm{o}_{(M)}[n_{(M)}]$
$\mathrm{w}_{(m-1)}[0,0]$
$\mathrm{w}_{(m-1)}[1,0]$
$\mathrm{w}_{(m-1)}[N_{(m)}-1,0]$
$\mathrm{w}^{\mathrm{bias}}_{(m-1)}[0]$
$\mathrm{w}^{\mathrm{bias}}_{(m-1)}[1]$
$\mathrm{w}^{\mathrm{bias}}_{(m-1)}[N_{(m)}-1]$
Nodes
Bias
1
Input layer
Hidden layers
Output layer

Figure 2.11: Multilayer perceptron graph.

$$\delta_{(M)}\left[n_{(M)}\right] = -\left(\mathrm{o}^{\mathrm{GT}}_{(M)}\left[n_{(M)}\right] - \mathrm{o}_{(M)}\left[n_{(M)}\right]\right)\mathrm{f}_{\mathrm{act}}{}'\left(\hat{\mathrm{o}}_{(M)}\left[n_{(M)}\right]\right) \tag{2.52}$$

Then (2.52) can be generalized for rest of the layers as in (2.53). But do not forget, we must have already calculated next error propagation $\delta_{(m+1)}\left[n_{(m+1)}\right]$ for calculating current one $\delta_{(m)}\left[n_{(m)}\right]$.

$$\begin{aligned}\delta_{(m)}\left[n_{(m)}\right] = -&\left(\sum_{n_{(m+1)}=0}^{N_{(m+1)}}\delta_{(m+1)}\left[n_{(m+1)}\right]\mathrm{w}_{(m)}\left[n_{(m)}, n_{(m+1)}\right]\right)\\ &\mathrm{f}_{\mathrm{act}}{}'\left(\hat{\mathrm{o}}_{(m)}\left[n_{(m)}\right]\right)\end{aligned} \tag{2.53}$$

Then we can update the weights using (2.54).

$$\mathrm{w}^{\mathrm{update}}_{(m)}\left[n_{(m)}, n_{(m+1)}\right] = \mathrm{w}_{(m)}\left[n_{(m)}, n_{(m+1)}\right] - \lambda\delta_{(m)}\left[n_{(m)}\right]\mathrm{o}_{(m)}\left[n_{(m)}\right] \tag{2.54}$$

This weight updating progress should be iterative until $\mathrm{Q}_{(M)}$ converges to a certain value determined by user, according to the use case.

2.4.2. Convolutional Neural Networks

The MLP is described in Section 2.4.1. CNN is a type of classifier that comprises two main parts: the convolutional network and the MLP. The convolutional part serves to reduce connections and weights, addressing computational complexity. CNNs are widely employed in image recognition applications, but they can also be utilized with orthogonal and frequency domain transforms of time series such as the DFT, DCT-II, and DWT.

Convolutional layers in CNNs are composed of a grid of neurons, necessitating that the preceding layer also be a grid of neurons. Each neuron receives inputs from a rectangular portion of the prior layer, with identical weights applied to this portion for every neuron in the convolutional layer. As a result, the convolutional layer executes an image convolution of the preceding layer, with the weights defining the convolution filter. Moreover, within each convolutional layer, there could be numerous grids, wherein each grid obtains inputs from all the grids in the prior layer, potentially employing diverse filters.

Convolutional layer equation can be defined as in (2.55) and (2.56), the graph can be seen in Figure 2.12.

$$\hat{\mathrm{O}}^{c_{(m)}}_{(m)} = \mathrm{W}^{\text{bias}}_{(m-1)} \left[c_{(m)}\right] + \sum_{c_{(m-1)}=0}^{C_{(m-1)}-1} \mathrm{O}^{c_{(m-1)}}_{(m-1)} \star \mathrm{W}_{(m-1)} \left[c_{(m-1)}, c_{(m)}\right] \tag{2.55}$$

$$\mathrm{O}^{c_{(m)}}_{(m)} = \mathrm{f}_{\text{act}} \left(\hat{\mathrm{O}}^{c_{(m)}}_{(m)}\right) \tag{2.56}$$

where m is layer number, $c_{(m)}$, $C_{(m)}$ channel index and count of channels in layer m respectively. $\mathrm{W}_{(m-1)} \left[c_{(m-1)}, c_{(m)}\right]$ is kernel tensor of layer m and $\mathrm{W}^{\text{bias}}_{(m-1)} \left[c_{(m)}\right]$ is bias weight. $\mathrm{f}_{\text{act}} (\cdot)$ is activation function to provide nonlinearity to the network.

There are three main concepts in CNN: Convolution, pooling, flatten, MLP. Each convolution layer has specific amount of filters, these are feature extractors from input image. Convolution layer also has an activation function to create nonlinearity and rectified linear unit (ReLU) [163] is the most common one for information preserving

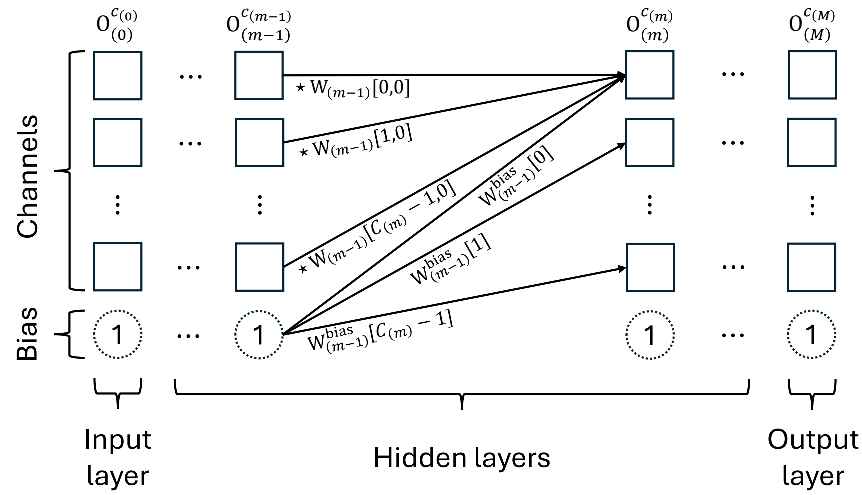


Figure 2.12: Convolutional network graph.

property and being fast on graphical processing units (GPU). After each convolution layer, usually there is a pooling layer to reduce image resolution for getting more general images to extract general features. At the end of convolution-pooling sequence, there is a flattening layer to make features suitable for input of fully connected layer a.k.a. multilayer perceptron. After flattening, multilayer perceptron learns from features and classifies the input image. You Only Look Once v3 (YOLOv3) [164] is a great example for CNN usage.

This concept is proposed for getting rid of hand made features and extracting features automatically from input images. There are several architectures like LeNet-5 [165], AlexNet [166], VGG-1 [167], Inception-v1 [168], Inception-v3 [169], ResNet-50 [170], Xception [171], Inception-v4 [172], ResNeXt-50 [173] etc. which uses and advances this concept.

2.4.3. Recurrent Neural Networks

RNN, and Long-Short Term Memory (LSTM) and Gated Recurrent Units (GRU) networks are all popular choices for handling sequential data in deep learning [160]. Each architecture comes with its own cons and pros for different types of tasks.

RNNs are the most fundamental of the three, designed to process sequences of data

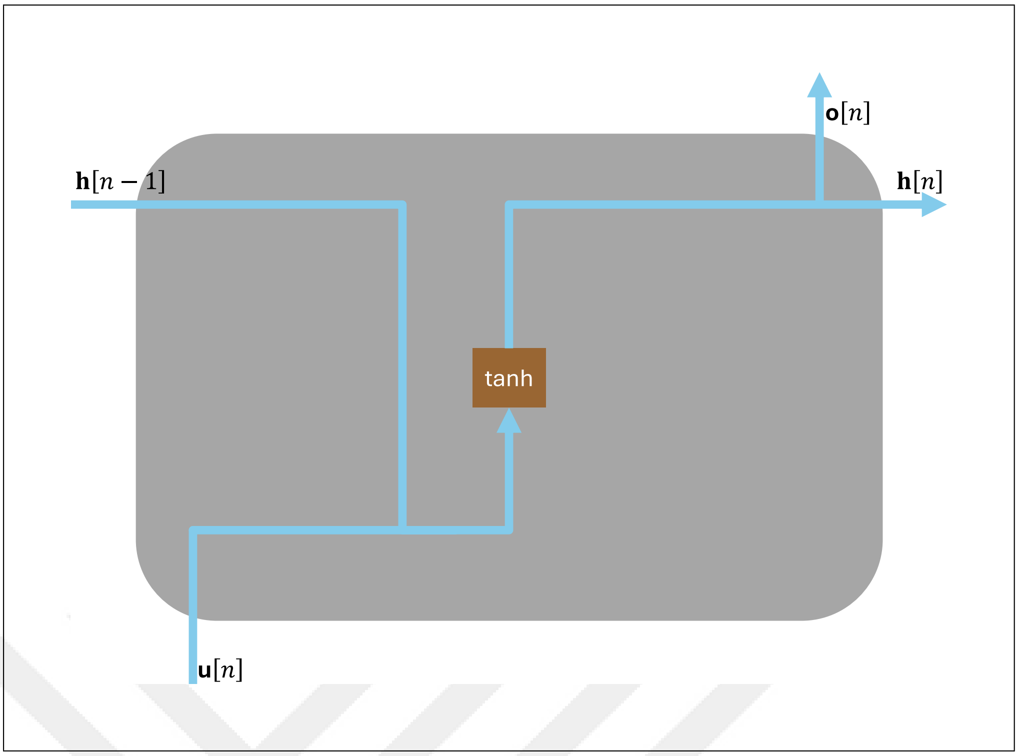


Figure 2.13: RNN cell.

by passing information from one step to the next. As can be seen from Figure 2.13, the difference from MLP node is RNN cells have history record. However, they suffer from the vanishing gradient problem, which limits their ability to capture long-term dependencies in sequences. As a result, RNNs often struggle to retain information over long distances within a sequence, which can hinder performance on tasks requiring such memory, such as language translation or speech recognition.

Equations for RNN cell are given for hidden state in (2.57),

$$\mathbf{h}[n] = \tanh(\boldsymbol{b}_{\mathbf{h}} + \boldsymbol{W}_{\mathbf{hu}} \cdot \mathbf{u}[n] + \boldsymbol{W}_{\mathbf{hh}} \cdot \mathbf{h}[n-1]) \tag{2.57}$$

Output gate in (2.58),

$$\mathbf{o}[n] = \boldsymbol{b}_{\mathbf{o}} + \boldsymbol{W}_{\mathbf{oh}} \cdot \mathbf{h}[n] \tag{2.58}$$

where $\mathbf{u}[n]$ input vector at sample step n. Also, $\boldsymbol{W}_{\mathbf{hu}}$, $\boldsymbol{W}_{\mathbf{hh}}$, $\boldsymbol{W}_{\mathbf{oh}}$ are weight matrices, and $\boldsymbol{b}_{\mathbf{h}}$, $\boldsymbol{b}_{\mathbf{o}}$ bias vectors which are all learned during training.

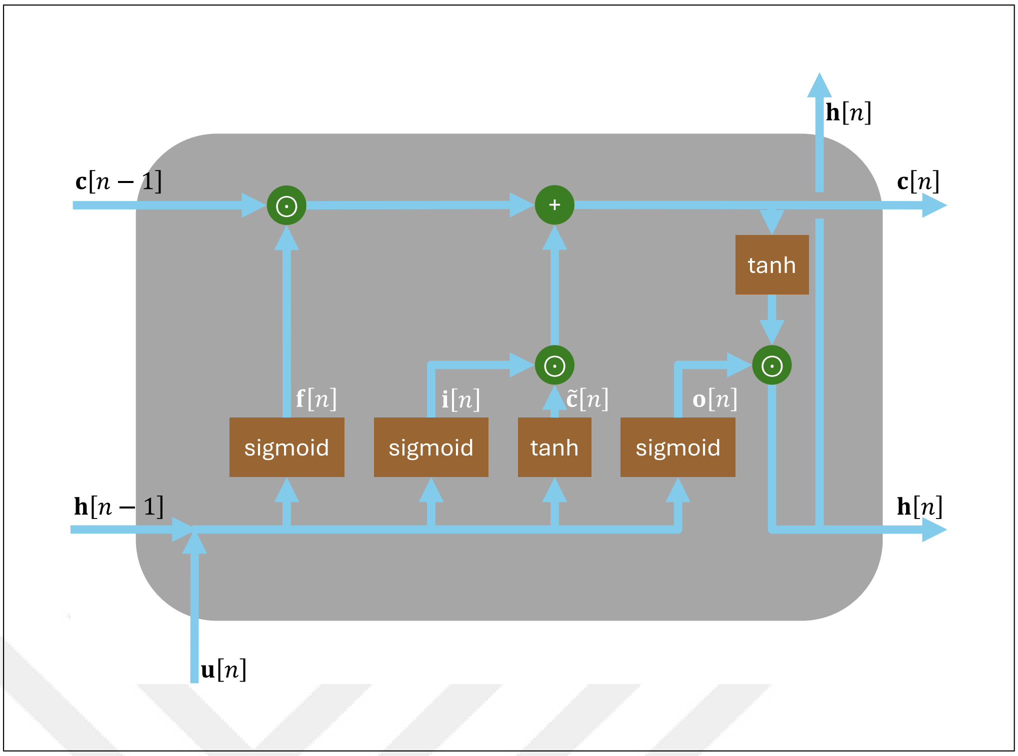


Figure 2.14: LSTM cell.

To overcome the vanishing gradient problem of RNNs, LSTM networks were introduced. LSTM networks incorporate specialized memory cells that can maintain information over long sequences, making them particularly effective for tasks involving long-term dependencies. These memory cells are equipped with input, forget, and output gates, allowing the network to selectively update and retain information. This gating mechanism enables LSTMs to better capture context and relationships within sequential data, leading to improved performance on tasks like speech recognition and sentiment analysis.

Equation for LSTM cell are given for forget gate, which controls how much of the previous cell state to retain or forget, in (2.59),

$$\mathbf{f}[n] = \text{sigmoid}(\boldsymbol{b}_\mathbf{f} + \boldsymbol{W}_\mathbf{f} \cdot \text{concat}(\mathbf{h}[n-1], \mathbf{u}[n])) \tag{2.59}$$

Input gate, controls amount of new information should be stored in cell state, in (2.60),

$$\mathbf{i}[n] = \text{sigmoid}(\boldsymbol{b}_\mathbf{i} + \boldsymbol{W}_\mathbf{i} \cdot \text{concat}(\mathbf{h}[n-1], \mathbf{u}[n])) \tag{2.60}$$

Candidate cell state, holds the new information to be added to the cell state, in (2.61),

$$\tilde{\mathbf{c}}[n] = \tanh(\boldsymbol{b}_{\mathbf{c}} + \boldsymbol{W}_{\mathbf{c}} \cdot \text{concat}(\mathbf{h}[n-1], \mathbf{u}[n])) \tag{2.61}$$

Update cell state, which is updated by combining the previous cell state with the new candidate cell state with forget gate controlling how much of the previous state to retain and the input gate controlling how much new information to add, in (2.62),

$$\mathbf{c}[n] = \mathbf{f}[n] \odot \mathbf{c}[n-1] + \mathbf{i}[n] \odot \tilde{\mathbf{c}}[n] \tag{2.62}$$

Output gate, which determines how much of the updated cell state should be output as the hidden state, in (2.63),

$$\mathbf{o}[n] = \text{sigmoid}(\boldsymbol{b}_{\mathbf{o}} + \boldsymbol{W}_{\mathbf{o}} \cdot \text{concat}(\mathbf{h}[n-1], \mathbf{u}[n])) \tag{2.63}$$

Hidden state in (2.64),

$$\mathbf{h}[n] = \mathbf{o}[n] \odot \tanh(\mathbf{c}[n]) \tag{2.64}$$

where $\mathbf{u}[n]$ input vector at sample step n. Also, $\boldsymbol{W}_{\mathbf{f}}$, $\boldsymbol{W}_{\mathbf{i}}$, $\boldsymbol{W}_{\mathbf{c}}$, $\boldsymbol{W}_{\mathbf{o}}$ are weight matrices, and $\boldsymbol{b}_{\mathbf{f}}$, $\boldsymbol{b}_{\mathbf{i}}$, $\boldsymbol{b}_{\mathbf{c}}$, $\boldsymbol{b}_{\mathbf{o}}$ bias vectors which are all learned during training. The operator $\odot$ element-wise multiplication.

GRU networks, similar to LSTMs, are designed to address the vanishing gradient problem using gating mechanisms. However, GRUs are computationally less complex than LSTMs because they have fewer parameters and lack separate memory cells. GRUs feature an update gate, which combines the functions of the input and forget gates, and they merge the cell state and hidden state into a single state. This streamlined architecture makes GRUs more efficient to train, often resulting in quicker convergence compared to LSTMs.

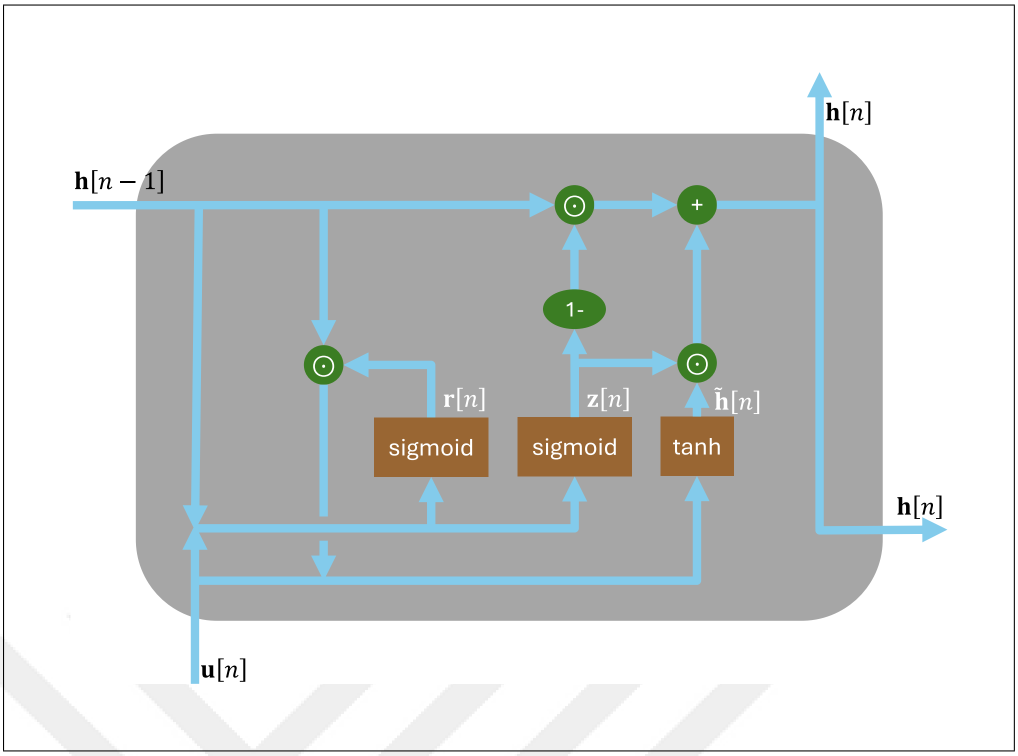


Figure 2.15: GRU cell.

Equation for GRU cell are given for update gate, which controls the amount of recent information to keep and amount of the new input to let through, in (2.65),

$$\mathbf{z}[n] = \text{sigmoid}(\boldsymbol{b}_{\mathbf{z}} + \boldsymbol{W}_{\mathbf{z}} \cdot \text{concat}(\mathbf{h}[n-1], \mathbf{u}[n])) \tag{2.65}$$

Reset gate, which determines amount of the recent information to forget, in (2.66),

$$\mathbf{r}[n] = \text{sigmoid}(\boldsymbol{b}_{\mathbf{r}} + \boldsymbol{W}_{\mathbf{r}} \cdot \text{concat}(\mathbf{h}[n-1], \mathbf{u}[n])) \tag{2.66}$$

Candidate activation, which computes the new candidate hidden state, in (2.67),

$$\tilde{\mathbf{h}}[n] = \tanh(\boldsymbol{b}_{\mathbf{h}} + \boldsymbol{W}_{\mathbf{h}} \cdot \text{concat}(\mathbf{r}[n] \odot \mathbf{h}[n-1], \mathbf{u}[n])) \tag{2.67}$$

Hidden state in (2.68),

$$\mathbf{h}[n] = (1 - \mathbf{z}[n]) \odot \mathbf{h}[n-1] + \mathbf{z}[n] \odot \tilde{\mathbf{h}}[n] \tag{2.68}$$

where $\mathbf{u}[n]$ input vector at sample step n. Also, $\boldsymbol{W}_{\mathbf{r}}$, $\boldsymbol{W}_{\mathbf{z}}$, $\boldsymbol{W}_{\mathbf{h}}$ are weight matrices, and $\boldsymbol{b}_{\mathbf{r}}$, $\boldsymbol{b}_{\mathbf{z}}$, $\boldsymbol{b}_{\mathbf{h}}$ bias vectors which are all learned during training. The operator $\odot$ element-wise multiplication.

All equations for different recurrent cell describe the flow of information through a single RNN unit at each sample step n in a sequence. As default, we can accept $\mathbf{c}[0] = \mathbf{0}$ and $\mathbf{h}[0] = \mathbf{0}$ as initial states, when training a model from scratch.

While RNNs are the simplest and most basic architecture for sequential data processing, they often struggle with long-term dependencies. LSTMs address this issue by incorporating sophisticated memory cells with separate gates for controlling information flow. GRUs offer a simpler alternative to LSTMs, achieving comparable performance with fewer parameters and less computational complexity. The choice between these architectures depends on factors such as the specific task, dataset size, computational resources, and desired trade-offs between model complexity and performance.

2.4.4. Convolutional Recurrent Neural Networks

ConvLSTM [174] and Convolutional GRU (ConvGRU) [175] are both specialized architectures designed to process spatiotemporal data efficiently, each with its unique advantages. ConvLSTM built on the mechanism of Fully Connected LSTM (FC-LSTM) [176] by incorporating convolutional operations, maintaining a memory cell and gates to manage the flow of data. This architecture is particularly effective in capturing long-term dependencies and precise temporal relationships in sequences, making it well-suited for tasks such as video prediction and action recognition where modeling complex dynamics is essential. ConvLSTM cell is given in Figure 2.16, and ConvGRU in Figure 2.17.

Equation for ConvLSTM cell are given for forget gate, which determines amount of the previous cell state to retain or forget, in (2.69),

$$\mathbf{F}[n] = \text{sigmoid}(\boldsymbol{b}_{\mathbf{F}} + \boldsymbol{W}_{\mathbf{UF}} \star \mathbf{U}[n] + \boldsymbol{W}_{\mathbf{HF}} \star \mathbf{H}[n-1] + \boldsymbol{W}_{\mathbf{CF}} \odot \mathbf{C}[n-1]) \quad (2.69)$$

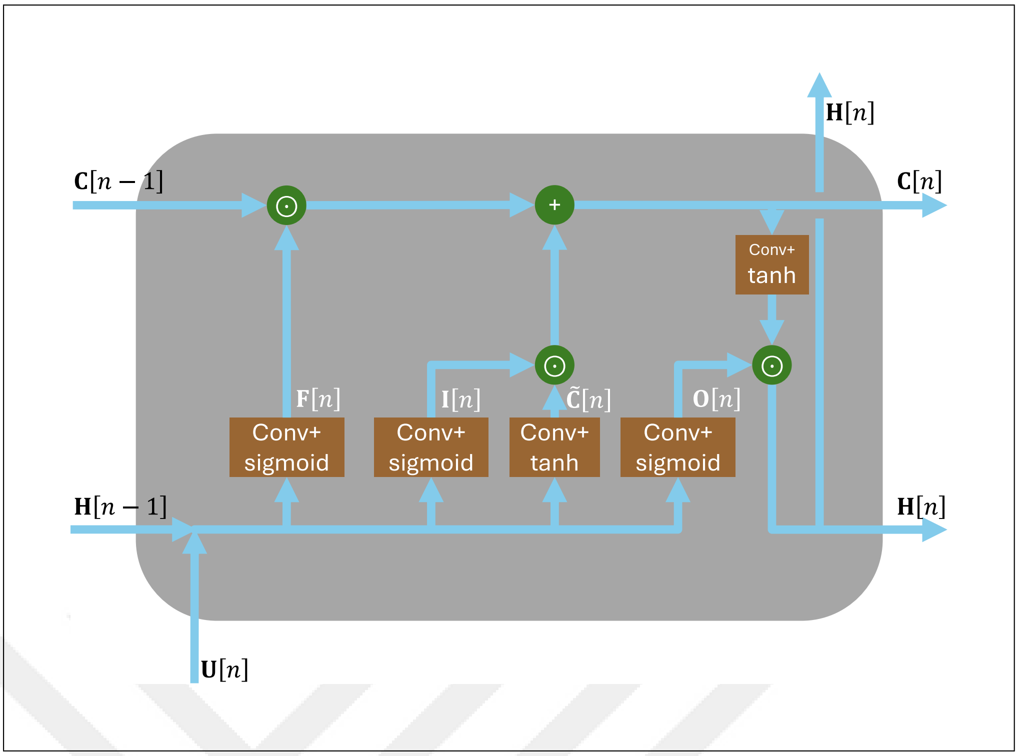


Figure 2.16: ConvLSTM cell.

Input gate, controls amount new information should be stored in cell state, in (2.70),

$$\mathbf{I}[n] = \text{sigmoid}(\boldsymbol{b}_{\mathbf{I}} + \boldsymbol{W}_{\mathbf{UI}} \star \mathbf{U}[n] + \boldsymbol{W}_{\mathbf{HI}} \star \mathbf{H}[n-1] + \boldsymbol{W}_{\mathbf{CI}} \odot \mathbf{C}[n-1]) \qquad (2.70)$$

Candidate cell state, represents the new information to be added to cell state, in (2.71),

$$\tilde{\mathbf{C}}[n] = \tanh(\boldsymbol{b}_{\mathbf{C}} + \boldsymbol{W}_{\mathbf{UC}} \star \mathbf{U}[n] + \boldsymbol{W}_{\mathbf{HC}} \star \mathbf{H}[n-1]) \qquad (2.71)$$

Update cell state, which is updated by combining the previous cell state with the new candidate cell state with forget gate controlling amount of the previous state to retain and the input gate controlling amount new information to add, in (2.72),

$$\mathbf{C}[n] = \mathbf{F}[n] \odot \mathbf{C}[n-1] + \mathbf{I}[n] \odot \tilde{\mathbf{C}}[n] \qquad (2.72)$$

Output gate, which controls amount of the updated cell state should be output as the hidden state, in (2.73),

$$\mathbf{O}[n] = \text{sigmoid}(\boldsymbol{b}_{\mathbf{O}} + \boldsymbol{W}_{\mathbf{UO}} \star \mathbf{U}[n] + \boldsymbol{W}_{\mathbf{HO}} \star \mathbf{H}[n-1] + \boldsymbol{W}_{\mathbf{CO}} \odot \mathbf{C}[n-1]) \quad (2.73)$$

Hidden state image in (2.74),

$$\mathbf{H}[n] = \mathbf{O}[n] \odot \tanh(\mathbf{C}[n]) \quad (2.74)$$

where $\mathbf{U}[n]$ input image at sample step n. Also, $\boldsymbol{W}_{\mathbf{UF}}$, $\boldsymbol{W}_{\mathbf{HF}}$, $\boldsymbol{W}_{\mathbf{CF}}$, $\boldsymbol{W}_{\mathbf{UI}}$, $\boldsymbol{W}_{\mathbf{HI}}$, $\boldsymbol{W}_{\mathbf{CI}}$, $\boldsymbol{W}_{\mathbf{UC}}$, $\boldsymbol{W}_{\mathbf{HC}}$, $\boldsymbol{W}_{\mathbf{UO}}$, $\boldsymbol{W}_{\mathbf{HO}}$, $\boldsymbol{W}_{\mathbf{CO}}$ are weight matrices, and $\boldsymbol{b}_{\mathbf{F}}$, $\boldsymbol{b}_{\mathbf{I}}$, $\boldsymbol{b}_{\mathbf{C}}$, $\boldsymbol{b}_{\mathbf{O}}$ bias vectors which are all learned during training. The operator $\odot$ element-wise multiplication.

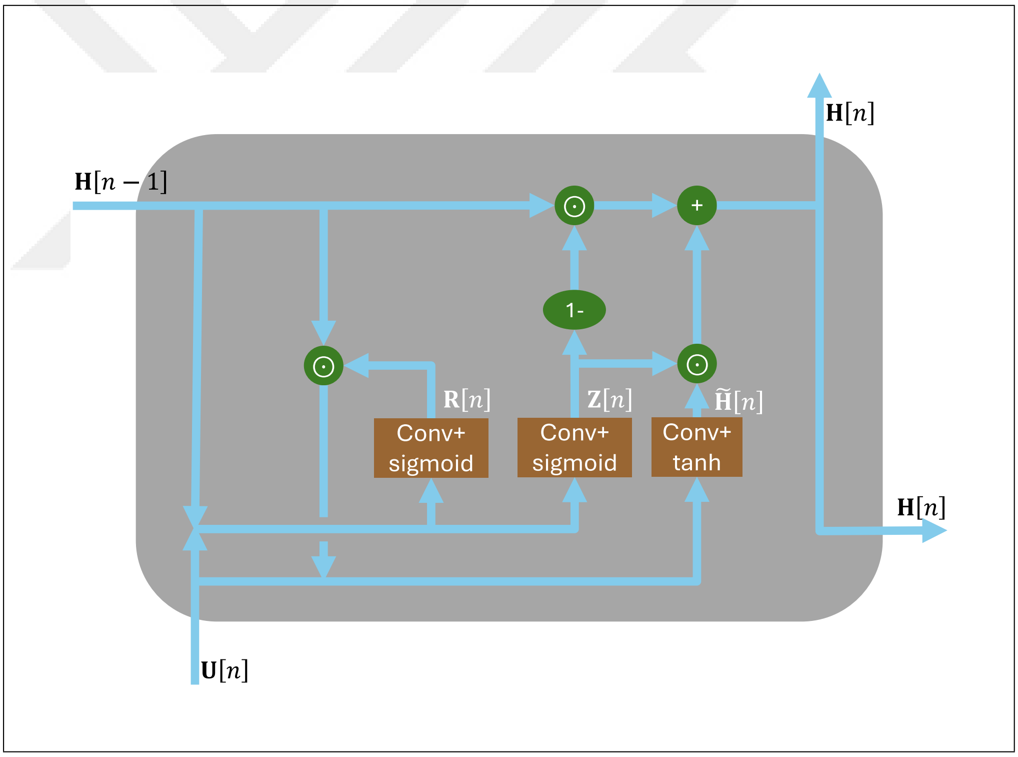


Figure 2.17: ConvGRU cell.

On the other hand, ConvGRU introduces convolutional operations into the GRU architecture, utilizing reset and update gates to control information flow. Compared to ConvLSTM, ConvGRU typically offers a simpler computational structure, making it more efficient for training and inference. This efficiency makes ConvGRU a preferred choice for applications where computational resources are limited, while still maintaining the capability to capture spatial and temporal dependencies effectively, such as video processing and spatiotemporal data analysis.

Equation for ConvGRU cell are given for update gate, which controls amount of the past information to keep and amount of the new input to let through, in (2.75),

$$\mathbf{Z}[n] = \text{sigmoid}(\boldsymbol{b}_{\mathbf{Z}} + \boldsymbol{W}_{\mathbf{UZ}} \star \mathbf{U}[n] + \boldsymbol{W}_{\mathbf{HZ}} \star \mathbf{H}[n-1]) \tag{2.75}$$

reset gate, which controls amount of the past information to forget, in (2.76),

$$\mathbf{R}[n] = \text{sigmoid}(\boldsymbol{b}_{\mathbf{R}} + \boldsymbol{W}_{\mathbf{UR}} \star \mathbf{U}[n] + \boldsymbol{W}_{\mathbf{HR}} \star \mathbf{H}[n-1]) \tag{2.76}$$

candidate activation, which computes the new candidate hidden state, in (2.77),

$$\tilde{\mathbf{H}}[n] = \tanh(\boldsymbol{b}_{\mathbf{H}} + \boldsymbol{W}_{\mathbf{UH}} \star \mathbf{U}[n] + \mathbf{R}[n] \odot (\boldsymbol{W}_{\mathbf{HH}} \star \mathbf{H}[n-1])) \tag{2.77}$$

hidden state image in (2.78),

$$\mathbf{H}[n] = (1 - \mathbf{Z}[n]) \odot \mathbf{H}[n-1] + \mathbf{Z}[n] \odot \tilde{\mathbf{H}}[n] \tag{2.78}$$

where $\mathbf{U}[n]$ input image at sample step n. Also, $\boldsymbol{W}_{\mathbf{UR}}$, $\boldsymbol{W}_{\mathbf{HR}}$, $\boldsymbol{W}_{\mathbf{UZ}}$, $\boldsymbol{W}_{\mathbf{HZ}}$, $\boldsymbol{W}_{\mathbf{UH}}$, $\boldsymbol{W}_{\mathbf{HH}}$ are weight filters, and $\boldsymbol{b}_{\mathbf{R}}$, $\boldsymbol{b}_{\mathbf{Z}}$, $\boldsymbol{b}_{\mathbf{H}}$ bias vectors which are all learned during training. The operator $\star$ convolution and $\odot$ element-wise multiplication.

Ultimately, the choice of ConvLSTM and ConvGRU changes with specific requirements of the task. ConvLSTM may be preferred for tasks that demand precise modeling of long-term dependencies, while ConvGRU might be favored in scenarios where computational efficiency is a primary concern without sacrificing performance in capturing spatiotemporal patterns. Both architectures have demonstrated effectiveness in various applications, offering flexibility in addressing different challenges in processing sequential data.

2.5. Localization

Localization stands as a pivotal task within security applications, aiming to ascertain the position of a target [37]. It distinctively differs from tracking, as it solely focuses on determining estimated positions of targets over time. Tracking, in applications like ours, can be regarded as a post-processing step following localization.

In the literature, seismic sensors have been employed for localizing two distinct types of targets: vehicle-pedestrian interactions [29], [59], [147], [177] and earthquake-like seismic events [152], [178].

Various methods are used specifically for the SITEX02 dataset to determine location, including common range-based approaches such as Received Signal Strength Index (RSSI), Angle of Arrival (AOA), Time Difference of Arrival (TDOA), Direction of Arrival (DOA) and Time of Arrival (TOA). Boettcher et al. introduced the TDOA algorithm for acoustic localization using the SITEX00 dataset [63], [64]. Li et al. suggested a target localization algorithm based on energy, assuming isotropic exponential attenuation for the energy source. They created circles from energy readings ratios, converting the problem into a nonlinear least squares issue [61], [179]. When a passive infrared sensor detects a target, its location is determined at the intersection of the PIR sensor's line-of-sight and the road. This technique was applied in SITEX02 by Ashraf et al. [180]. Boettcher et al. integrated panoramic cameras with acoustic bearing estimation for localization [65]. Brooks et al. used differences in detection time and activated sensor locations to estimate the target's position [66]. Sheng et al. proposed the CFAR to detect events in small sensor areas with collaborative signal processing, using an acoustic energy-based method to locate the target [67].Duarte et al. proposed a decision fusion technique based on distance, which leverages the correlation between sensor-to-target distance, signal-to-noise ratio, and classification rate. This approach necessitates reduced communication and attains superior classification rates for regions compared to traditional fusion methods based on majority voting [68]. Rahman et al. used Optimized Maximum Likelihood for acoustic-based localization with the SITEX02 dataset [69]. In their study of the SITEX02 dataset, Le Borgne et al. explored centroid schemes and distributed regression methods for localization [70]. Sindhu et al. analyzed data from 19 nodes from the same run, and the extracted features were used to monitor and confirm when a particular vehicle has passed a given sensor node [71].

It is widely acknowledged that a sensor cluster comprising at least three sensor nodes is essential for accurately estimating target position in a sensor network. Additionally, studies have demonstrated that clustering nodes can enhance classification performance. Taheri et al. utilized a time-varying autoregressive model (TVAR) alongside clustered nodes to enhance classification performance [78]. Zwartjes et al. proposed a method for adaptive learning to classify military vehicles named QUantile Estimation after Supervised Training (QUEST) for [181].

In this section, we introduce triangulation and trilateration, two methods frequently employed in localization. We also derive their equations.

2.5.1. Triangulation

Triangulation is AOA based localization method as we can see from Figure 2.18. We require minimum 2 sensor positions, and these sensors must measure angle of target.

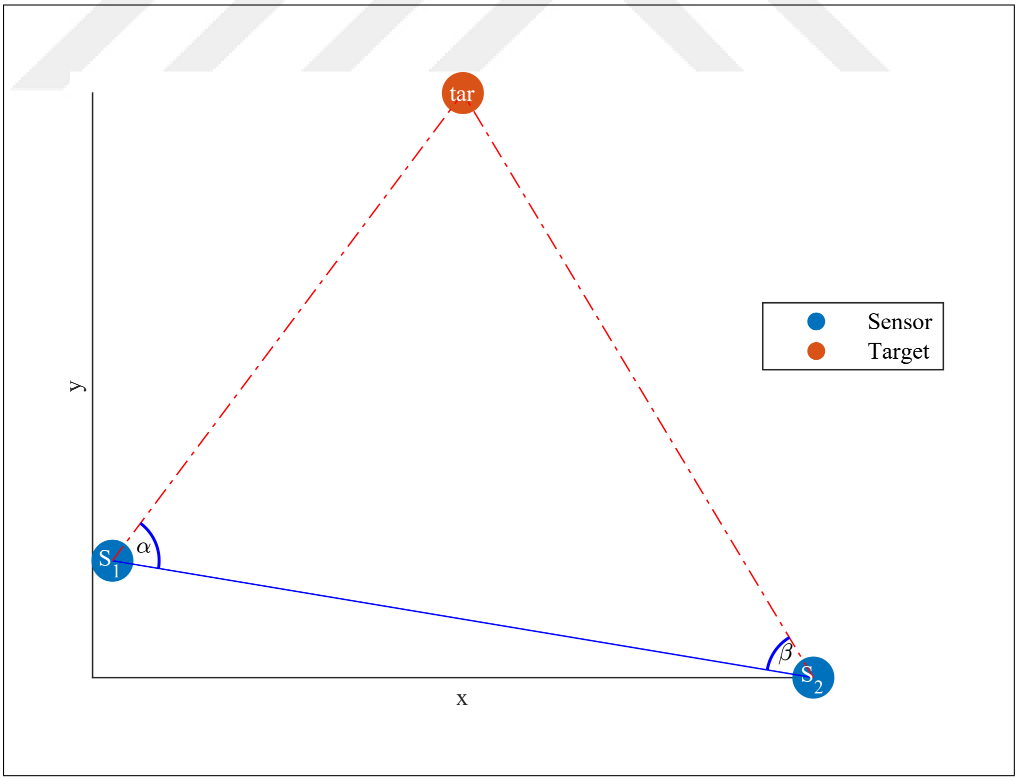


Figure 2.18: Triangulation.

The relation between actual target position $\boldsymbol{x}_{\text{tar}} = [x_{\text{tar}}, y_{\text{tar}}]^{\text{T}}$ and the sensor position

$\boldsymbol{x}_{s_i} = [x_{s_i}, y_{s_i}]^{\mathrm{T}}$ is given in (2.80) using law of sines in (2.79).

$$\frac{\sin(\theta_1)}{a_1} = \frac{\sin(\theta_2)}{a_2} = \frac{\sin(\theta_3)}{a_3} \tag{2.79}$$

$$\frac{\sin(\alpha)}{\mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_2}, \boldsymbol{x}_{\mathrm{tar}}\right)} = \frac{\sin(\beta)}{\mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{tar}}\right)} = \frac{\sin(\pi - \alpha - \beta)}{\mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{s}_2}\right)} \tag{2.80}$$

where α, β and $\mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{s}_2}\right)$ are known quantities as in Figure 2.18.

Following equations will $\mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{tar}}\right)$, where $\sin(\pi - \alpha - \beta) = \sin(\alpha + \beta)$.

$$\mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{tar}}\right) = \frac{\sin(\beta)}{\sin(\alpha + \beta)} \mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{s}_2}\right) \tag{2.81}$$

Then express the dot product in (2.82) and norm of the cross product in (2.83)

$$\mathrm{s}_1\vec{\mathrm{s}}_2 \cdot \mathrm{s}_1\vec{\mathrm{s}}_3 = \mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{tar}}\right) \mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{s}_2}\right) \cos(\alpha) \tag{2.82}$$

$$\|\mathrm{s}_1\vec{\mathrm{s}}_2 \times \mathrm{s}_1\vec{\mathrm{s}}_3\| = \mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{tar}}\right) \mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{s}_2}\right) \sin(\alpha) \tag{2.83}$$

Then express the dot product and norm of the cross product by using coordinate notations in (2.84) and (2.85) respectively.

$$\begin{aligned}(x_{s_2} - x_{s_1})(x_{\mathrm{tar}} - x_{s_1}) + (y_{s_2} - y_{s_1})(y_{\mathrm{tar}} - y_{s_1})\\ = \mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{tar}}\right) \mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{s}_2}\right) \cos(\alpha)\end{aligned} \tag{2.84}$$

$$\begin{aligned}(x_{s_2} - x_{s_1})(y_{\mathrm{tar}} - y_{s_2}) - (y_{s_2} - y_{s_1})(x_{\mathrm{tar}} - x_{s_2})\\ = \mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{tar}}\right) \mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{s}_2}\right) \sin(\alpha)\end{aligned} \tag{2.85}$$

We can do replacement in (2.81) for (2.84) and (2.85). Then we will obtain (2.86) and (2.87) respectively.

$$\begin{aligned}(x_{s_2} - x_{s_1})(x_{\mathrm{tar}} - x_{s_1}) + (y_{s_2} - y_{s_1})(y_{\mathrm{tar}} - y_{s_1})\\ = \frac{\sin(\beta)\cos(\alpha)}{\sin(\alpha + \beta)} \mathrm{d}_2\left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{s}_2}\right)^2\end{aligned} \tag{2.86}$$

$$\begin{aligned} &(x_{s_2} - x_{s_1})(y_{\text{tar}} - y_{s_2}) - (y_{s_2} - y_{s_1})(x_{\text{tar}} - x_{s_2}) \\ &\qquad = \frac{\sin(\beta)\sin(\alpha)}{\sin(\alpha+\beta)} \mathrm{d}_2 \left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{s}_2}\right)^2 \end{aligned} \tag{2.87}$$

Equations (2.86) and (2.87) can form an equation system, then solution will lead us to $\boldsymbol{x}_{\text{tar}}$.

$$\begin{aligned} &x_{\text{tar}}(x_{s_2} - x_{s_1}) + y_{\text{tar}}(y_{s_2} - y_{s_1}) \\ &= x_{s_1}(x_{s_2} - x_{s_1}) + y_{s_1}(y_{s_2} - y_{s_1}) + \frac{\sin(\beta)\cos(\alpha)}{\sin(\alpha+\beta)} \mathrm{d}_2 \left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{s}_2}\right)^2 \end{aligned} \tag{2.88}$$

$$\begin{aligned} &-x_{\text{tar}}(y_{s_2} - y_{s_1}) + y_{\text{tar}}(x_{s_2} - x_{s_1}) \\ &= y_{s_2}(x_{s_2} - x_{s_1}) - x_{s_2}(y_{s_2} - y_{s_1}) + \frac{\sin(\beta)\sin(\alpha)}{\sin(\alpha+\beta)} \mathrm{d}_2 \left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{s}_2}\right)^2 \end{aligned} \tag{2.89}$$

The equations above can be converted to matrix form $\boldsymbol{A}\boldsymbol{x}_{\text{tar}} = \boldsymbol{b}$, where $\boldsymbol{x}_{\text{tar}} = [x_{\text{tar}}, y_{\text{tar}}]^{\mathrm{T}}$ using (2.90) and (2.91).

$$\boldsymbol{A} = \begin{bmatrix} (x_{s_2} - x_{s_1}) & (y_{s_2} - y_{s_1}) \\ (y_{s_1} - y_{s_2}) & (x_{s_2} - x_{s_1}) \end{bmatrix} \tag{2.90}$$

$$\boldsymbol{b} = \begin{bmatrix} x_{s_1}(x_{s_2} - x_{s_1}) + y_{s_1}(y_{s_2} - y_{s_1}) + \frac{\sin(\beta)\cos(\alpha)}{\sin(\alpha+\beta)} \mathrm{d}_2 \left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{s}_2}\right)^2 \\ y_{s_2}(x_{s_2} - x_{s_1}) - x_{s_2}(y_{s_2} - y_{s_1}) + \frac{\sin(\beta)\sin(\alpha)}{\sin(\alpha+\beta)} \mathrm{d}_2 \left(\boldsymbol{x}_{\mathrm{s}_1}, \boldsymbol{x}_{\mathrm{s}_2}\right)^2 \end{bmatrix} \tag{2.91}$$

Sensor locations x_{s_i} and y_{s_i} are known variables. If sensor-target angles α and β are known, target position can be found as follows.

$$\boldsymbol{x}_{\text{tar}} = \boldsymbol{A}^{-1}\boldsymbol{b} \tag{2.92}$$

Equation (2.92) is valid to acquire $\boldsymbol{x}_{\text{tar}} = [x_{\text{tar}}, y_{\text{tar}}]^{\mathrm{T}}$.

2.5.2. Trilateration

Trilateration is as a fundamental method for range-based localization. Circular trilateration, illustrated in Figure 2.19, involves three sensors, namely S_1, S_2, and S_3, situated at distinct locations. For example, TOA, TDOA or RSS can be employed to estimate the location of the target tar by processing the signals from these sensor nodes [36].

Actual target position $(x_{\text{tar}}, y_{\text{tar}})$ and the sensor position (x_{s_i}, y_{s_i}) relation equation is given in (2.93)

$$(x_{s_i} - x_{\text{tar}})^2 + (y_{s_i} - y_{\text{tar}})^2 = r_i^2 \tag{2.93}$$

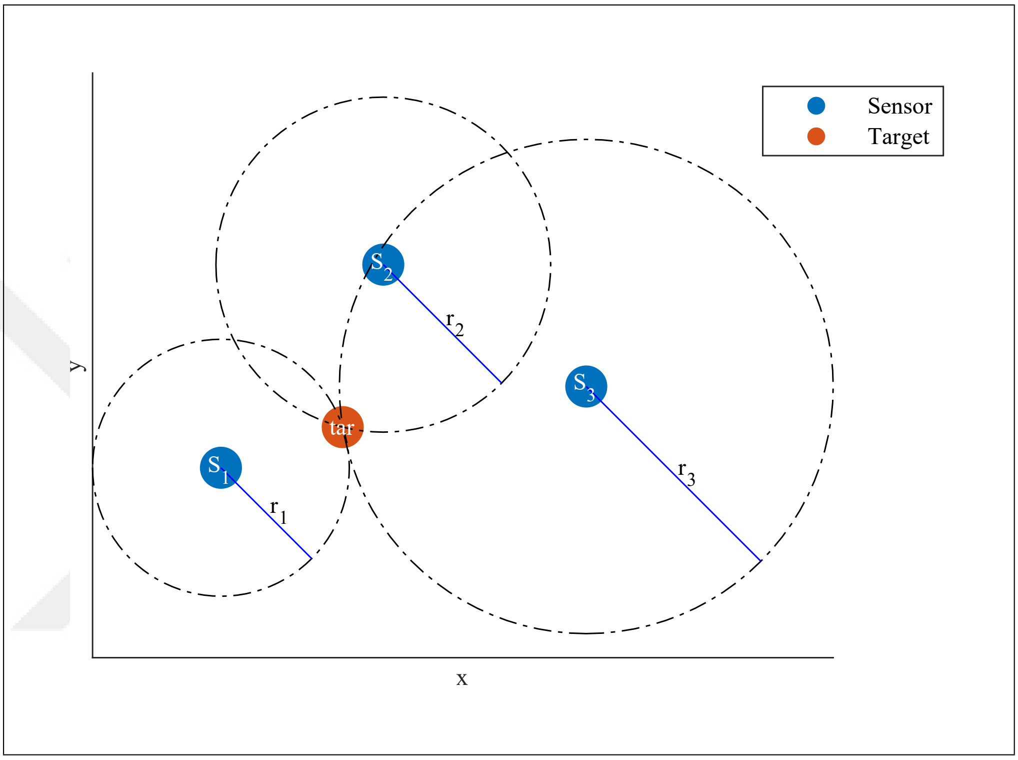


Figure 2.19: Trilateration.

where r_i is sensor-target distance and $i = 1, 2, 3$. Difference between r_i^2 and r_j^2 can be used to eliminate the squares of x_{tar} and y_{tar} as given in Eq (2.94) and (2.95).

$$(x_{s_i} - x_{\text{tar}})^2 + (y_{s_i} - y_{\text{tar}})^2 - \left(x_{s_j} - x_{\text{tar}}\right)^2 - \left(y_{s_j} - y_{\text{tar}}\right)^2 = r_i^2 - r_j^2 \tag{2.94}$$

$$x_{\text{tar}} \left(2x_{s_j} - 2x_{s_i}\right) + y_{\text{tar}} \left(2y_{s_j} - 2y_{s_i}\right) = r_i^2 - r_j^2 - (x_{s_i}^2 + y_{s_i}^2) + (x_{s_j}^2 + y_{s_j}^2) \tag{2.95}$$

where $i, j = 1, 2, 3$ and $j \neq i$. The equations above can be converted to matrix form $\boldsymbol{A}\boldsymbol{x}_{\text{tar}} = \boldsymbol{b}$, where $\boldsymbol{x}_{\text{tar}} = [x_{\text{tar}}, y_{\text{tar}}]^{\text{T}}$ and

$$\boldsymbol{A} = 2\begin{bmatrix} (x_{s_2} - x_{s_1}) & (y_{s_2} - y_{s_1}) \\ (x_{s_3} - x_{s_1}) & (y_{s_3} - y_{s_1}) \\ (x_{s_3} - x_{s_2}) & (y_{s_3} - y_{s_2}) \end{bmatrix} \tag{2.96}$$

$$\boldsymbol{b} = \begin{bmatrix} r_1^2 - r_2^2 - (x_{s_1}^2 + y_{s_1}^2) + (x_{s_2}^2 + y_{s_2}^2) \\ r_1^2 - r_3^2 - (x_{s_1}^2 + y_{s_1}^2) + (x_{s_3}^2 + y_{s_3}^2) \\ r_2^2 - r_3^2 - (x_{s_2}^2 + y_{s_2}^2) + (x_{s_3}^2 + y_{s_3}^2) \end{bmatrix} \tag{2.97}$$

Sensor locations (x_{s_i}, y_{s_i}) are known information. If also we know r_i, (2.98) is valid to acquire $\boldsymbol{x}_{\text{tar}} = [x_{\text{tar}}, y_{\text{tar}}]^{\text{T}}$ position under the assumption that matrix $\boldsymbol{A}^{\text{T}}\boldsymbol{A}$ is invertible.

$$\boldsymbol{x}_{\text{tar}} = \boldsymbol{A}^{+}\boldsymbol{b} = \left(\left(\boldsymbol{A}^{\text{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{A}^{\text{T}}\right)\boldsymbol{b} \tag{2.98}$$

2.6. Tracking

Tracking is the process of refining the estimated positions of targets derived from GPS data or various localization algorithms. This section introduces fundamental tracking algorithms that are critical for enhancing the accuracy of these estimations.

The first algorithm we discuss is the Alpha-Beta(-Gamma) ($\alpha - \beta(-\gamma)$) Filter. This filter is extensively used in fields such as radar tracking, navigation systems, and other applications that require precise estimation of position, velocity, and acceleration. Its popularity stems from its simplicity and effectiveness. The $\alpha - \beta(-\gamma)$ filter operates on principles similar to those of the Kalman Filter, providing a reliable method for tracking and predicting the movements of dynamic systems. By incorporating acceleration into its calculations, this filter improves both accuracy and responsiveness, making it suitable for a wide range of applications.

Next, we examine the Kalman Filter [182]. In contrast to the $\alpha - \beta(-\gamma)$ Filter, the Kalman Filter models measurements, current state estimations, and future state estimations as normally distributed random variables, defined by their mean and variance. This probabilistic approach enables the Kalman Filter to achieve higher accuracy in its estimations compared to the $\alpha - \beta(-\gamma)$ Filter. Moreover, there are advanced versions of the Kalman Filter, such as the Extended Kalman Filter [183] and the Unscented Kalman Filter [184], which offer enhanced performance in handling non-linear systems.

2.6.1. Alpha-Beta(-Gamma) Filter

The $\alpha - \beta(-\gamma)$ Filter is an extension of the Alpha-Beta ($\alpha - \beta$) filter, which is used for smoothing time series in dynamic systems. It incorporates an additional term to account for acceleration, improving the accuracy of the estimates.

The main idea of filter starts from averaging N previous measurements of $\mathbf{x}[n]$ as in (2.99), where $n, N \in \mathbb{N}$. We need to convert mean equation (2.99) to a transitional form from $\mathbf{x}_{\text{flt}}[n-1|n-1]$ to $\mathbf{x}_{\text{flt}}[n|n]$ for our needs. For this, equation (2.99) can be transformed to (2.104), using the steps (2.100)-(2.103).

$$\mathbf{x}_{\text{flt}}[N|N] = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}[n]) \tag{2.99}$$

$$= \frac{1}{N}\sum_{n=1}^{N-1}(\mathbf{x}[n]) + \frac{1}{N}\mathbf{x}[N] \tag{2.100}$$

$$= \frac{N-1}{N}\frac{1}{N-1}\sum_{n=1}^{N-1}(\mathbf{x}[n]) + \frac{1}{N}\mathbf{x}[N] \tag{2.101}$$

$$= \frac{N-1}{N}\mathbf{x}_{\text{flt}}[N-1|N-1] + \frac{1}{N}\mathbf{x}[N] \tag{2.102}$$

$$= \mathbf{x}_{\text{flt}}[N-1|N-1] - \frac{1}{N}\mathbf{x}_{\text{flt}}[N-1|N-1] + \frac{1}{N}\mathbf{x}[N] \tag{2.103}$$

$$\mathbf{x}_{\text{flt}}[N|N] = \mathbf{x}_{\text{flt}}[N-1|N-1] + \frac{1}{N}(\mathbf{x}[N] - \mathbf{x}_{\text{flt}}[N-1|N-1]) \tag{2.104}$$

The filter equations are based on the state variables: position $\mathbf{x}[n]$, velocity $\dot{\mathbf{x}}[n]$, and acceleration $\ddot{\mathbf{x}}[n]$. The motion equations for the state variables for constant position:

$$\mathbf{x}[n+1] = \mathbf{x}[n] \tag{2.105}$$

Constant velocity:

$$\mathbf{x}[n+1] = \mathbf{x}[n] + \dot{\mathbf{x}}[n]\Delta t \tag{2.106}$$

$$\dot{\mathbf{x}}[n+1] = \dot{\mathbf{x}}[n] \tag{2.107}$$

And for constant acceleration:

$$\mathbf{x}[n+1] = \mathbf{x}[n] + \dot{\mathbf{x}}[n]\Delta t + \frac{1}{2}\ddot{\mathbf{x}}[n]\Delta t^2 \tag{2.108}$$

$$\dot{\mathbf{x}}[n+1] = \dot{\mathbf{x}}[n] + \ddot{\mathbf{x}}[n]\Delta t \tag{2.109}$$

$$\ddot{\mathbf{x}}[n+1] = \ddot{\mathbf{x}}[n] \tag{2.110}$$

First case is estimation under constant position with noisy measurement conditions. This makes predicted state equal to estimated state as in (2.111) from (2.105). Equation (2.104) becomes (2.112), which is called state update equation.

$$\mathbf{x}_{\text{flt}}[N|N-1] = \mathbf{x}_{\text{flt}}[N-1|N-1] \tag{2.111}$$

$$\mathbf{x}_{\text{flt}}[N|N] = \mathbf{x}_{\text{flt}}[N|N-1] + \frac{1}{N}(\mathbf{x}[N] - \mathbf{x}_{\text{flt}}[N|N-1]) \tag{2.112}$$

The constant $\frac{1}{N}$ in equation (2.112) is called filter gain, and for generalization, we will call it α. And the term $(\mathbf{x}[N] - \mathbf{x}_{\text{flt}}[N|N-1])$ is called innovation, because it contains new information. Alpha (α) filter state update equation will be (2.113), where we use n instead on N to make it generic.

$$\mathbf{x}_{\text{flt}}[n|n] = \mathbf{x}_{\text{flt}}[n|n-1] + \alpha(\mathbf{x}[n] - \mathbf{x}_{\text{flt}}[n|n-1]) \tag{2.113}$$

Second case is estimation under constant velocity with noisy measurement conditions.

This makes predicted velocity state equal to estimated velocity state as in (2.115) from (2.107). Also position state will be (2.114) from (2.106).

$$
\begin{aligned}
\mathbf{x}_{\text{flt}}[N|N-1] &= \mathbf{x}_{\text{flt}}[N-1|N-1] + \Delta t \dot{\mathbf{x}}_{\text{flt}}[N-1|N-1] && (2.114)\\
\dot{\mathbf{x}}_{\text{flt}}[N|N-1] &= \dot{\mathbf{x}}_{\text{flt}}[N-1|N-1] && (2.115)
\end{aligned}
$$

Alpha-Beta ($\alpha - \beta$) filter state update equations will be (2.116) and (2.117). The gain parameters α and β control the extent of the correction for position and velocity.

$$
\begin{aligned}
\mathbf{x}_{\text{flt}}[n|n] &= \mathbf{x}_{\text{flt}}[n|n-1] + \alpha(\mathbf{x}[n] - \mathbf{x}_{\text{flt}}[n|n-1]) && (2.116)\\
\dot{\mathbf{x}}_{\text{flt}}[n|n] &= \dot{\mathbf{x}}_{\text{flt}}[n|n-1] + \beta \left(\frac{\mathbf{x}[n] - \mathbf{x}_{\text{flt}}[n|n-1]}{\Delta t} \right) && (2.117)
\end{aligned}
$$

Third case is estimation under constant acceleration with noisy measurement conditions. This makes predicted acceleration state equal to estimated acceleration state as in (2.120) from (2.110). Also position state will be (2.118) from (2.108), and velocity state will be (2.119) from (2.109).

$$
\begin{aligned}
\mathbf{x}_{\text{flt}}[N|N-1] =& \mathbf{x}_{\text{flt}}[N-1|N-1] + \Delta t \dot{\mathbf{x}}_{\text{flt}}[N-1|N-1] && (2.118)\\
&+ \frac{1}{2} \Delta t^2 \ddot{\mathbf{x}}_{\text{flt}}[N-1|N-1]\\
\dot{\mathbf{x}}_{\text{flt}}[N|N-1] =& \dot{\mathbf{x}}_{\text{flt}}[N-1|N-1] + \Delta t \ddot{\mathbf{x}}_{\text{flt}}[N-1|N-1] && (2.119)\\
\ddot{\mathbf{x}}_{\text{flt}}[N|N-1] =& \ddot{\mathbf{x}}_{\text{flt}}[N-1|N-1] && (2.120)
\end{aligned}
$$

Alpha-Beta(-Gamma) ($\alpha - \beta(-\gamma)$) filter state update equations based on the measurement $\mathbf{x}[n]$ will be (2.121), (2.122) and (2.123). The gain parameters α, β, and γ control the extent of the correction for position, velocity, and acceleration.

$$\mathbf{x}_{\text{flt}}[n|n] = \mathbf{x}_{\text{flt}}[n|n-1] + \alpha(\mathbf{x}[n] - \mathbf{x}_{\text{flt}}[n|n-1]) \tag{2.121}$$

$$\dot{\mathbf{x}}_{\text{flt}}[n|n] = \dot{\mathbf{x}}_{\text{flt}}[n|n-1] + \beta\left(\frac{\mathbf{x}[n] - \mathbf{x}_{\text{flt}}[n|n-1]}{\Delta t}\right) \tag{2.122}$$

$$\ddot{\mathbf{x}}_{\text{flt}}[n|n] = \ddot{\mathbf{x}}_{\text{flt}}[n|n-1] + \gamma\left(\frac{\mathbf{x}[n] - \mathbf{x}_{\text{flt}}[n|n-1]}{\Delta t^2}\right) \tag{2.123}$$

The gain parameters α, β, and γ are chosen to optimize the filter performance. They determine the responsiveness and stability of the filter. These values can be tuned according to the specific application and desired performance. Typical values can be; α: 0.2 to 0.85, β: 0.1 to 0.5, γ: 0.01 to 0.1.

2.6.2. Kalman Filter

The Kalman Filter is more advanced method used for smoothing time series in dynamic systems. Measurements, current state estimation, and next state estimation are normally distributed random variables for Kalman Filter, which are described by mean and variance. It has two steps: prediction and correction.

Prediction step involves state extrapolation equation and covariance extrapolation equation. State extrapolation is given in (2.124), where $\mathbf{x}_{\text{flt}}[n+1|n]$ predicted system state at $n+1$, $\mathbf{x}_{\text{flt}}[n|n]$ estimated system state at n, $\mathbf{u}[n]$ measurable input variable, $\mathbf{w}[n]$ unmeasurable process noise variable, $\boldsymbol{F}$ transition matrix, $\boldsymbol{G}$ input transition or control matrix.

$$\mathbf{x}_{\text{flt}}[n+1|n] = \boldsymbol{F}\mathbf{x}_{\text{flt}}[n|n] + \boldsymbol{G}\mathbf{u}[n] + \mathbf{w}[n] \tag{2.124}$$

Second prediction step equation is covariance extrapolation equation as given in (2.125), where $\mathbf{P}[n|n]$ is covariance matrix of the current state given in (2.126), $\mathbf{P}[n+1|n]$ covariance matrix of next state, $\boldsymbol{F}$ transition matrix, $\boldsymbol{Q}$ process noise matrix.

$$\mathbf{P}[n+1|n] = \boldsymbol{F}\mathbf{P}[n|n]\boldsymbol{F}^{\mathrm{T}} + \boldsymbol{Q} \tag{2.125}$$

$$\mathbf{P}[n|n] = \mathrm{E}\left[(\mathbf{x}_{\text{flt}}[n|n] - \mu_{\mathbf{x}_{\text{flt}}[n|n]})(\mathbf{x}_{\text{flt}}[n|n] - \mu_{\mathbf{x}_{\text{flt}}[n|n]})^{\mathrm{T}}\right] \tag{2.126}$$

$$\mathbf{Q}[n] = \mathrm{E}\left[\mathbf{w}[n]\mathbf{w}^{\mathrm{T}}[n]\right] \tag{2.127}$$

If we consider 2-D controlled and measured acceleration scenario without process noise, equations (2.128), (2.129) and (2.130) will be estimated state, input variable, transition matrix and control matrix respectively. Equation (2.132) will be predicted system state. The result will be kinematic equations under zero initial values.

$$\mathbf{x}_{\text{flt}}[n|n] = \begin{bmatrix} x_{\text{flt}}[n|n] & y_{\text{flt}}[n|n] & \dot{x}_{\text{flt}}[n|n] & \dot{y}_{\text{flt}}[n|n] \end{bmatrix}^{\mathrm{T}} \tag{2.128}$$

$$\mathbf{u}[n] = \begin{bmatrix} \ddot{x}[n] & \ddot{y}[n] \end{bmatrix}^{\mathrm{T}} \tag{2.129}$$

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.130}$$

$$\mathbf{G} = \begin{bmatrix} \frac{1}{2}\Delta t^2 & 0 \\ 0 & \frac{1}{2}\Delta t^2 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \tag{2.131}$$

$$\mathbf{x}_{\text{flt}}[n+1|n] = \boldsymbol{F}\mathbf{x}_{\text{flt}}[n|n] + \boldsymbol{G}\mathbf{u}[n] = \\ \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{\text{flt}}[n] \\ y_{\text{flt}}[n] \\ \dot{x}_{\text{flt}}[n] \\ \dot{y}_{\text{flt}}[n] \end{bmatrix} + \begin{bmatrix} \frac{1}{2}\Delta t^2 & 0 \\ 0 & \frac{1}{2}\Delta t^2 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \begin{bmatrix} \ddot{x}[n] \\ \ddot{y}[n] \end{bmatrix} \tag{2.132}$$

Correction step involves first computing Kalman gain, then state update equation, then covariance update equation, which are given in (2.133), (2.134) and (2.135) respectively, where $\mathbf{K}[n]$ is Kalman gain, $\boldsymbol{H}$ is observation matrix-which depends on application-, $\mathbf{R}[n]$ in (2.137) is measurement covariance, $\mathbf{z}[n]$ in (2.136) is measurement, and $\mathbf{v}[n]$ is measurement noise.

$$\mathbf{K}[n] = \mathbf{P}[n|n-1]\boldsymbol{H}^{\mathrm{T}}\left(\boldsymbol{H}\mathbf{P}[n|n-1]\boldsymbol{H}^{\mathrm{T}} + \mathbf{R}[n]\right)^{-1} \tag{2.133}$$

$$\mathbf{x}_{\text{flt}}[n|n] = \mathbf{x}_{\text{flt}}[n|n-1] + \mathbf{K}[n]\left(\mathbf{z}[n] - \boldsymbol{H}\mathbf{x}_{\text{flt}}[n|n-1]\right) \tag{2.134}$$

$$\mathbf{P}[n|n] = (\boldsymbol{I} - \mathbf{K}[n]\boldsymbol{H})\mathbf{P}[n|n-1](\boldsymbol{I} - \mathbf{K}[n]\boldsymbol{H})^{\mathrm{T}} + \mathbf{K}[n]\mathbf{R}[n]\mathbf{K}^{\mathrm{T}}[n] \tag{2.135}$$

$$\mathbf{z}[n] = \boldsymbol{H}\mathbf{x}[n] + \mathbf{v}[n] \tag{2.136}$$

$$\mathbf{R}[n] = \mathrm{E}\left[\mathbf{v}[n]\mathbf{v}^{\mathrm{T}}[n]\right] \tag{2.137}$$

Process should start with initial estimates of $\mathbf{x}_{\text{flt}}[0|0]$ and $\mathbf{P}[0|0]$ then prediction step, then correction step should execute sequentially, until the end of process. At the end of each step, (2.134) will provide us filtered time series at sample n.

3. PROPOSED METHOD

This chapter is for describing our method for localization in seismic sensor networks using the SITEX02 dataset. As mentioned in Section 2.5, traditional methods like AOA, TDOA, DOA, and RSSI are difficult to use in seismic sensor networks due to the complexity of seismic signals. However, with data-driven methods, we can model the average behavior of seismic signals when a vehicle passes in front of a sensor group.

In this chapter, we will start by examining the behavior of seismic signals in Section 3.1 through the frequency domain. Then, we will further analyze the dataset through classification in Section 3.2 using the features extracted from the analysis. Classification and feature extraction are quite popular in seismic signal processing literature, allowing us to compare our results with previous studies.

Next, we will present our localization method in Section 3.3, which uses a combination of RNNs and CNNs to capture both the temporal and spectral aspects of the signal, helping us determine the position of vehicles, whether they are tracked or trackless. We will explain sensor clustering, feature extraction, and the network model in this section.

After detailing our methodology for localization, we will present our experimental results in Chapter 4.

3.1. Data Analysis

The SITEX02 dataset [57] consists of 23 nodes equipped with acoustic, seismic, and passive infrared sensors, along with deployment location information. Figure 3.1 shows deployed nodes and their locations for the SITEX02 dataset. In dataset, sampling frequency is $f_s = 4960$ Hz in 16-bit signed integer format. Data divided into time blocks of $T_b = 0.75$ s, which makes $T_b \cdot f_s = N_b = 3720$ samples for each time block. Additionally, every time block includes detection information from CFAR algorithm.

The dataset comprises nine experiments for AAV and eleven for DW, each consisting of three distinct trajectories, supported by GPS data recordings. However, there exists a lack of data for nodes 6, 60, and 61 [181]. Due to financial limitations and reliance on the SITEX02 dataset, the investigation into real-world deployment issues in seismic sensor networks was constrained.

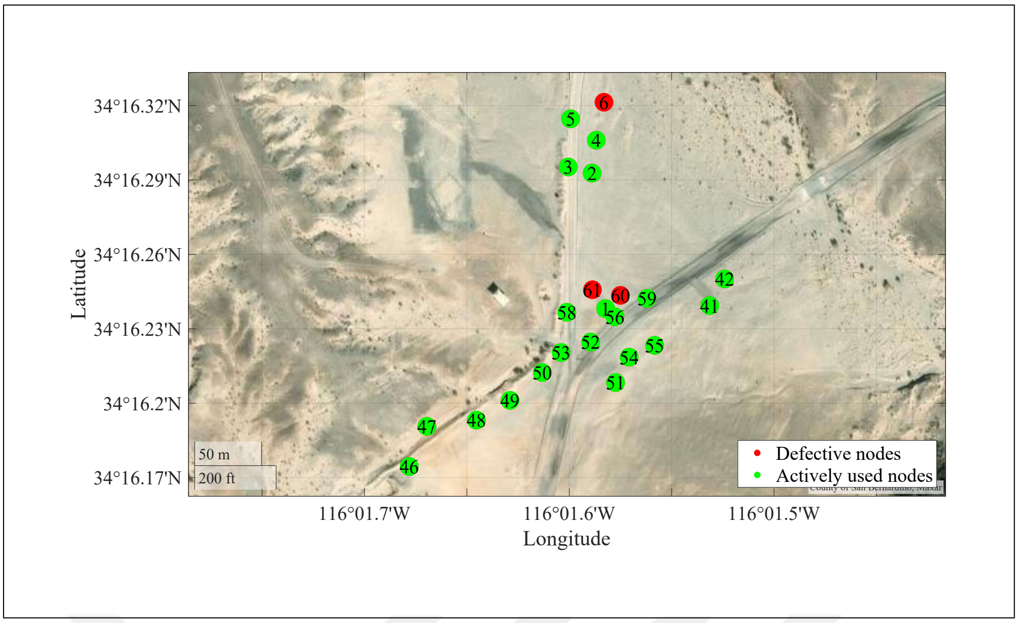
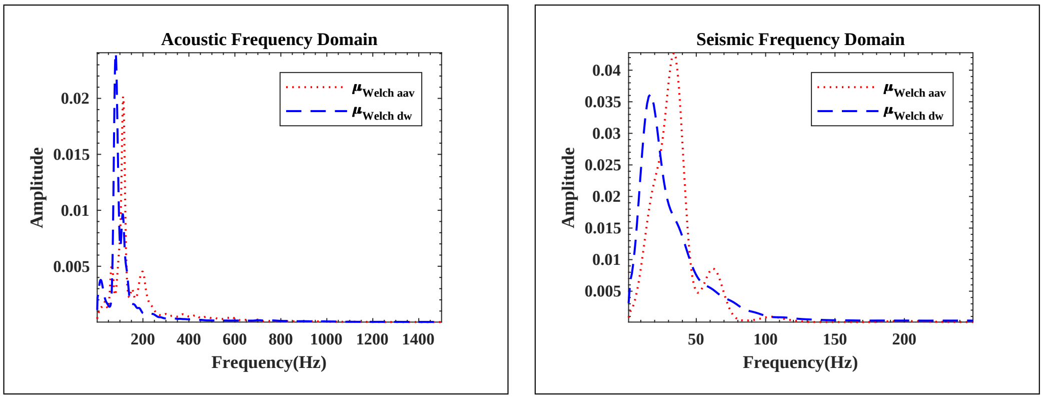


Figure 3.1: Deployed nodes and the test site for SITEX02 dataset.

Duarte et al. [57] examined the frequency response of seismic modality triggered by various events and carried out a DFT analysis focusing on frequencies as high as 484 Hz, aimed at classifying vehicles. They also extended their study to other frequency ranges. As an initial step in processing, the zero frequency, or direct current component, was removed from these signals. To reduce spectral leakage during spectrum estimation, a Hamming window was applied to segments of the time series data. After this, the signals were normalized according to their energy content within each window. In terms of vehicle classification, it was noted that acoustic signals mainly contained energy in the 0-250 Hz range, whereas seismic signals primarily had energy in the 0-100 Hz range, as shown in Figure 3.2. Frequencies outside these specific ranges were not considered further due to their low energy contributions.

Therefore, we exclusively use the MUSIC algorithm to determine the component frequencies without employing its pseudo-spectrum for discrimination. Instead, we utilize the Welch algorithm for spectrum estimation and then calculate the energy of frequency bands centered on the frequencies pinpointed by the MUSIC algorithm. Given that the Welch algorithm reduces variance in the estimated PSD by permitting segment overlap, we argue that it provides more robust features compared to the Fourier transform. This is because in the Fourier transform, both frequency and amplitude of components can vary greatly due to measurement noise.

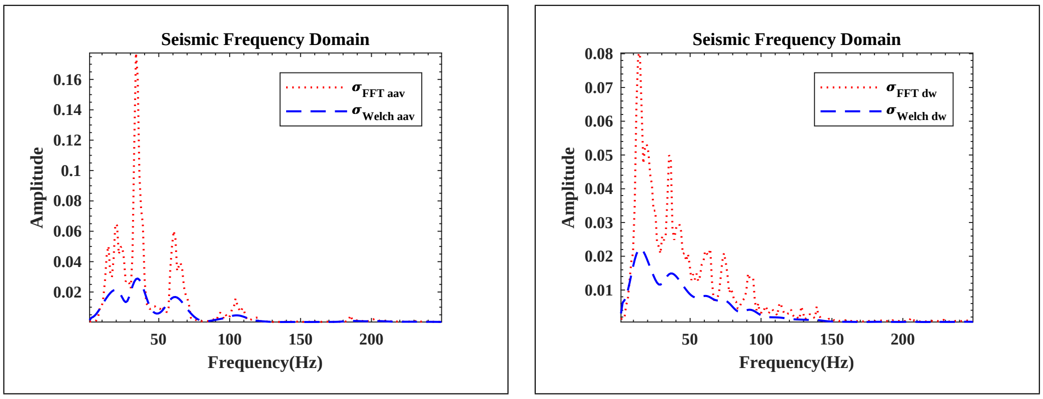


(a) Acoustic.

(b) Seismic.

Figure 3.2: Average Welch spectral densities.

This is evident in Figure 3.3. The left figure illustrates DFT magnitude and PSD estimated by the Welch algorithm for the seismic signal of an AAV vehicle, while the right figure depicts the same for a DW vehicle. Although seismic data is exclusively showcased in this illustration, similar trends are observed for acoustic signals: Fourier transform exhibits substantial standard deviation and considerable variation across the frequency spectrum for training data of the two vehicles under study. Peaks in variance coincide with frequencies of sinusoidal components, suggesting that heightened variance at these frequencies indicates less reliable features. This analysis underscores the superior reliability of PSD provided by the Welch algorithm over features derived from DFT.



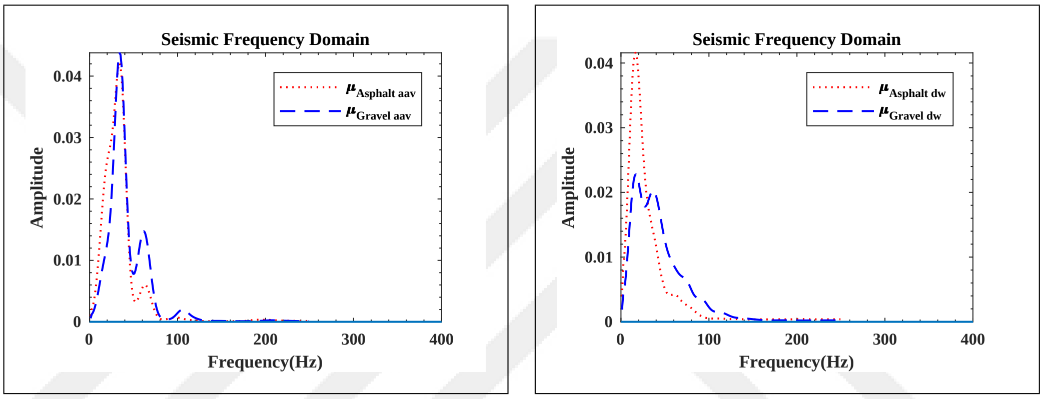
(a) AAV.

(b) DW.

Figure 3.3: Standart deviation of seismic Welch spectral densities.

Further investigation into frequency response on diverse road surfaces reveals distinct characteristics of seismic signals. Figure 3.4 illustrates spectral densities of signals for two vehicle types on asphalt and gravel roads, as inferred from satellite images in Figure 3.1. As evident, seismic signals exhibit greater energy at lower frequencies on asphalt roads and higher frequencies on gravel roads.

Additionally, a noteworthy observation is the similarity in spectral density between AAVs on asphalt roads and DWs on gravel roads, contrasting with their dissimilarity on gravel roads alone. This suggests that discriminating between these vehicle types on asphalt roads poses a greater challenge. Detailed findings are presented in Section 4.1.2.



(a) AAV.

(b) DW.

Figure 3.4: Spectral densities on asphalt and gravel roads.

3.2. Classification

In this section, we will explain the approach used in [185] because classification is crucial for analyzing data before starting the localization task. This will demonstrate our method for compressing frequency domain data to lower frequencies than the actual sampling frequency f_s.

We utilized five distinct descriptive features extracted from the time domain signals: skewness, kurtosis [123], energy, peak-to-peak value, and zero crossing density. These were combined with frequency domain features, and their individual contributions to overall performance were analyzed. To derive frequency domain features, we initially estimated the frequencies of spectral components. The MUSIC algorithm was employed

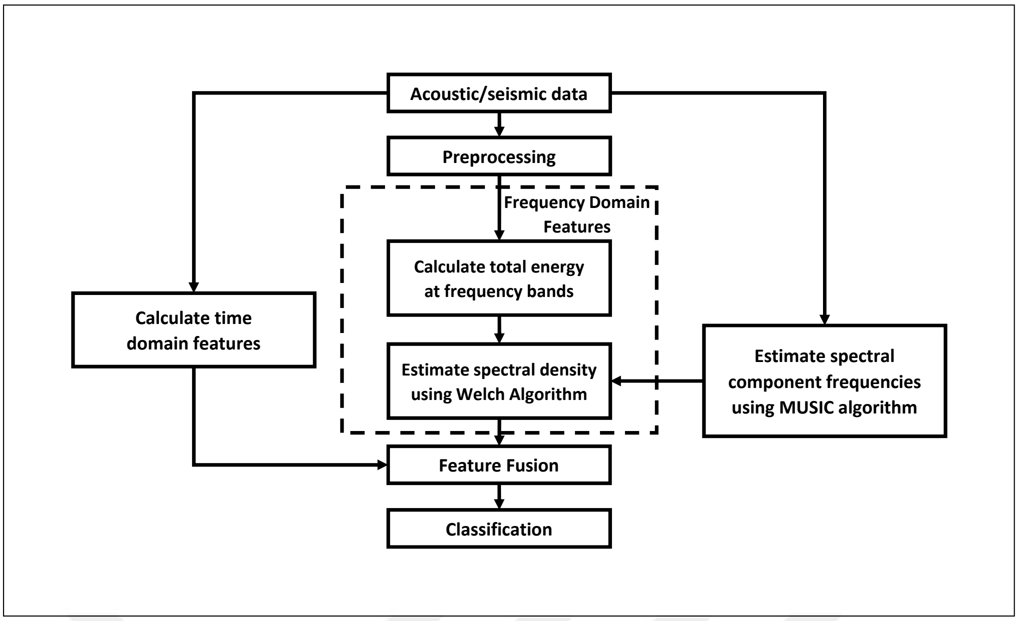

Acoustic/seismic data
Preprocessing
Frequency Domain Features
Calculate total energy at frequency bands
Estimate spectral density using Welch Algorithm
Calculate time domain features
Estimate spectral component frequencies using MUSIC algorithm
Feature Fusion
Classification

Figure 3.5: Flow graph for vehicle classification.

to identify roots and locate peaks in the estimated spectrum. The resulting acoustic frequency bands determined by the MUSIC algorithm are 6–23 Hz, 53–70 Hz, 73–89 Hz, 107–123 Hz, 125–141 Hz, 144–161 Hz, and 191–208 Hz. Meanwhile, seismic frequency bands are 6–23 Hz, 25–42 Hz, 41–58 Hz, 54–71 Hz, and 71–87 Hz. Some bands exhibit overlap due to high variance. Subsequently, we applied the Welch method to extract frequency domain features. The entire process is summarized in Figure 3.5.

We developed a two-class classifier featuring a single hidden layer comprising 3 nodes to differentiate between AAV and DW type vehicles. Previous studies have utilized kNN, MLP, ESN and SRC techniques for vehicle classification using the SITEX02 dataset. However, in this investigation, we employed a MLP.

Our classifier incorporates an adaptive learning algorithm using Stochastic Gradient Descent (SGD) and cosine annealing learning rate schedule [186], which employs cosine annealing across epochs to mitigate the risk of encountering local minima and issues stemming from random initialization points. It employs Exponential Linear Unit (ELU) [187] as activation for the hidden layers, which stands out as one of the most effective solutions for vanishing gradient problem, and hyperbolic tangent as the output layers activation.

Weight initialization is a critical factor affecting the speed of convergence. Improper initial weights can delay the convergence of an MLP. Rather than choosing weights from a uniform distribution, we opted to initialize the weights using a normal distribution. The mean of this distribution is zero, and its standard deviation is adjusted according to the number of nodes [188], [189].

3.3. ConvLSTM Based Localization

Trilateration is a core technique for range-based localization and forms the basis of our method. In our scenario, the coordinates x_{s_i} and y_{s_i} of the sensors are known. With the sensor-target distance r_i also known, as detailed in Section 2.5.2, the position of the target can be pinpointed by setting up a system of linear equations with (2.93) for $i = 1, 2, 3$ [37]. If r_i is unknown, additional data linearly related to r_i must be used. Trilateration often integrates with RSSI in mobile networks for this reason [190]. Building on this idea, Le Borgne et al. [70] experimented with seismic signal energy instead of RSSI. They found that using seismic energy alone for trilateration has limited effectiveness. This is due to the lack of a precise RSSI-distance function for seismic signals and variations in signal patterns caused by different velocities, accelerations, soil conditions, and vehicle types. Therefore, they chose a different strategy, utilizing all the sensors in the SITEX02 dataset and applying distributed regression to raw data, which led to better results.

Therefore, we opted to replace traditional methodologies with ConvLSTM [174] to leverage the temporal dynamics of vehicle movements and their effects on the seismic frequency domain features [191]. This innovative method merges Convolutional and LSTM layers into a cohesive framework, expanding the analytical dimensions into both spatial and temporal realms. By fusing Convolutional and LSTM models, this approach significantly improves the performance of data-driven techniques. The model processes frequency domain features derived from raw seismic data and the geographic coordinates of seismic sensors as distinct inputs, and outputs the positions of targets relative to the centralized cluster of sensor nodes. The SITEX02 dataset, with its strategic node placement, serves as an excellent example for demonstrating the clustering and feature extraction techniques essential for target localization with just three sensors in the field. In the following sections, an overview of the dataset will be presented along with detailed insights into the localization algorithm being proposed.

3.3.1. Network Architecture

ConvLSTM is employed to capture time-frequency features of seismic signatures from moving vehicles within a sensor network setting, as detailed in Section 3.3, making it a pivotal component in our architecture. Our proposed flow is shown in Figure 3.6, while Figure 3.7 is the base model of our proposed method. The $SeismicFourier$ block utilizes frequency domain features. Initially, we utilized a ConvLSTM with a temporal length of $M = 3$, employing filters with a $(3, 3)$ shape and 32 filters, followed by a continuation with filters of $(1, 3)$ shape and 64 filters, akin to Darknet-19 [192]. Average pooling layers with a shape of $(1, 3)$ were incorporated after each convolution layer. This configuration was selected due to its optimal balance between compactness and performance, as determined through extensive experimentation.

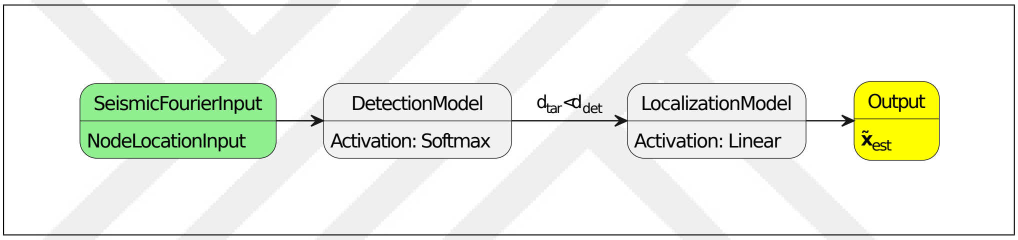


Figure 3.6: Flow of proposed approach.

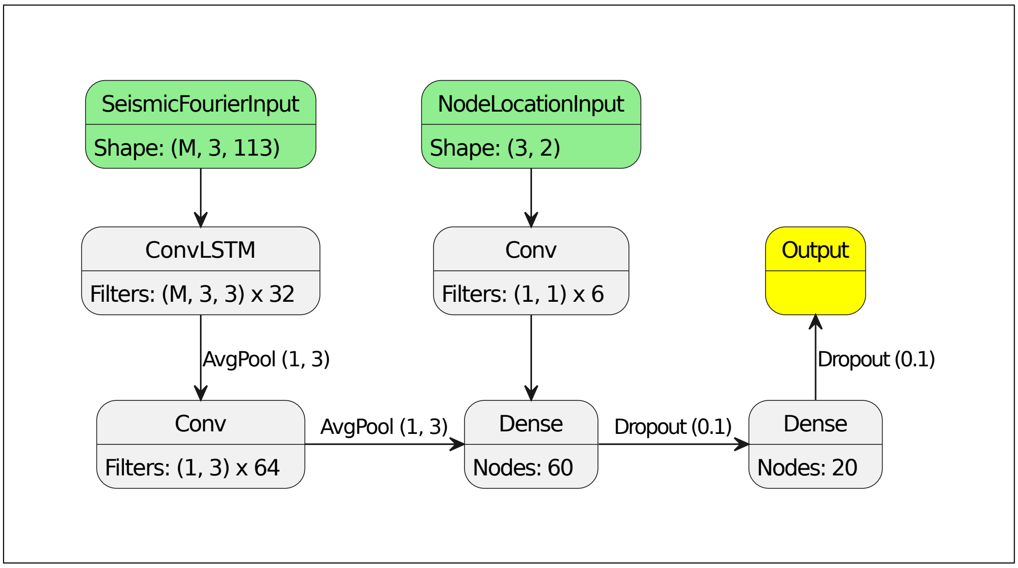


Figure 3.7: Base network for detection and localization models.

The base model described thus far serves as the foundation for both the target detection and localization models, differing only in their output layers. For the detection model, cross-entropy is employed as the loss function, depicted in (3.1), which is well-suited for Softmax activation [193] as specified in (3.2). The loss function $\mathrm{L}_{\mathrm{CE}}\left(\mathrm{f}_{\mathrm{SM}}(\boldsymbol{z}), \tilde{\boldsymbol{z}}\right)$ is utilized for the output layer.

$$\mathrm{L}_{\mathrm{CE}}\left(\boldsymbol{z}, \tilde{\boldsymbol{z}}\right) = -\tilde{\boldsymbol{z}}^{\mathrm{T}} \log(\boldsymbol{z}) \tag{3.1}$$

$$\mathrm{f}_{\mathrm{SM}}(\boldsymbol{z}) = \frac{e^{\boldsymbol{z}}}{\sum_{z \in \boldsymbol{z}} e^{z}} \tag{3.2}$$

The selection of suitable loss function holds significant importance in the context of the localization model. Since classification loss functions are not applicable, we need to pick one of regression loss functions, including Log-cosh, Cosine Similarity, Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Squared Logarithmic Error (MSLE), and Huber [194]. We opt for the Huber loss function due to robustness against outliers, introduced by Peter Jost Huber [194]. The Huber loss L_{δ} is given in (3.3), where $\boldsymbol{z}$ represents output of network, $\tilde{\boldsymbol{z}}$ denotes the ground truth value, and δ outlier handling parameter. Loss function for the output layer is only $\mathrm{L}_{\delta}\left(\boldsymbol{z}, \tilde{\boldsymbol{z}}\right)$, because of linear activation at output layer.

$$\mathrm{L}_{\delta}\left(\boldsymbol{z}, \tilde{\boldsymbol{z}}\right) = \sum_{\substack{z \in \boldsymbol{z} \\ \tilde{z} \in \tilde{\boldsymbol{z}}}} \begin{cases} \frac{1}{2}\left(z - \tilde{z}\right)^2 & , \left|z - \tilde{z}\right| \leq \delta \\ \delta\left(\left|z - \tilde{z}\right| - \frac{1}{2}\delta\right) & , \left|z - \tilde{z}\right| > \delta \end{cases} \tag{3.3}$$

The activation functions for the hidden layers are Leaky Rectified Linear Units (LReLU) [195], employed to mitigate issues such as gradient vanishing and the dead-ReLU problem, as described in (3.4), where z represents a node output, with leakiness parameter as α, which we choose as 1e-3.

$$\mathrm{f}_{\alpha}\left(z\right) = \begin{cases} \alpha z & , z < 0 \\ z & , z \geq 0 \end{cases}, \mathrm{f}_{\alpha}'\left(z\right) = \begin{cases} \alpha & , z < 0 \\ 1 & , z \geq 0 \end{cases} \tag{3.4}$$

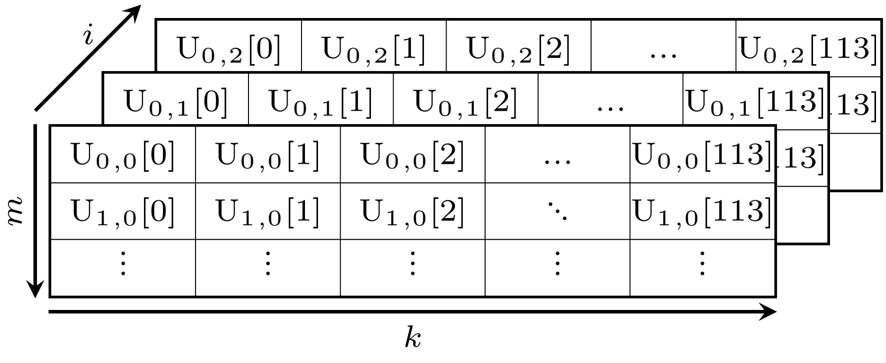
3.3.2. Frequency Domain Input

Given $\mathrm{u}[n]$ as the seismic input with a duration of T_b seconds. We compute its N-point DFT $\mathrm{U}[k]$ utilizing equation (3.5), as discussed in Section 2.2.1.

$$\mathrm{U}\left[k\right] = \left| \sum_{n=0}^{N-1} \mathrm{u}\left[n\right] e^{-j2\pi \frac{n}{N} k} \right| \tag{3.5}$$

The tensor representing features from frequency domain has three dimensions: the first dimension corresponds to the time index m, the second dimension denotes the sensor number i, and the third dimension represents the frequency index k, where $0 \leq K_1 \leq k \leq K_2 < N$. Here, m serves as a sequential time block index, providing temporal context about the event occurrence to the Convolutional Neural Network (CNN), with $0 \leq m < M$. This information can be encapsulated within the three-dimensional tensor $\mathrm{U}_{m,i}[k]$.

Figure 3.8 shows our first input tensor for two time blocks, three sensor nodes, and features from frequency domain, with indices ranging from 0 to 113 for a 3720-point DFT. The limitation on the frequency index is justified by the fact that the majority of seismic signal energy is typically concentrated between 0 to 150 Hz [185].



i
$\mathrm{U}_{0,2}[0]$ $\mathrm{U}_{0,2}[1]$ $\mathrm{U}_{0,2}[2]$ $\dots$ $\mathrm{U}_{0,2}[113]$
$\mathrm{U}_{0,1}[0]$ $\mathrm{U}_{0,1}[1]$ $\mathrm{U}_{0,1}[2]$ $\dots$ $\mathrm{U}_{0,1}[113]$
$\mathrm{U}_{0,0}[0]$ $\mathrm{U}_{0,0}[1]$ $\mathrm{U}_{0,0}[2]$ $\dots$ $\mathrm{U}_{0,0}[113]$
$\mathrm{U}_{1,0}[0]$ $\mathrm{U}_{1,0}[1]$ $\mathrm{U}_{1,0}[2]$ $\ddots$ $\mathrm{U}_{1,0}[113]$
$\vdots$ $\vdots$ $\vdots$ $\vdots$ $\vdots$
m
k

Figure 3.8: Frequency domain input tensor.

3.3.3. Sensor Location Input

We organize the sensor nodes into triangular clusters. It is essential during this process to form clusters in geographically dispersed areas to ensure a comprehensive representation of the sensor field. Although various clustering algorithms [196]–[200] are available, the limited number of sensors in our setup permits manual clustering. After establishing clusters, we calculate normalized positions using the average location of all sensors. The configuration of the triangulated nodes is shown in Figure 3.9. We have formed four clusters: S_1 with nodes s_5, s_3, s_4, S_2 with s_{47}, s_{46}, s_{48}, S_3 with s_{58}, s_{53}, s_{52}, and S_4 with s_{59}, s_{55}, s_{41}, collectively referred to as $\boldsymbol{S} = S_j | j = 0, 1, 2, 3$.

To ensure the model's insensitivity to the geographic location of the site, we utilize the relative locations of the sensors rather than their absolute positions. This is achieved by computing the relative locations through the subtraction of the mean location of the sensor cluster set, as depicted in equation (3.6). Here, $\boldsymbol{\mu}_{\boldsymbol{x}}^{(\boldsymbol{S})}$ denotes the center of the cluster set $\boldsymbol{S}$ and can be computed using equation (3.7).

$$\tilde{\boldsymbol{x}}_{s_i} = \begin{bmatrix} \tilde{x}_{s_i} \\ \tilde{y}_{s_i} \end{bmatrix} = \begin{bmatrix} x_{s_i} \\ y_{s_i} \end{bmatrix} - \boldsymbol{\mu}_{\boldsymbol{x}}^{(\boldsymbol{S})} \tag{3.6}$$

$$\boldsymbol{\mu}_{\boldsymbol{x}}^{(\boldsymbol{S})} = \frac{1}{\sum_{S_j \in \boldsymbol{S}} |S_j|} \sum_{S_j \in \boldsymbol{S}} \sum_{s_i \in S_j} \begin{bmatrix} x_{s_i} \\ y_{s_i} \end{bmatrix} \tag{3.7}$$

Figure 3.10 displays a tensor structure example for two coordinates across three sensor nodes, specifically from cluster $S_1 = s_5, s_3, s_4$. This structure comprises sensor location features arranged into two dimensions: the first dimension corresponds to the sensor index i, and the second dimension pertains to the axis index l, with $l = 0$ indicating the x-axis and $l = 1$ pointing to the y-axis. These dimensions are encapsulated within a two-dimensional tensor denoted as $V_{i,l}$.

3.3.4. Output Format

The training and inference processes of our algorithm rely on the detection of the target in the sensor cluster S_j, which we designate as activated. Upon target detection, we utilize $\tilde{\boldsymbol{x}}_{s_i} \in S_j$ as the sensor location input $V_{i,l}$ along with the feature input $U_{m,i}[k]$ for

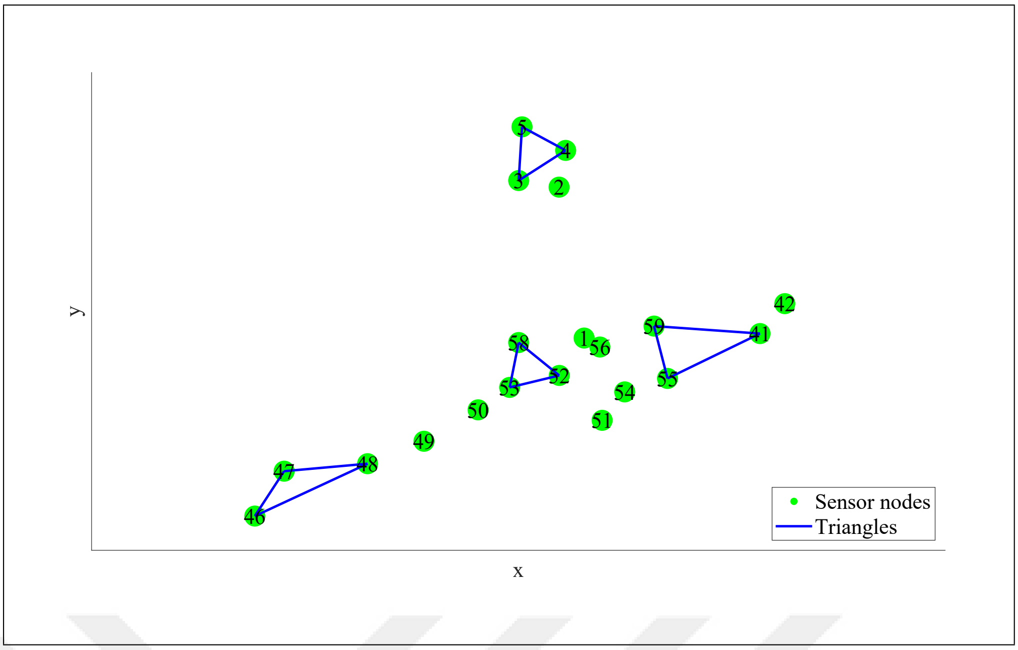


Figure 3.9: Sensor node clusters.

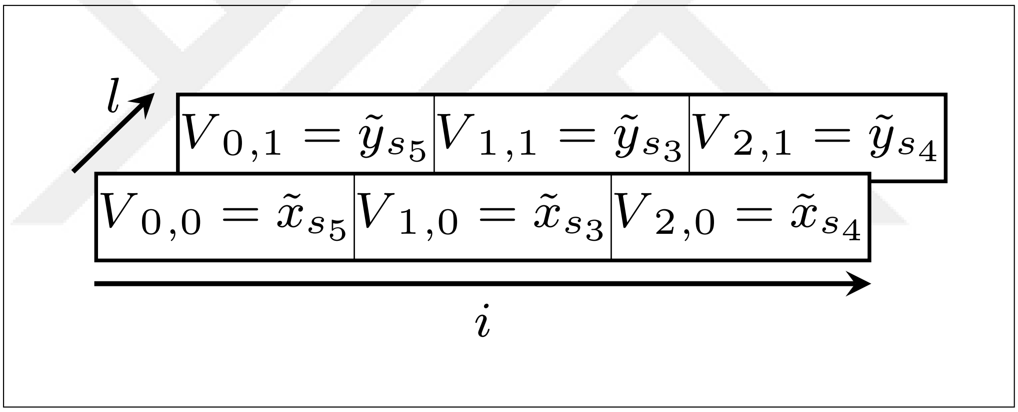

l
$V_{0,1} = \tilde{y}_{s_5}$
$V_{1,1} = \tilde{y}_{s_3}$
$V_{2,1} = \tilde{y}_{s_4}$
$V_{0,0} = \tilde{x}_{s_5}$
$V_{1,0} = \tilde{x}_{s_3}$
$V_{2,0} = \tilde{x}_{s_4}$
i

Figure 3.10: Sensor location input tensor.

both training and inference steps. This approach ensures that our model can effectively process both spatial and feature data essential for localization. The amalgamation of these inputs empowers the algorithm to develop a comprehensive understanding of the environment and target dynamics.

During the training phase, we normalize the ground truth target location $\boldsymbol{x}_{\text{tar}} = [x_{\text{tar}}, y_{\text{tar}}]^{\text{T}}$ to be zero-mean across sensor clusters, promoting generalization. This normalization step is pivotal as it enables the model to generalize across various sensor clusters, enhancing its versatility and reducing sensitivity to specific configurations. The target location $\tilde{\boldsymbol{x}}_{\text{tar}}$ relative to the center of the activated sensor cluster S_j can be computed using (3.8), where $\boldsymbol{\mu}_{\boldsymbol{x}}^{(S_j)}$ represents the center of the cluster S_j and can be determined using (3.9). This approach ensures consistent and accurate target localization relative to the center of the activate sensor cluster.

$$\tilde{\boldsymbol{x}}_{\text{tar}} = \begin{bmatrix} \tilde{x}_{\text{tar}} \\ \tilde{y}_{\text{tar}} \end{bmatrix} = \begin{bmatrix} x_{\text{tar}} \\ y_{\text{tar}} \end{bmatrix} - \boldsymbol{\mu}_{\boldsymbol{x}}^{(S_j)} \tag{3.8}$$

$$\boldsymbol{\mu}_{\boldsymbol{x}}^{(S_j)} = \frac{1}{|S_j|} \sum_{s_i \in S_j} \begin{bmatrix} x_{s_i} \\ y_{s_i} \end{bmatrix} \tag{3.9}$$

During the inferencing or testing phase, the output of the algorithm $\tilde{\boldsymbol{x}}_{\text{est}} = [\tilde{x}_{\text{est}}, \tilde{y}_{\text{est}}]^{\text{T}}$ can be denormalized using (3.10) to compute the estimated location in the activated cluster S_j. This denormalization step transforms the model's output back to the original coordinate system, offering a practical and interpretable location estimate. The capability to accurately convert the model outputs to real-world coordinates is crucial for validating the model's performance and ensuring its relevance in practical applications.

$$\boldsymbol{x}_{\text{est}} = \begin{bmatrix} x_{\text{est}} \\ y_{\text{est}} \end{bmatrix} = \begin{bmatrix} \tilde{x}_{\text{est}} \\ \tilde{y}_{\text{est}} \end{bmatrix} + \boldsymbol{\mu}_{\boldsymbol{x}}^{(S_j)} \tag{3.10}$$

3.4. Summary

To summarize, in this chapter, we introduced the data and its frequency response in different environments with tracked and trackless vehicles in Section 3.1. We proposed a filter bank for feature extraction and a simple neural network for classification to demonstrate the performance of our analysis in Section 3.2. In Section 4.1, we will see the quality of our classification model using only 23 features. Based on this analysis, we explained how we used downsampled seismic signals as input for the localization model in Section 3.3.2. Then, we described the ConvLSTM-based model in Section 3.3.1. We concluded by explaining how to interpret the output layer of the ConvLSTM model in Section 3.3.4. Chapter 4 contains the test results and performance comparisons of our proposed approach with existing approaches in the literature.

4. TEST RESULTS

The test results section presents findings for both classification and localization. We began with classification, examining the fusion of acoustic and seismic features and their performance across different road types inferred from GPS data in Section 4.1. These results demonstrated that using a maximum frequency of 150 Hz for seismic data is adequate for localization. However, this time we will employ ConvLSTM to identify the appropriate filter bank.

Localization test results are provided in Section 4.2. We examined the effects of varying target-sensor cluster distances, the number of sensors in a cluster, the impact of the detection method, and the temporal length for ConvLSTM on detection and localization. Additionally, we demonstrated the benefit of using a Kalman Filter in the 100 m detection case to reduce localization error.

In summary, we provided comments and conclusions in Section 4.3.

4.1. Classification

We tested our experiments on the SITEX02 dataset, as described in Section 3.1. Specifically, we utilized trials 3rd 6th and 9th trials of AAV and DW vehicles. We excluded the data partitions not detected from the CFAR outputs [57], which subtracts the background. The training portion comprised 70% of the available data, validation accounted for 15%, and the remaining 15% was allocated for testing.

We evaluated the test set performance using the True Positive Ratio (TPR) and False Positive Ratio (FPR) metrics. TPR is computed according to (4.1), where TP is the true positive count and FN is the false negative count. FPR is calculated using (4.2), where FP is the false positive count and TN is the true negative count. For each combination of different sensor types and domains, we carried out 10 experimental trials and documented the highest performance achieved in each scenario.

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{4.1}$$

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \tag{4.2}$$

Our primary classifier was an MLP. However, we also utilized Bagging and Gentle Boosting methods to validate the results obtained with the MLP classifier. Each combination of sensor modality and feature domain was systematically tested to assess the contribution of each feature type and sensor modality. Detailed test procedures are outlined in this section, while comparisons to previous studies are presented in the subsequent section.

4.1.1. Fusion Results

This section will investigate the fusion of different modalities and the impact of combining features from different domains. Features were extracted as detailed in Section 3.2. Fusion results will be evaluated if there is a positive interaction or synergy between acoustic and seismic modalities and between frequency and time domain features.

Three classifiers were utilized to evaluate the features quality, but results from only MLP are reported here. In the following section, performances of the other two classifiers will also be provided. Training was conducted using the training portion and was concluded when performance began to decline on the validation portion. The classification performance for all scenarios is listed in Table 4.1 and 4.2.

Table 4.1 shows the benefits of integrating sensor features and applying features from both the time and frequency domains. The use of time-domain features is particularly advantageous when working with seismic sensors alone. Conversely, vehicles of the AAV type can be identified with an accuracy of 82.4% using only frequency domain features. However, when time-domain features are added to the mix, classification accuracy improves by 7%. Acoustic sensors, when used in the frequency domain alone, yield a TPR of 95.9% and a FPR of 3.9%. Both TPR and FPR see an improvement of 1.4% when combined with seismic frequency domain features. Using only the acoustic modality, the best results achieved combine frequency and time-domain features, resulting in a TPR of 96.2% and an FPR of 3.8%. The highest overall performance, with a TPR of 98.6% and an FPR of 1.4%, is achieved by merging both acoustic and seismic modalities with frequency and time-domain features. This indicates that while the acoustic modality alone can be effective, combining modalities is crucial for minimizing FPR in critical applications.

Table 4.1: MLP classifier performance after fusion for overall.

		AAV+DW		
		Acoustic	Seismic	Both
Frequency Domain	TPR %	95.9	85.5	97.3
	FPR %	3.9	14.5	2.5
Time Domain	TPR %	90.4	76.7	92.5
	FPR %	9.3	23.3	10.4
Both Domains	TPR %	96.2	89.6	98.6
	FPR %	3.8	10.4	1.4

Table 4.2: MLP classifier performance after fusion for AAV and DW.

		AAV			DW		
		Acoustic	Seismic	Both	Acoustic	Seismic	Both
Frequency Domain	TPR %	95.9	82.4	97.1	95.9	88.6	97.6
	FPR %	3.7	11.4	2.0	4.1	17.7	2.9
Time Domain	TPR %	92.9	80.6	94.7	87.8	72.8	90.2
	FPR %	12.2	27.2	9.8	6.5	19.4	5.3
Both Domains	TPR %	96.5	89.4	97.7	95.9	89.8	99.6
	FPR %	4.7	10.2	0.4	3.5	10.6	2.4

4.1.2. Different Road Types

Figure 3.4 indicates that developing a distinct classifier for each road type can enhance performance. Accordingly, two classifier specifically trained for asphalt and gravel roads seperately with seismic frequency domain features. Results are given in Table 4.3. Classifier for gravel roads achieved a TPR of 88.7% with a FPR of 11.4%. TPR on asphalt was slightly lower, at 85.5%. The classifier trained on a unified dataset, encompassing both asphalt and gravel roads, also recorded a performance of 85.5%. Road-specific training increases classification accuracy on particular road types, especially for gravel roads in this dataset.

Table 4.3: Performance of the classifiers for different road types.

	Training on Asphalt Data		Training on Gravel Data		Training on Unified Data	
	TPR %	FPR %	TPR %	FPR %	TPR %	FPR %
AAV	85.5	14.7	90.9	13.6	82.4	11.4
DW	85.4	14.5	86.4	9.1	88.6	17.7
Overall	85.5	14.6	88.7	11.35	85.5	14.5

4.1.3. Comparison with Previous Works

Various methods have been developed to classify vehicles in distributed networks, primarily utilizing either time domain or frequency domain features, with a focus on acoustic sensors. This section compares our results to previous studies on the SITEX02 dataset. Table 4.5 details results using the seismic modality alone, and Table 4.4 illustrates results achieved through the fusion of acoustic and seismic sensors from other studies. Our method significantly enhances performance, showing approximately 10% TPR improvement.

Table 4.4: Acoustic and seismic sensors results literature comparison.

	AAV		DW		Both	
	TPR %	FPR %	TPR %	FPR %	TPR %	FPR %
Duarte et al. [57]	85.9	3.0	81.8	10.7	83.6	6.9
Mazarakis et al. [75]	100.0	-	77.0	-	88.5	-
Kornaropoulos et al. [76]	94.3	11.1	88.9	5.7	91.6	8.4
Wang et al. [77]	100.0	-	90.0	-	95.0	-
Ntalampiras [154]	95.8	-	96.7	-	96.3	-
Gentle Boosting	97.7	1.6	98.4	2.4	98.0	2.0
Bagging	99.4	1.6	98.4	0.6	98.9	1.1
MLP	97.1	0.4	99.6	2.4	98.6	1.4

Ntalampiras [154] recently reported a TPR of 96.3% using the acoustic modality with 50 features, and Kalra et al. [81] attained a TPR of 92.1% using the seismic modality with a CNN. By contrast, our method achieves comparable performance using only 13 acoustic and 10 seismic features.

Table 4.5: Only seismic sensors results literature comparison.

	AAV		DW		Both	
	TPR %	FPR %	TPR %	FPR %	TPR %	FPR %
Duarte et al.[57]	58.0	48.6	56.8	47.6	57.4	48.1
Mazarakis et al.[75]	87.0	-	69.0	-	78.0	-
Jin et al. [157]	-	-	-	-	89.7	-
Bin et al. [158]	-	-	-	-	92.1	-
Kalra et al. [81]	86.0	14.0	76.0	24.0	81.0	19.0
Bagging	84.7	11.0	89.0	15.3	86.9	13.1
Gentle Boosting	84.1	12.2	87.8	15.9	86.0	14.0
MLP	89.4	10.2	89.8	10.6	89.7	10.4

4.2. Localization

For localization testing, we utilized the 4th 5th and 6th trials of AAV and DW vehicles, while other trials were allocated for training purposes. This partitioned the dataset of 11,700 samples into 70% for training and 30% for testing, chosen because the 4th, 5th, and 6th trials feature different vehicle routes within the sensor field.

The models were trained using the Keras deep learning library [201]. We set the training duration to 50 epochs for the detection model and 200 epochs for the localization model, with batch sizes of 1000 and 20 samples, respectively, for the detection and localization phases. The training was repeated ten times with shuffled data, and the results were averaged for reporting purposes. The Adam optimizer [202], along with a cyclical learning rate [203] ranging from $1e-6$ to $1e-3$ without decay and a step size of 100, was employed for the detection model. Conversely, the localization model used the SGD optimizer [162], with the learning rate cycling from $1e-5$ to $1e-2$ and including a decay formula 0.9^{lr} applied after every 50-step cycle.

4.2.1. Sensor Clustering

We explored the impact of the sensor configuration and detection method on performance, as detailed in Table 4.6. The detection method $Original$ utilizes target detectedion information based on CFAR from [57]. The detection method $Distance\ Based$ refers to our model, which is trained to estimate x_{tar}. As illustrated in Figure 3.6, our model disregards all measurements where $d_{\text{tar}} > d_{\text{det}}$, as specified in (4.3). This approach was inspired by the study of Duarte et al. [68], which employed a distance-based decision fusion with $d_{\text{det}} = 50$ m.

$$d_{\text{tar}} = \mathrm{d}_2\left(\boldsymbol{x}_{\text{tar}}, \boldsymbol{\mu}_{\boldsymbol{x}}^{(S_j)}\right) \tag{4.3}$$

Table 4.6: Different sensor configurations.

Detection	Error Statistics	Sensor Cluster Size		
Method	for Localization	1	2	3
$Original$	μ_{loc} (m)	41.1	23.7	15.3
	σ_{loc} (m)	30.2	21.5	14.0
$DistanceBased$	μ_{loc} (m)	23.2	14.6	9.7
	σ_{loc} (m)	16.6	10.7	5.9

We also examined the impact of the sensor node count in sensor clusters on localization accuracy. Findings, as shown in Table 4.6, indicate that increasing the count of sensor nodes in each sensor cluster leads to a reduction in localization error. Furthermore, the results suggest that excluding sensors that are far from the target during the training process significantly improves estimation accuracy.

4.2.2. Target-Sensor Cluster Distance Analysis

We examined the effects of implementing a maximum detection distance d_{det} between the target vehicle and sensor cluster during training, with results presented in Table 4.7. We utilized the F1 score, defined in (4.4), as a measure of detection performance due to the imbalanced data. The F1 score is calculated based on counts of TP, FP, and

FN. Findings indicate that a distance of 50 m is one of the optimal settings for d_{det}, corroborating the earlier research by Duarte et al. [68].

$$\text{F1}_{\text{det}} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}} \tag{4.4}$$

Table 4.7: Different detection distance results.

d_{det} (m)	10	20	30	40	50	60	70	80	90	100
F1_{det}	0.79	0.89	0.91	0.94	0.94	0.94	0.94	0.93	0.93	0.93
μ_{loc} (m)	9.86	9.48	9.47	9.50	9.70	9.95	11.10	13.42	13.96	15.26
σ_{loc} (m)	8.81	6.76	7.02	6.72	5.91	6.42	9.01	15.24	13.38	16.13

4.2.3. Temporal Length Analysis

The choice of the temporal length M is crucial and must be carefully considered. Performance metrics for different temporal lengths are illustrated in Figure 4.1. It appears that a temporal length of $M = 3$ provides the best balance, as this setting yielded the highest performance. The average distances between sensors within each cluster were recorded as follows: 29.95 m for S_1, 43.68 m for S_2, 27.36 m for S_3, and 44.52 m for S_4. These measurements are shown in Figure 3.9. Furthermore, Table 4.8 lists the average velocities from three trials for AAV and DW. Selecting values of M greater than 5 can cause the vehicle to travel distances exceeding 30 m, based on an average velocity of 7.09 m/T_b, which surpasses the minimal sensor distance within a cluster. As a result, performance declines as depicted in Figure 4.1 for larger M values. When the target's speed increases, reducing M is advised to ensure accurate localization.

Table 4.8: Estimated target vehicle velocities from GPS data.

	AAV4	AAV5	AAV6	DW4	DW5	DW6
μ_{vel} (m/T_b)	6.38	7.46	7.08	6.98	7.61	7.04
μ_{vel} (km/h)	30.63	35.82	33.98	33.54	36.54	33.79
σ_{vel} (km/h)	3.28	1.15	0.69	0.60	0.60	0.81

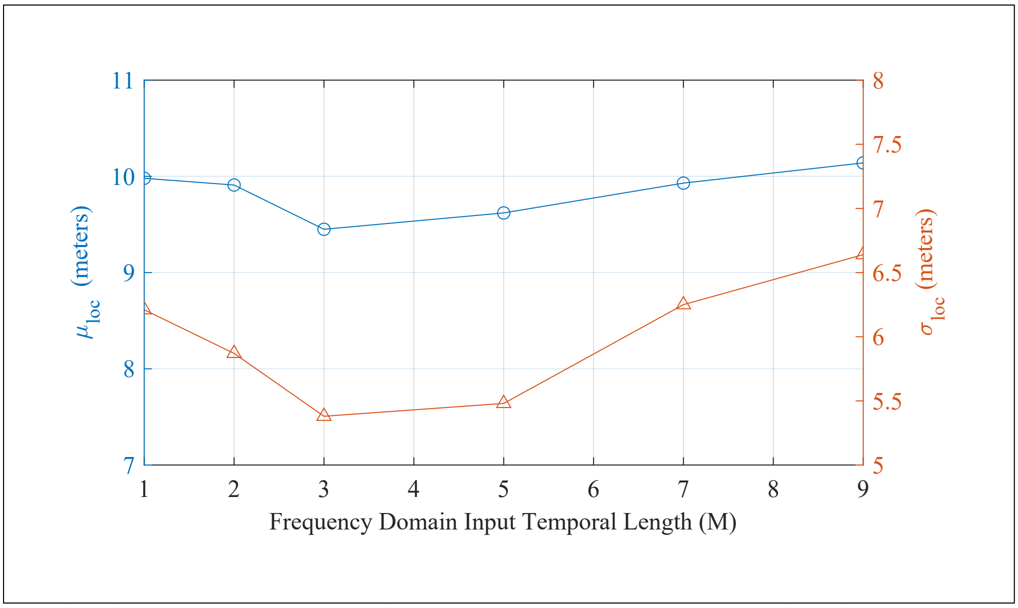


Figure 4.1: LSTM performance according to temporal length for $d_{\text{det}} = 50$ m.

4.2.4. Detection Results

Detection results for clusters of three node sensors with $M = 3$ and d_{det} = 50 m using ConvLSTM for the 4th AAV trial are analyzed in this section. Based on the route shown in Figure 4.14, the first activated cluster is $S_2 = \{s_{47}, s_{46}, s_{48}\}$. The detection results for these nodes are presented in Figure 4.3, Figure 4.2, and Figure 4.4 respectively. The SITEX02 detection results vary for each node as it uses independent detection for each. This means that each sensor operates on its own, which can lead to inconsistencies in detection results. However, since we employed collaborative detection with ConvLSTM, according to (4.3), the detection results are consistent for nodes within the same cluster.

The second activated cluster is $S_3 = \{s_{58}, s_{53}, s_{52}\}$. The detection results for these nodes are shown in Figure 4.7, Figure 4.6, and Figure 4.5. Similarly, the third activated cluster is $S_1 = \{s_5, s_3, s_4\}$, with the detection results presented in Figure 4.10, Figure 4.8, and Figure 4.9. In both cases, we observe that detection is not capturing all the signal, because it is capturing the event in sensor-wise collaborative way.

The inactive cluster is $S_4 = \{s_{59}, s_{55}, s_{41}\}$ because it is not on the route of the target. The detection results for these nodes are shown in Figure 4.13, Figure 4.12, and Figure 4.11. This cluster did not detect the target because it was not on the trajectory of the vehicle.

The model only detected certain parts of the signal, focusing on center of the cluster for detection. If the target is far from center of the cluster, it won't be detected, even if it is close to one of the sensors. This limitation highlights the need for careful consideration of cluster configuration to ensure comprehensive coverage of the monitored area. Additionally, the SITEX02 detections for cluster S_4 are inconsistent, showing variability in detection performance. However, our approach effectively filters out these low SNR sections, improving the overall robustness and reliability of the detection system. This filtering process ensures that only high-quality signals are used for localization, reducing the likelihood of false detections and improving accuracy of the system.

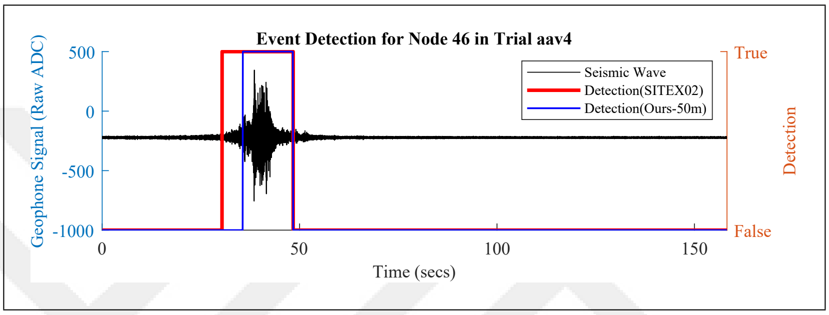


Figure 4.2: AAV 4th trial, target detection at node s_{46}, d_{det}=50 m.

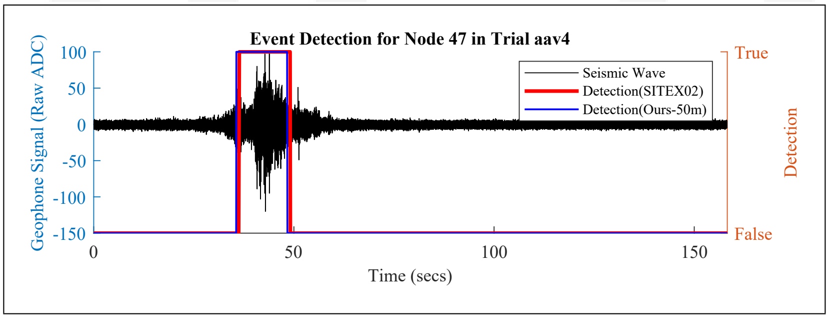


Figure 4.3: AAV 4th trial, target detection at node s_{47}, d_{det}=50 m.

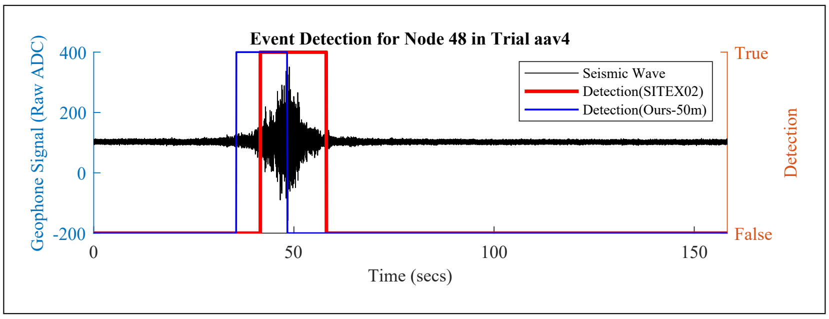


Figure 4.4: AAV 4th trial, target detection at node s_{48}, d_{det}=50 m.

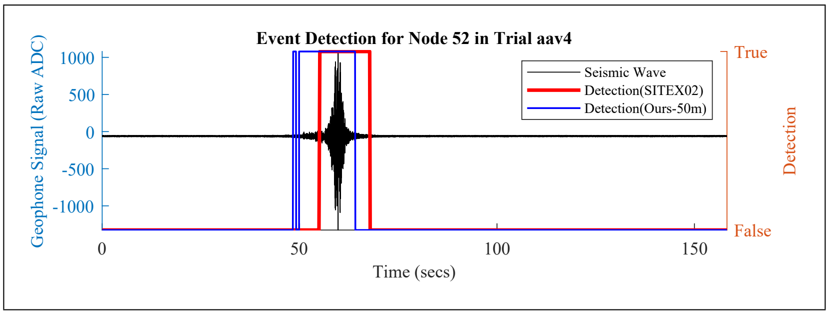


Figure 4.5: AAV 4th trial, target detection at node s_{52}, d_{det}=50 m.

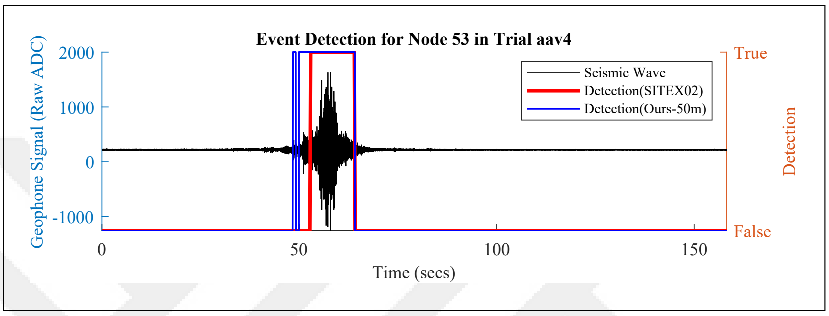


Figure 4.6: AAV 4th trial, target detection at node s_{53}, d_{det}=50 m.

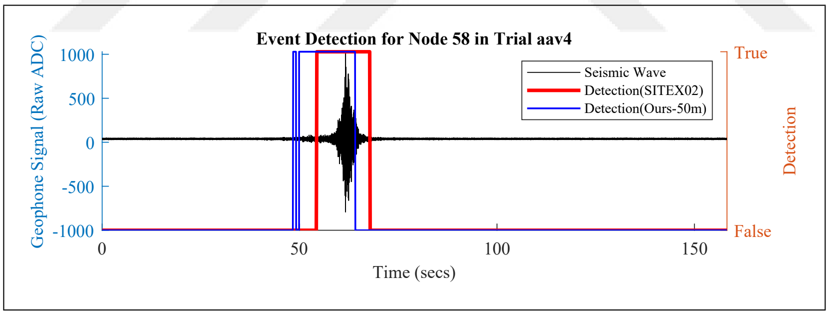


Figure 4.7: AAV 4th trial, target detection at node s_{58}, d_{det}=50 m.

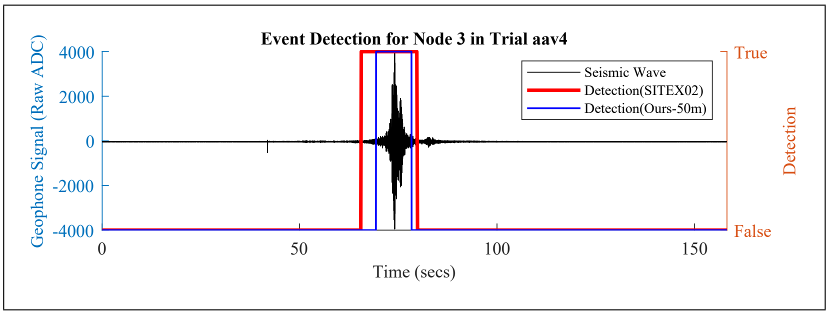


Figure 4.8: AAV 4th trial, target detection at node s_3, d_{det}=50 m.

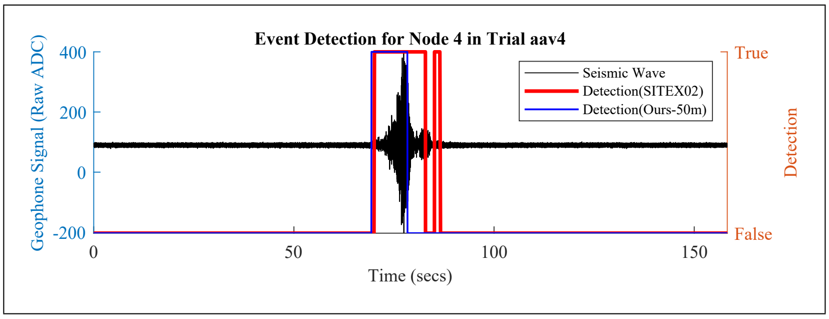


Figure 4.9: AAV 4th trial, target detection at node s_4, d_{det}=50 m.

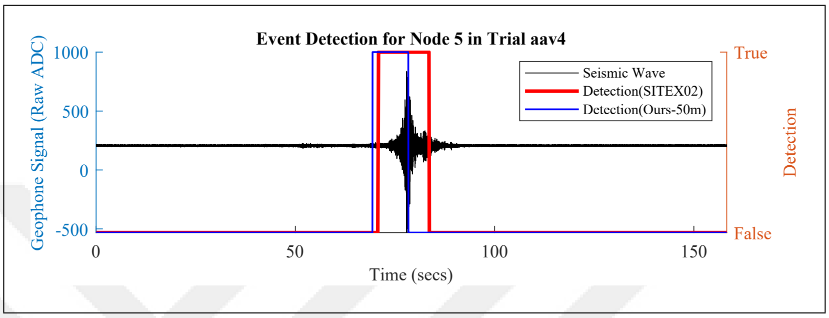


Figure 4.10: AAV 4th trial, target detection at node s_5, d_{det}=50 m.

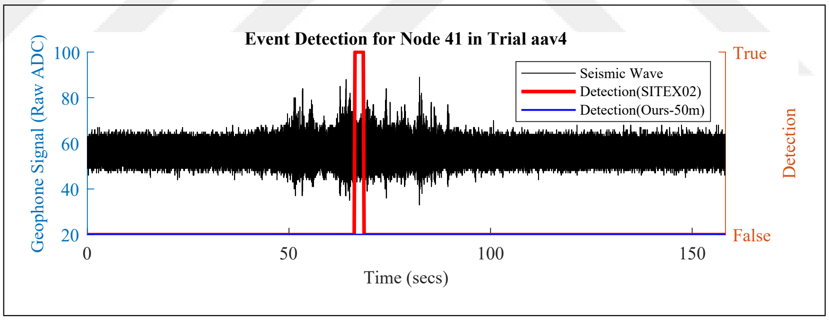


Figure 4.11: AAV 4th trial, target detection at node s_{41}, d_{det}=50 m.

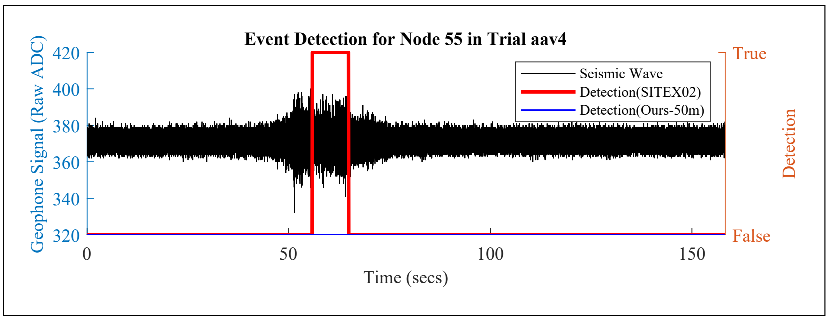


Figure 4.12: AAV 4th trial, target detection at node s_{55}, d_{det}=50 m.

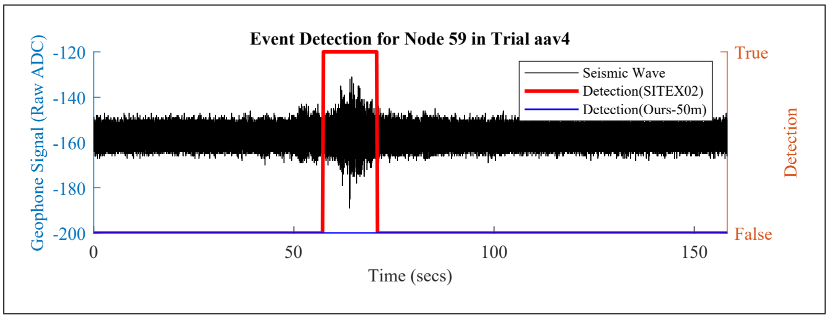


Figure 4.13: AAV 4th trial, target detection at node s_{59}, d_{det}=50 m.

4.2.5. Estimated Target Locations for 50 m Detection Distance

We examined the estimated paths for the AAV and DW vehicles across three different routes, as detailed in Section 4.2.4 with $d_{\text{det}} = 50$ m, which Table 4.7 suggests is the ideal detection distance. The path estimations for the 4th, 5th, and 6th trials of the AAV and DW vehicles are illustrated in Figures 4.14, 4.15, 4.16, 4.17, 4.18, and 4.19. These figures show that while the 5th trial of AAV and the 5th and 6th trials of DW display some positional deviations, their trajectories generally align well with the actual routes. This alignment is supported by statistical data presented in Table 4.9. Overall, the model demonstrates a higher efficacy for AAV, except for a noted anomaly in the 5th trial of DW. The trajectory during the 5th trial of DW closely matches with Figure 4.17.

Table 4.9: Trial statistics for $d_{\text{det}} = 50$ m.

	AAV4	AAV5	AAV6	DW4	DW5	DW6
μ_{loc} (m)	7.12	10.44	8.65	7.5	17.19	10.64
σ_{loc} (m)	4.26	6.30	7.10	3.89	5.37	3.13

Error plots for estimated paths across three different routes for both AAV and DW are provided in Figures 4.20, 4.21, 4.22, 4.23, 4.24, and 4.25. These plots confirm the accuracy of the path figures discussed earlier. Error plot for the 5th trial of DW, which shows a positional deviation, is specifically illustrated in Figure 4.23. In general, the distribution of errors over time is consistent, with exceptions noted at specific instances, such as at the 49th second for the 4th trial of AAV, the 79th second for the 5th trial of DW, and the 41st second for the 6th trial of AAV, where there are noticeable spikes.

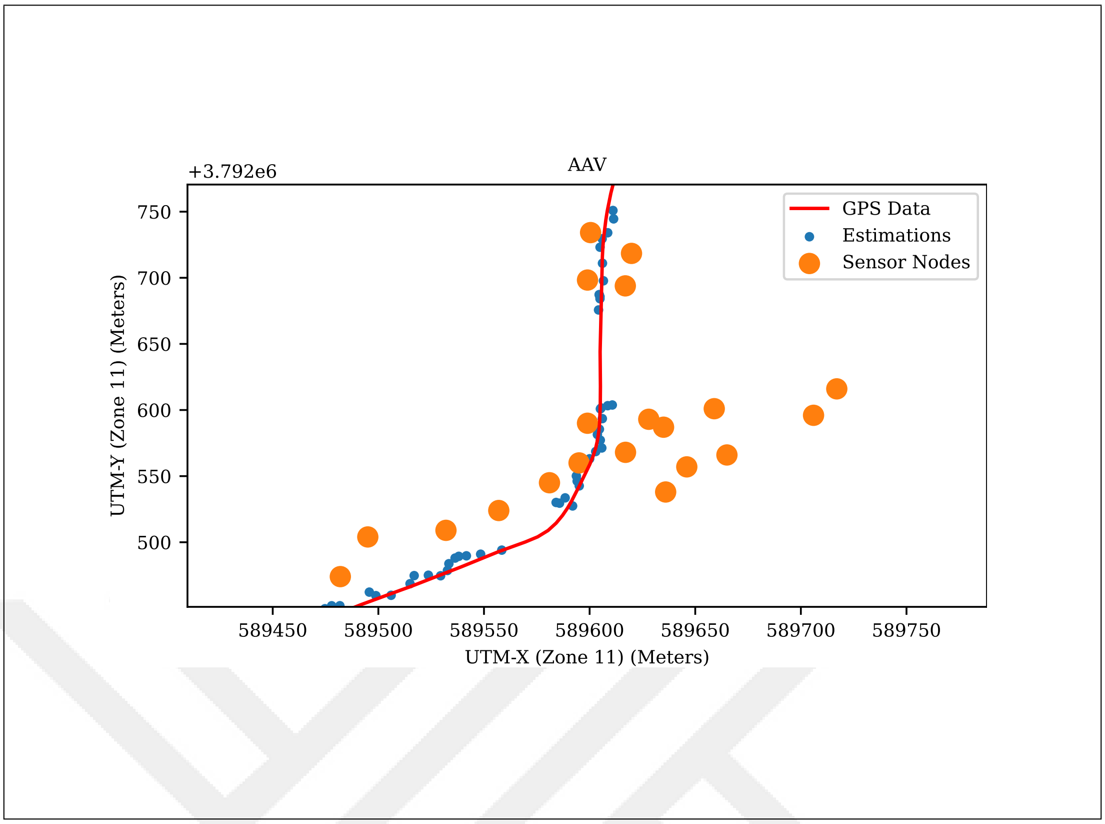


Figure 4.14: AAV 4th trial, target localization, $d_{\mathrm{det}} = 50$ m.

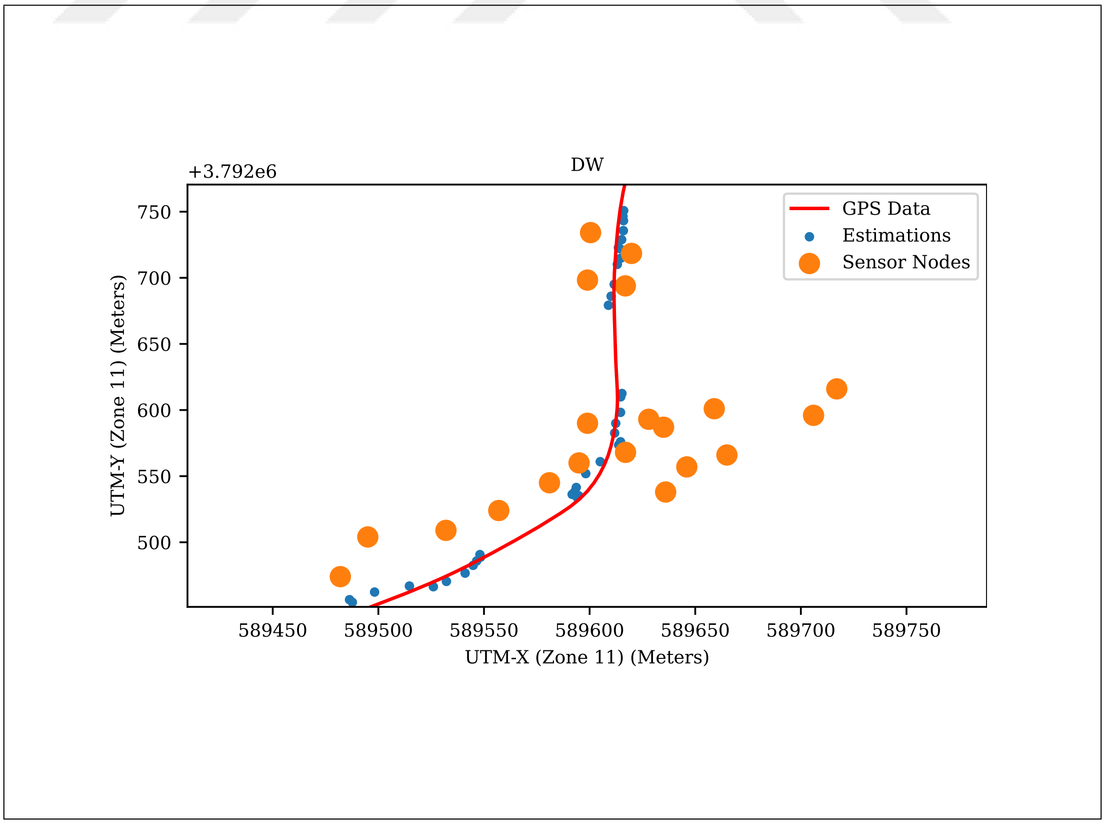


Figure 4.15: DW 4th trial, target localization, $d_{\mathrm{det}} = 50$ m.

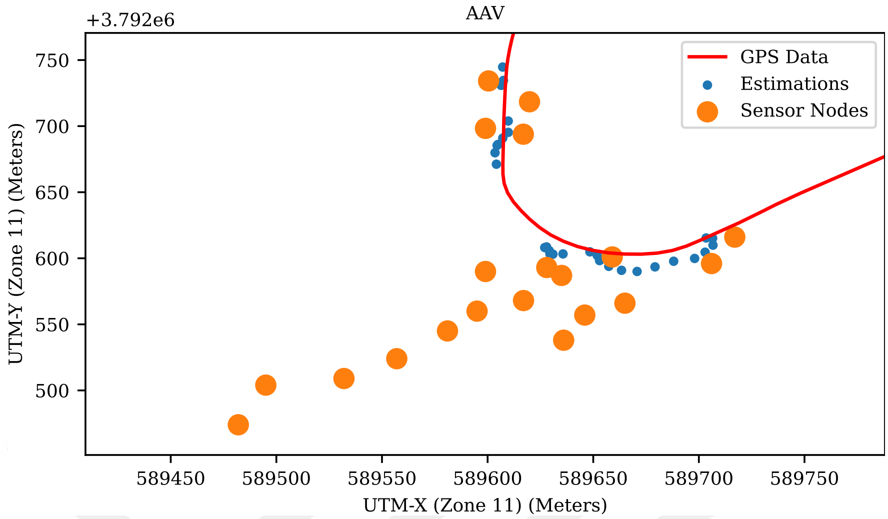


Figure 4.16: AAV 5th trial, target localization, $d_{\mathrm{det}} = 50$ m.

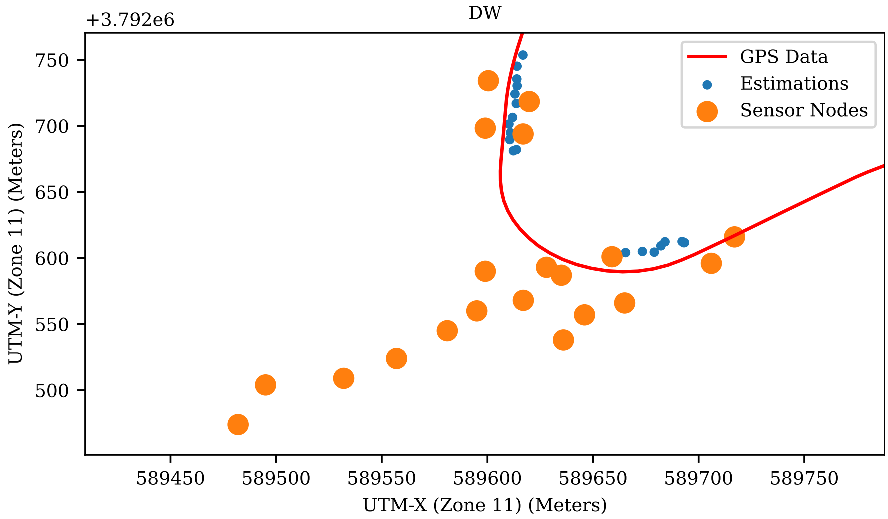


Figure 4.17: DW 5th trial, target localization, $d_{\mathrm{det}} = 50$ m.

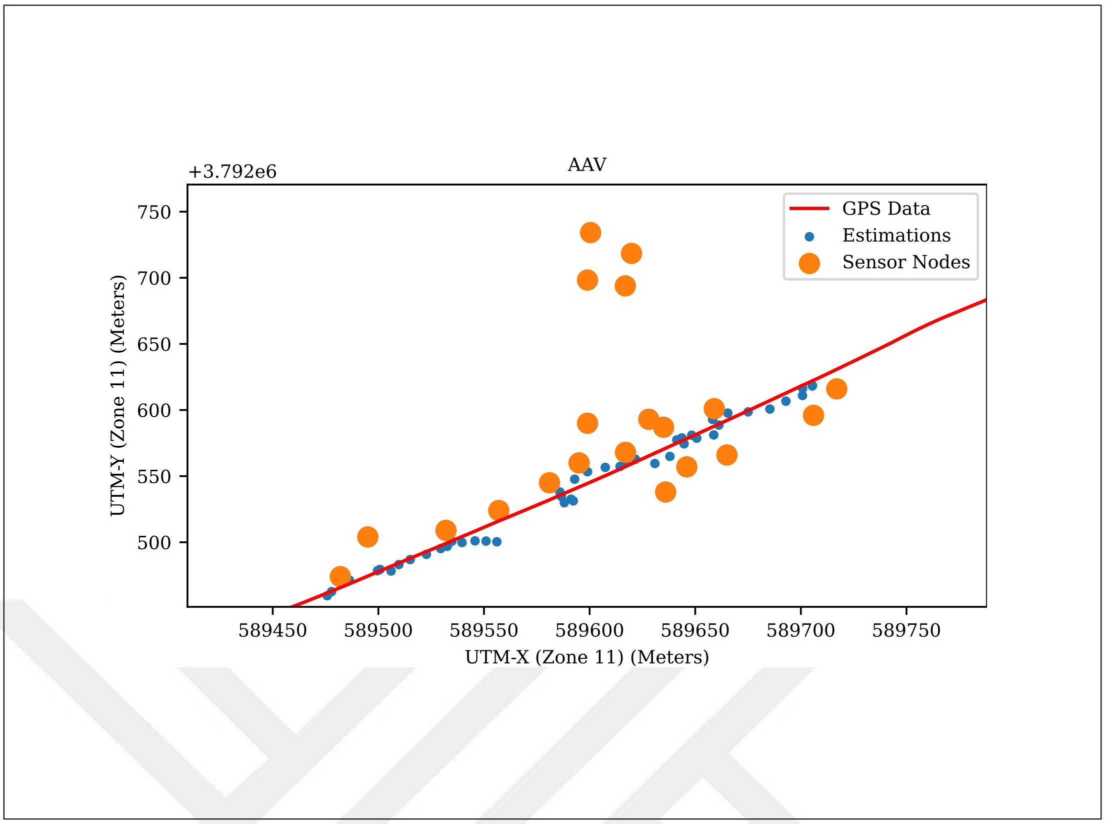


Figure 4.18: AAV 6th trial, target localization, $d_{\text{det}} = 50$ m.

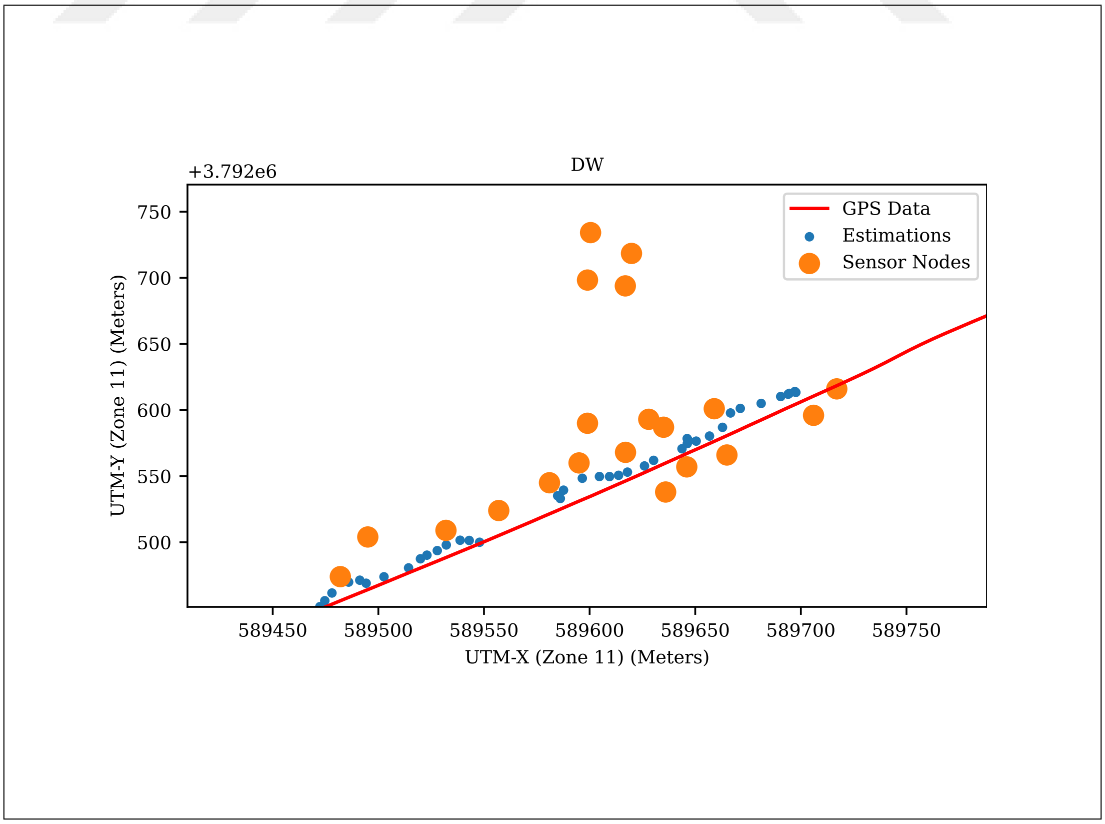


Figure 4.19: DW 6th trial, target localization, $d_{\text{det}} = 50$ m.

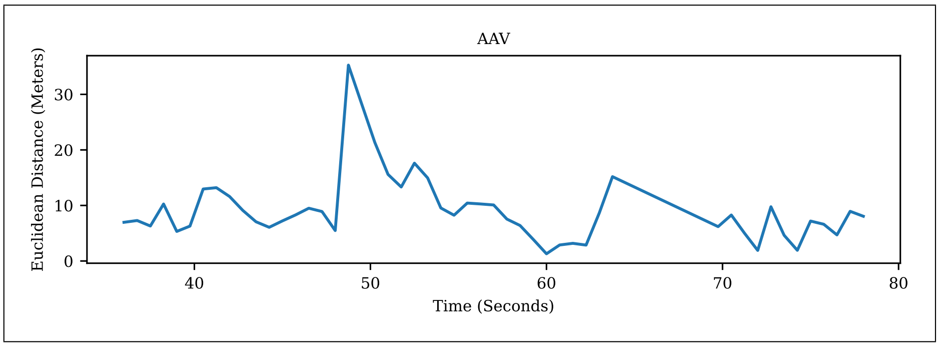


Figure 4.20: AAV 4th trial, localization error, $d_{\text{det}} = 50$ m.

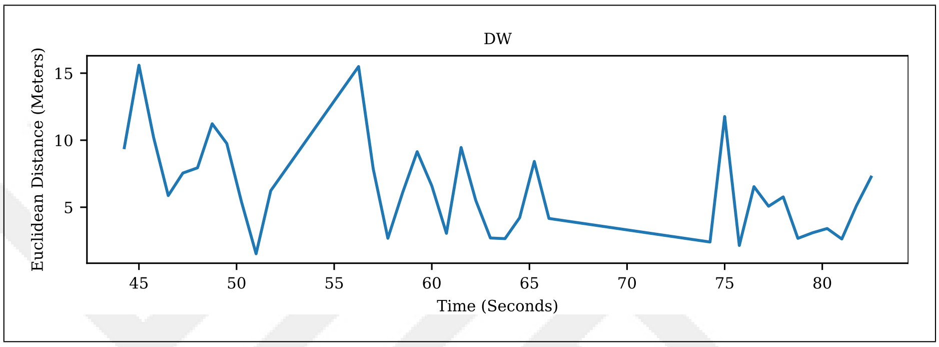


Figure 4.21: DW 4th trial, localization error, $d_{\text{det}} = 50$ m.

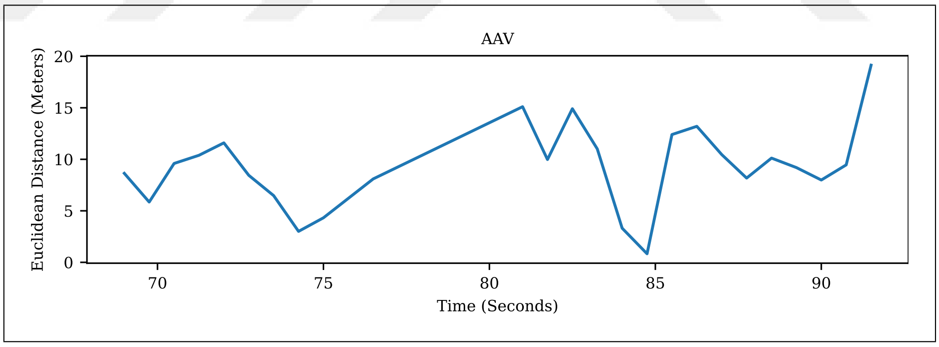


Figure 4.22: AAV 5th trial, localization error, $d_{\text{det}} = 50$ m.

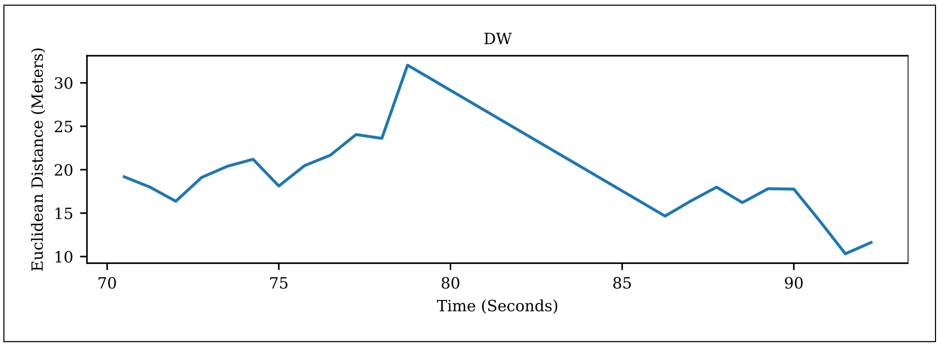


Figure 4.23: DW 5th trial, localization error, $d_{\text{det}} = 50$ m.

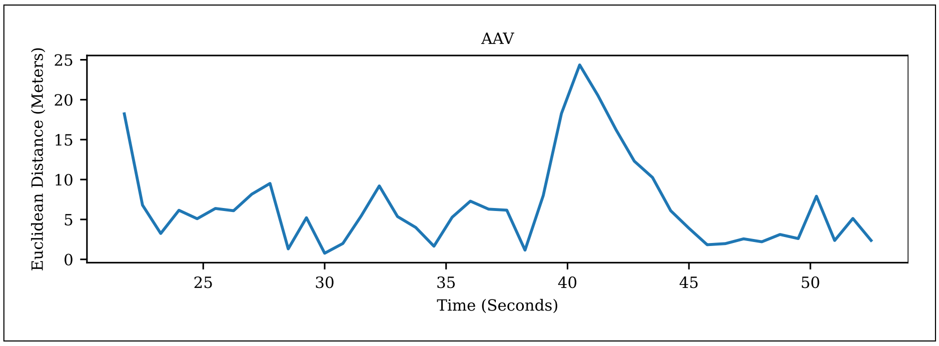


Figure 4.24: AAV 6th trial, localization error, d_{det} = 50 m.

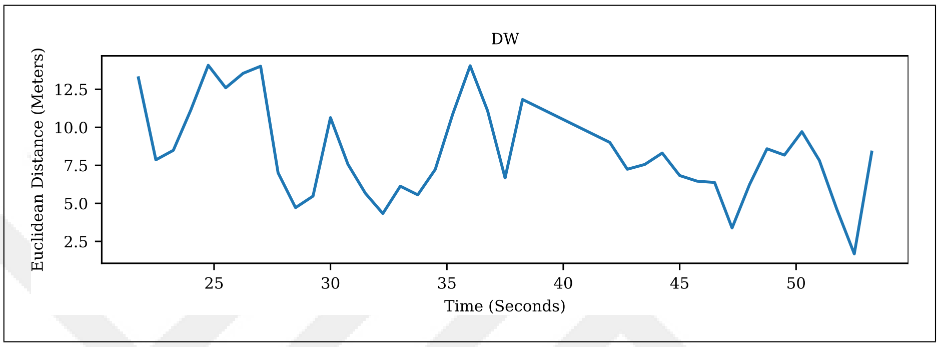
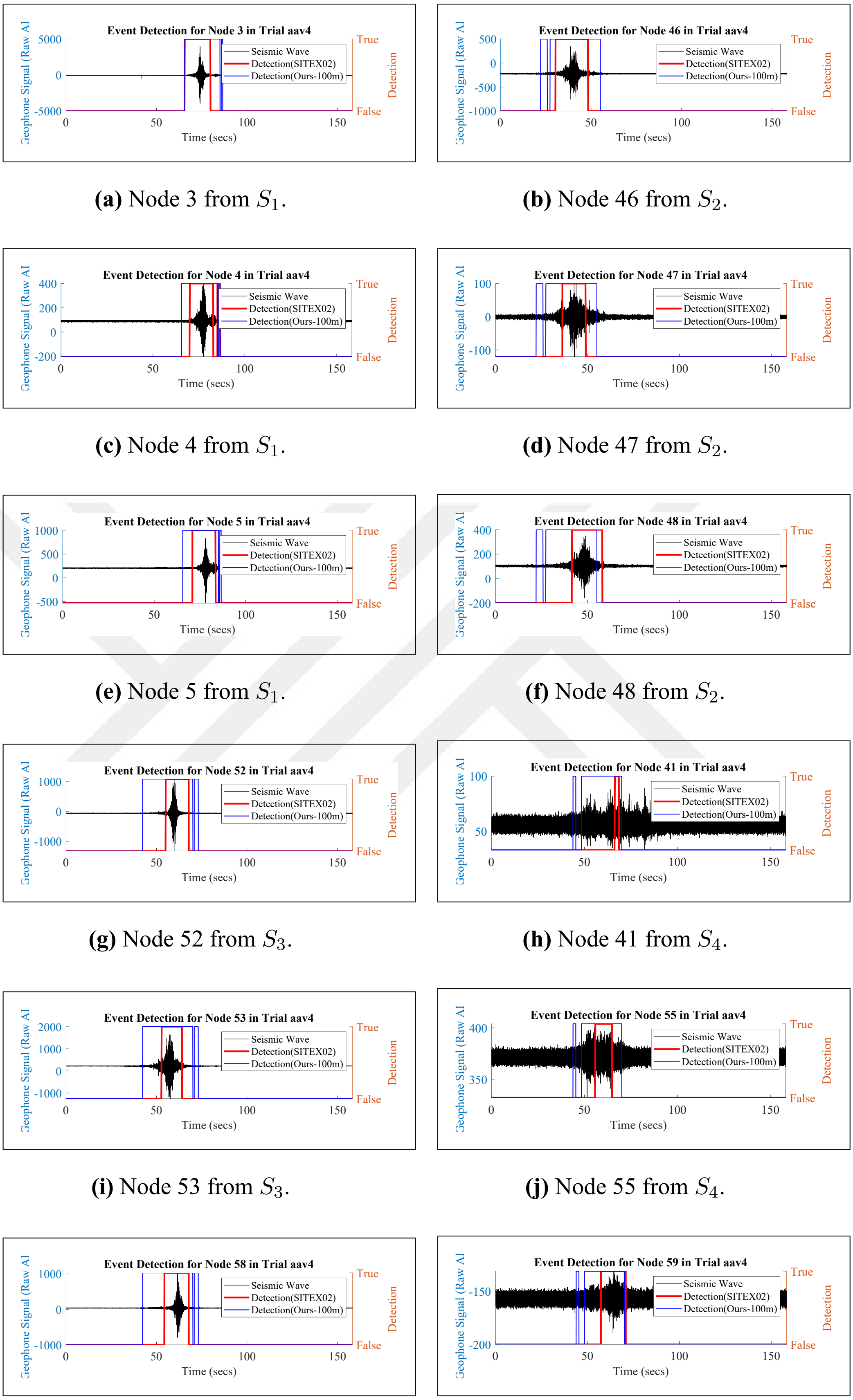


Figure 4.25: DW 6th trial, localization error, d_{det} = 50 m.

4.2.6. Estimated Target Locations for 100 m Detection Distance

In this section, we present the results for the maximum detection distance $d_{\text{det}} = 100$ m. Figure 4.26 shows the detection results for the fourth trial of the AAV vehicle, indicating that the detection begins to include noisy parts of the sequence with a weak correlation to the target location. As the detection range increases to 100 meters, the algorithm picks up extraneous signals, introducing more noise.

Due to the increased number of points and error generated by $d_{\text{det}} = 100$ m, we utilized MATLAB implementation of Interacting Multiple Model (IMM) with an Extended Kalman Filter. The IMM filter is designed for tracking highly maneuverable objects. Trials 4, 5, and 6, shown in Figures 4.28, 4.29, 4.30, 4.31, 4.32, and 4.33, respectively, demonstrate that although more points can be generated, precision decreases with increased detection distance. Table 4.10 shows that accuracy at $d_{\text{det}} = 100$ m is lower than at 50 m due to increased noise. However, this noise can be mitigated with advanced tracking and filtering algorithms, improving accuracy and reliability even at greater distances, thus making extended detection range beneficial in practical applications.



(a) Node 3 from S_1.

(b) Node 46 from S_2.

(c) Node 4 from S_1.

(d) Node 47 from S_2.

(e) Node 5 from S_1.

(f) Node 48 from S_2.

(g) Node 52 from S_3.

(h) Node 41 from S_4.

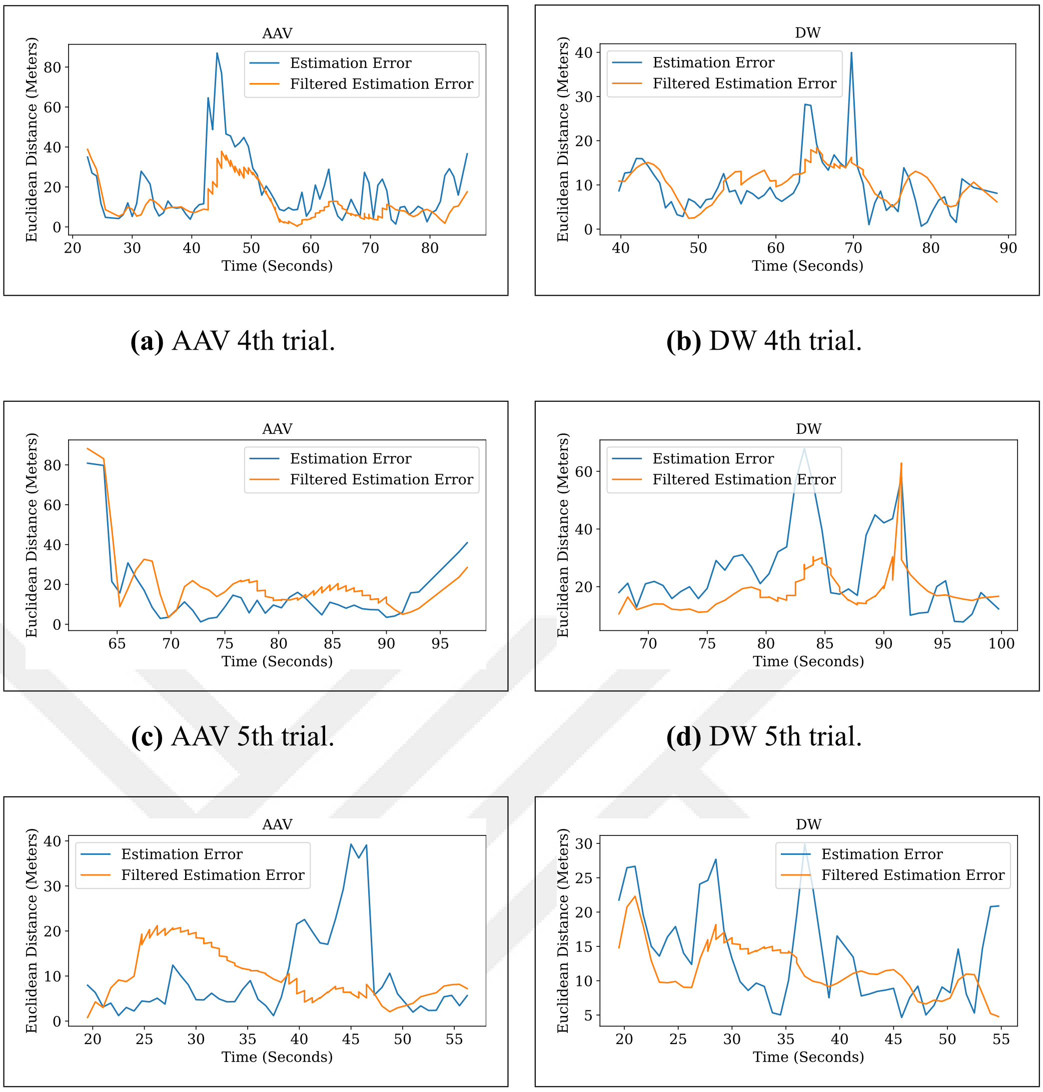
(i) Node 53 from S_3.

(j) Node 55 from S_4.

(k) Node 58 from S_3.

(l) Node 59 from S_4.

Figure 4.26: Target detection for AAV4 at S_1, S_2, S_3 and S_4, d_{det} = 100 m.



(a) AAV 4th trial.

(b) DW 4th trial.

(c) AAV 5th trial.

(d) DW 5th trial.

(e) AAV 6th trial.

(f) DW 6th trial.

Figure 4.27: Target location estimation errors after filtering, d_{det} = 100 m.

We chose the IMM parameters using the statistics for $d_{\text{det}} = 100$ m from Table 4.7. The initial covariance matrix is a diagonal matrix diag$[1, 256, 1, 256, 1, 256]$, and the measurement error is 256 due to σ_{loc} being 16 m for $d_{\text{det}} = 100$ m. We set the transition probability to 0.99 and the maximum number of smoothing steps to 5. The initial state is the first measurement. Filters for the motion models, set as constant velocity, constant acceleration, and constant turn, are all Extended Kalman Filters.

From comparison between Table 4.10 and Table 4.11, we can see that filtering reduced μ_{loc} from 15.26 m to 13.73 m and σ_{loc} from 16.13 m to 8.89 m. As shown in Figure 4.27, only the error for AAV5 is the only trial that shows performance degradation from

Table 4.10 to Table 4.11. This degradation is attributed to some misdetections at the start of AAV5 due to low SNR.

Table 4.10: Trial statistics for $d_{\text{det}} = 100$ m.

	AAV4	AAV5	AAV6	DW4	DW5	DW6
μ_{loc} (m)	17.78	16.19	13.03	12.88	22.10	13.24
σ_{loc} (m)	15.92	14.24	6.40	7.07	15.34	4.46

Table 4.11: Trial statistics for $d_{\text{det}} = 100$ m after filtering.

	AAV4	AAV5	AAV6	DW4	DW5	DW6
μ_{loc} (m)	12.58	18.50	10.66	11.13	19.19	12.33
σ_{loc} (m)	9.65	13.35	5.83	3.78	7.73	3.58

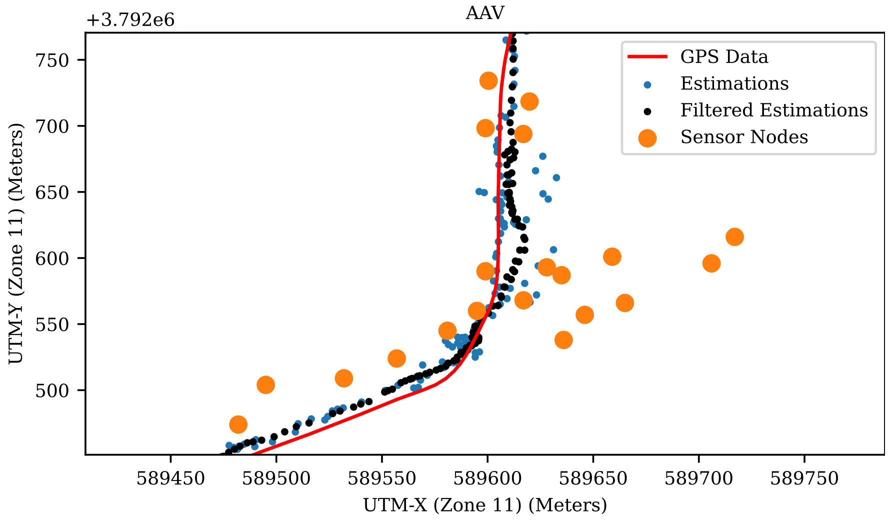


Figure 4.28: Filtered AAV4 target location estimation, $d_{\text{det}} = 100$ m.

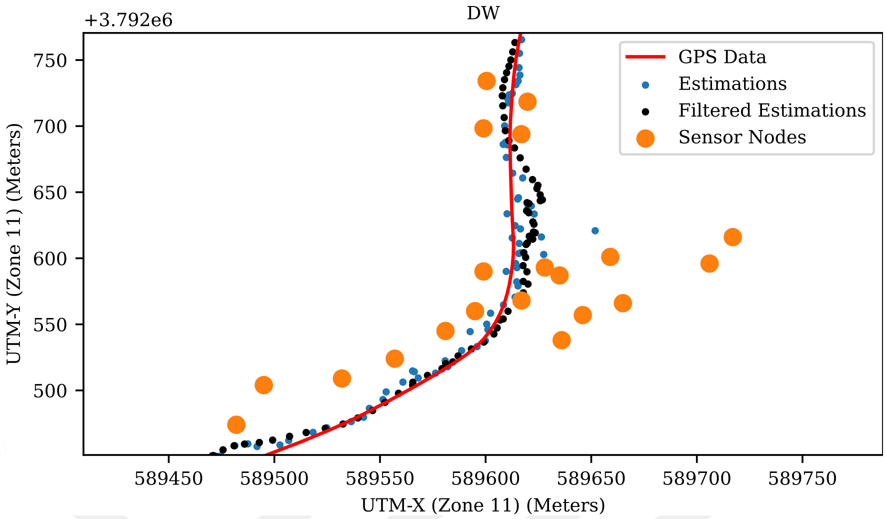


Figure 4.29: Filtered DW4 target location estimation, d_{det} = 100 m.

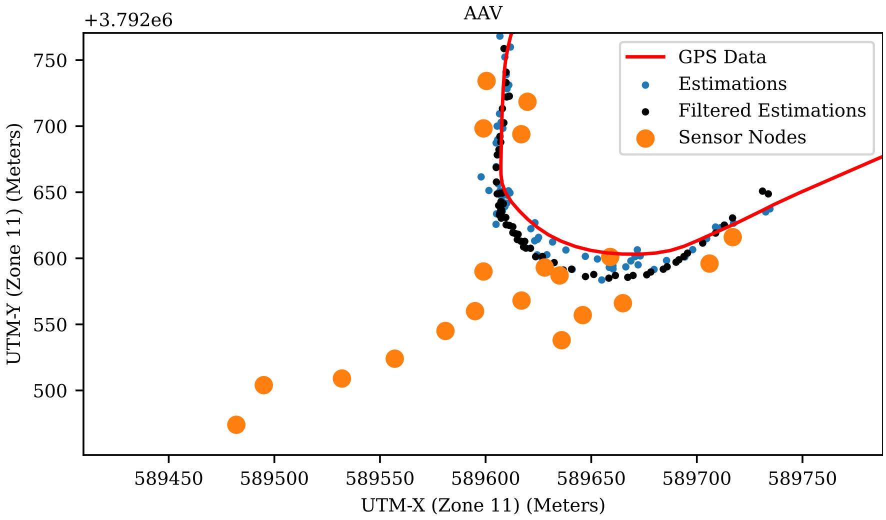


Figure 4.30: Filtered AAV5 target location estimation, d_{det} = 100 m.

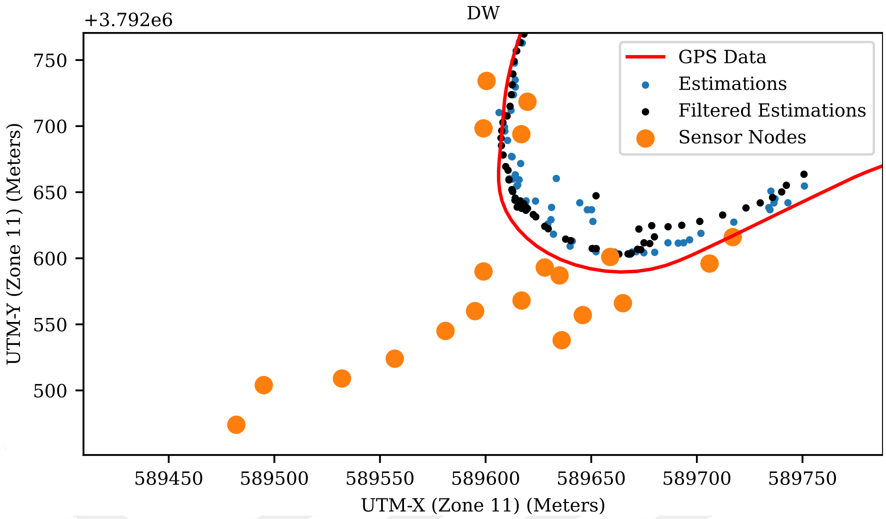


Figure 4.31: Filtered DW5 target location estimation, d_{det} = 100 m.

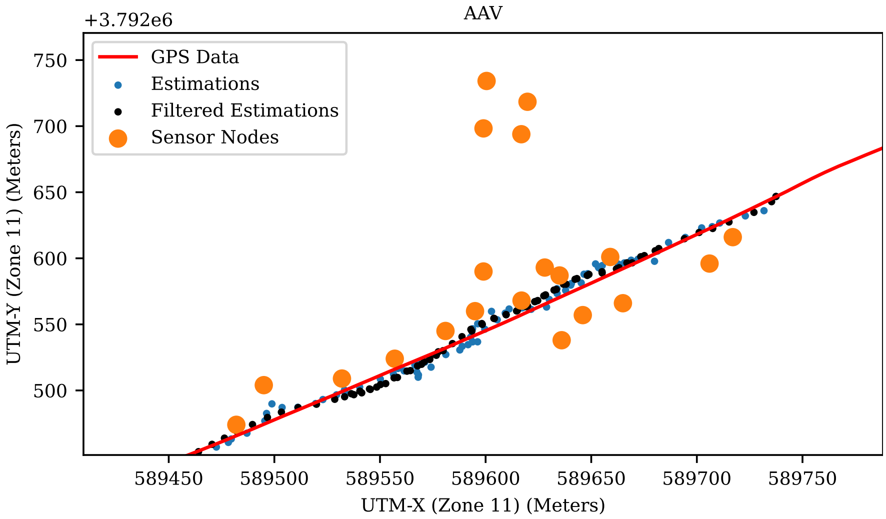


Figure 4.32: Filtered AAV6 target location estimation, d_{det} = 100 m.

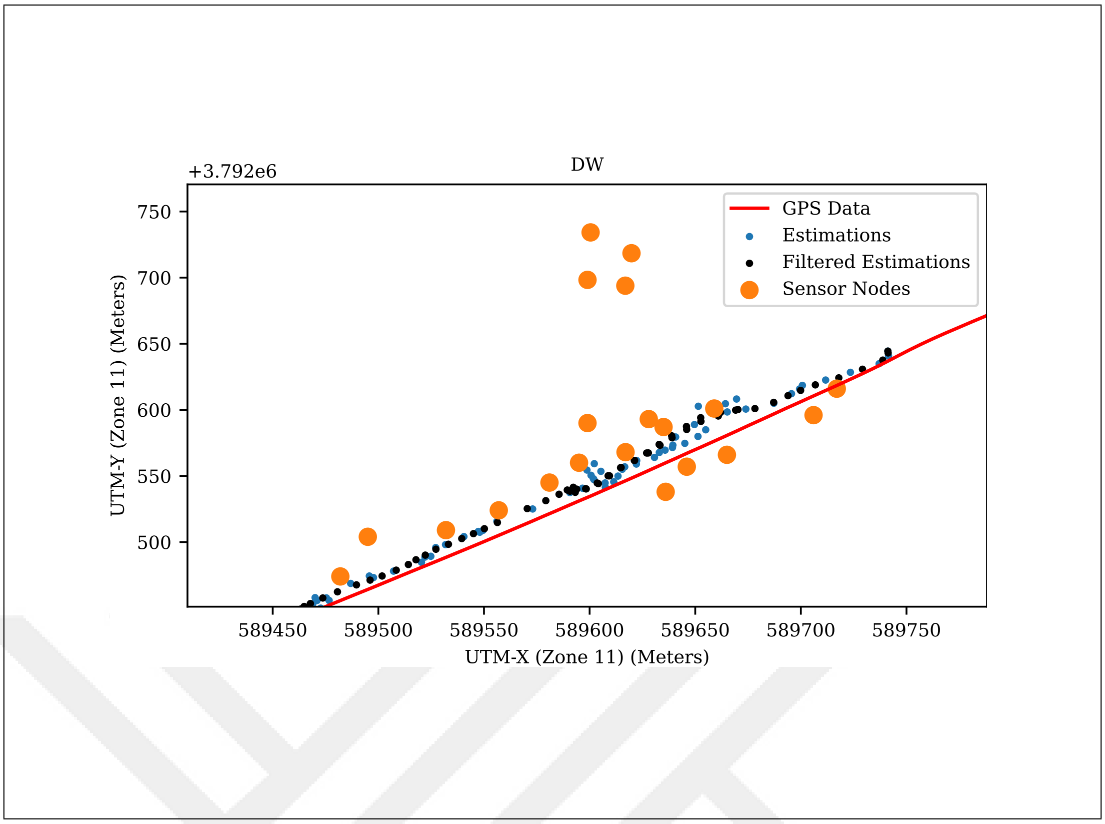


Figure 4.33: Filtered DW6 target location estimation, $d_{\text{det}} = 100$ m.

4.3. Summary

The demonstrated approach showed superior accuracy, attaining a mean localization error of 9.70 m and a standard deviation of 5.91 m for $d_{\text{det}} = 50$ m. When expanded to $d_{\text{det}} = 100$ m and enhanced with IMM filtering, the method exhibited errors with a mean of 13.73 m and a standard deviation of 8.8 m, outperforming existing strategies documented in the academic literature. This method represents a substantial advancement over the distributed regression and centroid techniques previously developed by Le Borgne et al. [70], with comparative results detailed in Table 4.12. Additionally, the work of Sindhu et al. [71], which introduced Symbolic Dynamic Filtering to track specific vehicles near sensor nodes rather than precisely locating them, achieved an average tracking accuracy of 82.4%. This makes direct comparison with our approach challenging. Furthermore, Lee et al. utilized Classification and Localization using Estimated Dynamics and Multimodal Data (CLEDM) [147], leveraging a dataset exclusive to the US Army Research Laboratory [58].

Assumptions underlying the localization errors include the precision of GPS data from SITEX02, estimated to be around 10 meters [63]. Despite some sensor nodes experiencing performance degradation [181], the algorithm remains effective. Additionally, the dimensions of AAV vehicles are documented as 8.15 meters in length and 3.28 meters in width [204], while DW vehicles measure 3.32 meters wide by 7.72 meters long [205]. These dimensions indicate an average vibration source width of approximately 3.3 meters and length of 7.5 meters. Given these factors, the proposed algorithm proves robust for this dataset.

Table 4.12: Localization methods in the literature for SITEX02.

Method	μ_{loc} (m)	σ_{loc} (m)
Trilateration with Energy [70]	175.0	168.0
Centroid [70]	30.0	19.0
Regression [70]	22.0	14.0
Our Method, $d_{\text{det}} = 50$ m	9.7	5.9
Our Method, $d_{\text{det}} = 100$ m with IMM filter	13.7	8.9

A pivotal element in performance evaluation is the process of model inferencing. The feature extraction phase entails executing $N \log_2 N = 49152$ Multiply-Accumulate operations (MACs), which consume merely 52 μs on an AMD Ryzen 7 7745HX—this duration is trivial in contrast to the time required for model inferencing. The detection and regression models both employ a 32-bit floating-point representation, occupying a model size of 700 kB and utilizing 3.5 MB of memory. These models necessitate 48 Mega MACs, culminating in a processing time of 4.2 ms when run as a single-threaded operation on the AMD Ryzen 7 7745HX. Thus, for each block of $T_b M = 2.25$ s, model inferencing requires 8.4 ms, affirming that our model operates in real-time. Given the total computation time of 2.25 seconds and an average vehicle speed of 34 km/h, our model adeptly captures a location shift of up to 19 meters within a detection range of $d_{\text{det}} = 50$ m and 100 m. In contrast to the resource-demanding YOLOv3[164], a prominent model for mobile applications [5], our proposed method manifests significantly lower complexity, requiring just about 33000 Mega MACs. Consequently, our approach is well-suited for deployment on mobile devices.

5. CONCLUSIONS AND FUTURE WORK

5.1. Conclusions

This thesis presents a comprehensive approach for identifying and locating moving targets with the use of distributed networks of seismic sensors. Initially, we developed a method to classify vehicles in a wireless sensor network by utilizing acoustic and seismic data. This method effectively refined frequency domain features while keeping a high rate of correct classification. We used the MUSIC algorithm to locate frequency components accurately and the Welch algorithm to estimate the complete PSD with reduced frequency location variance. Our research showed that the type of road affects the classification accuracy due to the distinct spectral properties of different surfaces.

Additionally, we carried out an in-depth analysis of how time and frequency domain features contribute to the accuracy of vehicle classification. Our results show that combining time and frequency domain features improves classification accuracy and reduces the FPR, particularly for seismic signals, with little effect on acoustic signals. Merging different sensor modalities significantly lowered the FPR and increased the classification rate. Our method achieved a TPR of 98.6% and a FPR of 1.4%, using only 23 features when both acoustic-seismic features from boty time and frequency domain were combined. This strategy of reducing data and features was crucial for further localization tasks.

We introduced a trilateration alternative, using sensor locations and spatio-temporal seismic features as inputs to a ConvLSTM based network for detecting and locating military vehicles in seismic sensor networks. Our study indicated that the number of sensor nodes in clusters and the detection range greatly affect performance. Compared to the distributed regression method, which uses the original detection method, our model with 100 m detection range showed a 30% improvement in localization accuracy. Additionally, using a 50 m detection range resulted in more than a 50% reduction in both the mean and standard deviation of localization errors. This significant improvement underlines the effectiveness of our method in enhancing the accuracy and reliability of localization.

5.2. Future Works

In this section, possible future research problems are presented, based on the findings of this research. Several areas warrant further investigation to enhance the robustness and applicability of the proposed method:

Developing real-time implementation strategies for the proposed algorithms, including optimization for embedded systems and low-power devices, will be crucial for practical deployment.

Conducting extensive field tests in diverse environments to validate the system's performance and reliability under different operational scenarios.

Exploring other machine learning models and architectures, such as Transformer networks, to potentially enhance the detection and localization capabilities.

Investigating the scalability of the proposed method for larger sensor networks, including the development of efficient data fusion and communication protocols.

By addressing these areas, future research can further advance the capabilities of seismic sensor networks for moving target localization, making them more versatile and effective for military and civilian applications alike.

REFERENCES

[1] Sivaraman S., Trivedi M. M., (2013), “Looking at vehicles on the road: a survey of vision-based vehicle detection, tracking, and behavior analysis”, IEEE Transactions on Intelligent Transportation Systems, 14(4), 1773–1795.

[2] Liu K., Mattyus G., (2015), “Fast multiclass vehicle detection on aerial images”, IEEE Geoscience and Remote Sensing Letters, 12(9), 1938–1942.

[3] Sun Y., Cao B., Zhu P., Hu Q., (2022), “Drone-based RGB-infrared cross-modality vehicle detection via uncertainty-aware learning”, IEEE Transactions on Circuits and Systems for Video Technology, 32(10), 6700–6713.

[4] Fayyad J., Jaradat M. A., Gruyer D., Najjaran H., (2020), “Deep learning sensor fusion for autonomous vehicle perception and localization: A review”, Sensors, 20(15), 1–35.

[5] Martinez-Alpiste I., Golcarenarenji G., Wang Q., Alcaraz-Calero J. M., (2022), “Smartphone-based real-time object recognition architecture for portable and constrained systems”, Journal of Real-Time Image Processing, 19(1), 103–115.

[6] Padmavathi G., Shanmugapriya D., Kalaivani M., (2010), “A study on vehicle detection and tracking using wireless sensor networks”, Wireless Sensor Network, 2(1), 173–185.

[7] Yang B., Lei Y., (2015), “Vehicle detection and classification for low-speed congested traffic with anisotropic magnetoresistive sensor”, IEEE Sensors Journal, 15(2), 1132–1138.

[8] Nellore K., Hancke G. P., (2016), “A survey on urban traffic management system using wireless sensor networks”, Sensors, 16(2), 1–25.

[9] Rashid B., Rehmani M. H., (2016), “Applications of wireless sensor networks for urban areas: A survey”, Journal of Network and Computer Applications, 60(1), 192–219.

[10] Odat E., Shamma J. S., Claudel C., (2017), “Vehicle classification and speed estimation using combined passive infrared/ultrasonic sensors”, IEEE Transactions on Intelligent Transportation Systems, 19(5), 1593–1606.

[11] Bernas M., Płaczek B., Korski W., Loska P., Smyła J., Szymała P., (2018), “A survey and comparison of low-cost sensing technologies for road traffic monitoring”, Sensors, 18(10), 1–27.

[12] Atrey P. K., Maddage N. C., Kankanhalli M. S., (2006), “Audio based event detection for multimedia surveillance”, IEEE International Conference on Acoustics Speech and Signal Processing Proceedings, 813–816, Toulouse, FRA, 14-19 May.

[13] Kakar V. K., Kandpal M., (2013), "Techniques of acoustic feature extraction for detection and classification of ground vehicles", International Journal of Emerging Technology and Advanced Engineering, 3(2), 419–426.

[14] Allen R. M., Melgar D., (2019), "Earthquake early warning: Advances, scientific challenges, and societal needs", Annual Review of Earth and Planetary Sciences, 47(1), 361–388.

[15] Aljaafreh A., Al-Fuqaha A., (2010), "Multi-target classification using acoustic signatures in wireless sensor networks: A survey", Signal Processing: An International Journal, 4(4), 175–200.

[16] Mayvan A. D., Beheshti A. A., Masoom M. H., (2015), "Classification of vehicles based on audio signals using quadratic discriminant analysis and high energy feature vectors", International Journal on Soft Computing, 6(1), 53–64.

[17] Wieczorkowska A., Kubera E., Słowik T., Skrzypiec K., (2018), "Spectral features for audio based vehicle and engine classification", Journal of Intelligent Information Systems, 50(1), 265–290.

[18] Xing K., Wang N., Wang W., (2022), "A ground moving target detection method for seismic and sound sensor based on evolutionary neural networks", Applied Sciences, 12(18), 9343.

[19] Ashhad M., Ahmed O., Ambat S. K., Haq Z. A., Alam M., (2023), "MVD: A novel methodology and dataset for acoustic vehicle type classification", arXiv:2309.03544, 1–6.

[20] Ajdler T., Kozintsev I., Lienhart R., Vetterli M., (2004), "Acoustic source localization in distributed sensor networks", Asilomar Conference on Signals, Systems and Computers, 1328–1332, Pacific Grove, CA, USA, 7-10 November.

[21] Cobos M., Antonacci F., Alexandridis A., Mouchtaris A., Lee B., (2017), "A survey of sound source localization methods in wireless acoustic sensor networks", Wireless Communications and Mobile Computing, 2017(1), 1–25.

[22] Meng W., Xiao W., (2017), "Energy-based acoustic source localization methods: a survey", Sensors, 17(2), 1–20.

[23] Correia S. D., Tomic S., Beko M., (2021), "A feed-forward neural network approach for energy-based acoustic source localization", Journal of Sensor and Actuator Networks, 10(2), 1–16.

[24] Grumiaux P.-A., Kitić S., Girin L., Guérin A., (2021), "Improved feature extraction for CRNN-based multiple sound source localization", European Signal Processing Conference, 231–235, Dublin, IRL, 23-27 August.

[25] Tan T.-H., Lin Y.-T., Chang Y.-L., Alkhaleefah M., (2021), "Sound source localization using a convolutional neural network and regression model", Sensors, 21(23), 1–17.

[26] Wu Y., Ayyalasomayajula R., Bianco M. J., Bharadia D., Gerstoft P., (2021), "SSLIDE: Sound source localization for indoors based on deep learning", International Conference on Acoustics, Speech and Signal Processing, 4680–4684, Toronto, ON, CAN, 06-11 June.

[27] Grumiaux P.-A., Kitić S., Girin L., Guérin A., (2022), "A survey of sound source localization with deep learning methods", Journal of the Acoustical Society of America, 152(1), 107–151.

[28] Bina K., Lina J., Tonga X., Zhang X., Wanga J., Luoa S., (2021), "Moving target recognition with seismic sensing: a review", Measurement, 181(1), 1–17.

[29] Choudhary P., Goel N., Saini M., (2023), "A survey on seismic sensor based target detection, localization, identification, and activity recognition", ACM Computing Surveys, 55(11), 1–36.

[30] Succi G. P., Prado G., Gampert R., Pedersen T. K., Dhaliwal H., (2000), "Problems in seismic detection and tracking", AeroSense: Unattended Ground Sensor Technologies and Applications II, 165–174, Orlando, FL, USA, 24-28 April.

[31] Houston K. M., McGaffigan D. P., (2003), "Spectrum analysis techniques for personnel detection using seismic sensors", AeroSense: Unattended Ground Sensor Technologies and Applications V, 162–174, Orlando, FL, USA, 21-25 April.

[32] Bahrampour S., Ray A., Sarkar S., Damarla T., Nasrabadi N. M., (2013), "Performance comparison of feature extraction algorithms for target detection and classification", Pattern Recognition Letters, 34(16), 2126–2134.

[33] Cheng L., Wu C., Zhang Y., Wu H., Li M., Maple C., (2012), "A survey of localization in wireless sensor network", International Journal of Distributed Sensor Networks, 8(12), 1–12.

[34] Han G., Xu H., Duong T. Q., Jiang J., Hara T., (2013), "Localization algorithms of wireless sensor networks: a survey", Telecommunication Systems, 52(4), 2419–2436.

[35] Mesmoudi A., Feham M., Labraoui N., (2013), "Wireless sensor networks localization algorithms: a comprehensive survey", International Journal of Computer Networks & Communications, 5(6), 1–20.

[36] Chowdhury T. J., Elkin C., Devabhaktuni V., Rawat D. B., Oluoch J., (2016), "Advances on localization techniques for wireless sensor networks: A survey", Computer Networks, 110(1), 284–305.

[37] Paul A. K., Sato T., (2017), "Localization in wireless sensor networks: A survey on algorithms, measurement techniques, applications and challenges", Journal of Sensor and Actuator Networks, 6(4), 24.

[38] Naidu P. S., (2018), “Distributed Sensor Arrays: Localization”, 1st Edition, CRC Press.

[39] Xing H., Li F., Xiao H., Wang Y., Liu Y., (2007), “Ground target detection, classification, and sensor fusion in distributed fiber seismic sensor network”, Photonics Asia: Advanced Sensor Systems and Applications, 1–10, Beijing, CHN, 11-15 November.

[40] Narayanaswami R., Gandhe A., Tyurina A., Mehra R. K., (2010), “Sensor fusion and feature-based human/animal classification for unattended ground sensors”, IEEE International Conference on Technologies for Homeland Security, 344–350, Waltham, MA, USA, 08-10 November.

[41] Zubair M., Hartmann K., (2011), “Target classification based on sensor fusion in multi-channel seismic network”, IEEE International Symposium on Signal Processing and Information Technology, 438–443, Bilbao, ESP, 14-17 December.

[42] Zhang W., Zhang Z., (2015), “Belief function based decision fusion for decentralized target classification in wireless sensor networks”, Sensors, 15(8), 20524–20540.

[43] Li Y., Jha D. K., Ray A., Wettergren T. A., (2016), “Information fusion of passive sensors for detection of moving targets in dynamic environments”, IEEE Transactions on Cybernetics, 47(1), 93–104.

[44] Guo H., Hong H., (2019), “Research on filtering algorithm of MEMS gyroscope based on information fusion”, Sensors, 19(16), 1–20.

[45] Telford W. M., Geldart L. P., Sheriff R. E., (1990), “Applied Geophysics”, 2nd Edition, Cambridge University Press.

[46] Krylov V. V., (2001), “Generation of ground elastic waves by road vehicles”, Journal of Computational Acoustics, 9(3), 919–933.

[47] Lacombe J., Peck L., Anderson T., Fisk D., (2006), “Seismic detection algorithm and sensor deployment recommendations for perimeter security”, Defense and Security Symposium: Unattended Ground, Sea, and Air Sensor Technologies and Applications VIII, 1–10, Orlando, FL, USA, 17-21 April.

[48] Anderson T. S., Moran M. L., Ketcham S. A., Lacombe J., (2004), “Tracked vehicle simulations and seismic wavefield synthesis in seismic sensor systems”, Computing in Science & Engineering, 6(6), 22–28.

[49] Evans N., (2010), “Automated Vehicle Detection and Classification Using Acoustic and Seismic Signals”, PhD Thesis, University of York.

[50] William P., (2011), “Low Complexity Feature Extraction for Classification of Harmonic Signals”, PhD Thesis, University of Nebraska–Lincoln.

[51] Mohammad Shafiei A., (2015), “An Energy-Efficient Target Tracking Protocol Using Wireless Sensor Networks”, PhD Thesis, University of Ottawa.

[52] Balid W., (2016), “Fully Autonomous Self-Powered Intelligent Wireless Sensor for Real-Time Traffic Surveillance in Smart Cities”, PhD Thesis, University of Oklahoma.

[53] Vilhelmsson R., (2017), “Vibration Analysis for Speed Estimation of Wheeled Indoor Vehicles”, MSc Thesis, Linköping University.

[54] Poston J. D., (2018), “ILoViT: Indoor Localization via Vibration Tracking”, PhD Thesis, Virginia Tech.

[55] Wu Q., (2018), “Algorithms for Multiple Ground Target Tracking”, PhD Thesis, McMaster University.

[56] Altmann J., Linev S., Weiß A., (2002), “Acoustic–seismic detection and classification of military vehicles—developing tools for disarmament and peace-keeping”, Applied Acoustics, 63(10), 1085–1107.

[57] Duarte M. F., Hu Y. H., (2004), “Vehicle classification in distributed sensor networks”, Journal of Parallel and Distributed Computing, 64(7), 826–838.

[58] Nabritt S. M., Damarla T., Chatters G., (2015), “Personnel and Vehicle Data Collection at Aberdeen Proving Ground (APG) and its Distribution for Research”, Technical Report No: ARL-MR-0909, US Army Research Laboratory, Adelphi, MD, USA.

[59] Gouda A., Heinrich D., Hünnefeld M., Priyanta I. F., Reining C., Roidl M., (2022), “A grid-based sensor floor platform for robot localization using machine learning”, International Instrumentation and Measurement Technology Conference, 1–6, Kuala Lumpur, MYS, 22-25 May.

[60] Yuan L., Chen H., Ewing R., Blasch E. P., Li J., (2023), “3-D Indoor positioning based on passive radio frequency signal strength distribution”, IEEE Internet of Things Journal, 10(15), 13933–13944.

[61] Li D., Wong K. D., Hu Y. H., Sayeed A. M., (2002), “Detection, classification, and tracking of targets”, IEEE Signal Processing Magazine, 19(2), 17–29.

[62] Kumar S., Shepherd D., (2001), “SensIT: Sensor information technology for the warfighter”, International Conference on Information Fusion, 1–7, Montréal, QC, CAN, 7-10 August.

[63] Boettcher P. W., Shaw G. A., Sherman J., (2001), “A distributed time-difference of arrival algorithm for acoustic bearing estimation”, International Conference on Information Fusion, 1–6, Montréal, QC, CAN, 7-10 August.

[64] Boettcher P. W., Sherman J. A., Shaw G. A., (2002), “Target localization using acoustic time-difference of arrival in distributed sensor networks”, AeroSense: Battlespace Digitization and Network-Centric Warfare, 180–191, Orlando, FL, USA, 1-5 April.

[65] Boettcher P. W., Shaw G. A., (2003), “Energy-constrained collaborative processing for target detection, tracking, and geolocation”, Information Processing in Sensor Networks, 254–268, Palo Alto, CA, USA, 22-23 April.

[66] Brooks R. R., Ramanathan P., Sayeed A. M., (2003), “Distributed target classification and tracking in sensor networks”, Proceedings of the IEEE, 91(8), 1163–1171.

[67] Sheng X., Hu Y. H., (2003), “Collaborative source localization in wireless sensor network system”, Technical report no:, San Diego State University, Masters of Homeland Security, San Diego, CA ,USA.

[68] Duarte M., Hu Y.-H., (2004), “Distance-based decision fusion in a distributed wireless sensor network”, Telecommunication Systems, 26(2), 339–350.

[69] Rahman M. Z., Habibi D., Ahmad I., (2008), “Source localisation in wireless sensor networks based on optimised maximum likelihood”, Australasian Telecommunication Networks and Applications Conference, 235–239, Adelaide, SA, AUS, 7-10 December.

[70] Le Borgne Y.-A., Nowé A., Abughalieh N., Steenhaut K., (2010), “Distributed regression for high-level feature extraction in wireless sensor networks”, International Conference on Networked Sensing Systems, 249–252, Kassel, DEU, 15-18 June.

[71] Sindhu N., Empran S., Uttarakumari M., Badiger S. D., (2018), “Target classification and tracking using symbolic dynamic filtering”, International Conference on Computing, Communication and Networking Technologies, 1–6, Bengaluru, IND, 10-12 July.

[72] https://cdn.sparkfun.com/datasheets/Sensors/Accelerometers/SM-24%20Brochure.pdf. Access Date: 10.03.2024.

[73] https://www.nxp.com/docs/en/data-sheet/MMA7361L.pdf. Access Date: 10.03.2024.

[74] https://www.sparkfun.com/datasheets/Sensors/Flex/MiniSense_100.pdf. Access Date: 10.03.2024.

[75] Mazarakis G. P., Avaritsiotis J. N., (2007), “Vehicle classification in sensor networks using time-domain signal processing and neural networks”, Microprocessors and Microsystems, 31(6), 381–392.

[76] Kornaropoulos E. M., Tsakalides P., (2009), “A novel KNN classifier for acoustic vehicle classification based on alpha-stable statistical modeling”, IEEE/SP Workshop on Statistical Signal Processing, 1–4, Cardiff, GBR, 31 August-03 September.

[77] Wang K., Wang R., Feng Y., Zhang H., Huang Q., Jin Y., Zhang Y., (2014), “Vehicle recognition in acoustic sensor networks via sparse representation”, IEEE International Conference on Multimedia and Expo Workshops, 1–4, Chengdu, CHN, 14-18 July.

[78] Taheri S. M., Nosrati H., (2014), “Acoustic signature identification using distributed diffusion adaptive networks”, International Symposium on Communication Systems, Networks and Digital Signal Processing, 943–948, Manchester, GBR, 23-25 July.

[79] Wang R., Guo S., Li Y., Zhang Y., (2017), “Fisher discriminative dictionary learning for vehicle classification in acoustic sensor networks”, Journal of Signal Processing Systems, 86(1), 99–107.

[80] Ghosh R., Sardana H. K., (2020), “Multi-feature optimization strategies for target classification using seismic and acoustic signatures”, SPIE Defense + Commercial Sensing: Automatic Target Recognition, 1–4, Online Only, CA, USA, 27 April - 9 May.

[81] Kalra M., Kumar S., Das B., (2021), “Analysis of instantaneous amplitude and frequency of EWT modes for automatic target classification”, IEEE Bombay Section Signature Conference, 1–4, Gwalior, IND, 18-20 November.

[82] Rahman M., (2011), “Applications of Fourier transforms to generalized functions”, 1st Edition, WIT Press.

[83] Cooley J. W., Tukey J. W., (1965), “An algorithm for the machine calculation of complex Fourier series”, Mathematics of Computation, 19(90), 297–301.

[84] Bluestein L., (1970), “A linear filtering approach to the computation of discrete Fourier transform”, IEEE Transactions on Audio and Electroacoustics, 18(4), 451–455.

[85] O’Leary G. C., (1970), “Nonrecursive digital filtering using cascade fast Fourier transformers”, IEEE Transactions on Audio and Electroacoustics, 18(2), 177–183.

[86] Groginsky H. L., Works G. A., (1970), “A pipeline fast Fourier transform”, IEEE Transactions on Computers, C-19(11), 1015–1019.

[87] Frigo M., Johnson S. G., (2005), “The design and implementation of FFTW3”, Proceedings of the IEEE, 93(2), 216–231.

[88] Garrido M., (2022), “A survey on pipelined FFT hardware architectures”, Journal of Signal Processing Systems, 94(11), 1345–1364.

[89] Wigner E., (1932), “On the quantum correction for thermodynamic equilibrium”, Physical Review, 40(5), 749–759. Publisher: APS.

[90] Wold H., (1938), “A Study in The Analysis of Stationary Time Series”, 2nd Edition, Almqvist & Wiksell.

[91] Udny Yule G., (1927), “On a method of investigating periodicities in disturbed series, with special reference to Wolfer’s sunspot numbers”, Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character, 226(1), 267–298.

[92] Walker G. T., (1931), “On periodicity in series of related terms”, Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character, 131(818), 518–532.

[93] Kay S. M., (1988), “Modern Spectral Estimation: Theory and Application”, 1st Edition, Pearson Education.

[94] Davis S., Mermelstein P., (1980), “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences”, IEEE Transactions on Acoustics, Speech, and Signal Processing, 28(4), 357–366.

[95] Abdul Z. K., Al-Talabani A. K., (2022), “Mel frequency cepstral coefficient and its applications: A review”, IEEE Access, 10(1), 122136–122158.

[96] Guo S., Wang R., Liu B., Wei Q., Li Y., (2016), “Vehicle classification in acoustic sensor networks based on hybrid dictionary learning”, IEEE International Symposium on Dependable, Autonomic and Secure Computing, 861–865, Auckland, NZL, 08-12 August.

[97] Zhou X., Garcia-Romero D., Duraiswami R., Espy-Wilson C., Shamma S., (2011), “Linear versus mel frequency cepstral coefficients for speaker recognition”, IEEE Workshop on Automatic Speech Recognition & Understanding, 559–564, Waikoloa, HI, USA, 11-15 December.

[98] Chakroborty S., Roy A., Saha G., (2008), “Improved closed set text-independent speaker identification by combining MFCC with evidence from flipped filter banks”, International Journal of Electronics and Communication Engineering, 2(11), 2554–2561.

[99] Zhou Q., Tong G., Xie D., Li B., Yuan X., (2012), “A seismic-based feature extraction algorithm for robust ground target classification”, IEEE Signal Processing Letters, 19(10), 639–642.

[100] Wang H., Quan W., Wang Y., Miller G. R., (2014), “Dual roadside seismic sensor for moving road vehicle detection and characterization”, Sensors, 14(2), 2892–2910.

[101] Ghosh R., Akula A., Kumar S., Sardana H., (2015), “Time–frequency analysis based robust vehicle detection using seismic sensor”, Journal of Sound and Vibration, 346(1), 424–434.

[102] Nhat T. D., Phuong M. N. T., (2018), “Analyzing seismic signal using support vector machine for vehicle motion detection”, International Conference on Industrial Networks and Intelligent Systems, 178–190, Da Nang, VNM, August 27–28.

[103] Bernas M., Korski W., Płaczek B., Smyła J., (2019), “Vehicle in motion weighing based on vibration data collected from sensor network”, International Conference on Computer Networks, 208–219, Kamień Śląski, POL, 25-27 June.

[104] Kalra M., Kumar S., Das B., (2020), “Seismic signal analysis using empirical wavelet transform for moving ground target detection and classification”, IEEE Sensors Journal, 20(14), 7886–7895.

[105] Peck L., Lacombe J., (2007), “Seismic-based personnel detection”, IEEE International Carnahan Conference on Security Technology, 169–175, Ottawa, ON, CAN, 8-11 October.

[106] Pakhomov A., Goldburt T., (2008), “Zero false alarm seismic detection and identification systems”, Defense and Security Symposium: Sensors, and Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Security and Homeland Defense VII, 1–12, Orlando, FL, USA, 16-20 March.

[107] Ekimov A., Sabatier J. M., (2009), “Human motion characterization”, Human Light Vehicle and Tunnel Detection Workshop, 1–10, Hanover, DEU, June 16-17.

[108] Faghfouri A. E., Frish M. B., (2011), “Robust discrimination of human footsteps using seismic signals”, SPIE Defense, Security, and Sensing: Unattended Ground, Sea, and Air Sensor Technologies and Applications, 1–9, Orlando, FL, USA, 25-29 April.

[109] Koç G., Yegin K., (2013), “Footstep and vehicle detection using seismic sensors in wireless sensor network: Field tests”, International Journal of Distributed Sensor Networks, 9(10), 1–8.

[110] Anchal S., Mukhopadhyay B., Kar S., (2017), “UREDT: Unsupervised learning based real-time footfall event detection technique in seismic signal”, IEEE Sensors Letters, 2(1), 1–4.

[111] Mukhopadhyay B., Anchal S., Kar S., (2017), “Detection of an intruder and prediction of his state of motion by using seismic sensor”, IEEE Sensors Journal, 18(2), 703–712.

[112] Anchal S., Mukhopadhyay B., Kar S., (2020), “Person identification and imposter detection using footstep generated seismic signals”, IEEE Transactions on Instrumentation and Measurement, 70(1), 1–11.

[113] Goodman G. L., (1999), “Detection and classification for unattended ground sensors”, Information, Decision and Control: Signal Processing and Communications Symposium, 419–424, Adelaide, SA, AUS, 8-10 February.

[114] Gramann R. A., Bennett M. B., O’Brien T. D., (1999), “Vehicle and personnel detection using seismic sensors”, Enabling Technologies for Law Enforcement and Security: Sensors, C3I, Information, and Training Technologies for Law Enforcement, 74–86, Boston, MA, USA, 1-6 November.

[115] Pakhomov A., Goldburt T., (2006), “Seismic systems for unconventional target detection and identification”, Defense and Security Symposium: Sensors, and Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Security and Homeland Defense V, 1–12, Orlando, FL, USA, 17-21 April.

[116] Jin X., Gupta S., Ray A., Damarla T., (2011*a*), “Symbolic dynamic filtering of seismic sensors for target detection and classification”, American Control Conference, 5151–5156, San Francisco, CA, USA, 29 June-1 July.

[117] Jin X., Sarkar S., Ray A., Gupta S., Damarla T., (2011*b*), “Target detection and classification using seismic and PIR sensors”, IEEE Sensors Journal, 12(6), 1709–1718.

[118] Küçükbay S. E., Sert M., Yazici A., (2017), “Use of acoustic and vibration sensor data to detect objects in surveillance wireless sensor networks”, International Conference on Control Systems and Computer Science, 207–212, Bucharest, ROU, 29-31 May.

[119] Ghosh R., Vajpeyi A., Akula A., Shaw V., Kumar S., Sardana H., (2019), “Performance evaluation of a real-time seismic detection system based on CFAR detectors”, IEEE Sensors Journal, 20(7), 3678–3686.

[120] Morozov Y. V., Rajfeld M. A., Spektor A. A., (2019), “Non-parametric detection of seismic signals by two parts of the same amplitude spectrum”, Spatial Data Processing for Monitoring of Natural and Anthropogenic Processes, 137–141, Berdsk, RUS, 26-30 August.

[121] Moriya H., (2011), “Phase-only correlation of time-varying spectral representations of microseismic data for identification of similar seismic events”, Geophysics, 76(6), WC37–WC45.

[122] Ruano A. E., Madureira G., Barros O., Khosravani H. R., Ruano M. G., Ferreira P. M., (2014), “Seismic detection using support vector machines”, Neurocomputing, 135(1), 273–283.

[123] Lara-Cueva R., Bernal P., Saltos M. G., Benítez D. S., Rojo-Álvarez J. L., (2015), “Time and frequency feature selection for seismic events from Cotopaxi Volcano”, Asia-Pacific Conference on Computer Aided System Engineering, 129–134, Quito, ECU, 14-16 July.

[124] Wu S., Wang Y., Zhan Y., Chang X., (2016), “Automatic microseismic event detection by band-limited phase-only correlation”, Physics of the Earth and Planetary Interiors, 261(A), 3–16.

[125] Moriya H., (2018), “Identification of similar seismic events using a phase-only correlation technique”, Construction and Building Materials, 168(1), 984–987.

[126] Namitha C., Uma Mahesh V., Anusha M., Koteswara Rao S., Vaishnavi Chandra T., (2018), “Frequency estimation using minimum norm algorithm on seismic data”, International Conference on Microelectronics, Electromagnetics and Telecommunications, 153–163, Hyderabad, TS, IND, 9-10 September.

[127] Wu Y., Lin Y., Zhou Z., Bolton D. C., Liu J., Johnson P., (2018), “DeepDetect: A cascaded region-based densely connected network for seismic event detection”, IEEE Transactions on Geoscience and Remote Sensing, 57(1), 62–75.

[128] Tian Y., Qi H., Wang X., (2002), “Target detection and classification using seismic signal processing in unattended ground sensor systems”, IEEE International Conference on Acoustics Speech and Signal Processing, 4172–4172, Orlando, FL, USA, 13-17 May.

[129] Keally M., Zhou G., Xing G., (2010), “Watchdog: Confident event detection in heterogeneous sensor networks”, IEEE Real-Time and Embedded Technology and Applications Symposium, 279–288, Stockholm, SWE, 12-15 April.

[130] Kalra M., Kumar S., Das B., (2018), “Target detection using smooth pseudo Wigner-Ville distribution”, IEEE Recent Advances in Intelligent Computational Systems, 6–10, Thiruvananthapuram, IND, 6-8 December.

[131] Kalra M., Kumar S., Das B., (2019), “Moving ground target detection with seismic signal using smooth pseudo Wigner–Ville distribution”, IEEE Transactions on Instrumentation and Measurement, 69(6), 3896–3906.

[132] Abidi N. A., Karim N., Althahabi A. M., Mezaal Y. S., Ali M. J., (2023), “Poignant ground target recognition with Smooth Pseudo Wigner-Ville Distribution by analysis of time-frequency techniques”, International Conference on Computational Intelligence, Communication Technology and Networking, 690–694, Ghaziabad, IND, 20-21 April.

[133] Nathani N., Bera U. N., Bhalerao S., (2023), “Investigation of seismic signal processing for detection and classification of moving unarmoured vehicles”, International Conference on Innovations in High Speed Communication and Signal Processing, 21–24, Bhopal, IND, 4-5 March.

[134] Bin K., Long Y., Tong X., Lin J., (2021), “Ground moving target detection with seismic fractal features”, IEEE Geoscience and Remote Sensing Letters, 19(1), 1–5.

[135] Richards M. A., (2022), “Fundamentals of Radar Signal Processing”, 3rd Edition, McGraw-Hill.

[136] Finn H. M., Johnson R. S., (1968), “Adaptive detection mode with threshold control as a function of spatially sampled clutter-level estimates”, RCA Review, 29(3), 414–464.

[137] Kuglin C. D., (1975), “The phase correlation image alignment method”, IEEE International Conference on Cybernetics and Society, 163–165, San Francisco, CA, USA, 23-25 September.

[138] Takita K., Aoki T., Sasaki Y., Higuchi T., Kobayashi K., (2003), “High-accuracy subpixel image registration based on phase-only correlation”, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, E86-A(8), 1925–1934.

[139] Ito I., Kiya H., (2007), “DCT sign-only correlation with application to image matching and the relationship with phase-only correlation”, IEEE International Conference on Acoustics, Speech and Signal Processing, I1237–I1240, Honolulu, HI, USA, 15-20 April.

[140] Asaari M. S. M., Suandi S. A., Rosdi B. A., (2014), “Fusion of band limited phase only correlation and width centroid contour distance for finger based biometrics”, Expert Systems with Applications, 41(7), 3367–3382.

[141] Yoo J.-C., Han T. H., (2009), “Fast normalized cross-correlation”, Circuits, Systems and Signal Processing, 28(1), 819–843.

[142] Pan S., Yu T., Mirshekari M., Fagert J., Bonde A., Mengshoel O. J., Noh H. Y., Zhang P., (2017), “FootprintID: Indoor pedestrian identification through ambient structural vibration sensing”, Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 1(3), 1–31.

[143] Xu T., Wang N., Xu X., (2019), “Seismic target recognition based on parallel recurrent neural network for unattended ground sensor systems”, IEEE Access, 7(1), 137823–137834.

[144] Lan J., Lan T., Nahavandi S., (2004), “A novel application of a microaccelerometer for target classification”, IEEE Sensors Journal, 4(4), 519–524.

[145] Lan J., Nahavandi S., Lan T., Yin Y., (2005), “Recognition of moving ground targets by measuring and processing seismic signal”, Measurement, 37(2), 189–199.

[146] Du K., Fang X., Zhang W.-p., Ding K., (2016), “Fractal dimension based on morphological covering for ground target classification”, Shock and Vibration, 2016(4), 1–5.

[147] Lee K., Riggan B. S., Bhattacharyya S. S., (2018), “A joint target localization and classification framework for sensor networks”, International Conference on Acoustics, Speech and Signal Processing, 3076–3080, Calgary, AB, CAN, 15-20 April.

[148] Zhang X., Yang X., Song E., Li B., Yuan X., (2012), “A distance segmented classification algorithm for ground moving targets based on seismic-acoustic sensor arrays and parallel fuzzy classifier”, Journal of Computational Information Systems, 8(7), 2789–2798.

[149] Huang J., Zhou Q., Zhang X., Song E., Li B., Yuan X., (2013), “Seismic target classification using a wavelet packet manifold in unattended ground sensors systems”, Sensors, 13(7), 8534–8550.

[150] Büyüktepe I. C., Hocaoğlu A. K., (2022), “Classification of moving ground targets using measurement from accelerometer on road surface”, Signal Processing and Communications Applications Conference, 1–4, Safranbolu, TUR, 15-18 May.

[151] Langer H., Falsaperla S., Powell T., Thompson G., (2006), “Automatic classification and a-posteriori analysis of seismic event identification at Soufriere Hills volcano, Montserrat”, Journal of Volcanology and Geothermal Research, 153(1), 1–10.

[152] Zhang G., Lin C., Chen Y., (2020), “Convolutional neural networks for microseismic waveform classification and arrival picking”, Geophysics, 85(4), 227–240.

[153] Kim Y., Jeong S., Kim D., López T. S., (2009), “An efficient scheme of target classification and information fusion in wireless sensor networks”, Personal and Ubiquitous Computing, 13(7), 499–508.

[154] Ntalampiras S., (2018), “Moving vehicle classification using wireless acoustic sensor networks”, IEEE Transactions on Emerging Topics in Computational Intelligence, 2(2), 129–138.

[155] Jin G., Ye B., Wu Y., Qu F., (2018), “Vehicle classification based on seismic signatures using convolutional neural network”, IEEE Geoscience and Remote Sensing Letters, 16(4), 628–632.

[156] Zhang H., Pan Z., Zhang W., (2018), “Acoustic–seismic mixed feature extraction based on wavelet transform for vehicle classification in wireless sensor networks”, Sensors, 18(6), 1862.

[157] Jin G., (2019), “Vehicle classification based on seismic signatures with weighted intrinsic mode functions”, arXiv:1902.09981, 1–4.

[158] Bin K., Lin J., Tong X., (2021), “Edge intelligence-based moving target classification using compressed seismic measurements and convolutional neural networks”, IEEE Geoscience and Remote Sensing Letters, 19(1), 1–5.

[159] LeCun Y., Bengio Y., (1998), “Convolutional Networks for Images, Speech, And Time Series”, In: M. A. Arbib, Editor, “The Handbook of Brain Theory and Neural Networks”, MIT Press.

[160] Fernández S., Graves A., Schmidhuber J., (2007), “An application of recurrent neural networks to discriminative keyword spotting”, International Conference on Artificial Neural Networks, 220–229.

[161] Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser L., Polosukhin I., (2017), “Attention is all you need”, Advances in Neural Information Processing Systems, 1–15, Long Beach, CA, USA, 4-9 December.

[162] Robbins H., Monro S., (1951), “A stochastic approximation method”, Annals of Mathematical Statistics, 22(3), 400–407.

[163] Clevert D.-A., Unterthiner T., Hochreiter S., (2015), “Fast and accurate deep network learning by exponential linear units (ELUs)”, International Conference on Learning Representations, 1–14, San Juan, PRI, 2-4 May.

[164] Redmon J., Farhadi A., (2018), “YOLOv3: An incremental improvement”, arXiv:1804.02767, 1–6.

[165] LeCun Y., Bottou L., Bengio Y., Haffner P., (1998), “Gradient-based learning applied to document recognition”, Proceedings of the IEEE, 86(11), 2278–2324.

[166] Krizhevsky A., Sutskever I., Hinton G. E., (2012), “Imagenet classification with deep convolutional neural networks”, Advances in Neural Information Processing Systems, 1097–1105, Lake Tahoe, NV, USA, 3-6 December.

[167] Simonyan K., Zisserman A., (2014), “Very deep convolutional networks for large-scale image recognition”, International Conference on Learning Representations, 1–14, San Diego, CA ,USA, 7-9 May.

[168] Szegedy C., Liu W., Jia Y., Sermanet P., Reed S., Anguelov D., Erhan D., Vanhoucke V., Rabinovich A., (2015), “Going deeper with convolutions”, IEEE Conference on Computer Vision and Pattern Recognition, 1–9, Boston, MA, USA, 7-12 June.

[169] Szegedy C., Vanhoucke V., Ioffe S., Shlens J., Wojna Z., (2016), “Rethinking the inception architecture for computer vision”, IEEE Conference on Computer Vision and Pattern Recognition, 2818–2826, Las Vegas, NV, USA, 27-30 June.

[170] He K., Zhang X., Ren S., Sun J., (2016), “Deep residual learning for image recognition”, IEEE Conference on Computer Vision and Pattern Recognition, 770–778, Las Vegas, NV, USA, 27-30 June.

[171] Chollet F., (2017), “Xception: Deep learning with depthwise separable convolutions”, Conference on Computer Vision and Pattern Recognition, 1251–1258, Honolulu, HI, USA, 21-26 July.

[172] Szegedy C., Ioffe S., Vanhoucke V., Alemi A., (2017), “Inception-v4, inception-resnet and the impact of residual connections on learning”, AAAI Conference on Artificial Intelligence, 1–12, San Francisco, CA, USA, 4-9 February.

[173] Xie S., Girshick R., Dollár P., Tu Z., He K., (2017), “Aggregated residual transformations for deep neural networks”, IEEE Conference on Computer Vision and Pattern Recognition, 1492–1500, Honolulu, HI, USA, 21-26 July.

[174] Shi X., Chen Z., Wang H., Yeung D.-Y., Wong W.-K., Woo W.-c., (2015), “Convolutional LSTM network: A machine learning approach for precipitation nowcasting”, Advances in Neural Information Processing Systems, 1–9, Montréal, QC, CAN, 7-12 December.

[175] Ballas N., Yao L., Pal C., Courville A., (2016), “Delving deeper into convolutional networks for learning video representations”, International Conference on Learning Representations, 1–11, San Juan, PRI, 2-4 May.

[176] Graves A., (2013), “Generating sequences with recurrent neural networks”, arXiv:1308.0850, 1–43.

[177] Yoo J., Kim H. J., (2014), “Target tracking and classification from labeled and unlabeled data in wireless sensor networks”, Sensors, 14(12), 23871–23884.

[178] Satriano C., Lomax A., Zollo A., (2008), “Real-time evolutionary earthquake location for seismic early warning”, Bulletin of the Seismological Society of America, 98(3), 1482–1494.

[179] Li D., Hu Y. H., (2003), “Energy-based collaborative source localization using acoustic microsensor array”, EURASIP Journal on Advances in Signal Processing, 2003(4), 1–17.

[180] Ashraf A., Hu Y.-H., (2003), “Region based distributed sensor network signal processing”, Technical report no:, San Diego State University, Masters of Homeland Security, San Diego, CA ,USA.

[181] Zwartjes A., Havinga P. J., Smit G. J., Hurink J. L., (2016), “QUEST: Eliminating online supervised learning for efficient classification algorithms”, Sensors, 16(10), 1–23.

[182] Kalman R. E., (1960), “A new approach to linear filtering and prediction problems”, Transactions of the ASME–Journal of Basic Engineering, 82(1), 35–45.

[183] Jazwinski A. H., (2007), “Stochastic Processes and Filtering Theory”, 1st Edition, Courier Corporation.

[184] Julier S. J., Uhlmann J. K., (2004), “Unscented filtering and nonlinear estimation”, Proceedings of the IEEE, 92(3), 401–422.

[185] Köse E., Hocaoğlu A. K., (2019), “A new spectral estimation-based feature extraction method for vehicle classification in distributed sensor networks”, Turkish Journal of Electrical Engineering & Computer Sciences, 27(2), 1120–1131.

[186] Loshchilov I., Hutter F., (2016), “SGDR: Stochastic gradient descent with warm restarts”, International Conference on Learning Representations, 1–16.

[187] Trottier L., Giguere P., Chaib-draa B., (2016), "Parametric exponential linear unit for deep convolutional neural networks", IEEE International Conference on Machine Learning and Applications, 207–214, Cancun, MEX, 18-21 December.

[188] Fernández-Redondo M., Hernandez-Espinosa C., (2001), "Weight initialization methods for multilayer feedforward", European Symposium on Artificial Neural Networks, 119–124, Bruges, BEL, 25-27 April.

[189] Glorot X., Bengio Y., (2010), "Understanding the difficulty of training deep feedforward neural networks", International Conference on Artificial Intelligence and Statistics, 249–256, Sardinia, ITA, 13-15 May.

[190] Luomala J., Hakala I., (2022), "Adaptive range-based localization algorithm based on trilateration and reference node selection for outdoor wireless sensor networks", Computer Networks, 210(1), 1–17.

[191] Köse E., Hocaoğlu A. K., (2023), "ConvLSTM-based vehicle detection and localization in seismic sensor networks", IEEE Access, 11(1), 139306–139313.

[192] Redmon J., Farhadi A., (2017), "YOLO9000: better, faster, stronger", Conference on Computer Vision and Pattern Recognition, 7263–7271, Honolulu, HI, USA, 21-26 July.

[193] Bridle J., (1989), "Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters", Advances in Neural Information Processing Systems, 211–217, Denver, CO, USA, 27-30 November.

[194] Huber P. J., (1992), "Robust Estimation of a Location Parameter", In: S. Kotz N. L. Johnson, Editors, "Breakthroughs in Statistics: Methodology and Distribution", Springer.

[195] Maas A. L., Hannun A. Y., Ng A. Y., (2013), "Rectifier nonlinearities improve neural network acoustic models", International Conference on Machine Learning, 3–11, Atlanta, GA, USA, 16-21 June.

[196] Ester M., Kriegel H.-P., Sander J., Xu X., (1996), "A density-based algorithm for discovering clusters in large spatial databases with noise", International Conference on Knowledge Discovery and Data Mining, 226–231, Portland, OR, USA, 2-4 August.

[197] Hinneburg A., Gabriel H.-H., (2007), "DENCLUE 2.0: Fast clustering based on kernel density estimation", International Symposium on Intelligent Data Analysis: Advances in Intelligent Data Analysis, 70–80.

[198] Ruan S., Mehmood R., Daud A., Dawood H., Alowibdi J. S., (2017), "An adaptive method for clustering by fast search-and-find of density peaks: Adaptive-DP", International World Wide Web Conference, 119–127, Perth, WA, AUS, 3-7 April.

[199] Gueorguieva N., Valova I., Georgiev G., (2017), "M&MFCM: Fuzzy c-means clustering with mahalanobis and minkowski distance metrics", Procedia Computer Science, 114(1), 224–233.

[200] Köse E., Hocaoğlu A. K., (2021), "Clustering with density based initialization and Bhattacharyya based merging", Turkish Journal of Electrical Engineering & Computer Sciences, 30(3), 502–517.

[201] Brownlee J., (2016), "Deep Learning with Python: Develop Deep Learning Models with Tensorflow and Keras", 2nd Edition, Machine Learning Mastery.

[202] Kingma D. P., Ba J., (2014), "Adam: A method for stochastic optimization", International Conference on Learning Representations, 1–15, San Diego, CA, USA, 7-9 May.

[203] Smith L. N., (2017), "Cyclical learning rates for training neural networks", IEEE Winter Conference on Applications of Computer Vision, 464–472, Santa Rosa, CA, USA, 24-31 March.

[204] https://www.baesystems.com/en-media/uploadFile/20210406061804/1434593116846.pdf. Access Date: 19.05.2023.

[205] https://archive.org/download/TM9-1767C/TM9-1767C.pdf. Access Date: 19.05.2023.

BIOGRAPHY

Erdem KÖSE received the B.Sc. degree from Gebze Institute of Technology, Kocaeli, Türkiye, in 2014 and the M.Sc. degree from Istanbul Technical University, Istanbul, Türkiye, in 2017, both in electronics engineering. He is currently working toward the Ph.D. degree in electronics engineering in graduate school at Gebze Technical University, Kocaeli, Türkiye. He is currently employed as a senior systems engineer at Qualcomm Technology, Cork, Ireland. His research interests include statistical signal processing, machine learning and their realizations and implementations on real-time systems.

PUBLICATIONS WITHIN the SCOPE of THESIS STUDY

[1] Köse E., Hocaoğlu A. K., (2023), “ConvLSTM-based vehicle detection and localization in seismic sensor networks”, IEEE Access, 11(1), 139306–139313.

[2] Köse E., Hocaoğlu A. K., (2021), “Clustering with density based initialization and Bhattacharyya based merging”, Turkish Journal of Electrical Engineering & Computer Sciences, 30(3), 502–517.

[3] Köse E., Hocaoğlu A. K., (2019), “A new spectral estimation-based feature extraction method for vehicle classification in distributed sensor networks”, Turkish Journal of Electrical Engineering & Computer Sciences, 27(2), 1120–1131.