Verifying RADAR Data Using Two-Dimensional QIM-based Data Hiding

by

Brandon Fedoruk

**A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Engineering
(Computer Engineering)
in the University of Michigan-Dearborn
2021**

Master's Thesis Committee:

**Professor Hafiz Malik, Chair
Associate Professor Samir Rawashdeh
Assistant Professor Alireza Mohammadi**

ACKNOWLEDGEMENTS

I would like to offer my sincere gratitude my advisor, Dr. Hafiz Malik. He was instrumental in offering guidance and support whenever I needed it, always welcoming and patient. Without your help, this would not be possible. I would like to offer a my gratitude to Raghavendar Changalvala, a PHD research student at The University of Michigan - Dearborn. Raghavendar and I collaborated over the summer doing joint research and work on this topic(*Changalvala et al.*, 2020). It resulted in a joint publication with MDPI Sensors. His experience and knowledge was very helpful with implementing type II embedding techniques and fusion algorithms. I would also like to thank friends and colleagues who assisted me during this project.

TABLE OF CONTENTS

CHAPTER

LIST OF FIGURES

Figure

ABSTRACT

Modern vehicles have evolved into supporting advanced internal networks and connecting System Based Chips (SBC), System in a Package (SiP) solutions or traditional micro controllers to foster an electronic ecosystem for high speed data transfers, precision and real-time control. The use of Controller Area Networks (CAN) is widely adopted as the backbone of internal vehicle communication infrastructure. Automotive applications such as ADAS, autonomous driving, battery management systems, power train systems, telematics and infotainment, all utilize CAN transmissions directly or through gateway management. The network transmissions lack robust integrity verification mechanisms to validate authentic data payloads, making it vulnerable to packet replay, spoofing, insertion, deletion and denial of service attacks. Additional methods exist to secure network data such as traditional cryptography. Utilizing this method will increase the computational complexity, processing latency and increase overall system cost. This thesis proposes a robust, light and adaptive solution to validate the authenticity of automotive sensor data using CAN network protocol. We propose using a two-dimensional Quantization Index Modulation (QIM) data hiding technique, to create a means of verification. Analysis of the proposed framework will be conducted in a sensor transmission scenario for RADAR sensors in an autonomous vehicle setting. The detection and effects of distortion on the application are tested through the implementation of sensor fusion algorithms and the results are observed and analyzed. The proposed framework offers a needed capability to maintain transmission integrity without the compromise of data quality and low design complexity. This framework could also be applied to different network architectures, as well as its operational scope could be modified to operate with more abstract types of data.

CHAPTER I: INTRODUCTION

Within the last three decades, the automotive industry has seen rapid growth with the utilization of in vehicle network controllers like ECUs. This introduced capabilities that people enjoy in modern vehicles such as dynamic vehicle control, ADAS features and infotainment controls. These features are dependent on external sensors and integrated advanced communication networks. Autonomous vehicles, in particular, rely heavily on sensory peripherals and data transmission networks to realize the capability of automated driving functions (*Sarmento et al.*, 2017). Current autonomous vehicles rely on sensory input to determine real-time decisions classified as level 2 and above (SAE J3016-201806). A typical loadout contains sensor peripherals such as RADAR, LiDAR, ultrasonic and vision systems. Sensor data is transmitted to centralized data processing units called Advanced Driver Assistance System (ADAS) module and flows through different network topologies consisting of ECUs and gateways; bridging internal vehicle networks as well as external wireless transmission sources. These external transmission sources, typically cell and wireless networks, allow vehicles to be connected to one another and subsequentially introduce additional attack surfaces for malicious actors. For vehicles designed to support SAE level 3 or higher, the severity of cyber-attacks exploiting vulnerabilities with secure data transfer from vehicle sensors to ADAS modules, gateways and associated peripherals, is greatly increased; thus, posing a significant risk to the safety and well being of the vehicle's occupants. Hardening sensor data transmissions is a critical component in the secure operation of autonomous vehicles (*Longxiang et al.*, 2017).

Current market ready sensors can be generalized into two categories, smart and raw sensors (*Jo et al.*, 2015). Devices that have high data rates such as Camera and LiDAR will transmit packet-based data over ethernet to an ADAS module; These are considered raw sensors. Sensors that typically support post processing infrastructure for object tracking and transmit over deterministic and fault tolerant networks such as CAN/CAN-FD, are classified as smart sensors. The transmitted data is still processed by an ADAS module, which is used as input constraints for proprietary sensor fusion and decision-making algorithms. Shortcomings in CAN protocol pertaining to broadcast message formatting, in the clear data transfers and lack of native message authentication and security, exposes CAN networks to packet replay, spoofing, injection, deletion and denial of service attacks (*Lin and Sangiovanni-Vincentelli*, 2012). Numerous demonstrations exploiting common attack vectors targeting vehicles over CAN networks, has been successfully attempted within automotive security (*Miller and Valasek*, 2015). Currently solutions exist to mitigate known vulnerabilities such as certificate based payloads, encryption, Message Authentication Code (MAC) and frame ID filtering (*Woo et al.*, 2015). A common implementation for transport layer security is utilizing Message Authentication Code and cryptography to maintain data integrity. AUTOSAR, which is a widely adopted software architecture used within automotive, incorporates MAC based authentication on classic platform architectures called secure on-board communication (SecOC) (AUTOSAR CP Release 4.3.1). Vehicle Architectures utilizing cryptography to secure data transmissions within vehicle networks will have to address and overcome operational constraints and deficiencies in their design such as processing overhead, key management, authentication failures and required dependencies for operation. Additionally, HS-CAN and MS-CAN may not be sufficient to implement traditional cryptography due to bandwidth and message length limitations. A common solution is upgrading the network architecture to CAN-FD, which offers increased bandwidth and message length constraints (*Woo et al.*, 2016).

Given current limitations of traditional data verification methods for information integrity, we introduce a new data hiding based watermark approach. This method addresses challenges associated with resource constrained architectures and real-time applications. The proposed algorithms are low computational complexity and bypass bandwidth limits of the systems architecture. This can be easily achieved as the size of the original payload transmitted is not changed. This proposed method utilizes a Quantization Index Modulation (QIM) technique to embed the generated water mark using a lightweight software algorithm. Utilizing the data hiding concepts in an automotive

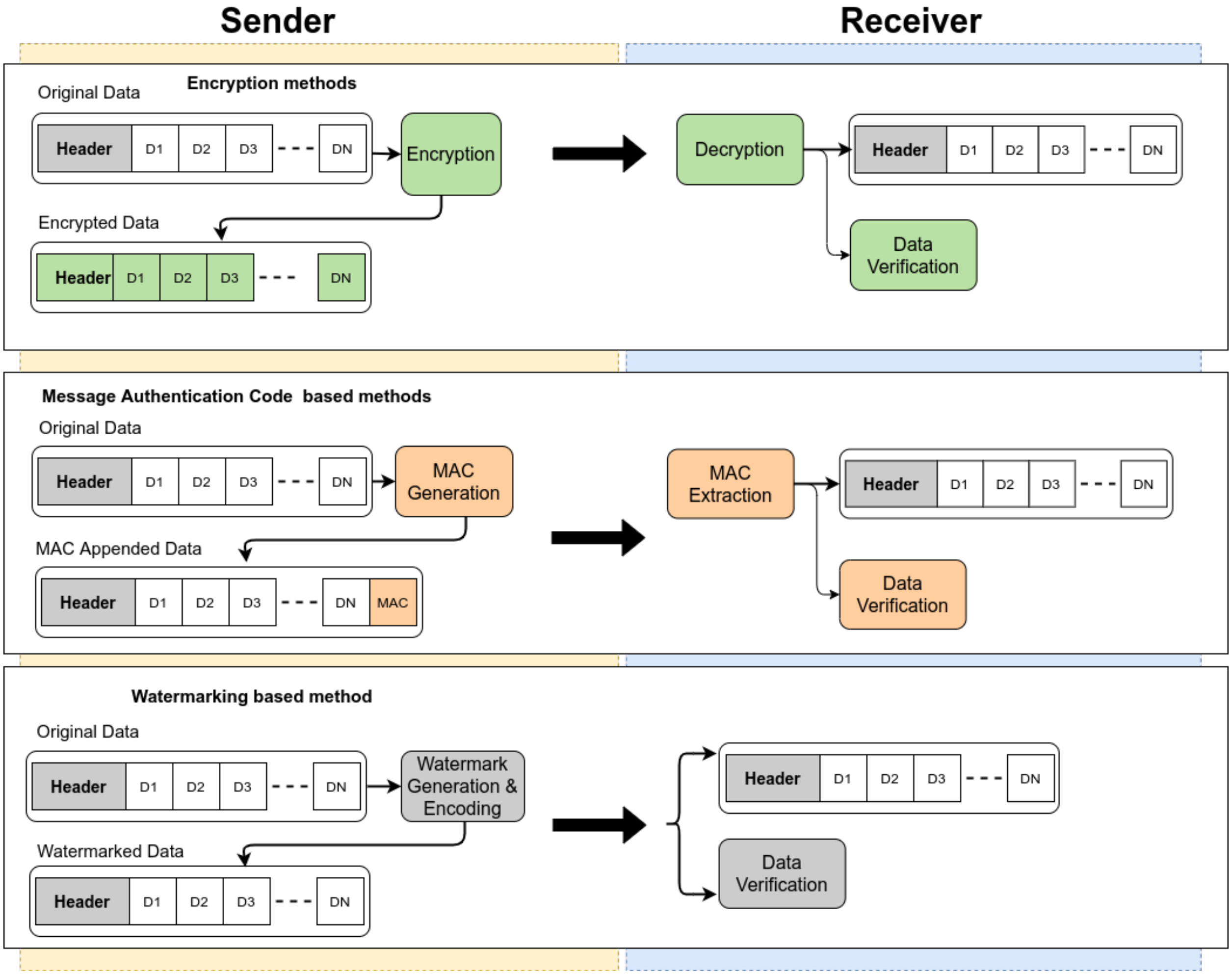

Figure 1.1: Comparison of different methods to achieve sensor data integrity.

environment for verifying transmitted sensor data, was introduced by (*Changalvala and Malik*, 2019). Additionally, utilizing the techniques applied in centralized autonomous vehicle architectures was introduced by (*Jo et al.*, 2014). In this paper, we will analyze a different technique. We will consider the application of smart sensor classes, focusing on RADAR as the primary sensor peripheral that produces the data. The watermarking method proposed is applied to the processed data transmitted from the RADAR sensor, evaluated and verified on the outcome of our sensor fusion algorithm.

1.1 Watermarking Advantages

As we progress further into the 21st century, the adoption and utilization of social media, connected devices and data acquisition, is unlocking potentials society struggled to fathom a century prior. With large scale data transfers, opportunities needed for data security and verification is paramount. Attacks on information infrastructure ranging from telecom network to automotive vehicle networks can be devastating and could potentially pose risk to human life. The concept of network information security focuses on securing data during transmission, storage and processing against common attack vectors like Man-in-The-Middle (MiTM) attacks, side channeling and tampering (*Lu and Guo*, 2017). Traditionally, cryptographic applications are used to secure the integrity of the communication channel. Encrypting communication networks ensure privacy, valid authentication and verification through symmetric, asymmetric and hybrid key sharing. Although cryptography is used to mitigate network security issues, there are disadvantages to utilizing this application. Encrypting a communication channel can implicitly draw attention to itself. If a communication channel is encrypted, then something sensitive and of high importance could be propagating through that channel. If the encryption being used is defeated, the attacker now has unimpeded access to the information on that channel. It's still possible for an attacker to invalidate the data on a communication channel by modifying the payload without defeating the channels encryption(*Lu and Guo*, 2017). Within the past two decades, data hiding techniques regained focus as a potential mechanism to prevent unauthorized accesses to critical data while remaining hidden by design. These methods are typically stealthed which offer unique applicability that traditional encryption could not do. They can be used as standalone measures or in tandem with encryption algorithms. A clear distinction between stenography and encryption is, the access to data being protected. With standard encryption algorithms, the data is obfuscated and only the intended recipient can decode and disseminate the information. Stenography does not obfuscate the data and restrict access. Instead, the embedded signature is hidden within the existing data and used to validate the authenticity without being detected (*Lu and Guo*, 2017). In environments where computational resources are limited, like in edge computing applications, standard cryptography applications are limited or unable to be implemented due to the resource constraints; especially in real-time environments. Edge computing systems are additionally burdened by the use of key management systems, network bandwidth limits and trade export restrictions. These combined effects increase the design complexity of the system thus increasing the overall cost of the product. Consider an edge computing application scenario within an autonomous vehicle environment. Two external RADAR sensors are connected to an ADAS unit on the vehicles CAN network. The ADAS module is responsible for processing the inbound RADAR data packet as well as additional sensor data where information will be combined in a fusion processor for increased modality. Securing this data from unauthorized access and manipulation using cryptography requires coordination of trust anchors and accelerators for key storage and processing. In addition, MAC based implementations will attribute to the increased unit cost of each module. This also limits the ability to implement MAC based devices on legacy networks, further limiting its application.

Watermarking applications are typically less computationally intensive and resource constrained. They have the capability of embedding a traceable, reversible or non-reversible signature into existing data that can be handled natively by the end process or application. This is an advantage for data processing in real-time application environments. Figure 1.1 shows an abstraction of three methods, encryption, Message Authentication Code and Digital Watermarking. These security applications can be implemented to maintain data integrity. In contrast to digital watermarking, both methods of encryption and Message Authentication Code require additional computational steps before the data can be processed. This additional step has to be factored into the design of current real-time applications. In MAC based implementations, the message payload increases based on the interface bandwidth. In CAN networks this type of implementation can double bandwidth requirements *Woo et al.* (2016) *Zou et al.* (2017). Data based watermarking works in parallel with the native data through minimal perturbation, eliminating the need to augment the payload size on the network and reducing consumption of available computational resources.

Utilizing a watermarking implementation also provides needed traceability and security for autonomous driving applications as well as additional use cases. A unique advantage to watermarking

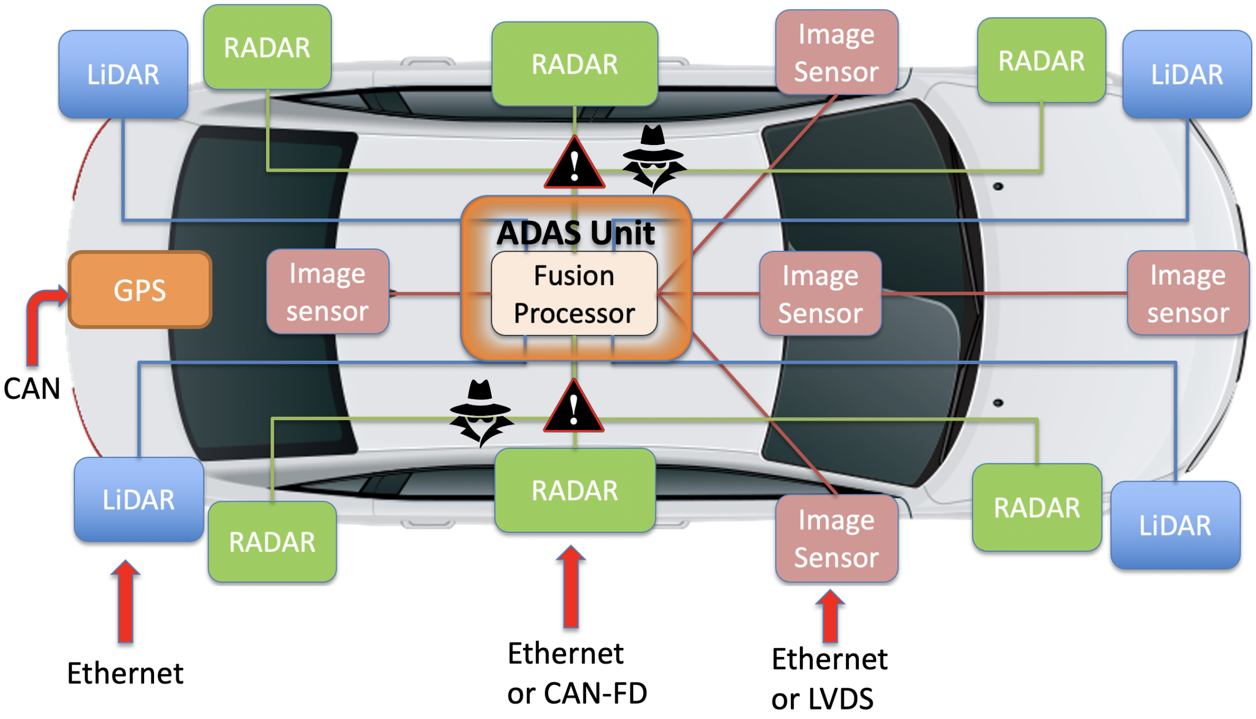

Figure 1.2: Block-diagram of problem statement.

data, is the embedded signature remaining in the payload until the data is received and processed. Passive applications such as bus traffic monitoring, dataloggers and upload mechanisms for centralized data analytics can utilize this characteristic for data integrity within edge computing applications.

1.2 Principal Contribution

Sensor networks designed for automotive applications could be grouped into numerous general constraints. However, the following two will be an area of focus:

- **Bandwidth limitations**: The communication link between the sensor and ECU is bandwidth limited in automotive vehicles. Most sensors use an HS-CAN interface which has a payload size of 8 bytes(*Zou et al.*, 2017). This is eight times smaller than CAN-FD which can support a payload of up to 64 bytes. CAN-FD would allow additional security measures to be implemented such as Message Authentication Code (MAC). Devices that use AUTOSAR with

SecOC will utilize. CAN-FDs bandwidth requirements can handle this but scaling multiple sensors on the same network will consume additional bandwidth due to increased transmission frequency, payload size or both. Legacy Networks will frequently encounter bandwidth limitations, given increased data throughput from sensors needed to construct high resolution fields of view for autonomous driving features.

- **Real-time data verification**: Operating conditions in autonomous vehicles often require sensor data to be processed in a real-time environment. This constraint adds design challenges to data security when using cryptography, considering processing overhead.

Given the existing limitations with CAN/CAN-FD in terms of data security, vehicle networks remain in a compromised state. Looking at figure 1.2, a range of attack vectors are available to gain access to CAN networks as an attack surface. This thesis proposes a watermarking solution which performs under these constraints while verifying the integrity of the sensor data before processing. Without modifications, standard watermarking types are vulnerable to estimation based watermark attacks. To mitigate this risk, we propose introducing variable watermark signatures based on available data from the in-vehicle network. This could range from GPS timestamp data to IMU metadata. This data would be used as a foundation to generate a unique watermark signature which would be embedded in the information intended to be secured. Mirroring the generation algorithm between the transmitter and receiver nodes, eliminates the need to share the watermarking scheme over any secured channel. In addition, when using watermark embedding applications, the impact embedded induced distortion has on the data you're trying to secure. A significant portion of this research is providing an analysis on the impact of embedded induced distortion on sensor data and fusion processing algorithms downstream.

CHAPTER II: RELATED WORK

Within the scope of resource constrained systems, digital watermarking has an advantage over traditional cryptography with respect to the impact on system bandwidth and resource consumption. Current research on implementing watermarking concepts to secure information on resource limited systems, directs you to Wireless Sensor Networks (WSN). WSN share similar bandwidth and processing limitations that exist in legacy automotive networks. The versatility and scalability introduced inherent vulnerabilities to a range of malicious replay, jamming, selective forwarding, general tampering and delay to sender attacks. As a result, significant research has been performed to mitigate these attack vectors with digital watermarking. Wireless Sensor Networks started adopting integrity verification for transmitted data back in 2003, with a publication proposing embedding cryptographically generated signatures into an existing data payload, written by (*Feng and Potkonjak*, 2003). A fragile watermarking technique was introduced in the publication (*Kamel and Juma*, 2011). This technique was designed with the goal of securing transmissions from insertion, replay and deletion attacks. Their proposal detailed a generation method by which a serial number is appended to each payload, making them uniquely identifiable mitigating the risk of insertion attacks being performed successfully. The generation algorithm used is a simple hashing function. An interesting publication by (*Ibaida et al.*, 2011) introduced watermarked methods to include patient information embedded in the EKG waveform, while maintaining the identifiable characteristics of the waveform. A variation of Least Significant Bit (LSB) QIM was used to embed the watermark. A single bit watermark was generated using an XOR operation for a homogeneous sensor network in(*Tiwari et al.*, 2013). A watermark generation algorithm was proposed in (*Sun et al.*, 2013) and detailed the generation of a one way hashing function. The hashed output was then grouped by an XOR function before being embedded in the data at predetermined and redundant locations. This was done to increase survivability. A linear interpolation watermark generation technique was proposed in (*Lalem*, 2016). Additional data to generate the watermark is not needed in this design, instead they propose to use fixed points in the data to generate the watermark. This poses a security risk and increases the probability of successful estimation attacks being performed on this method. The watermark generation used in (*Zhang et al.*, 2017) is based on known information shared between the sender and receiver; in this case a key. To reduce computational complexity, they assume the candidate data has header information and is time synchronized. The embedding process takes a bitstream and randomly places bits throughout the data. That signature is then extracted and validated against an expected signature generated from the shared key. (*Alromih et al.*, 2018) propose using a Randomized Watermark Filtering Scheme (RWFS) for connected devices. The generated watermark is randomly embedded using a Pseudo Random Number Generator (PRNG). The seed value used for the PRNG is defined by the cluster from which the packet is generated from. Using key sharing, the generated watermark is compared with the embedded watermark after extracting it from the payload for validation. Their proposed method provided end to end confidentiality from encrypted key sharing and protection from replay and modification attacks. (*Bahirat and Prabhakaran*, 2017) discuss the concept of insider attacks on the LiDAR point cloud. Multiple approaches that exploit the resolution and occlusion consistency between the tampered and clean data frames are proposed. In (*Changalvala and Malik*, 2019), Raghu and Hafiz introduce a watermarking based approach to secure raw LiDAR sensor data in autonomous vehicles.

CHAPTER III: SYSTEM & ATTACK MODEL

Within the system model, it is assumed that an ADAS processing unit is centralized within the vehicle architecture connected to external sensors over the vehicle network as referenced in figure 1.2. The inbound information is combined within the ADAS module and performs the applicable information extraction from the object detection lists. It's also assumed that all applicable peripherals and modules are clean initially. Attacks are launched during the transmission time windows between from the sensor to the ADAS module, as referenced in figure 1.2. The perceived threat model exposes numerous entry points into the vehicle's network which makes it more enticing for an attacker to potentially gain entry from. The system model in this particular case requires a GPS receiver transmitting periodic timestamp data, all sensors as well as the ADAS module will have access to this data. Assuming a data formatting as referenced in figure 3.2 for RADAR sensor information transmitted on the intended network, each payload will begin with a header that contains metadata.

The metadata details tracked object quantity, unique Data Identifier, etc. Following the header, the body of the payload will consist of data elements which will vary in contents based on the capability of each RADAR smart sensor. However, for simplicity it's assumed the baseline content contained is the derived Cartesian Coordinates (x, y) of the tracked object. The tracked objects position data will be used to embed the watermark. The QIM based watermarking method proposed will modify the data directly, which makes the added embedded induced distortion a critical attribute to monitor. The importance of achieving a low distortion rate cannot be expressed enough in order to maintain the efficacy of the original data. Therefore, preserving the desired results or behavior of the intended application that consumes this data is paramount. For the purpose of this endeavor, an extended Kalman Filter (EKF), which is widely used in autonomous vehicle applications is utilized to analyze the effects of the embedded induced distortion.

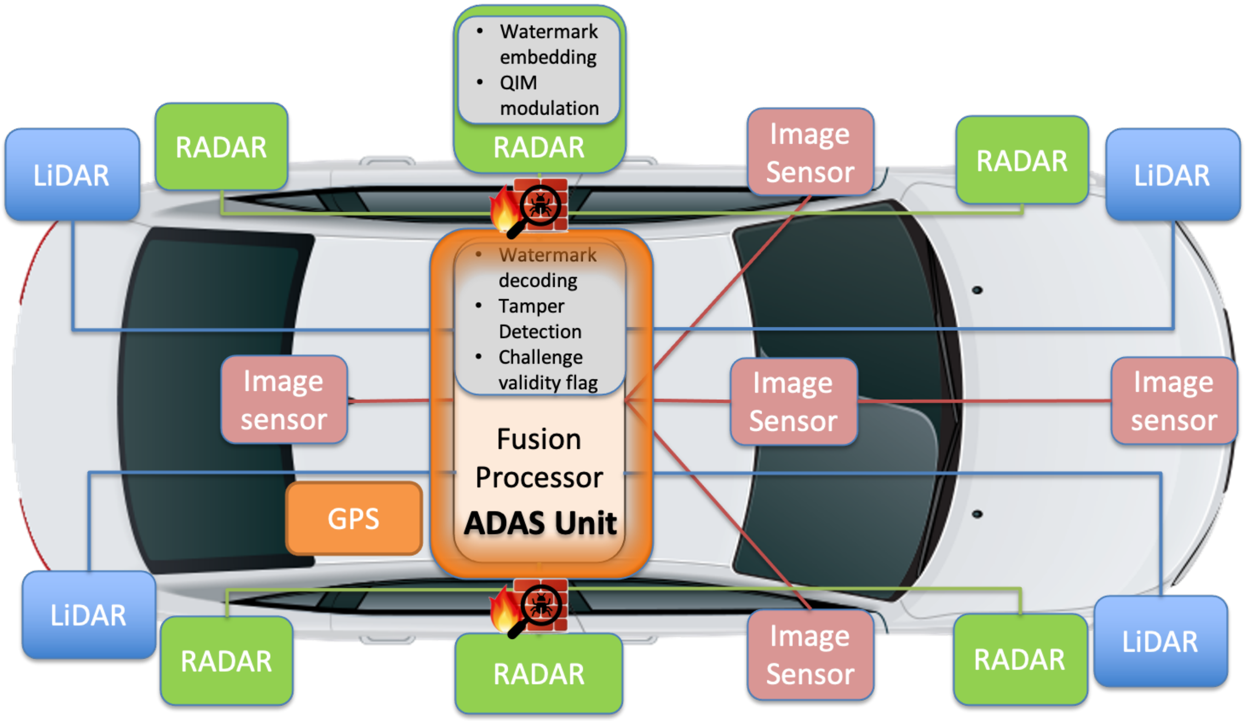

Figure 3.1: Block-diagram of proposed method.

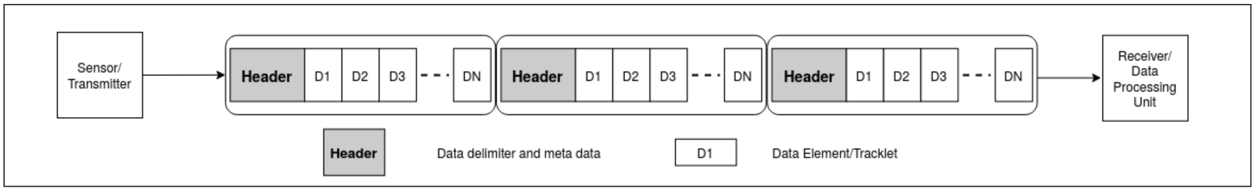

Figure 3.2: Radar data stream.

3.1 Sensor Fusion Data Model

To test the proposed framework, an open sourced dataset by Mercedes Benz's autonomous driving utility *Technologies* (2018). The dataset contains pedestrian information including the predicted path of the individual, captured by the vehicle from one or more of the on-board RADAR and LiDAR sensors shown figure 3.3. The RADAR data is broken down into position and velocity vectors, represented in polar coordinates (ρ, ϕ, $\dot{\rho}$). ρ is the radial measure from sensor to target, ϕ is the measured lateral angle from sensor to target and $\dot{\rho}$ is the rate of change ρ. To begin processing, the polar Coordinates are converted to Cartesian coordinates.

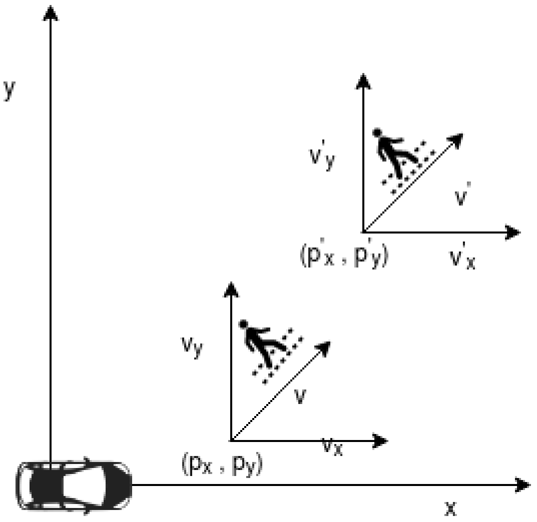

Figure 3.3: State vector for pedestrian motion.

$$
\begin{aligned}
x &= \rho * cos(\phi) \\
y &= \rho * sin(\phi)
\end{aligned}
\tag{3.1}
$$

The LiDAR measurements contained in the dataset are represented as position coordinates (x, y). In conjunction with the LiDAR data entries, GPS timestamp is recorded at the time of acquisition. The settling time between data acquisition is $\Delta t = 50$ ms. At each sample interval Δ, the ground truth for pedestrian position vectors (p_x, p_y, v_x, v_y) are calculated based on a constant velocity model. The model is a 2D bicycle model with the yaw rate assumed to be zero and described with the equations below:

$$
\begin{aligned}
&\dot{\theta} = 0 \\
&x^{'} = x + v \cdot \Delta t \cdot cos(\theta) \\
&y^{'} = y + v \cdot \Delta t \cdot cos(\theta)
\end{aligned}
\tag{3.2}
$$

where v is the target velocity, Δt is the elapsed time and θ is the yaw (x, y) & $(x^{'}, y^{'})$ are respectively initial and final positions. From the ground truth values, Gaussian white noise is added to the data to simulate configurable variance. This is represented by the equations below:

$$m_{t_k} = g_{t_k} + \epsilon \tag{3.3}$$

where m_{t_k}, g_{t_k}, represent the ground truths at t_k and ϵ is the measurement error represented by an independent and identical distribution (i.i.d) Gaussian noise with zero mean and covariance matrix $R > 0$, i.e. $\epsilon = N(0, R)$.

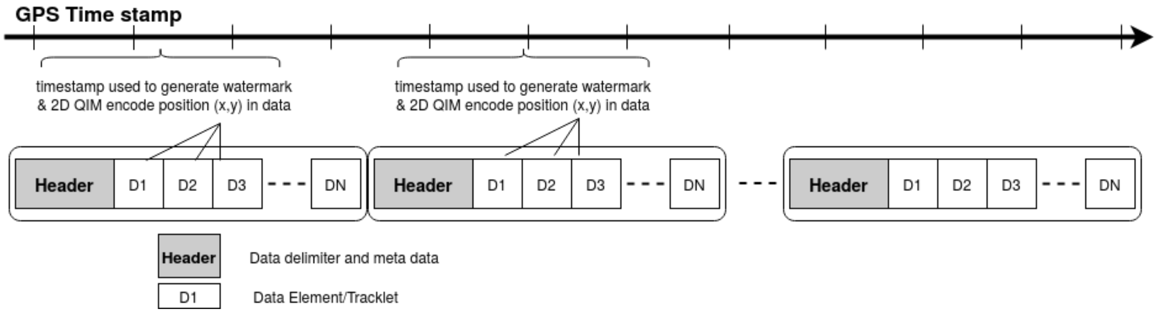
GPS Time stamp
timestamp used to generate watermark & 2D QIM encode position (x,y) in data
timestamp used to generate watermark & 2D QIM encode position (x,y) in data
Header
D1
D2
D3
DN
Header
D1
D2
D3
DN
Header
D1
D2
D3
DN
Header
Data delimiter and meta data
D1
Data Element/Tracklet

Figure 3.4: Proposed framework and 2D QIM embedding process.

3.2 Watermarking Data Model

In this model, the detector will be unaware of the original embedded signature. This closely fits the blind decoding method. Although, for verification purposes, the encoding message sequence will be regenerated and compared with the decoded signature. The host signal represented as c_0 is considered a member of the set of n p-dimensional vectors $c \in \{c_0, \cdots, c_n\}$, where n is signal length. Depending on the sensor used, the dimensionality of the data can vary. For this use case, the data is represented two dimensionally as (x, y), with $p = 2$. With p indicating the dimensionality of the signal. Typically, in data embedding schemes, the host signal c_0 is transformed to a new value based on the characteristics of the watermark. Numerous values are generated within the hosts dataspace that are virtually indistinguishable from the original data space. To render a conceptual visualization of this process, imagine a region around the host c_0 data space in which every vector corresponds to the RADAR position vector; virtually indistinguishable from the original vector. This region is known as the region of acceptable fidelity (*Cox et al.*, 2008).

Once the embedding algorithm has transformed the original data into the watermarked data m, the detection algorithm will operate within the scope of the detection region. This region is defined by the message m and the generated watermark represented as a key k. The message and key are represented as a set of position vectors. The region is derived by a given correlation algorithm to determine how similar each position vector is to the key. Linear correlation could be used to compare the data from the detectors input and the original message m, to derive a relationship value. During the embedding process, an algorithm maps the message into a pattern of the same type and dimension as the host signal. This is represented by the coefficient w_r with a potential dependence on key k. A linear correlation is derived by the detection algorithm from the input of the received signal and the reference pattern w_r. This is comparable to the orthogonal projection of the received signal c_{wn} into the reference pattern w_r. Let, c_1 and c_2 be two p-dimensional vectors in the dataset. If we create a limit of τ_{mse} on the function measuring the variance (perceived distance between the vectors), the region of acceptable fidelity is an n-dimensional sphere centered around the host signal c_0. The radius of such a sphere is defined as $\sqrt{n\tau_{mse}}$. The measure of success for the embedding process is determined from the intersection of the of the detection regions and acceptable fidelity regions. This entire process will essentially perturbate the host signal to an extent the modified host signal is similar enough to fall within the intersection regions. This minimizes the induced distortion while making it possible for recovery or verification. If the perturbed signal exceeds the intersection thresholds, then the signal has been tampered with or corrupted in some way. This will ultimately damage the watermark and provide an origin point. A fragile watermarking scheme similar to what was just described is used with QIM to detect and localize data tampering.

CHAPTER IV: FRAMEWORK

The proposed data integrity framework is designed on the following assumptions. First, the sensor and ADAS module contain the same watermark generation algorithm. Second, the ADAS module is configured to collect, process and fuse LiDAR and RADAR data into detection lists. The framework proposed in figure 3.1 is partitioned into three parts and further conceptualized in figure 4.1:

- Watermark generation
- Watermark embedding
- Watermark decoding

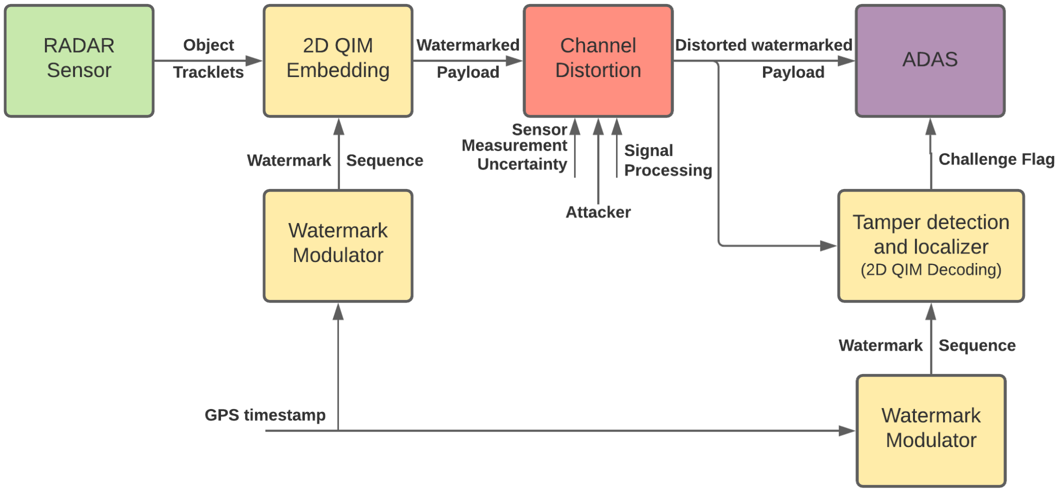

Figure 4.1: Framework Abstraction.

Generating the watermark begins with the utilization of the GPS timestamp information as a baseline for the generation algorithm. The GPS timestamp is converted into a binary sequence

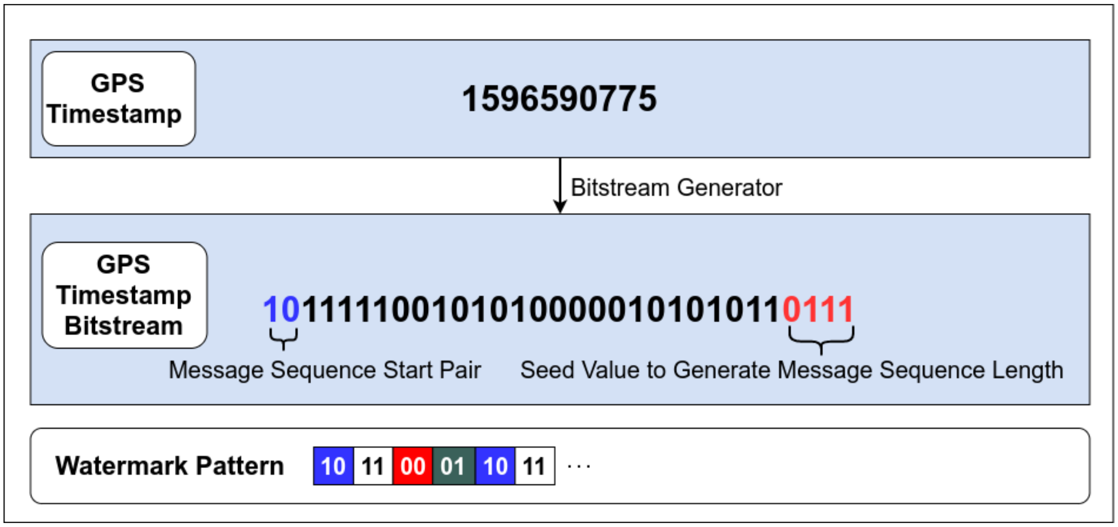

Figure 4.2: Time-stamp conversion.

represented as $m_e = f(t_{gps})$ and is embedded into the position data of the processed object detection lists using 2D QIM. The watermarked data is then transmitted to the ADAS unit over an in-vehicle communication network such as CAN. The ADAS module then receives watermarked data in conjunction with the GPS timestamp information. This data is given as an input to the sensor fusion and verification algorithm, operating in parallel with the fusion processors. The verification algorithm will generate the embedded message sequence m_e using an identical method as the sensor. The embedded message sequence from the received frames is extracted from m_e using the decoding algorithm and represented as m_d. The decoding message sequence is compared against the embedded message sequence for anomalies as mentioned in section V. If tampering is detected a validity flag is set, challenging the authenticity of the fused detection list. In figure 3.1, this is identified as a fire-wall between the ADAS module and Motion Control Module. The integrity verification mechanism can operate passively in the background, comparing embedded signatures until an anomaly is detected; referenced in figure 3.1. In which case, the kernel on the ADAS module will be notified of a challenge and will take the necessary action dictated by applicable failure modes.

A core component of the integrity verification framework is our watermark generation technique. The proposed model leverages GPS timestamp data to generate a binary sequence which

Algorithm 1: Watermark Sequence Generator.

```
Result: $generatedSequence[\,]$
$generatedPair \leftarrow LSbits(b(t_0))$;
$generatedSequence[\,] \leftarrow generatedPair$;
$randomNum \leftarrow pseudorandom(MSnibble(b(t_0)))$;
$num_{Pairs} \leftarrow floor(randomNum)$;
while $size(generatedSequence) <= num_{Pairs}$ do
    $generatedPair+ = b01$;
    $generatedPair = generatedPair\%4$;
    $generatedSequence[] \leftarrow generatedPair$;
end
```

becomes embedded into the host data before transmitting downstream. The diagram in figure 4.2 displays a visual representation of a GPS timestamp converted into a bitstream format. Assuming the architecture supports little endianness, the two least significant bits (LSBs) and the most significant nibble is parsed and stored in a secured buffer. This is utilized in the sequence generator described in Algorithm 1. The LSB pair previously stored is used to determine the starting bit pair for the generated sequence. This adds a level of obfuscation to the generated sequence by altering the starting bit pair of the binary sequence before embedding into the host signal. Additionally, the most significant nibble is converted to a decimal representation and utilized as a seed value to generate a random number; bounded by the theoretical maximum for potential data elements generated in one payload.

4.1 Watermark Generation

The sequence generator described by Algorithm 1 will use the information derived from the GPS timestamp explained previously to generate a deterministic sequence. The process involves taking the range limited random number x, derived from the seed value of the most significant nibble, where $num_{Pairs} = \lfloor x \rfloor$ is used to determine the sequence length. A two-bit value is incremented and appended to the generated sequence buffer. The proposed 2D QIM embedding method allows for a message sequence of integer values bounded between $[0 \leq m_{val} \leq 4]$. Therefore, the remainder of the generated bit pairs $(m_{val} \leq mod(4))$ is used to keep the sequence bounded. By design, the desired length is dependent on the seed value calculated from parsing the GPS timestamp. If the sequence is shorter than the number of elements in the payload, the sequence will be reused. The added randomness is created from the dynamic start and sequence length; Both can be created by the receiver for verification.

4.2 Watermark Embedding

We use a 2D QIM-based data hiding method for watermark embedding. Quantization Impulse Modulation is a non-linear, data hiding, semi-fragile watermarking method that is widely used in digital forensics and steganography (*Brian and W*, 2001). During the embedding process, a host signal $S = \{s_1, s_2, \cdots, s_N\}$ is quantized based on the embedded message symbols $M = \{m_1, m_2, \cdots, m_N\}$. If we consider a trivial implementation of a binary QIM scheme, where $m_i \in \{0, 1\}$, the modified or watermarked host signal S_w can be represented as:

$$S_w = q_{m_i}(s_i, \Delta), \text{ where } i = 1, 2, \cdots, N \tag{4.1}$$

where, $q_{m_i}(\cdot)$ denotes a uniform quantizer. With a quantization step-size Δ and a perturbation of $\Delta/2$, this uniform quantizer is represented as:

$$q_{m_i}(s_i, \Delta) = \text{round}\left(\frac{s_i}{\Delta}\right) \cdot \Delta \pm m_i \cdot \Delta/2 \tag{4.2}$$

Observing equation 4.1 & 4.2, the host signal is modified post embedding while the distortion level is proportional to the perturbation level. This feature offers flexability in scaling the distortion level to achieve a desired result for your end application. This motivated us to select QIM over other available watermarking methods.

Within the QIM embedding process, the quantization phase uses a set of quantizers that form a reconstruction grid for point mapping (*Joachim and Bernd*, 2002). The dimensionality of the reconstruction grid depends on the message symbol size. An n-dimensional symbol will result in

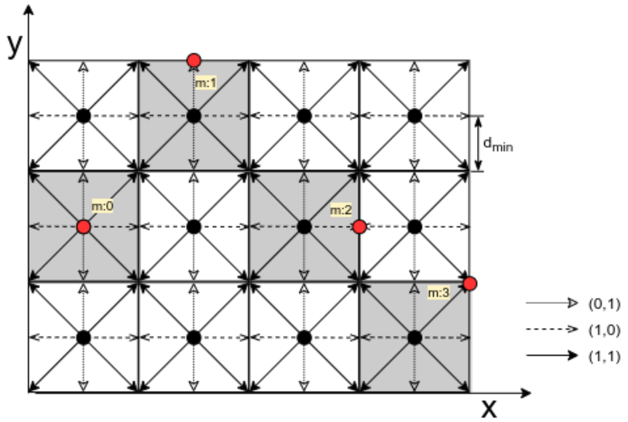

Figure 4.3: 2D QIM scheme.

a $\log_2(n)$-dimensional reconstruction grid. As an example, if a binary message m is defined by $m \in \{0, 1\}$, where a one dimensional reconstruction grid is represented with $n = 2$. Consider, extending the aforementioned concept using datasets with two dimensional vectors (x,y) and a four dimensional message symbol $m \in \{0, 1, 2, 3\}$. The message symbol m can now be used to hide data, requiring a 2D reconstruction grid. The corners of the generated grids can be used as reconstruction points in recovering the embedded message. In reference to figure 4.3, a sample dataset is shown with a variation in embedded message symbols. After initial quantization, the datapoints within the highlighted polygons are represented by a black dot $c_k = \{x_k, y_k\}$. Based on the embedded message symbol $m_k = \{m_{xk}, m_{yk}\}$ in 2D QIM, the midpoint is translated to one of eight fixed locations along the polygon's boundary, represented with a red dot. d_{min}, the minimum separation distance between the reconstruction points, determines the robustness for the of the framework and the channel noise. One of the advantages of QIM based watermarking is the configurability d_{min} as a tunable parameter. This increases framework portability, allowing adaptation to different application environments. Additionally, due to the non-intersecting reconstruction points, host signal interference rejection further increases reliability in 2D QIM schemes (*Chen and Wornell*, 1998). The resulting watermarked signal s_w is represented by:

$$s_w(s_{c_k}, m_k) = q_{m_k}(s_{c_k}, \Delta) \tag{4.3}$$

where $q_{m_k}(\cdot)$, denotes 2D QIM quantizer which is expressed as:

$$q_{m_k}(c_k, \Delta) = \text{round}\left(\frac{c_k}{\Delta}\right) \cdot \Delta \pm \frac{\Delta}{2} \cdot m_k \tag{4.4}$$

Within the proposed framework, an algorithm will parse the generated message symbols, applying the quantizer values to the target data. In this case, the RADAR position data referenced in equation 4.4. The GPS timestamp obtained by the sender is processed based on the explanation in section 4.1; The generated watermark is embedded into the data as shown in figure 3.4 and figure 4.1. The now modified data is transmitted over the network with additional meta-data included in the header, acting as a delimiter to the data elements. The embedded watermark will remain with the payload regardless of the transport protocols used to transmit data to the receiver. Additionally, since there isn't any added data, such as a MAC, the bandwidth requirements remain the same.

4.3 Watermark Decoding

On the receiver side, the object detection lists from the RADAR sensor can be processed by the fusion algorithm in the ADAS module, without additional modifications beforehand. The tamper detection and isolation algorithms can be executed in parallel, offering an advantage to using a watermarking method over an encryption one to maintain data integrity and localize intrusions. The decoding operation is similar to the embedding operation in section 4.1, in which the received position values are quantized using the same process as embedding to generate the reconstruction points. In this case, the reconstruction points will be a set of four vectors. The derived reconstruction points are then compared with the received values and the point with the least difference is considered part of the decoded message. This is shown in equation 4.5.

$$m_d =_{i \in 0,1} |s_w^{'}(s_i, m_d) - s_w(s_i, m_i)| \tag{4.5}$$

In the above equation, $s_w^{'}(s_i, m_d)$ represents a distorted signal, while m_i is the embedded message and m_d is the decoded message. During this process, it's assumed the sensors are time-synchronized by a trusted universal timestamp authority, in this case a GPS sensor (AUTOSAR CP R19-11).

4.4 Attack Detection - Data Injection

This section discusses the different attack profiles, assuming a malicious actor is able to compromise and gain access to the vehicles network. Additionally, we discuss in detail how the proposed framework can detect and localize the attack profiles mentioned in this section. For each attack profile, the attacker is assumed to have working knowledge of vehicle network protocols and a background in automotive electrical systems. Furthermore, the attacker is assumed to have the capability to monitor active network traffic, cherry pick and replay messages on a CAN/CAN-FD network. Three attack profiles were identified and classified as:

- Data injection
- Data deletion
- Data modification

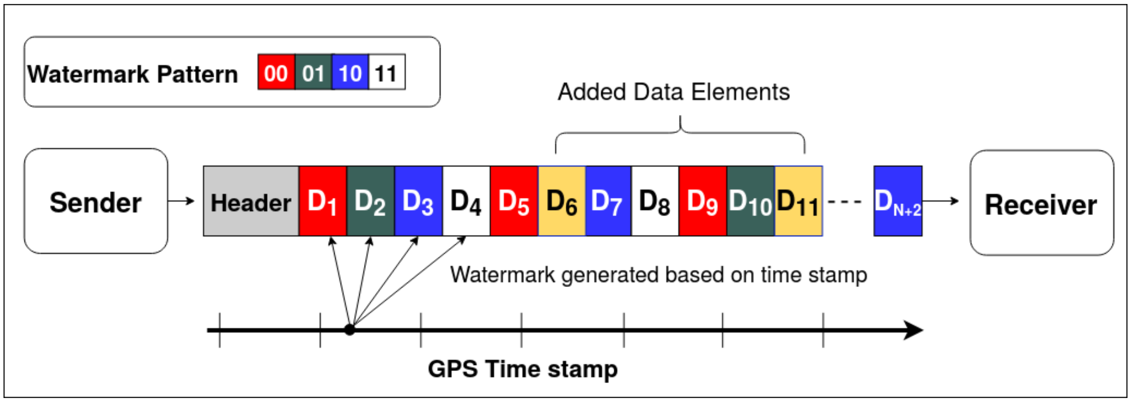

Figure 4.4: Data addition attack vector depiction.

Each attack profile analysis is performed on the capability of the proposed framework for detection and localization. The attacker modifies the processed RADAR tracklets with additional arbitrary data elements at their discretion. Figure 4.4 depicts an attack scenario where the data elements D_6 and D_{11} are injected into the original data payload. This ultimately increases the total element

Algorithm 2: Find Added Indices

```
Result: $addedIndices[\,]$
$addedIndices \leftarrow 0$;
$expectedListIndex \leftarrow 0$;
$modListIndex \leftarrow 0$;
while $modListIndex < len(modList)$ & $expectedIndex < len(expectedList)$ do
    if $expectedList[expectedListIndex] == modList[modListIndex]$ then
        $expectedListIndex+ = 1$;
        $modListIndex+ = 1$;
    end
    else
        $addedIndices[\,] \leftarrow expectedListIndex$;
        $modListIndex+ = 1$;
    end
end
```

count from n to $n+2$. Within the framework, the two-dimensional position vectors encoded with a GPS timestamp derived sequence, as explained in section 4.1, uses 2D QIM embedding to encode the color coded watermark pattern in figure 4.4. To help aid in conceptualization, the sequence length of the watermark is limited to four elements. During an attack, additional data elements are added, causing a disruption in the message sequence. This will happen even if the injection attack just replicates an existing message or data elements within a CAN payload. The encoded message sequence will be disrupted, and an inspection can localize the entry point. Assuming a breach has occurred and an injection attack is in progress, the receiver is expecting a green color coded sequential element D_6 and a blue color coded sequential element for D_{11}. However, in this case a deviation is detected in the expected watermark sequence. In this example, we assume the receiver has knowledge of the message length beforehand. Knowing this information as well as metadata from a sensor's modality, such as a GPS timestamp as mentioned in section 4.1, allows the decoder to generate the modulated watermark used in the embedding process. From the length of the encoded l_{encode} and decoded l_{decode} message sequence, the type of attack performed can be identified based on the characteristics of the manipulated message at the receiver. A simple algorithm can determine if data was injected into the message if $l_{decode} > l_{encode}$ holds true. In algorithm 2 the expected and decoded message sequence is compared to isolate the entry point of the injection attack. The algorithmic complexity in this case is $O(N)$. The referenced algorithm operates on the assumption that the injected elements pattern will differ from the existing adjacent elements in the message sequence. The framework was subjected to simulated external interference in the form of additive uniform noise, as depicted in figure 5.1. The results offer a quantitative observation of the robustness of the tamper detection algorithm in the presence of channel noise and indicate that QIM based schemes are able to recover the watermark signature if the below equation holds true for the channel noise.

$$d_{min}^2 > 4 \cdot N \cdot \sigma_n^2 \tag{4.6}$$

In the above equation, σ is the standard deviation of channel noise, N is the dimensionality or encoding bits, d_{min} is the minimum distance between the reconstruction points(*Chen and Wornell*, 1998). Observing the detection algorithm in figure 5.1, the accuracy is 100% when channel noise is within the bounds defined in equation 4.6, for a step size of $\Delta = 1\ cm$.

4.5 Data Deletion

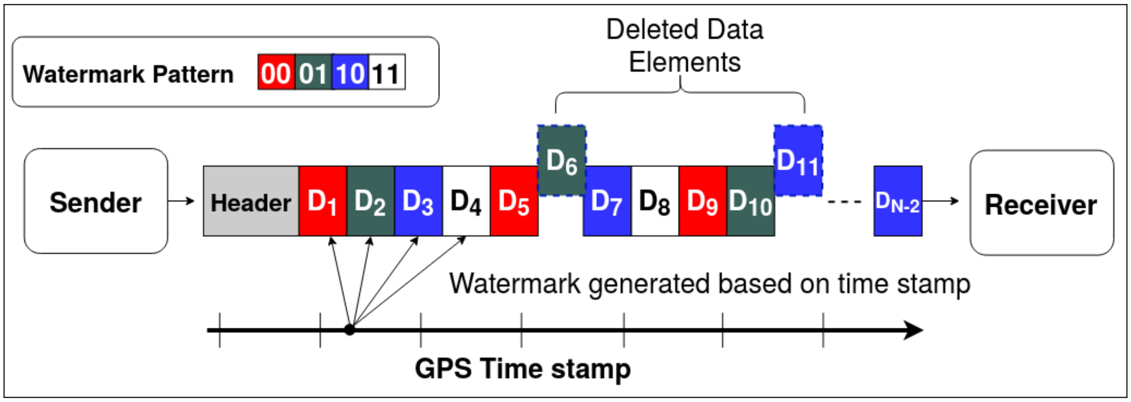

Figure 4.5: Data deletion attack vector depiction.

Within the attack profile shown in figure 4.5, the attacker can obstruct elements of the object detection lists or entire payloads if they choose. A plausible attack is depicted in figure 4.5, as elements D_6 and D_{11} are removed from the original sequence; modifying the total element count from n to $n-2$. Within the framework, the two-dimensional position vectors encoded with a GPS timestamp derived sequence, as explained in section 4.1, uses 2D QIM embedding to encode the color coded watermark pattern in figure 4.4. When elements are deleted, the embedded message sequence is broken. Assuming an attack is in progress, the receiver is expecting Green color coded sequential element D_6 and a Blue color coded sequential element D_{11}. The receiver detects the received elements D_7 and D_{12} as mismatched, indicating a potential breach. After the lengths of the encoded l_{encode} and decoded l_{decode} message sequence are calculated, this type of attack can be determined if $l_{decode} < l_{encode}$ holds true for the incoming message. The tamper localization algorithm in figure 3 compares the decoded message sequence with the expected sequence, to detect and locate the deleted data elements. Algorithmic complexity in this case, is $O(N)$. Observing the results of this algorithm shown in figure 5.1, the accuracy is 100% when channel noise is within the bounds defined in equation 4.6, for a step size of $\Delta = 1\ cm$. Additionally, as the noise variance increases, the accuracy diminishes for the given step-size.

Algorithm 3: Find Deleted Indices

```
Result: $missingIndices[\,]$
$missingIndices \leftarrow 0$;
$expectedListIndex \leftarrow 0$;
$modListIndex \leftarrow 0$;
while $modListIndex < len(modList)$ & $expectedIndex < len(expectedList)$ do
    if $expectedlist[expectedListIndex] \neq modList[modListIndex]$ then
        $missingIndices[\,] \leftarrow expectedListIndex$;
        $expectedListIndex + = 1$;
    end
    else
        $expectedListIndex + = 1$;
        $modListIndex + = 1$;
    end
end
```

4.6 Data Modification

The attack profile shown in figure 4.6 demonstrates a scenario where the attacker modifies the existing data (object detection lists) to achieve a desired outcome. Within the framework, the two-dimensional position vectors encoded with a GPS timestamp derived sequence, as explained in section 4.1, uses 2D QIM embedding to encode the color coded watermark pattern in figure 4.4. To help aid in conceptualization, the sequence length of the watermark is limited to four elements.

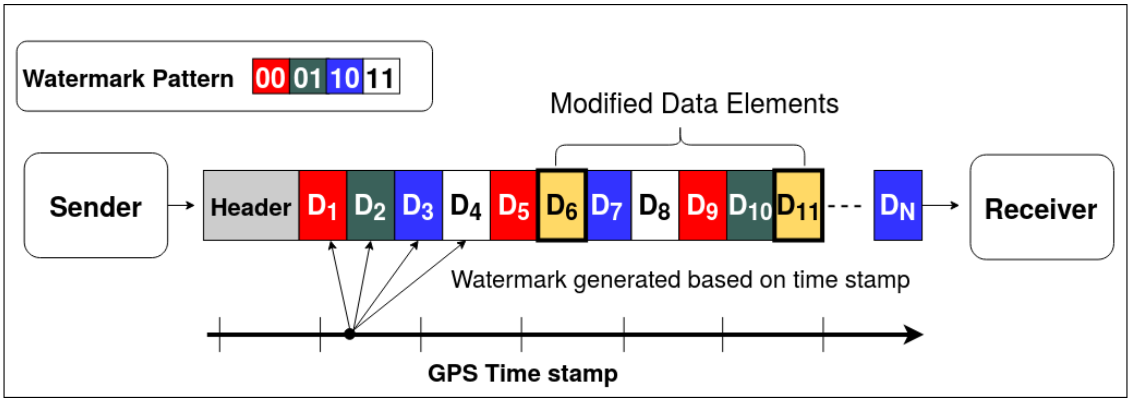

Figure 4.6: Data modification attack vector depiction.

Algorithm 4: Find Modified Indices.

```
Result: $modifiedIndices[\,]$
$modifiedIndices \leftarrow 0$;
$expectedListIndex \leftarrow 0$;
$modListIndex \leftarrow 0$;
while $expectedIndex < len(expectedList)$ do
    if $expectedList[expectedListIndex] \neq modList[modListIndex]$ then
        $modifiedIndices[\,] \leftarrow expectedListIndex$;
    end
    else
        $expectedListIndex+ = 1$;
        $modListIndex+ = 1$;
    end
end
```

The elements D_6 and D_{11} are modified without changing the total count of elements. At the receiver side, the verification algorithm is expecting a green color coded data element D_6 and a blue color coded data element D_{11}, but the sequence is broken from the modifications and an anomaly is detected. The data modifications can be localized by using Algorithm 4, which compares the decoded message sequence with the expected message sequence. This algorithm assumes the channel noise is within the parameters specified in the two previous algorithms and the modified data form the attackers occurs within the payload. Observing figure 5.1, as the noise variance increases, the localization accuracy of the algorithm decreases. However, when the noise is within bounds, the detection and localization accuracy of the modified data elements is 100%.

CHAPTER V: PERFORMANCE EVALUATION

Frameworks utilizing data-hiding based techniques for sensor integrity is complex, opposed to traditional cryptography. A chief concern when embedding signatures into existing data is the unavoidable distortion added during the embedding process. As a result, the effects of the embedded induced distortion have on the given applications ability to utilize the information as intended, under normal operating conditions, is a use case that needs to be tested and analyzed. The results will help aid industry OEMs and suppliers in adaptation of watermarking techniques for sensors and other applications within a vehicle. In this chapter, an analysis on the effects of embedded RADAR object data using 2D QIM embedding framework, on a fusion algorithm processor is performed. The fusion algorithm selected is an Extended Kalman Filter (EKF). Typically, in a vehicle environment, Kalman filters are used to estimate the state of dynamic systems such as position estimation, feature tracking, cluster tacking, data fusion and much more. Kalman filters are lightweight fusion algorithms in the sense of requiring current and previous observations (a robust causal filter) to make their determination(*Jetto et al.*, 1999) (*Rigatos*, 2010) (*Madhavan and Schlenoff*, 2003). In addition, Kalman filters are designed to mitigate sensor noise by taking inputs from at least two sensors and predict a position vector for the object currently tracked. This process is repeated as new data is processed by the EKF. For this experiment, as explained in 3.1, measurements utilized from vehicle LiDAR and RADAR sensors estimate the state of pedestrian's movements in front of the vehicle. Both sensors will detect the same object, which an Extended Kalman Filter fuses the data to predict the position of the pedestrian.

$$x = [p_x \quad p_y \quad v_x \quad v_y]^T \tag{5.1}$$

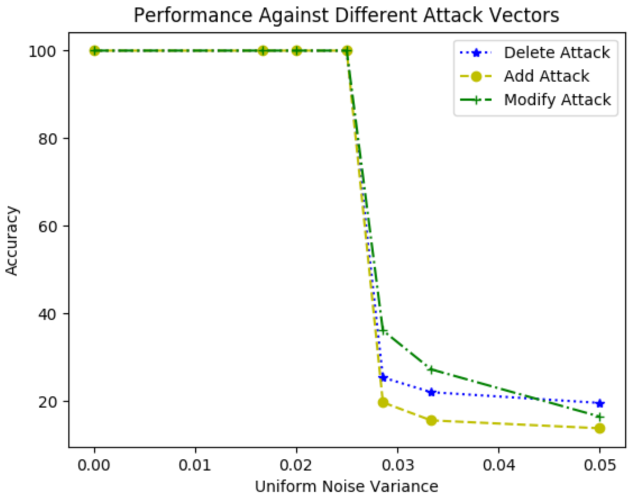

Figure 5.1: Tamper localization algorithm performance under varying channel noise.

Where (p_x, p_y) are (x, y) components of position and (v_x, v_y) are (x, y) components of his velocity at a given time t_k for the pedestrian. Internally, a Kalman filter can be categorized into two steps, prediction and update. During prediction, the state vector $x^{'}$ at time t_k is estimated, also producing an error or uncertainty vector $P^{'}$. The uncertainty is based on values of x and P at previous time t_{k-1} and the new data is updated. P is represented as gaussian random noise, affecting the accuracy of the prediction step. The state vector $x^{`}$ is represented as:

$$x^{'} = f(x, \mu) \tag{5.2}$$

where, μ is the stochastic part, represented as $N(0, Q)$, this can be alternatively expressed as:

$$\begin{aligned} x^{'} &= Fx + \mu \\ P^{'} &= FPF^T + Q \end{aligned} \tag{5.3}$$

where the state transition matrix F, models the transitions from previous time t_{k-1} to current time t_k. μ is the added noise and Q is the co-variance matrix that's modeling the stochastic part of the state transitions. As mentioned previously, the linear motion model with constant velocity, defines the state transition matrix F. The next position at time t_k is derived from:

$$p'_{t_k} = p_{t_{k-1}} + v_{t_{k-1}} * \delta t \tag{5.4}$$

where $\delta t = t_k - t_{k-1}$ and since the model assumes constant velocity, at the next time step, the velocity is given as

$$v'_{t_k} = v_{t_{k-1}} \tag{5.5}$$

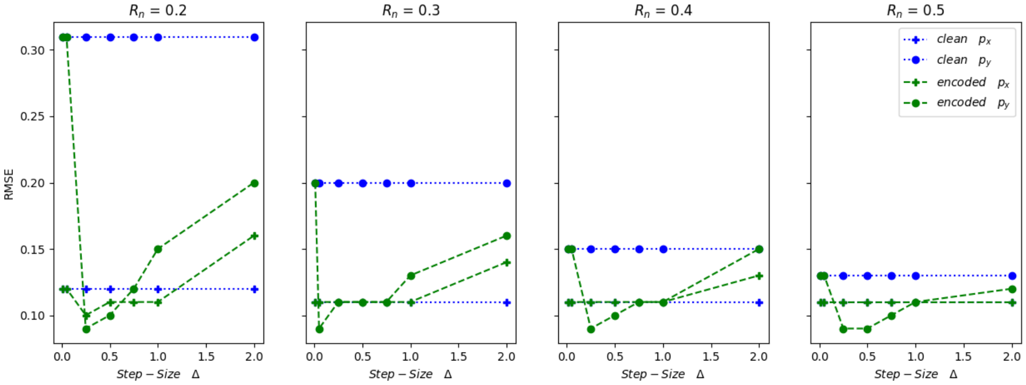

Figure 5.2: RMSE comparison at $R_m = 0.4$.

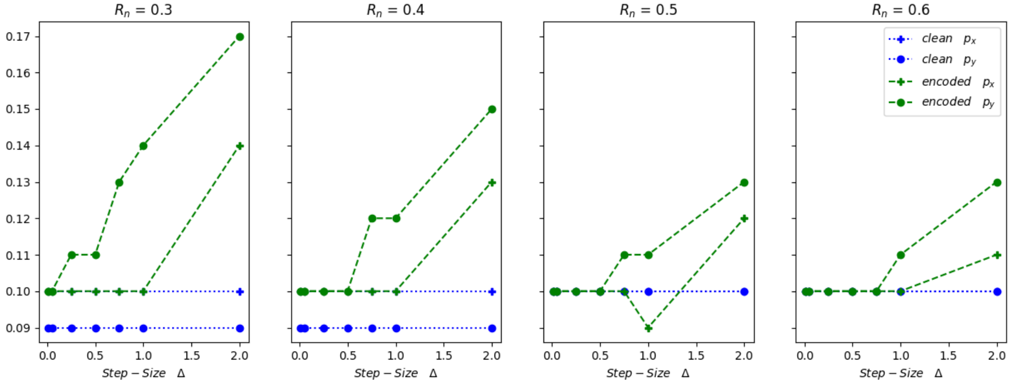

Figure 5.3: RMSE comparison at $R_m = 0.5$.

Based on the above model, the Kalman filter uses the estimated state to predict the pedestrian position. During the update phase, sensor measurements correct the predicted states to produce a position estimate with greater accuracy. The equation for expressing the pedestrian's position within the measurement function can be expressed as:

$$z = [p^{'}]^T \tag{5.6}$$

The measurement phase of the filter utilizes the measurement model, matrix H and covariance matrix R to correctly estimate the measurement vector z. Transformation of the measurement vector utilizes the matrix vector, which produces the state vector shown in equation 5.1. The measurement function can now be expressed as:

$$z = Hx + \omega \tag{5.7}$$

where H, the measurement matrix, providing the objects raw position. ω is the measurement error, encompassing uncertainties in measurements from the sensor. This is represented as a zero mean Gaussian distribution and covariance matrix R, $\omega \approx N(0, R)$. Assuming the measurement components have not been cross correlated yet, the covariance matrix R becomes a diagonal matrix. The dimensionality of R will depend on the size of the measurement vector z. This results in two for LiDAR and three for RADAR. Hence, R becomes a $3x3$ diagonal matrix for RADAR and a $2x2$ diagonal matrix for LiDAR. Ultimately, the measurement matrix will vary depending on the sensor data used by the fusion algorithm. For RADAR measurements, an EKF variant of the Kalman filter is used. Consider the application of LiDAR measurements to achieve a similar result. After post processing, LiDAR will ultimately measure the position of the target in Cartesian coordinates (x, y). The state to measurement vector transition will be linear and the measurement calculation of H is relatively straight forward, after discarding the velocity comment from the state vector. Due to the linear nature of the measurement vector transitions, a standard Kalman filter can be used for LiDAR measurements. However, in the case of RADAR, the transitions are non-linear since the application of RADAR will obtain measurements ρ, ϕ, $\dot{\rho}$ of the object. During the update phase an EKF variant of the standard Kalman filter is used to handle the non-linear functions required to make an accurate measurement prediction. Kalman filters are linear estimators and the non-linear variation of that design is called the Extended Kalman Filter (*Madhavan and Schlenoff*, 2003). The non-linear state observed in EKF equations are linearized using Jacobian matrices. For the case of processing the RADAR data, the Jacobian of H is evaluated to obtain the linear approximation. Once z has been determined, the correction or update phase is executed, taking the latest measurements to update the state estimates and uncertainties. This is expressed as:

$$y = z - Hx^{'} \tag{5.8}$$

Here y is the error value or the difference between the prediction and actual measurement at a given time step. The estimation error S is evaluated as:

$$S = HP^{'}H^{T} + R \tag{5.9}$$

The Kalman gain K is evaluated as:

$$K = P^{'}H^{T}S^{-1} \tag{5.10}$$

After the computation of the Kalman gain, the predictions are updated using the following equations and these steps are repeated for the entire drive cycle.

$$x = x^{'} + Ky \tag{5.11}$$

$$P = (I - KH)P^{'} \tag{5.12}$$

Observing the EKF uncertainty in both the process and measurements, are taken into consideration. In general, the measurement uncertainty or the measurement noise covariance matrix R in equation 5.7, is the inherent sensor behavior and hence provided by the sensor manufacturer. Whereas, the process uncertainty Q in equation 5.3, is defined based on the motion model and other application related assumptions. If we use the EKF as the sensor fusion algorithm, three configuration parameters can affect the algorithm outcome in the proposed framework. The first being the measurement noise matrix R, second the overall process noise Q, and third, the embedding step-size Δ. In this experiment, we analyze the impact of the watermark embedding using the 2D QIM method on the sensor fusion algorithm's output under different configuration scenarios. Two types of RADAR data are used as an input to the EKF algorithm, predicting the state vector of the pedestrian. The first type is the clean and unmodified data, and the second type is 2D QIM modified data. The resulting predictions from EKF are compared against the ground truth position vectors in both cases using Root Mean Square Error (RMSE) in the following equation:

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(x_t^{gt} - x_t^{pred})^2} \tag{5.13}$$

where x_t^{gt} & x_t^{pred} are the ground truth and predicted position vectors respectively at a given time t and n is the length of data. The RMSE value is used to determine the accuracy of the prediction. Low RMSE value indicates the sensor fusion algorithm predicted the tracked objects position accurately throughout the tracking duration. The RMSE values of position vector (p_x, p_y) predictions generated from clean and watermarked inputs to the EKF are shown in Fig. 5.2 & 5.3.

The measurement input into the EKF has a noise component, dependent on the intrinsic electronic characteristics of the sensor in use. This can be represented as an additive Gaussian noise ω as shown in equation 5.7. The measurement noise covariance R represents the deviation of the sensor measured values from the true values. This deviation is estimated during the calibration phase by the sensor manufacturer. If sensor manufacturer data is not available, an estimation can also be obtained using analytical methods (*Park et al.*, 2019). In order to compensate for measurement noise in an EKF, the R value for the sensors used must be estimated or known and used in equation 5.7.

Consider $\omega_{R_m} \approx N(0, R_m)$ as the known or calculated measurement uncertainty and $\omega_{R_n} \approx N(0, R_n)$ as the overall measurement uncertainty used in the sensor fusion EKF algorithm in equation 5.7. Consider, an EKF provides accurate predictions when the value of $R_n \geq R_m$. Here, it's better to keep the R_n & R_m values close to each other. If the EKF requires an inflated R_n value to incur correct predictions, then it could be concealing other anomalies in the measurements such as outliers and a noise distribution that is non-Gaussian. The measurement uncertainty values used in the EKF ω_{R_n} can be represented as a combination of two or more different noise distributions with data satisfying the i.i.d criteria. Let's say, $\omega_{R_n^1} \approx N(0, R_n^1)$ and $\omega_{R_n^2} \approx N(0, R_n^2)$ are two different noise distributions that contributed to the overall noise ω_{R_n}, then the resulting distribution can be represented as:

$$\omega_{R_n} = N(0, R_n^1 + R_n^2) \tag{5.14}$$

The RMSE results depicted in figure 5.2 & 5.3 show the 2D QIM embedded contributes added random noise to the overall measurement uncertainty, represented by R_n^1 or R_n^2 in equation 5.14. In this experiment, the RMSE values for clean and 2D QIM embedded RADAR data are calculated at different measurement noise covariance values $R_m \in (0.4, 0.5)$, $R_n \in (0.2, 0.3, 0.4, 0.5, 0.6)$ and varying embedded step sizes $\Delta \in (0.01, 0.05, 0.25, 0.50, 0.75, 1, 2)\ m$. Considering the R_m as the measurement error covariance provided by the sensor manufacturer, the EKF which accepts this RADAR sensor data should use a covariance matrix value R_n above or equal to R_m uncertainty. Observing figure 5.2, when $R_n \geq R_m$, the RMSE values of position vector for the 2D QIM embedded data is less than or equal to the RMSE values from clean data for step-size $\Delta < 0.75\ m$. With a given range of $p_x \approx 18.5\ m$ & $p_y \approx 12.5\ m$ in the data under test, the results show that the fusion algorithm can recover from position data perturbations of up-to 6%. As the R_n value drops below R_m, the RMSE of encoded data is less than the clean data, but only when $\Delta < 0.05\ m$. This shows the embedding induced distortion at higher step sizes is acting like additional uncompensated noise, introducing prediction errors. Similar results are observed for the state vector predictions in case of data with measurement covariance value $R_m = 0.5$, as shown in figure 5.3. It can be inferred from the results if the measurement covariance $R_n < R_m$, as the embedding step-size increases, the measurement noise value increases. Hence, the predictions of the embedded data elements are inaccurate. However, as the R_n value is increased above R_m, the embedding induced distortion is gracefully handled by the fusion algorithm. This results in low RMSE values even at larger step sizes. This behavior can be explained by equation 5.14. Here the embedded induced distortion acts like an additive Gaussian noise component. The inherent randomness in the watermark generation and embedding process, acting as noise, contributes to the randomness in the sensor noise. These two sources of error are independent of each other; hence the resultant effect is additive. This increases the RMSE value of the prediction error when the fusion algorithm fails to consider and compensate for this additional noise. These experiments, when repeated at different permissible values of process noise covariance values from $Q > 0$, showed similar results.

In addition to analysis performed on embedded induced distortion, two different performance metrics, Bit Error Rate (BER) and False Alarm Rate were conducted to measure performance of the detection framework.

5.1 Bit Error Rate Evaluation

The Bit Error Rate is used to analyze errors from the decoded bitstream in the presence of channel noise. The decoder generates a binary message stream $M_{x,y} = \{m^1_{x,y}, m^2_{x,y}, \cdots, m^N_{x,y}\}$, from the RADAR data elements. The BER calculation is performed by comparing each bit in the decoded message $m^i_{x,y} \in \{m^i_x, m^i_y\}$ with its associated embedded bit $\hat{m}^i_{x,y}$ expressed as:

$$BER = \frac{\sum_{i=1}^{n} \mathbf{I}_{m^i_{x,y} \neq \, \hat{m}^i_{x,y}}}{n} \tag{5.15}$$

where **I** is the indicator function and n is the size of the decoded message bitstream. When no additional noise is added to the RADAR data elements, the BER is close to 8.6%, corresponding to the noise from a probable attack. As the channel noise modeled by a uniform distribution is added to the data, the BER stays below 9.5% for the noise variance $\sigma < \Delta/5.65$, given a step-size Δ. As the noise variance increases beyond the threshold in equation 4.6, the BER value increases as shown in table 5.1. The robustness to the channel noise is directly proportional to the step-size Δ, which-in turn is directly proportional to the embedded induced distortion.

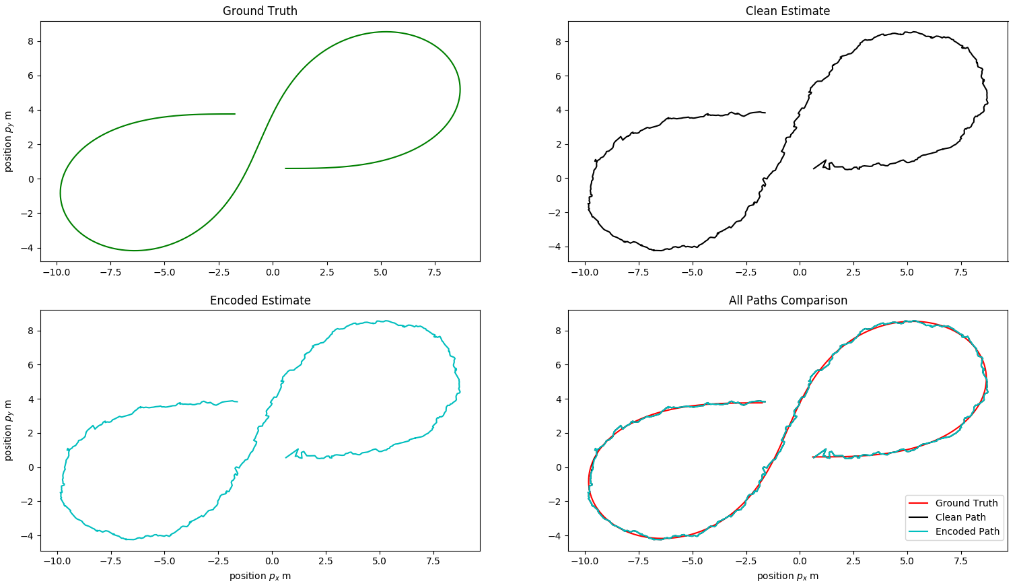

Figure 5.4: Comparison: EKF path prediction from clean and encoded data at R_m = 0.5, R_n = 0.5 & Δ = 0.01 m.

5.2 False Alarm Rate Evaluation

This performance metric measures the number of data elements classified by the framework as tampered, when in fact the data was unmodified or clean. The False Alarm Rate is determined by subjecting the framework to a predetermined amount modified and unmodified data elements. The $f_{AlarmRate}$ ratio is calculated from the equation below:

$$f_{AlarmRate} = N_{FalsePositive}/N_{DataElements} \tag{5.16}$$

where, $N_{FalsePositive}$ is the number of data elements the framework incorrectly classified as tampered and $N_{DataElements}$ is the total number of data elements tested. The experiment is repeated with different levels of additive uniform noise to replicate the channel noise. The results are shown in table 5.1. Observing the data, $f_{AlarmRate}$ stayed at 0% when the uniform noise variance $\sigma < d_{min}/(2 * \sqrt{(}N))$, $d_{min} = \Delta/2$ and $N = 2$ in our framework. As the noise variance increases beyond this threshold, the false positives increase resulting in a higher false alarm rate.

From the results, when the channel noise is within acceptable bounds, our framework can achieve 100% detection accuracy with zero false positives.

Table 5.1: BER and False-Alarm Rate at Different Noise Levels

Noise variance σ	BER %	FalseAlarm %
0.0	8.6	0.0
$\Delta/6$	9.2	0.0
$\Delta/5$	9.2	0.0
$\Delta/4$	8.6	0.0
$\Delta/3.5$	18.6	61.1
$\Delta/3$	28.6	75.0
$\Delta/2$	56.4	85.2

CHAPTER VI: FURTHER THOUGHTS

In any implementation there are always drawbacks to consider when evaluating a system like this for practical implementation. One of the limitations we discovered during testing was in this case the GPS timestamp. More generally speaking, the secondary data on the communication channel used to generate the watermark could be side channeled. If this happens, the attacker's likelihood to penetrate the affected network increases. Adding additional obfuscation techniques or utilizing additional information on the network to generate the watermark, will reduce the likelihood of an attacker detecting the watermark.

An additional watermark design we considered was to implement a reversible watermark in the event the distortion produced from the embedding process, would render the data unusable to fusion processing algorithms. Ultimately, we decided to implement a fragile watermarking scheme to limit computational complexity. However, if a design calls for a sensitive fusion algorithm, a reversible watermark can supplant the fragile watermarking scheme while the generation technique remains consistent.

A feedback loop could be introduced into our forward design. This could act as an adaptive distortion compensator, calibrating the quantization step size based on the performance of the fusion processing algorithm. This could provide a unique benefit for automotive applications in a dynamic system environment.

CHAPTER VII: CONCLUSION

Autonomous vehicles are inherently vulnerable to cyber-attacks, creating potential exploiting external sensors as attack surfaces to gain access into the vehicle and their respective transmission channels. This makes it necessary to verify the integrity of the sensor data before processing that information into actionable tasks. Traditionally, maintaining data integrity with forms of cryptography, can't be applied in their entirety due to resource constraints of the existing architecture. A pipeline based watermarking framework was proposed to detect and localize the tampering of sensor data in an autonomous vehicle. This framework was tested on the impact that embedded induced distortion has on simulated RADAR data used as an input into an EKF sensor fusion algorithm. The results concluded that the 2D QIM watermarking method has virtually no effect on the EKF predictions for small quantization step-sizes $\Delta \leq 0.05\,m$, which is a direct result to the minimal distortion induced from the 2D QIM embedding process. Often times a layered architecture is preferred in situations where an attack cannot be prevented; It can be detected to prevent the worst outcome. We believe that watermarking the sensor data adds another layer to the security scheme using some lightweight and efficient techniques. These implementations can be used either in a standalone application or in conjunction with traditional cryptography methods where-ever necessary, securing data transfers over any physical interface such as CAN/CAN-FD, Ethernet and other protocols. The research and testing conducted has demonstrated that tamper localization accuracy of our framework is virtually 100%, when the interface noise is zero. In automotive networks, sensor data interfaces like CAN and Ethernet are wired and the channel noise is minimal. Hence, using a 2D QIM approach with small step sizes yields effective tamper detection and localization accuracy with minimal data distortion.

REFERENCES

Alromih, A., M. Al-Rodhaan, and Y. Tian (2018), A randomized watermarking technique for detecting malicious data injection attacks in heterogeneous wireless sensor networks for internet of things applications, *Sensors*, *18*(12), doi:10.3390/s18124346.

AUTOSAR CP R19-11 (2019), Specification of Time Synchronization over CAN , *Standard*, AUTOSAR, AUTOSAR CP R19-11.

AUTOSAR CP Release 4.3.1 (2017), Specification of Secure Onboard Communication , *Standard*, AUTOSAR, AUTOSAR CP Release 4.3.1.

Bahirat, K., and B. Prabhakaran (2017), A Study On Lidar Data Forensics, *Proceedings - IEEE International Conference on Multimedia and Expo*, (July), 679–684, doi:10.1109/ICME.2017.8019395.

Brian, C., and W. G. W (2001), Quantization index modulation methods for digital watermarking and information embedding of multimedia, *Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, *27*(1-2), 7–33, doi:10.1023/A:1008107127819.

Changalvala, R., and H. Malik (2019), Lidar data integrity verification for autonomous vehicle using 3d data hiding, in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1219–1225, doi:10.1109/SSCI44817.2019.9002737.

Changalvala, R., B. Fedoruk, and H. Malik (2020), Radar data integrity verification using 2d qim-based data hiding, *Sensors*, *20*(19), doi:10.3390/s20195530.

Chen, B., and G. W. Wornell (1998), Digital watermarking and information embedding using dither modulation, in *1998 IEEE 2nd Workshop on Multimedia Signal Processing*, vol. 1998-Decem, pp. 273–278, doi:10.1109/MMSP.1998.738946.

Cox, I., M. Miller, J. Bloom, J. Fridrich, and T. Kalker (2008), *Digital Watermarking and Steganography*, 2 ed., Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Feng, J., and M. Potkonjak (2003), Real-time watermarking techniques for sensor networks, *Proceedings of SPIE - The International Society for Optical Engineering*, *5020*, doi:10.1117/12.479736.

Ibaida, A., I. Khalil, and R. van Schyndel (2011), A low complexity high capacity ecg signal watermark for wearable sensor-net health monitoring system, in *2011 Computing in Cardiology*, pp. 393–396.

Jetto, L., S. Longhi, and G. Venturini (1999), Development and experimental validation of an adaptive extended kalman filter for the localization of mobile robots, *IEEE Transactions on Robotics and Automation*, *15*(2), 219–229.

Jo, K., J. Kim, D. Kim, C. Jang, and M. Sunwoo (2014), Development of autonomous car—part i: Distributed system architecture and development process, *IEEE Transactions on Industrial Electronics*, *61*(12), 7131–7140.

Jo, K., J. Kim, D. Kim, C. Jang, and M. Sunwoo (2015), Development of autonomous car—part ii: A case study on the implementation of an autonomous driving system based on distributed architecture, *IEEE Transactions on Industrial Electronics*, *62*(8), 5119–5132.

Joachim, E., and G. Bernd (2002), *Informed Watermarking*, Kluwer Academic Publishers.

Kamel, I., and H. Juma (2011), A lightweight data integrity scheme for sensor networks, *Sensors (Basel, Switzerland)*, *11*(4), 4118–4136, doi:10.3390/s110404118, 22163840[pmid].

Lalem, F. (2016), Data authenticity and integrity in wireless sensor networks based on a watermarking approach.

Lin, C., and A. Sangiovanni-Vincentelli (2012), Cyber-security for the controller area network (can) communication protocol, in *2012 International Conference on Cyber Security*, pp. 1–7.

Longxiang, G., M. Sagar, L. Xuehao, and J. Yunyi (2017), Teaching autonomous vehicles how to drive under sensing exceptions by human driving demonstrations, *SAE Technical Paper 2017-01-0070*, doi:10.4271/2017-01-0070.

Lu, Z.-M., and S.-Z. Guo (2017), Chapter 1 - introduction, in *Lossless Information Hiding in Images*, edited by Z.-M. Lu and S.-Z. Guo, pp. 1 – 68, Syngress, doi:https://doi.org/10.1016/B978-0-12-812006-4.00001-2.

Madhavan, R., and C. Schlenoff (2003), Moving object prediction for off-road autonomous navigation, *Proceedings of SPIE - The International Society for Optical Engineering*, *5083*, doi:10.1117/12.485771.

Miller, C., and C. Valasek (2015), Remote Exploitation of an Unaltered Passenger Vehicle, *Black Hat USA, 2015.*

Park, S., M.-S. Gil, H. Im, and Y.-S. Moon (2019), Measurement noise recommendation for efficient kalman filtering over a large amount of sensor data, *Sensors*, *19*(5), 1168, doi:10.3390/s19051168.

Rigatos, G. G. (2010), Extended Kalman and Particle Filtering for sensor fusion in motion control of mobile robots, *Mathematics and Computers in Simulation*, *81*(3), 590–607, doi:10.1016/j.matcom.2010.05.003.

SAE J3016-201806 (2018), Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles , *Ground vehicle standard*, SAE International, United States.

Sarmento, A., B. Garcia, L. Coriteac, and L. Navarenho (2017), The challenges of the autonomous vehicle for emergent markets, *SAE Technical Paper 2017-36-0436*, doi:10.4271/2017-01-1406.

Sun, X., J. Su, B. Wang, and Q. Liu (2013), Digital watermarking method for data integrity protection in wireless sensor networks, *International Journal of Security and its Applications*, *7*, 407–416.

Technologies, M. (2018), Carnd-mercedes-sf-utilities, *GitHub repository*.

Tiwari, A., S. Chakraborty, and M. Mishra (2013), Secure data aggregation using irreversible watermarking in wsns, doi:10.1049/cp.2013.2337.

Woo, S., H. J. Jo, and D. H. Lee (2015), A practical wireless attack on the connected car and security protocol for in-vehicle can, *IEEE Transactions on Intelligent Transportation Systems*, *16*(2), 993–1006.

Woo, S., H. J. Jo, I. S. Kim, and D. H. Lee (2016), A practical security architecture for in-vehicle can-fd, *IEEE Transactions on Intelligent Transportation Systems*, *17*(8), 2248–2261.

Zhang, G., L. Kou, L. Zhang, C. Liu, Q. Da, and J. Sun (2017), A new digital watermarking method for data integrity protection in the perception layer of iot, *Security and Communication Networks*, *2017*, 1–12, doi:10.1155/2017/3126010.

Zou, Q., W. K. Chan, K. C. Gui, Q. Chen, K. Scheibert, L. Heidt, and E. Seow (2017), The study of secure can communication for automotive applications, in *WCX™ 17: SAE World Congress Experience*, SAE International, doi:https://doi.org/10.4271/2017-01-1658.