ALERT: Adding a Secure Layer in Decision Support for Advanced Driver Assistance System (ADAS)

Kanchan Bahirat, Umang Shah, Alvaro A. Cardenas and Balakrishnan Prabhakaran

The University of Texas at Dallas

Richardson, TX, USA

Kanchan.Bahirat,Umang.Shah,Alvaro.Cardenas,bprabhakaran@utdallas.edu

ABSTRACT

With the ever-increasing popularity of LiDAR (Light Image Detection and Ranging) sensors, a wide range of applications such as vehicle automation and robot navigation are developed utilizing the 3D LiDAR data. Many of these applications involve remote guidance - either for safety or for the task performance - of these vehicles and robots. Research studies have exposed vulnerabilities of using LiDAR data by considering different security attack scenarios. Considering the security risks associated with the improper behavior of these applications, it has become crucial to authenticate the 3D LiDAR data that highly influence the decision making in such applications.

In this paper, we propose a framework, ALERT (Authentication, Localization, and Estimation of Risks and Threats), as a secure layer in the decision support system used in the navigation control of vehicles and robots. To start with, ALERT tamper-proofs 3D LiDAR data by employing an innovative mechanism for creating and extracting a dynamic watermark. Next, when tampering is detected (because of the inability to verify the dynamic watermark), ALERT then carries out cross-modal authentication for localizing the tampered region. Finally, ALERT estimates the level of risk and threat based on the temporal and spatial nature of the attacks on LiDAR data. This estimation of risk and threats can then be incorporated into the decision support system used by ADAS (Advanced Driver Assistance System). We carried out several experiments to evaluate the efficacy of the proposed ALERT for ADAS and the experimental results demonstrate the effectiveness of the proposed approach.

CCS CONCEPTS

• **Computer systems organization** → **Real-time systems**; Robotics; • **Security and privacy** → *Network Security*;

KEYWORDS

LiDAR; 3D Watermarking; ADAS

ACM Reference Format:

Kanchan Bahirat, Umang Shah, Alvaro A. Cardenas and Balakrishnan Prabhakaran. 2018. ALERT: Adding a Secure Layer in Decision Support for Advanced Driver Assistance System (ADAS). In *2018 ACM Multimedia Conference (MM '18)*, October 22–26, 2018, Seoul, Republic of Korea. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3240508.3241912>

ACM acknowledges that this contribution was authored or co-authored by an employee, contractor, or affiliate of the United States government. As such, the United States government retains a nonexclusive, royalty-free right to publish or reproduce this article, or to allow others to do so, for government purposes only.

MM '18, October 22–26, 2018, Seoul, Republic of Korea

© 2018 Association for Computing Machinery.

ACM ISBN 978-1-4503-5665-7/18/10...\$15.00

<https://doi.org/10.1145/3240508.3241912>

1 INTRODUCTION

3D LiDAR (Light Imaging Detection and Ranging) sensors aid in sensing the environment where the distance is measured by projecting a laser beam on the target. Generally, LiDAR sensors use the *time of flight* approach for depth computation and the scanned data is stored as the list of points representing X, Y and Z coordinates (also termed as ‘unstructured 3D point cloud’). These sensors are typically accompanied with a pair of stereo cameras for obtaining the color information of the scene. LiDAR sensors outperform conventional color cameras due to their infrared signal based workflow which makes them useful at night time as well as in adverse environmental conditions. Therefore, most autonomous cars are equipped with LiDAR sensors.

However, considering the fatal crashes involving autonomous cars [20, 21] that occurred over a last couple of years, most of the self-driving car companies are adopting the concept of ‘teleoperation’ to keep a human in the loop. Aligned with this concept, California State has recently amended the regulation for testing autonomous vehicles that require a human remote operator to continuously supervise vehicle’s performance [31] using a two-way communication link between a vehicle and the remote operator [30]. One of the self-driving companies, Drive.ai suggests that starting with one operator per vehicle, one tele-choice operator will eventually be able to monitor multiple vehicles at the same time [1]. In such a scenario, it becomes difficult for a single operator to identify a forgery based on data consistency. Similarly, various mixed initiative remote robot control systems [14, 29] are introduced where a human can remotely guide a robot to complete a certain task without error.

In this context, as the data from a self-driving car or robot is communicated to a remote operator over a network, it is vulnerable to various forgery attacks. For example, the attacker may remove a vehicle in front of the self-driving car to create an illusion that there is no risk in increasing the speed and may result in a potential accident. Recently, [25, 26] showed that it is easy to stall a self-driving car just using simple commodity hardware. Thus, it becomes crucial to authenticate the data before utilizing it for decision making.

1.1 ALERT for ADAS

With this motivation, we propose a novel framework ALERT (Authentication, Localization, and Estimation of Risks and Threats) that adds a secure layer in the decision support systems used in Advanced Driver Assistance System (ADAS). Figure 1 describes an outline of the ALERT framework. As most autonomous cars use stereo RGB camera along with LiDAR, ALERT creates a dynamic watermark based on the stereo RGB data and employs fragile watermarking for tamper-proofing the 3D LiDAR data. The forgery

Figure 1: Overall workflow of the proposed ALERT framework.

detection in the proposed framework is a 2-fold setup that consists of a dynamic watermarking along with a cross-modal authentication and localization. It first identifies the attack by verifying the extracted watermark. If tampering is detected, the second step is triggered to localize the tampered region. ALERT also identifies the type of attacks and determines the risk factor associated with the data. While risk factor associated with an attack can be defined in different ways, the proposed formulation of the risk factor provides a holistic definition of the risk or threats based on various aspects of attacks including spatial and temporal consistency. ALERT provides a complete set of information to ADAS to make a more informed decision. Additionally, it can notify the tele-choice operator about the possible risk which is beneficial when one tele-choice operator is monitoring multiple vehicles. Experimental results show the effectiveness of the ALERT framework in terms of: a) imperceptibility and efficiency of the proposed dynamic watermarking; b) overall execution time taken by various steps in the framework and c) accuracy of forgery detection based on the discrepancy in the watermark and cross-modal authentication and localization.

Assumptions: As described in Figure 1, we assume that the stereo data is secured with traditional watermarking techniques [10, 18]. We focus on attacks targeting 3D LiDAR data.

Contributions: To summarize the principal contributions:

- Efficient and effective ordering scheme that overcomes the challenges posed by lack of surface information and sparse nature of 3D LiDAR data for tamper-proofing it.
- Dynamic nature of watermark generation and embedding that enhances the security.
- Cross-modal authentication and risk factor assessment that analyze the 3D LiDAR data thoroughly.

2 RELATED WORK

3D data authentication and verification have been done in two ways: a) Forensic analysis of the characteristics of the data to detect forgery attacks, and b) Embedding a secret data as a watermark to verify the authenticity of the 3D data.

2.1 Forensics Analysis

Over the last decade, forgery detection for images and videos has been well-studied [11, 22]. Forensic analysis of 3D data, however, is relatively less explored with the exception of a few methods [3, 28]. Raghuraman et al. [28] designed an anti-forensic framework to

capture and manipulate the live RGB-D data stream to create a realistic illusion of an individual performing the activities which they did not actually do. Presenting one of the key work in the area of 3D forensics, Bahirat et al. [3] identified various possible attacks on 3D LiDAR data as well as two forgery detection methods based on consistency in the key characteristics of LiDAR data such as: sampling density and occlusion effect.

2.2 3D Watermarking

Generally, watermarking methods can be broadly classified into two categories: robust and fragile methods. Robust methods [15, 27, 33, 37–39] are designed to reconstruct the watermark even in case of malicious attacks for providing ownership protection. Whereas fragile methods [16, 35] are designed to detect even a very slight modification in a data for authentication purpose. In this paper, we will mainly focus on the 3D fragile watermarking.

2.2.1 Fragile 3D Watermarking. Fragile watermarking methods should possess two desired properties: 1) vulnerability to even a slight modification of data; 2) localizing any kind of attack.

Spatial domain: Fragile techniques in the spatial domain generally modify the individual vertex positions to achieve vulnerability against different attacks. For example, methods [7, 17, 36] utilize an approach that identifies a new position for each vertex where two predefined hash functions have an identical value, in order to make all vertices valid for authentication [34]. However, these approaches suffer from causality problem (please refer to [34]) and need 1-ring neighbor information. Cayre and Macq [4] proposed a blind data-hiding algorithm for triangular meshes where the projection of a vertex on its opposite edge of the triangle is selected as a primitive for watermarking.

Transform domain: To achieve imperceptibility, researchers often embed the watermark in a spectral domain based on the 3D mesh multiresolution analysis. Chao et al. [6] proposed a fragile method that performs several decompositions of original triangular mesh and utilizes facets in the coarser mesh as authentication primitive. Wang et al [19] described another approach where after one wavelet decomposition, the norm and orientation of each wavelet coefficient vector are modified independently.

Point cloud watermarking: Among the early work, Ohbuchi [23] generated a non-manifold mesh from a point cloud and performed a mesh spectral analysis for embedding watermark into spectral coefficients. Cotting [9] designed a technique that: (a) partitions a point cloud data into patches; (b) transforms them into

Figure 2: Real-time 3D dynamic watermarking.

Figure 3: a) Cylinder enclosing the 3D space, b) Top view and the selected slice (highlighted with yellow) along with the corresponding 1D vector, c) Blockwise layout of the watermark and 1D vector corresponding to the selected block (marked in red).

discrete frequency bands; (c) embeds a watermark into the low-frequency components. Aggarwal et al. [2] proposed a blind watermarking that uses a quantization index of bit encoding to embed watermark bits into ordered points.

Limitations and Challenges: Most of the algorithms described are designed for 3D polygonal meshes using topological information, connectivity, and surface parameterization. Since LiDAR data do not have surface and connectivity information associated with it, we can not use most of the 3D mesh watermarking methods. Unlike existing fragile watermarking methods, the proposed watermarking is robust against re-ordering of points that is likely to occur during network transmission.

3 ALERT: AUTHENTICATION, LOCALIZATION, AND ESTIMATION OF RISKS AND THREATS

Most autonomous vehicles (and robots) are equipped with a LiDAR sensor as well as a pair of stereo color cameras. We propose a novel framework ALERT (Figure 1) that adds a secure layer for a navigation system that uses both LiDAR and stereo RGB data. It consists of three building blocks:

- A dynamic watermark based on stereo RGB data and embed it in 3D LiDAR data using fragile watermarking techniques.
- Cross-modal authentication and localization
- Risk factor computation

3.1 Dynamic 3D Watermarking

For achieving the desired tamper proofing of 3D LiDAR data, the proposed watermarking scheme (Figure 2) must possess the following key characteristics: a) **Dynamic watermark** is obtained

by creating a depth map from the stereo data. However, to correctly embed a 2D watermark into a list of 3D points, each 3D point must be mapped to a unique pixel in the 2D watermark. Such mapping is achieved by ordering 3D points as described later in this section. The additional benefit of the ordering strategy is the robustness against shuffling the points. Such a robustness against shuffling is advisable as merely re-ordering the 3D points does not affect the scene understanding and consequently the navigational decisions; b) **Imperceptibility** in embedding is attained by restricting the effect of embedding the watermark to the least significant bits (LSBs) of each 3D point; c) **Fragility** of the watermarking is very important to realize tamper-proofing of the data which is achieved by spreading the watermark across all 3D points. Such that any kind of tampering is easily reflected in the extracted watermark.

3.1.1 Watermark Generation: For increased security, instead of using a static watermark, we suggest utilizing a dynamic watermark that changes for every frame. We obtain the dynamic watermark by extracting the binarized depth map from the stereo data. To construct a depth map, first a disparity map from stereo data can be obtained using traditional stereo correspondence [24], followed by depth computation using a pin-hole camera model.

3.1.2 Watermark Embedding: The complete watermark embedding process consists of various steps:

Ordering 3D points: helps to create a mapping such that each 3D point is mapped to a unique pixel in a 2D watermark image. First, all the points are mapped to a Cylindrical coordinate system with the origin located at the position of the LiDAR sensor and z-axis aligned with the z-axis of the LiDAR (see Figure 3 a). Mathematically, the corresponding mapping for a 3D point (x, y, z) in the Cartesian coordinate system to the corresponding (r, θ, z) in the Cylindrical coordinate system is given by: $r = \sqrt{x^2 + y^2 + z^2}$, $\theta = \arctan(\frac{y}{x})$, $z = z$; where r , θ and z are radial, azimuth and height coordinates.

All the points represented in the Cylindrical coordinate system are then sorted based on the θ value and divided into different slices with angular width δ as shown in Figure 3 b. Hence, the total number of slices will be, $N_{slice} = \frac{360}{\delta}$. For each slice, all the corresponding points are first sorted with respect to z followed by r in the ascending order. This will form a 1D vector of all the 3D points in that slice (see Figure 3 b). The generated watermark is then divided into N_{slice} vertical blocks (see Figure 3 c).

Slice and Block Selection: As there are N_{slice} number of 3D data slices as well as watermark blocks, there is a one to one mapping possible between them. However, this correspondence between slices and blocks is randomized for enhanced security. Based

Figure 4: Watermark extraction and validation.

on the predetermined key (key_2), slice and the blocks are randomly chosen for embedding the watermark information.

Embedding: the watermark block into the corresponding slice depends on two factors: a) Watermark image size and b) Number of points in the slice, K . The size of the cropped watermarked image is selected based on the statistical analysis of the experimental dataset that determines the minimum number of points M in any slice. In the current experimental setup, the watermark image size is empirically determined to be $M \times N_{slice}$. Based on values of M and K various conditions are possible. When $M = K$, each watermark bit is simply embedded in the corresponding 3D point. If $M < K$, then a K points are divided into groups where each group contains M points except the last group. M bit watermark is replicated randomly across all groups. However, the last group with $< M$ number of points will contain only partial watermark information. If $M > K$ for any slice, it is merged with the next smallest slice. This process is repeated until the smallest slice has $K > M$. The information about slice merging is provided as a metadata at the receiver side. Please note that N_{slice} will also be updated based on merging operations.

3.1.3 Embedding Strategies: For embedding watermarking information in 3D points, one can use any traditional embedding strategies such as:

Low Bit Modulation (LBM): Due to simplicity, Low bit modulation is widely used in various watermarking techniques. In this strategy, the watermarked data is simply stored in the LSB (Least Significant Bit) of each 3D point which makes it imperceptible with least perturbation. To make sure the LSB only represents the watermark data, we first set the LSB of each 3D point to zero followed by replacing the LSB of each point with the corresponding watermark bit. Similarly on the receiver side, first the information stored in the LSB of the point cloud data is extracted. This LSB bit is then set to zero followed by the proposed ordering scheme. The mapping between 3D points and 2D pixels is then utilized to construct the extracted watermark image.

Quantization Index Modulation (QIM): QIM is a popular embedding strategy as it provides a good trade-off between imperceptibility, watermark capacity and robustness [5]. We adopt the basic QIM strategy proposed in [5] with slightly modified quantizers with quantization step Δ . To embed the watermark bit '1' in 'z' coordinate of point p , we quantize it by

$$Q_1(z) = \Delta \left\lfloor \frac{z}{\Delta} + \frac{1}{2} \right\rfloor. \quad (1)$$

To embed the watermark bit '0', we quantize the z coordinate by Q_0

Figure 5: Quantization index modulation. Each point is quantized to nearest reconstruction point for '0' or '1' bit.

If $z < Q_1$

$$Q_0(z) = \Delta \left\lfloor \frac{z}{\Delta} + \frac{1}{2} \right\rfloor - \frac{\Delta}{2}. \quad (2)$$

Else,

$$Q_0(z) = \Delta \left\lfloor \frac{z}{\Delta} + \frac{1}{2} \right\rfloor + \frac{\Delta}{2}. \quad (3)$$

Figure 5 illustrates the QIM in the case where one bit (0 or 1) is embedded per point. Thus, there will be two quantizers and the corresponding set of construction points in $\mathcal{R}$ are shown in Figure 5. So if we want to embed bit '1', the point p_1 will be quantized to n and to embed '0' it will be quantized to $(n + \frac{\Delta}{2})$.

At the receiver side, for extracting the watermark bit $wm_{extract}$ embedded in point $p(x, y, z)$, we solve for following equation:

$$wm_{extract}(p) = \underset{i=0,1}{\operatorname{argmin}} (Q_i(p(z)) - p(z)) \quad (4)$$

3.1.4 Meta data communication: For a successful operation of the proposed watermarking method, a metadata consisting of keys for slice and block selection, slice merging information and the size of watermark must be communicated to the receiver end via a secured channel. We suggest embedding this information in stereo data via a traditional image watermarking. To achieve this, the image watermarked must have additional capacity to store the metadata while satisfying the real-time constraints described later.

3.1.5 Watermark Extraction and Validation: Figure 4 describes a watermark extraction and validation process carried out at the receiver end of the system. First, the watermark information is extracted from the point cloud and saved as $wm_{extract}$. All the points in the received point cloud are mapped to a Cylindrical coordinate system and ordered as described in Section 3.1. The ordered set of points are divided into slices with angular width δ each. Using the predefined key (key_2) and metadata, the desired slice and corresponding block is selected to create a 2D binary watermark image from the extracted watermark data $wm_{extract}$. Simultaneously, a dynamic watermark is re-generated at the receiver end using stereo data. These two watermarks are then compared pixelwise to determine if the data is real or forged. Please note that, for the slices

Figure 6: Cross-modal authentication and localization.

Figure 7: Safety Region Illustration a) Side view, b) Top View (d_{safe} is the minimum stopping distance at current speed). Figure 8: Visualization for the temporal element of the Risk factor with $X_n = 5$ in case of a) Pattern 1, b) Pattern 2.

where $M < N$, the number of extracted bits are divided into groups of M bits and logical AND operation is performed to compute the watermark information in the corresponding block.

3.2 Cross-modal Authentication and Localization:

As the same scene is observed by two modalities: LiDAR sensor and stereo cameras, performing a cross-modal authentication is a natural choice where the data captured using one sensor can be validated using the data from another sensor. However, the data captured using these two modalities are in different domains. The stereo data is a set of 2D images while the point cloud is a set of 3D points. To bring them to the same domain for comparison, we process them as described in Figure 6. First, we generate a depth map from the stereo data and project it to 3D using the camera parameters. It will generate a corresponding point cloud data. As the estimated depth maps from stereo images can be noisy, we can not directly perform a point to point comparison between stereo and LiDAR point cloud. Hence, we apply an equirectangular projection to get a 2D depth map as shown in Figure 6. Similarly, we also apply equirectangular projection to the LiDAR point cloud. Now, as both the modalities are in same domain, we can verify if the objects

present in one modality are also there in another modality. Due to the 2D representation of data, we can easily identify the modified region using connected component analysis. As mentioned earlier, we assume that the stereo data has been authenticated using traditional video/image watermarking techniques.

3.3 Risk Factor Computation:

The decision making in autonomous vehicles is a complex system and involves multiple sensors. Such systems account for the fact that the performance of a sensor is highly dependent on the environmental conditions and sensor's reliability and precision might be affected by unfavorable conditions. Thus, for reliability, these systems need the self-assessment of each sensor where each sensor notes its own limitations and signals this information to the control unit [32]. Being consistent with existing decision-making systems, we design a 'Risk Factor' associated with the LiDAR data that conveys the reliability and risk associated with the data.

Needless to say, the risk factor can be defined in different ways. In this paper, we propose a risk factor R_f ranging between (0,1) where 0 indicates minimum risk while 1 indicates the highest risk. We consider a region (referred to as 'safety region') in front of the car that is more critical. The dimension of a safety region depends on the minimum stopping distance at current speed d_{safe} , the height of the car and width of lanes. Figure 7 illustrates the safety region and its coverage. To define the risk factor R_f , we mainly consider the following six key elements:

A distance of the attacked region from the vehicle, d . If the distance is d_{safe} , the risk is lower as we can probably stop before the attacked region. Hence, the corresponding factor is defined as $f_d = \max(0, \frac{(d_{safe} - d)}{d_{safe}})$.

The number of attacked points, n . Higher the density of attacked points, higher is the risk. Thus, the corresponding factor is $f_n = \frac{n}{N}$, where N is the total number of points in the safety region.

The type of attack, α . Different types of attack may pose different amount of risk. So, we define the corresponding factor as $f_\alpha = \alpha$, α can be between (0,1) based on the type of attack.

Angle of the attacked region with the forward direction of the vehicle, θ helps to identify if the attacked region is in the current lane or side lanes. $\theta = 0$ indicates that the attack is in the current lane which introduces higher risk. Hence, we formulate the corresponding factor as $f_\theta = \cos(\theta)$.

Spatial correlation of the attack, s . If the same region is attacked in two consecutive frames, the d and θ will be probably same across these frames. Hence to capture the spatial correlation, the corresponding factor is defined as $f_s = \text{average}_{i=0, X_n}(\text{corr}(s_i, s_{i-1}))$, where X_n is the total number of previous frames and $s_i = [d_i, \theta_i]$.

Figure 9: Illustration of different types of attacks on frame 6 from Drive 1: a) Original point cloud, b-g) With attack 1-6.

Temporal consistency of the attack, t . It includes two sub-factors: a) the percentage of frames being attacked and b) history information signifying the steadiness of the attack. The corresponding factor is $f_t = \frac{x_n}{X_n} \times \frac{c_n}{X_n}$, where x_n are the number of frames that are attacked, c_n holds the history information indicating the steadiness of attack. If the frame is attacked, we set $c_n = X_n$. But if there is no attack, it is updated as $c_n = c_n - 1$ until it reaches to 0. Among these elements, the first six elements are based on information in the current frame only, while spatial and temporal elements take a set of previous frames into account.

Definition: Based on the above parameters, risk factor R_f is defined as:

If the attack is detected, $R_f = f_d \times f_n \times f_\theta \times f_\alpha \times f_s \times f_t$

If no attack is detected, $R_f = f_s \times f_t$

For computing the temporal and spatial element of the risk factor, we consider X_n number of the previous frame. X_n is computed based on the current speed of the vehicle and time it would take to cover the distance of 100 meters (the distance for which one would like to observe the attack). Figure 8 illustrates the computation of c_n and f_t in various attacks. This formulation helps to distinguish between the attack patterns with the same number of attacked frames x_n but different attack distribution as shown in Figure 8.

4 ATTACKS

To evaluate the vulnerability of 3D LiDAR data, we consider 6 different possible attacks (See Figure 9) by an attacker trying to deceive the decision-making module in autonomous system.

Additive attacks (Attack 1): The attacker can introduce a new object to suddenly stop the autonomous vehicle (Figure 9 b). To perform such attacks, the region of interest in one point cloud is extracted and inserted into another by simply appending points to the initial list.

Subtractive Attack (Attack 2): As the objects that are near and in front of the LiDAR sensors are very important, we identify points in proximity of LiDAR sensor. These points are then removed to perform a subtraction attack (see Figure 9 c).

Simultaneous Additive and Subtractive attack (Attack 3): To increase the complexity of attack, we perform both additive attack and subtractive attack together (refer to Figure 9 d).

Replacement attack (Attack 4): In this case, the forged object is added in the scene to replace an existing object (Figure 9 e).

Translation (Attack 5): To trick the decision support module, the attacker may want to move the object a little farther to change the decision (Figure 9 f). We achieve it by identifying the nearest object and translate it away from the LiDAR sensor.

Rotation (Attack 6): The attacker may want to rotate the car parked on the one side of the road to alter the decision (Figure 9 g). This is achieved by identifying the nearest object, computing the

center of mass and translate it to the origin, rotate it about y-axis by a predefined angle and translate it back to its original position.

Attack Type Detection: All the attacks described earlier have different levels of risk associated with them. For example, the attack where the car in front is forgedly remove with a subtractive attack is more dangerous than the additive attack. Thus, determining the type of attack is crucial for determining the risk factor described in the earlier section. We broadly classified all the above attack into three categories: additive, subtractive and modification attack. All the attack 4-6 are termed as modification attack. To determine the type of attack the depth map obtained from the LiDAR data is masked with the difference image obtained using a cross-modal module. Finally, the type of the attack is determined based on the depth values in the masked region and corresponding area. For additive attack, the masked area will have non-zero depth values in the depth map. While, for subtractive and modification attack, the masked region will have a blob of pixels with zero depth values. However, the area of the zero depth patch will significantly differ in the case of subtractive and modification attack.

5 EXPERIMENTAL RESULTS

In this section, we demonstrate the efficiency of the ALERT framework. Mainly, we carried out 4 different set of experiments that: a) Demonstrate the imperceptibility of the proposed fragile watermarking method, b) Exhibit the efficiency of the proposed watermarking approach by computing a recovery rate, c) Verify the real-time performance by enlisting the execution time taken by various steps and d) Evaluate the accuracy of both the proposed fragile watermarking as well as cross-modal module under various attacks described in Section 4. All the attacks and algorithms are implemented in MATLAB and experiments are run on a CPU with Intel (R) Core (TM) i7-5820K with 3.30GHz speed and 32GB internal RAM. Various algorithm parameters used in the experiments are: a) With the statistical analysis of the experimental dataset (described later), the minimum number of points in any slice M is empirically determined as 1000; b) To get significant number of slices, angular width δ is set to 10; c) To keep the distortion level minimum due to watermark embedding, we use step size $\Delta = 0.0025$.

5.1 Dataset

We utilize LiDAR and stereo data provided by the KITTI Vision Benchmark Suite [12]. It consists of a variety of environments such as City, Residential, Road etc scanned using multiple sensors (please refer to [12]). In our experiments, we mainly considered the 3D point clouds scanned using Velodyne HDL-64E and stereo color images captured using 1.4 Megapixels Point Grey Flea 2 cameras. *Please note that the point cloud data and stereo data are synchronized and rectified.* We primarily focused on the drives in the ‘Residential’

Figure 10: Illustration of imperceptibility of the proposed watermarking: a) Original point cloud, b) Watermarked point cloud (with QIM embedding).

area as there are multiple cars parked on both sides of the road that allow us to create a significant number of attacks of type 2-6. The experimental dataset consists of three drives captured on 09-26-2011: drive 22, 39 and 64 (referred as Drive 1, 2 and 3, respectively) with 800, 395 and 570 frames each. Table 2 lists all the attacks performed to create forged data as described in Section 4. The total number of attacks created for Drive 1, Drive 2 and Drive 3 are 4135, 2130 and 2470 respectively. So overall in this experimental setup, we evaluated 8735 different types of attacks.

5.2 Evaluation

Imperceptibility. One of the key characteristics of the watermarking technique is the imperceptibility that requires the watermark embedding to introduce the least amount of distortion in the data. In the proposed approach, the desired imperceptibility is achieved by restricting the effect of embedding to the least significant bits of each point in the data. Figure 10 demonstrates the imperceptibility achieved using the proposed watermarking with QIM embedding. To quantitatively evaluate the imperceptibility, we compute a Hausdorff distance as described in [8] between the original point cloud and the watermarked point cloud for all the frames in each drive. The average Hausdorff distance across all drive is 0.001732 with LBM and 0.00125 using QIM.

Efficiency. To compute the efficiency of the proposed fragile watermarking method, we embed and extract the watermark with both the embedding strategies (LBM and QIM) under no attack condition. The recovery rate is recorded for every frame in each drive. The proposed method demonstrates 100% recovery rate in case of all the drives included in the experimental dataset.

Execution time. In the KITTI Vision Benchmark Suite [12], the LiDAR data is typically scanned at 10 fps while the synchronized and rectified stereo data is available only at 1 fps. Thus, we have 100 milliseconds (ms) to process each LiDAR frame for watermark embedding and extraction. Similarly, we have approximately 1 second for performing cross-modal authentication. The ‘real-time’ aspect of performance is defined based on the aforementioned time constraints. To evaluate the performance of the proposed system in terms of execution time, we measure the average execution time taken by different steps of the proposed framework for each drive. As can be seen from Table 1, we can process each frame for watermark embedding and extraction in approximately 100 ms each. Similarly, the execution time taken by cross-modal authentication module is approximately 200 ms satisfying the required time constraints. Please note that the reported times do not include watermark generation as it is considered as a pre-processing step.

	Average Execution time (in ms)		
	Drive 1	Drive 2	Drive 3
Ordering Points (b)	61.80	59.10	59.40
LBM Embedding (c)	38.00	35.80	36.80
QIM Embedding (d)	48.80	47.90	49.10
Execution time with LBM at Tx ($a + b + c$)	100.24	95.34	96.63
Execution time with QIM at Tx ($a + b + d$)	111.04	107.44	108.93
LBM Extraction (e)	47.50	45.40	47.20
QIM Extraction (f)	32.90	31.80	32.80
Watermark Comparison (g)	0.08	0.07	0.05
Execution time with LBM at Rx ($a + b + e + g$)	109.82	105.01	107.08
Execution time with QIM at Rx ($a + b + f + g$)	95.22	91.41	92.68
Depth From stereo (h)	172.98	174.12	173.24
Rectilinear projection (i)	12.35	12.21	12.45
Depth Comparison (i)	4.07	4.25	4.14
Execution time for cross-modal ($h + 2 \times i$)	201.75	202.79	202.28

Table 1: Execution time taken by various steps of ALERT.

Figure 11: Failure case of cross-modal authentication: a) Depth map from LiDAR data, b) Depth map from stereo data (noisy region marked with the red rectangle).

Accuracy. The proposed framework is evaluated by analyzing the accuracy of: a) the proposed fragile watermarking, and b) cross-modal authentication and localization. The original point cloud data is watermarked using the proposed watermarking with both embedding strategies (LBM and QIM). The data then undergoes various attacks listed in Section 4. The attacked data is processed through the step of watermark extraction and cross-modal module. Any discrepancy in the extracted watermark is detected as a forgery. Similarly, any difference between the LiDAR data and the stereo data is captured as forged data by a cross-modal module. Figure 12 illustrates the performance of the proposed cross-modal module in localizing various types of attacks. Table 2 shows the statistics of the attack detection for the proposed watermarking with LBM and QIM embedding strategy and for the proposed cross-modal authentication and localization. Overall, the proposed dynamic watermarking achieves 100.00% accuracy with both LBM and QIM embedding. The cross-modal module achieves 97.56% of overall accuracy. The reduced accuracy with a cross-modal module is mainly due to noisy depth estimation from stereo data as shown in Figure 11. Additional results are provided in the **supplementary material**.

6 DISCUSSION

Dynamic Watermark: As the algorithm is currently implemented on CPU in MATLAB, without any optimization, the execution time needed for the depth map generation from the stereo is approximately 170 milliseconds. GPU based implementation is very likely to reduce this execution time significantly. We can also consider generating a dynamic watermark for a LiDAR frame at time t in

Figure 12: Performance of cross-modal authentication and localization in case of: a) Original point cloud, b-g) With attack 1-6.

Type	Drive 1								Drive 2								Drive 3							
		LBM		QIM		Cross-modal			LBM		QIM		Cross-modal			LBM		QIM		Cross-modal				
	# of Frames	O	F	O	F	O	F		# of Frames	O	F	O	F	O	F	# of Frames	O	F	O	F	O	F		
Original	800	800	0	800	0	772	28	395	395	0	395	0	382	13	570	570	0	570	0	564	6			
Attack 1	800	0	800	0	800	11	789	395	0	395	0	395	1	394	570	0	570	0	570	0	570			
Attack 2	667	0	667	0	667	25	642	347	0	347	0	347	12	335	380	0	380	0	380	11	369			
Attack 3	667	0	667	0	667	0	667	347	0	347	0	347	2	345	380	0	380	0	380	1	379			
Attack 4	667	0	667	0	667	20	647	347	0	347	0	347	1	346	380	0	380	0	380	0	380			
Attack 5	667	0	667	0	667	24	643	347	0	347	0	347	9	338	380	0	380	0	380	13	367			
Attack 6	667	0	667	0	667	43	624	347	0	347	0	347	16	331	380	0	380	0	380	20	360			
Precision(%)		100.00		100.00		99.31		-	100.00		100.00		99.38		-	100.00		100.00		99.75				
Recall(%)		100.00		100.00		97.03		-	100.00		100.00		98.08		-	100.00		100.00		98.18				
Accuracy(%)		100.00		100.00		96.94		-	100.00		100.00		97.86		-	100.00		100.00		97.56				

Column 'O' refers to detected as Original and 'F' refers to detected as Forged.

Table 2: Statistics of attack detection for the proposed watermarking with LBM and QIM embedding strategy and for cross-modal authentication and localization.

two other ways: a) using the depth map obtained from stereo data at time $t - 1$ and b) using a 2D random noise pattern.

A trade-off between accuracy and time for depth from stereo: As described in Table 1, generating the depth map from stereo image pair is computationally expensive. If the depth map generation step needs to be expedited, it results in inaccurate depth estimates which consequently results in increased false positives during cross-modal authentication and localization. Thus, it becomes crucial to achieve a trade-off between the accuracy and execution time for depth map generation from stereo data.

Limited Localization: The current dataset includes the stereo data capturing only the frontal environment. Thus, based on the data availability, the current approach is limited to localize attacks occurring in the frontal 180° only. However, the full 360° localization can be obtained if multiple stereo camera pairs are installed for scanning the complete environment.

Decision making: As mentioned earlier, the proposed framework assist in autonomous navigation of self-driving cars and robots. It provides a complete set of information including whether the data has been attacked or not, attacked region, type of attacks and risk factor associated with the data. This complete set of information is provided to ADAS for making a more informed decision. For example, if the risk factor associated with the data is '1' and the attacked region is localized in the front of the vehicle and the side lane is marked as safe, then one can decide to pull over and stop. Although such decisions are subjected to the controversial debate of 'what is an ethical decision' [13], it is important to provide some preliminary decision for a reliable navigational system. Also for cases where one tele-choice operator is supervising multiple autonomous vehicles, ALERT can notify the tele-choice operator regarding the potential risk.

7 CONCLUSION

We have proposed a novel framework, ALERT, for assisting autonomous vehicles and robots. ALERT has 4 key features:

- Handles the sparse nature and lack of surface information in LiDAR data with the proposed ordering scheme.
- Creates a real-time, dynamic watermark based on stereo RGB data and embeds it on 3D LiDAR data using fragile 3D watermarking schemes.
- Localizes the attacked region using cross-modal authentication and localization along with risk factor assessment.
- Bestows a possible, complete solution for a secure navigation of autonomous systems.

ALERT is evaluated in terms of imperceptibility and efficiency of watermarking technique, execution time performance and accuracy of both watermarking as well as a cross-modal module. The experimental results demonstrate the effectiveness of ALERT and its possible utilization in assisting autonomous vehicles and robots. In future work, we plan to exploit various deep learning techniques to adaptively learn the forgery detection from the observed situation and bring more intelligence in ALERT.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation (NSF) under Grant No. 1012975 and US Army Research Office (ARO) Grant W911NF-17-1-0299. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF and ARO.

REFERENCES

- [1] Evan Ackerman. May 10, 2018. Drive.ai Launches Robot Car Pilot in Texas With a Focus on Humans. <https://www.dmv.ca.gov/portal/dmv/detail/vr/autonomous/testing>. (May 10, 2018).
- [2] Parag Agarwal and Balakrishnan Prabhakaran. 2007. Robust blind watermarking mechanism for point sampled geometry. In *Proceedings of the 9th workshop on Multimedia & security*. ACM, 175–186.
- [3] Kanchan Bahirat and Balakrishnan Prabhakaran. 2017. A study on lidar data forensics. In *Multimedia and Expo (ICME), 2017 IEEE International Conference on*. IEEE, 679–684.
- [4] François Cayre and Benoit Macq. 2003. Data hiding on 3-D triangle meshes. *IEEE Transactions on signal Processing* 51, 4 (2003), 939–949.
- [5] Brian Chen and Gregory W Wornell. 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 (2001), 7–33.
- [6] Wan-Hyun Cho, Myung-Eun Lee, Hyun Lim, and Soon-Young Park. 2004. Watermarking technique for authentication of 3-D polygonal meshes. In *International Workshop on Digital Watermarking*. Springer, 259–270.
- [7] Chang-Min Chou and Din-Chang Tseng. 2006. A public fragile watermarking scheme for 3D model authentication. *Computer-Aided Design* 38, 11 (2006), 1154–1165.
- [8] Paolo Cignoni, Claudio Rocchini, and Roberto Scopigno. 1998. Metro: Measuring error on simplified surfaces. In *Computer Graphics Forum*, Vol. 17. Wiley Online Library, 167–174.
- [9] Daniel Cotting, Tim Weyrich, Mark Pauly, and Markus Gross. 2004. Robust watermarking of point-sampled geometry. In *Shape Modeling Applications, 2004. Proceedings*. IEEE, 233–242.
- [10] Gwenael Doerr and Jean-Luc Dugelay. 2003. A guide tour of video watermarking. *Signal processing: Image communication* 18, 4 (2003), 263–282.
- [11] H. Farid. 2009. Image forgery detection. *Signal Processing Magazine, IEEE* 26, 2 (March 2009), 16–25. <https://doi.org/10.1109/MSP.2008.931079>
- [12] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. 2013. Vision meets Robotics: The KITTI Dataset. *International Journal of Robotics Research (IJRR)* (2013).
- [13] Edd Gent. 2017. Should Self-Driving Cars Make Ethical Decisions Like We Do? <https://singularityhub.com/2017/07/11/should-self-driving-cars-make-ethical-decisions-like-we-do/#sm.00014wpdjexpxee2vqmr24m3twq764>. (2017).
- [14] Frederik W Heger and Sanjiv Singh. 2006. Sliding autonomy for complex coordinated multi-robot tasks: Analysis & experiments. (2006).
- [15] L. Jing, W. Yinghui, H. Wenjuan, and L. Ye. 2014. A New Watermarking Method of 3D Mesh Model. *TELKOMNIKA Indonesian Journal of Electrical Engineering* 12, 2 (2014), 1610–1617.
- [16] H. Sean Lin, Hong-Yuan Mark Liao, C. Lu, and J. Lin. 2005. Fragile watermarking for authenticating 3-D polygonal meshes. *Multimedia, IEEE Transactions on* 7, 6 (2005), 997–1006.
- [17] H-YS Lin, H-YM Liao, Chun-Shien Lu, and Ja-Chen Lin. 2005. Fragile watermarking for authenticating 3-D polygonal meshes. *IEEE Transactions on Multimedia* 7, 6 (2005), 997–1006.
- [18] Yan Liu and Jiying Zhao. 2010. A new video watermarking algorithm based on 1D DFT and Radon transform. *Signal Processing* 90, 2 (2010), 626–639.
- [19] Michael Lounsbury, Tony D DeRose, and Joe Warren. 1997. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Transactions on Graphics (TOG)* 16, 1 (1997), 34–73.
- [20] Matt McFarland. 2016. Tesla driver killed in autopilot crash said the technology was 'great'. <http://money.cnn.com/2016/07/01/technology/tesla-driver-death-autopilot/index.html?iid=EL>. (2016).
- [21] Matt McFarland. 2018. Uber self-driving car kills pedestrian in first fatal autonomous crash. <http://money.cnn.com/2018/03/19/technology/uber-autonomous-car-fatal-crash/index.html?iid=EL>. (2018).
- [22] S. Milani, M. Fontani, P. Bestagini, M. Barni, A. Piva, M. Tagliasacchi, and S. Tubaro. 2012. An overview on video forensics. *APSIPA Transactions on Signal and Information Processing* 1 (2012). <https://doi.org/10.1017/ATSIP.2012.2>
- [23] Ryutarou Ohbuchi, Akio Mukaiyama, and Shigeo Takahashi. 2004. Watermarking a 3D shape model defined as a point set. In *Cyberworlds, 2004 International Conference on*. IEEE, 392–399.
- [24] Anders Olofsson. 2010. Modern stereo correspondence algorithms: investigation and evaluation. (2010).
- [25] J. Petit and S. Shladover. 2015. Potential cyberattacks on automated vehicles. *Intelligent Transportation Systems, IEEE Transactions on* 16 (2015), 546–556.
- [26] J. Petit, B. Stottelaar, M. Feiri, and F. Kargl. 2015. Remote Attacks on Automated Vehicles Sensors: Experiments on Camera and LiDAR. In *Black Hat Europe*.
- [27] Emil Praun, Hugues Hoppe, and Adam Finkelstein. 1999. Robust mesh watermarking. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 49–56.
- [28] S. Raghuraman, K. Bahirat, and B. Prabhakaran. 2015. Evaluating the efficacy of RGB-D cameras for surveillance.. In *ICME, IEEE*, 1–6. <http://dblp.uni-trier.de/db/conf/icmcs/icme2015.html#RaghuramanBP15>
- [29] Markus Sauer. 2011. Mixed-Reality for Enhanced Robot Teleoperation. (2011).
- [30] DMV State of California. February 26, 2018. Testing of Autonomous Vehicles. https://www.dmv.ca.gov/portal/wcm/connect/a6ea01e0-072f-4f93-aa6c-e12b844443cc/DriverlessAV_Adopted_Regulatory_Text.pdf?MOD=AJPERES. (February 26, 2018).
- [31] DMV State of California. February 26, 2018. Testing of Autonomous Vehicles with a Driver. <https://www.dmv.ca.gov/portal/dmv/detail/vr/autonomous/testing>. (February 26, 2018).
- [32] Christoph Stiller, Jochen Hipp, C Rössig, and A Ewald. 2000. Multisensor obstacle detection and tracking. *Image and vision Computing* 18, 5 (2000), 389–396.
- [33] J. Wang, J. Feng, and Y. Miao. 2012. A robust confirmable watermarking algorithm for 3D mesh based on manifold harmonics analysis. *The Visual Computer* 28, 11 (2012), 1049–1062.
- [34] Kai Wang, Guillaume Lavoué, Florence Denis, and Atila Baskurt. 2008. A comprehensive survey on three-dimensional mesh watermarking. *IEEE Transactions on Multimedia* 10, 8 (2008), 1513–1527.
- [35] B. Yeo and M. Yeung. 1999. Watermarking 3D objects for verification. *Computer Graphics and Applications, IEEE* 19, 1 (1999), 36–45.
- [36] Boon-Lock Yeo and Minerva M Yeung. 1999. Watermarking 3D objects for verification. *IEEE Computer Graphics and Applications* 19, 1 (1999), 36–45.
- [37] Kangkang Yin, Zhigeng Pan, Jiaoying Shi, and David Zhang. 2001. Robust mesh watermarking based on multiresolution processing. *Computers & graphics* 25, 3 (2001), 409–420.
- [38] Zhiqiang Yu, Horace HS Ip, and LF Kwok. 2003. A robust watermarking scheme for 3D triangular mesh models. *Pattern recognition* 36, 11 (2003), 2603–2614.
- [39] Stefanos Zafeiriou, Anastasios Tefas, and Ioannis Pitas. 2005. Blind robust watermarking schemes for copyright protection of 3D mesh objects. *IEEE Transactions on Visualization and Computer Graphics* 11, 5 (2005), 596–607.