ViewMap: Sharing Private In-Vehicle Dashcam Videos

Minho Kim, Jaemin Lim, Hyunwoo Yu, Kiyeon Kim, Younghoon Kim, and Suk-Bok Lee, *Hanyang University*

https://www.usenix.org/conference/nsdi17/technical-sessions/presentation/kim-minho

This paper is included in the Proceedings of the 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI '17).

March 27–29, 2017 • Boston, MA, USA

ISBN 978-1-931971-37-9

ViewMap: Sharing Private In-Vehicle Dashcam Videos

Minho Kim, Jaemin Lim, Hyunwoo Yu, Kiyeon Kim, Younghoon Kim, Suk-Bok Lee

Hanyang University

Abstract

Today, search for dashcam video evidences is conducted manually and its procedure does not guarantee privacy. In this paper, we motivate, design, and implement ViewMap, an automated public service system that enables sharing of private dashcam videos under anonymity. ViewMap takes a profile-based approach where each video is represented in a compact form called a view profile (VP), and the anonymized VPs are treated as entities for search, verification, and reward instead of their owners. ViewMap exploits the line-of-sight (LOS) properties of dedicated short-range communications (DSRC) such that each vehicle makes VP links with nearby ones that share the same sight while driving. ViewMap uses such LOS-based VP links to build a map of visibility around a given incident, and identifies VPs whose videos are worth reviewing. Original videos are never transmitted unless they are verified to be taken near the incident and anonymously solicited. ViewMap offers untraceable rewards for the provision of videos whose owners remain anonymous. We demonstrate the feasibility of ViewMap via field experiments on real roads using our DSRC testbeds and trace-driven simulations.

1 Introduction

A dashcam is an onboard camera that continuously records the view around a vehicle (Fig. 1). People install dashcams in their vehicles because irrefutable video evidence can be obtained in the event of an accident. Dashcams are becoming popular in many parts of Asia and Europe. For example, the rate of dashcam adoption has exceeded 60% in South Korea [1]. Other countries with high adoption rates include Russia and China. In these nations, the use of dashcams has now become an integral part of the driving experience of individuals.

Dashcams, as a side benefit, have tremendous potential to act as silent witnesses to others' accidents. Authorities such as police want to exploit the potential of dashcams because their videos, if collected, can greatly assist in the accumulation of evidence, providing a complete picture of what happened in incidents. For example, the police appeal for public to send in dashcam video evidences of certain traffic accidents [2, 3].

Figure 1: Dashcam installation on the windshield [4].

However, the current practice of volunteer-based collection limits participation of the general public due to the following reasons. First, people are reluctant to share their videos in fear of revealing their location history. Users want strong *anonymity*. Second, people are not interested in things that do not specifically concern them. Users want some form of *compensation* for provision of their videos. Third, people do not like to be annoyed by manual procedures, e.g., checking a wanted list, searching through their own videos, and sending in the matched videos all by themselves. Users want *automation* for hassle-free participation.

In this work, we aim to build a system that fulfills the key requirements above. There are, however, three challenges on the way. First, the authenticity of videos must be validated under users' anonymity. Verification of locations and times of videos should not rely on existing infrastructure such as 3G/4G networks where user identities may be exposed. Second, verifiable reward must be given without tracking users. The difficulty lies in making the reward double-spending proof while not linking it to users' videos. Third, irrelevant videos must be automatically rejected without human review. Under anonymity, attackers may simply upload an overwhelming number of fake videos, making it impractical to manually examine all by human eyes.

In this paper, we present ViewMap, a public service system (run by authorities) that addresses the challenges of dashcam sharing. To preserve user privacy, ViewMap takes a profile-based approach where each video is represented in a compact form called a *view profile* (VP), and the anonymized VPs are treated as entities for search, verification, and reward instead of their owners. Each VP makes verifiable links with nearby vehicles' VPs that share the same sight via line-of-sight (LOS) properties of DSRC radios. This VP linkage process also incorporates inter-vehicle path obfuscation for protection against tracking. The system accumulates anonymized VPs (from normal users) and trusted VPs (from authorities, e.g., police cars), and uses such LOS-based VP links to build a map of visibility on a given incident in the form of a mesh-like structure called a *viewmap*. Its key strengths are that: (i) trusted VPs do not need to be near the incident location; (ii) the system can pinpoint VPs whose original videos are worth reviewing, and reject, if any, fake VPs cheating locations and/or times via the unique linkage structure; and (iii) it leads minimal communication overhead because original videos are never transmitted unless verified taken near the incident. Users upload videos only when anonymously solicited by the system for further human checking. The system rewards users with virtual cash that is made untraceable based on blind signatures.

We demonstrate the feasibility of ViewMap via field experiments on real roads with our DSRC testbeds and trace-driven simulations. Our evaluations show that ViewMap provides: (i) users with strong location privacy (tracking success ratio $< 0.1\%$); and (ii) the system with high-accuracy verification ($> 95\%$) in face of an extremely large number of fake VPs cheating locations and times.

Besides the privacy of users sharing videos, there exists an additional privacy concern specific to dashcam video sharing. Video contents may threaten the privacy of others visible in the videos. We do not completely handle this, but provide some defense for video privacy. Specifically, we have implemented the realtime license plate blurring, which is integrated into ViewMap-enabled dashcams. However, other sensitive objects can still be captured (e.g., pedestrians walking into the view). This video privacy is not fully addressed in this work, and merits separate research.

In summary, we make the following contributions:

1. **New application:** We introduce a new application that enables sharing of dashcam videos. It poses unique challenges: combination of location privacy, location authentication, anonymous rewarding, and video privacy at the same time.
2. **Comprehensive solution package:** We present a solution suite that finds, verifies, and rewards private, location-dependent dashcam video evidence by leveraging DSRC-based inter-vehicle communications without resorting to existing infrastructure where user identities may be exposed.
3. **Prototype and evaluation:** We build a full-fledged prototype and conduct real road experiments using ViewMap-enabled dashcams with DSRC radios. It validates that LOS-based VP links are strongly associated with the shared "view" of videos in reality. Our evaluations show that ViewMap achieves strong privacy protection and high verification accuracy.

2 Background

Dashboard camera. Dashcams, installed on the windshield of a vehicle, continuously record in segments for a unit-time (1-min default) and store them via on-board SD memory cards. Once the memory is full, the oldest segment will be deleted and recorded over. For example, with 64 GB cards, videos can be kept for 2-3 weeks with 1-2 hours daily driving. Dashcams feature a built-in GPS system that provides vehicle speed and location. Some dashcams have an advanced feature of a dedicated parking mode, where videos can be recorded when the motion detector is triggered, even if a vehicle is turned off.

Dashcam video sharing. The most prevalent way to obtain dashcam video evidences today is public announcement, and users voluntarily hand in their videos. Some organizations adopt a more arranged approach where dashcam users have themselves registered for a pool of volunteers. For example, the police in South Korea operate such a system called shadow cops [5], but the number of the registered users is only a very small proportion of dashcam users in the country, less than 0.01% as of 2016. Recent studies [6, 7] report that privacy concerns and monetary motives are two major factors behind the sharing of dashcam videos for urban surveillance.

3 Motivation

3.1 Use Cases

Analyzing traffic accidents. When investigating a traffic accident, dashcam videos recorded by nearby vehicles are often valuable. While an accident vehicle may have its own video, it only offers one partial view and does not guarantee conclusive evidence. Nearby vehicles, on the other hand, have wider views each with different angle on the accident. However, the major impediment, besides lack of systematic search, is that people are reluctant to share videos due to privacy concerns and the associated hassle without compensation.

Investigating crimes. Dashcams can assist crime investigations and also have great potential for crime prevention. While CCTV cameras are installed in public places [8], there exist a countless number of blind spots. Dashcams are ideal complements to CCTVs since pervasive deployment—cars are everywhere—is possible. However, the difficulty here is that users are not often aware whether they have such video evidences. This is because criminal evidences are not as noticeable as traffic accidents. Thus, the current practice of volunteer-based collection has more serious limitation in this context.

3.2 Threat Model

User privacy. Users face privacy risks when providing personal, location and time-sensitive information. The system may wish to track users via such collected location samples. Time-series analysis on location samples, even if anonymized, could accumulate path information (i.e., following the footsteps) and eventually identify users location history [9, 10]. Such location tracking can further reveal users' very private information if the system can connect specific individuals to specific locations [11]. For example, an anonymous user at a hospital or other private location can be eventually identified if the user's path is tracked from his/her home (as a resident of a particular address). Besides, users face risk of revealing their identities and past locations when rewarded for their videos.

System security. On the other hand, if anonymity is provided, the system can becomes a target for various attacks. Dishonest users may claim rewards for fake videos cheating locations and/or times, or they may even fabricate video evidence. Human review may help identify and reject such videos. However, a manual review not only takes time, but also requires trained manpower. Anonymous attackers may launch denial-of-service attacks by simply uploading an overwhelming number of fake or dummy videos. Given limited resources of manpower, a manual review of all such videos would be impractical.

4 ViewMap Framework

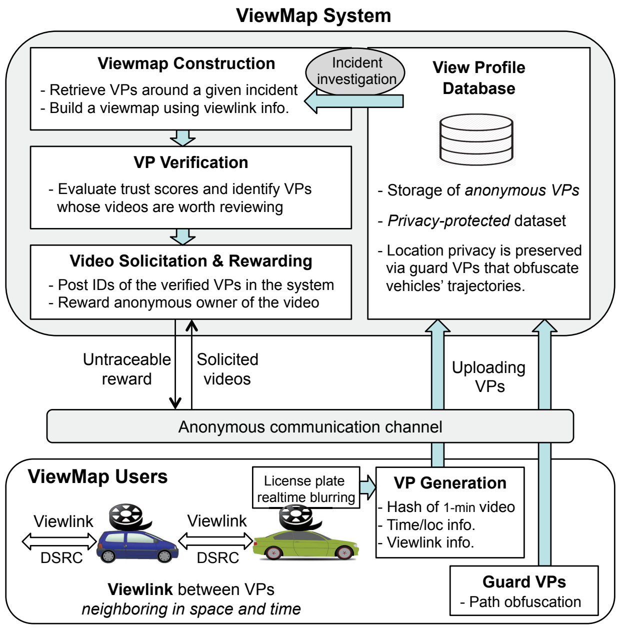
We first highlight the key features of ViewMap (Fig. 2).

Visual anonymization. Vehicles perform license plate blurring on their video stream while recording. Only such content anonymized videos are used in ViewMap, and hereafter referred to simply as "videos".

Profile-based anonymity. Each video (1-min default) is represented by its *view profile*, VP, that summarizes (i) time/location trajectory, (ii) video fingerprint, and (iii) fingerprints of videos taken by neighbor vehicles that share the same sight (via DSRC radios). We call such association between two neighbor VPs a *viewlink* (see Fig. 2). Vehicles anonymously upload their past VPs to the system whenever possible (e.g., WiFi or in-vehicle Internet access). These anonymized, self-contained VPs are stored in the VP database and collectively used for search and reward instead of their owners.

Location tracking protection. Vehicles also upload *guard VPs*, which are indistinguishable from actual VPs, to the system for protection against tracking. These guard VPs are created not for actual videos, but for plausible trajectories among neighbor vehicles such that their actual paths become indeterminable from the system's viewpoint. This is to guarantee location privacy in the VP database. Guard VPs are used only for path obfuscation, and vehicles delete them in their storage after submission.

Automated collection of video evidences. When video evidence is required, the system retrieves relevant anonymous VPs (from normal users) and trusted VPs (from authorities, e.g., police cars; not necessarily near



ViewMap System
Viewmap Construction
- Retrieve VPs around a given incident
- Build a viewmap using viewlink info.
Incident investigation
View Profile Database
VP Verification
- Evaluate trust scores and identify VPs whose videos are worth reviewing
- Storage of anonymous VPs
- Privacy-protected dataset
- Location privacy is preserved via guard VPs that obfuscate vehicles' trajectories.
Video Solicitation & Rewarding
- Post IDs of the verified VPs in the system
- Reward anonymous owner of the video
Untraceable reward
Solicited videos
Uploading VPs
Anonymous communication channel
ViewMap Users
License plate realtime blurring
VP Generation
- Hash of 1-min video
- Time/loc info.
- Viewlink info.
Viewlink
DSRC
Viewlink
DSRC
Guard VPs
- Path obfuscation
Viewlink between VPs neighboring in space and time

Figure 2: ViewMap framework.

the site), and builds viewmap(s) of a given incident using viewlinks among those VPs. The system exploits viewmap's unique structure to exclude, if any, fake VPs.[1] The verification criteria is: (i) 'trust scores' of the member VPs evaluated using trusted VPs as 'trust seeds' and (ii) their topological positions in the viewmap. The system solicits videos of the legitimate VPs (verified taken near the incident) by posting their VP identifiers,[2] without publicizing location/time of the investigation. Users upload anonymously the matched videos, if any. The videos are first validated against the system-owned VPs, and then reviewed by human investigators.

Untraceable reward. To reward users for provision of videos, their VP identifiers are likewise posted in the system. The users anonymously prove their ownership, and earn the virtual cash that is made untraceable via blind signatures. The authenticity of the virtual cash is self-verifiable, but never linked to users' VPs or videos.

5 Design of ViewMap

5.1 Protecting Location Privacy

5.1.1 Decoupling Users and Videos

Video recording. ViewMap-enabled dashcams are time-synched via GPS and continuously start recording new videos every minute on the minute. This is to facilitate the construction of viewmaps, each of which corre-

[1]We consider VPs (regardless of actual or guard VPs) that were created by users at their physical positions via the proper VP generation (Section 5.1) as *legitimate*; otherwise as *fake*.

[2]Both actual and guard VPs legitimately created near the incident may be on the requested video list, but the actual VPs only trigger video uploading. The guard VPs have already been deleted in users' storage.

sponds to a single unit-time (1-min) during an incident period. The recording procedure also performs license plate blurring in real time (detailed in Section 6.2) for video privacy (Fig. 3). The realtime plate blurring is chosen for two reasons: (i) post processing of videos, if allowed, opens the door for posterior fabrication of video evidence; and (ii) realtime visual anonymization can also alleviate visual privacy concerns[3] for the use of dashcams.

Video fingerprinting. Each vehicle, when completing 1-min recording of current video u, generates its unique view profile, VP_u that contains (i) time/location trajectory, (ii) fingerprints of u, and (iii) a summary (by a Bloom filter) of fingerprints of others' videos neighboring in space and time (via DSRC radios). To exchange fingerprints in the transient vehicular environment, each vehicle periodically (e.g., every second) produces and broadcasts a hash and associated information of its currently recording (for i secs: $1 \leq i \leq 60$) video u; we refer to such a cumulative fingerprint of u as *view digest*, VD_{u_i} (see Fig. 4). These VDs are exchanged between neighbors as below.

Broadcasting VDs. Every second, each vehicle A produces and broadcasts a view digest, VD_{u_i} of its video u currently being recorded for i secs using DSRC radio:

$$A \longrightarrow * : \quad T_{u_i}, L_{u_i}, F_{u_i}, L_{u_1}, R_u, H(T_{u_i}|L_{u_i}|F_{u_i}|H_{u_{i-1}}|u_i^{i-1}).$$

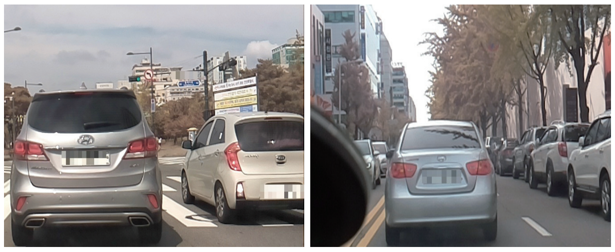
where T_{u_i}, L_{u_i}, and F_{u_i} are time, location, and byte-size of video u at ith second, respectively. L_{u_1} is the initial location of u used for guard VP generation (Section 5.1.2). R_u is VP identifier of u. $H_{u_{i-1}}$ is the hash of previous $\text{VD}_{u_{i-1}}$, and u_i^{i-1} is a newly recorded content from $(i-1)$th to ith seconds (see Fig. 4. Note: $H_{u_0} = R_u$). This cascaded hash operation facilitates the constant-time VD generation regardless of total file size. Note that original video u (within hash H) is not revealed in VD_u, and is later provided to the system only after VP_u is verified and anonymously solicited. The value R_u is further derived as $R_u = H(Q_u)$, where Q_u is file u's secret number chosen by A and is later used for untraceable rewarding.

Accepting neighbor VDs. Each vehicle A also receives VD_{x_j} broadcasted from nearby vehicle B:

$$B \longrightarrow * : \quad T_{x_j}, L_{x_j}, F_{x_j}, L_{x_1}, R_v, H(T_{x_j}|L_{x_j}|F_{x_j}|H_{x_{j-1}}|v_j^{j-1}).$$

where T_{x_j}, L_{x_j}, and F_{x_j} are time, location, and byte-size of video x at jth sec, respectively. A first validates VD_{x_j} by checking whether T_{x_j} and L_{x_j} are in acceptable ranges: T_{x_j} within the current 1-sec interval; L_{x_j} inside a radius of DSRC radios. If yes, A treats VD_{x_j} as a valid one. A temporarily stores at most two valid VDs per neighbor: the first and the last received VDs with same R value.

[3]Recording using a dashcam is strongly discouraged in some countries such as Austria and Switzerland due to visual privacy concerns.



(a) Dashcam video 1. **(b)** Dashcam video 2.

Figure 3: Videos recorded by our ViewMap-enabled dashcams that perform license plate blurring in real time.

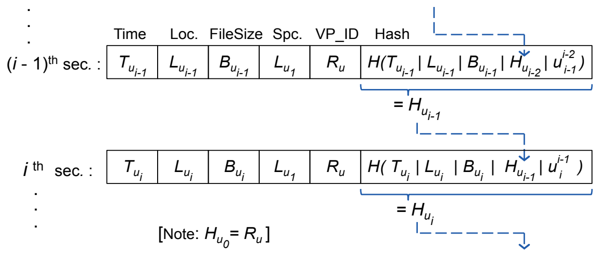


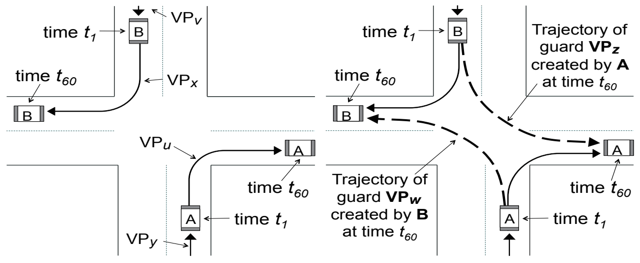
Figure 4: Example VDs of currently recording video u. A series of (sixty) VDs constitute a single (1-min) VP_u.

Generating actual VPs. Upon complete recording of current 1-min video u, each vehicle A generates its view profile VP_u. A first inserts the neighbor VDs (at most two VDs—the first and the last received VDs—per neighbor) into Bloom filter bit-array N_u. This reflects the neighbor VPs' cumulative fingerprints in part and their contact interval. A compiles its VDs (all VD_{u_i}: $1 \leq i \leq 60$) and Bloom filter bit-array N_u into VP_u, which will be anonymously uploaded to the system as described below.

5.1.2 Protection against Location Tracking

A collection of VPs, albeit each as anonymized location data, are subject to tracking. The system may follow a user's path by linking a series of VPs adjacent in space and time. For example in Fig.5a, the system tracking anonymous vehicle A can connect VP_y and VP_u using time-series location information. To obfuscate tracking, vehicles also generate and upload guard VPs as below.

Creating guard VPs. Each vehicle A generates guard VP(s) along with actual VP_u at the end of 1-min recording (e.g., time=t_{60} in Fig.5a). More specifically, among its m neighbor VPs, A randomly picks $\lceil \alpha \times m \rceil$ neighbor VPs ($0 < \alpha \leq 1$) and creates guard VPs for them (We use $\alpha = 0.1$, which is discussed in Section 6). For each chosen neighbor VP_x, A creates a guard VP (VP_z in Fig. 5b) whose trajectory starting at VP_x's initial location L_{x_1} (logged in its VDs) and ending at its own VP_u's last position. There are readily available on/offline tools that instantly return a driving route between two points on a road map. In this work, we make vehicles to use


time t_1
VP$_v$
time t_{60}
VP$_x$
B
VP$_u$
A
time t_{60}
A
time t_1
VP$_y$
Trajectory of guard **VP$_z$** created by **A** at time t_{60}
Trajectory of guard **VP$_w$** created by **B** at time t_{60}

(a) Actual VPs' trajectories. **(b)** Guard VPs's trajectories.

Figure 5: Protection against location tracking.

Google Directions API [12] for this purpose. In an effort to make guard VPs indistinguishable from actual VPs, we arrange their VDs variably spaced (within the predefined margin) along the given routes. Guard VPs are not for actual videos and thus, their hash fields are filled with random values. *A* makes neighborship between guard and actual VPs by inserting their VDs into each other's Bloom filter bit-arrays. *A* now clears all temporary memories, and resumes a new round of VP generation for the next recording video.

Cooperative privacy. Creating guard VPs for one another realizes continuously divergent paths from the viewpoint of observers, obfuscating the system's tracking of a vehicle's trajectory. Unlike the previous schemes like Mix-zones [13], this approach does not strictly require space-time intersections of vehicles' paths. Instead, vehicles can create path confusion any time, any place within DSRC range (up to 400 m). Note that they only need to be within DSRC range for a time to be considered as neighbors, not necessarily for the whole 1-min period.

Uploading VPs. Vehicles, whenever connected to the network (e.g., WiFi or in-vehicle Internet access), upload their actual and guard VPs anonymously to the system. We use Tor [14] for this purpose. More specifically, we make users constantly change sessions with the system, preventing the system from distinguishing among users by session ids. The submitted VPs are stored in the VP database of the system. Vehicles keep their actual VPs but delete guard VPs in their storage after submission.

5.2 Collecting Video Evidences

5.2.1 Viewmap Construction

In search of video evidences of an incident, the system builds a series of viewmaps each corresponding to a single unit-time (e.g., 1 min) during the incident period. We here describe the construction of a single viewmap at certain 1-min time interval at t, which is built only with VPs (actual and guard VPs)[4] whose times are t.

[4]From the system's perspective, actual and guard VPs are indistinguishable, thus treated equally. We hereafter refer to both as "VP".

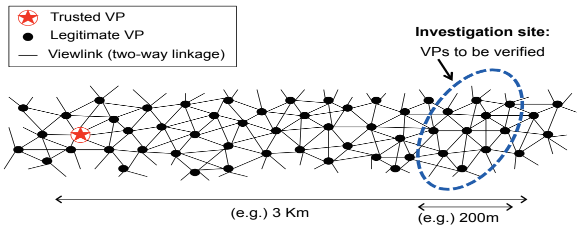


Figure 6: Illustration of a viewmap.

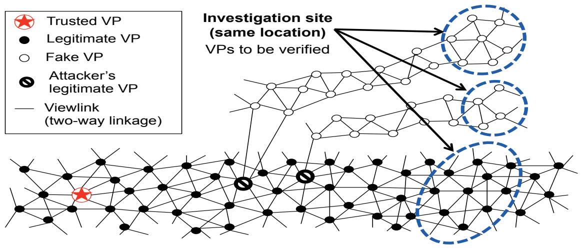


Figure 7: An example viewmap with fake VPs.

We first select the *trusted* VP$_s$ closest to the investigation site l, where VP$_s$ is a special VP from authorities such as police cars. The geographical coverage[5] of the viewmap spans an area C that encompasses l and VP$_s$ (Fig. 6). Then, all the VPs whose claimed locations are inside area C (anytime during incident time t) become members of the viewmap. For each member VP$_u$, we find its neighbor candidates VP$_v$ such that any of their time-aligned locations L_{u_i} and L_{x_j} are within the radius of DSRC radios. We then validate the mutual neighborship between VP$_u$ and VP$_v$ via membership queries (Bloom filter) using their element VDs on N_v and N_u. If none of the element VDs (of either VPs) passes the Bloom filter test, they are not mutual neighbor VPs. In this way, we build a viewmap by creating edges between 'two-way' neighbor VPs. We refer to such an edge as a *viewlink*, which signifies that two VPs are line-of-sight neighbors that share the same sight at some point in their times.

5.2.2 ViewProfile Verification

Given a constructed viewmap, there are certain VPs whose claimed locations (anytime during incident period) fall within an investigation site. The goal is to identify legitimate VPs among them. We consider VPs that join in a viewmap via the proper VP generation (as in Section 5.1) as *legitimate*; otherwise as *fake*.

Insights. A viewmap appears single-layered (Fig. 6) when all members are legitimate VPs. On the other hand, a viewmap with fake VPs results in multi-layered structure (Fig. 7), with only one layer containing a trust VP. This is because the validation of two-way linkage prevents attackers from creating arbitrary edges with other users' legitimate VPs without actual VD exchange. Thus,

[5]The coverage area of a viewmap is normally much larger than the size of the investigation site since trusted VPs (i.e., police cars) *were* not always very close to the site.

Algorithm 1 Verify_VPs(viewmap G)

```
Let $\mathbf{M}$ be the transition matrix on $G$
Let $\mathbf{d}$ be the trust distribution vector by the trusted VP
Let $\mathbf{P}$ be the trustRank-score vector, initially $\mathbf{P} = \mathbf{d}$
Let $\delta$ be a damping factor
// compute TrustRank scores
while not converged do
    $\mathbf{P} = \delta \cdot \mathbf{M} \cdot \mathbf{P} + (1-\delta) \cdot \mathbf{d}$
end while
// determine the legitimacy of VPs
Let $X$ be the all VPs within investigation site of $G$
Mark the highest scored VP $u \in X$ as 'LEGITIMATE'
Let $W (\subset X)$ be VPs reachable from $u$ only via $X$
for each VP $i \in W$ do
    Mark $i$ as 'LEGITIMATE'
end for
```

attackers can only inject fake VPs by linking them with their own legitimate VPs, if any. This is a very restrictive condition for attackers unless they happened to be actually in the vicinity of the incident. Moreover, the location proximity checking between neighbor VPs precludes long-distance edges and thus, forces attackers to create their own chain of fake VPs in order to place some of them in the target site.

Evaluating trust scores. To exploit the linkage structure of viewmaps, we adopt the TrustRank[6] algorithm tailored for our viewmap. In our case, a trusted VP (as a trust seed) has an initial probability (called trust scores) of 1, and distributes its score to neighbor VPs divided equally among all adjacent 'undirected' edges (unlike outbound links in the web model). Iterations of this process propagate the trust scores over all the VPs in a viewmap via its linkage structure. As shown in Algorithm 1, for a given viewmap G, trust scores P of all VPs are calculated via iterations of the following matrix operation:

$$\mathbf{P} = \delta \cdot \mathbf{M} \cdot \mathbf{P} + (1-\delta) \cdot \mathbf{d},$$

where $\mathbf{M}$ is the transition matrix representing VP linkage of G, $\mathbf{d}$ is the trust score distribution vector with an entry for trust VP only to 1, and δ is a damping factor empirically set to 0.8. The trust scores of all member VPs are obtained upon convergence of P.

Given a trusted VP z and an investigation site X, the VPs in X of z's layer are strongly likely to have higher trust scores than VPs in X of other layers. This is because the flow of trust scores has more chance of using edges within the base layer of z than using cross-layer edges (analyzed in Section 6). Accordingly, we identify the highest trust scored VP u in X as legitimate. All the VPs in X reachable from u strictly via VPs in X are determined legitimate as well.

[6]The TrustRank algorithm [15] outputs a probability distribution of likelihood that a person randomly clicking on links, starting from a certain 'seed' page, will arrive at any particular page.

5.2.3 Solicitation of Videos

Once VPs near a given incident is identified, the system solicits the videos. Owners are unknown and thus, they are requested via VP identifiers. More specifically, legitimate VP$_u$'s identifier R_u is marked as 'request for video' and posted in the system. Users check the list of solicited videos when accessing the system, and upload anonymously, if any, the matched video u along with its VP$_u$. The video is first validated via cascading hash operations against the system-owned VP, and then reviewed by human investigators.

5.3 Rewarding Anonymous Users

After human review, the system offers untraceable virtual cash to reward users for provision of videos based on their contributions. The system S posts VP identifier R_u (of reviewed video u) marked as 'request for reward', and corresponding user A obtains from S the virtual cash that is made untraceable using blind signatures [16] as follows: (i) A provides secret number Q_u of video u ($R_u = H(Q_u)$) to prove its ownership, (ii) A makes a blinded message $B(H(m_u), r_u)$ using a blinding secret r_u, (iii) S signs the message with its private key K_S^- not knowing the 'blinded' content m_u, and (iv) A unblinds the signed message using blinding secret r_u. This unblinded signature-message pair $(\{H(m_u)\}_{K_S^-}, m_u)$ results in virtual cash. When A presents it for payment, anyone can verify (i) its authenticity via S' signature and (ii) its freshness via double-spending checking on m_u, but fails to link A with his/her video u. Even the system cannot derive the link between A's virtual cash and video u's blinded message without the blinding secret r_u (only known to A). A more detailed description of the above procedure is provided in the Appendix.

6 Analysis of ViewMap

6.1 Overhead of ViewMap

Communication. The VP generation involves inter-vehicle communication to exchange up-to-the-second VDs via DSRC broadcast. The format of a VD message includes time and location information (8 bytes each), file size (8 bytes), VP identifier (16 bytes), and a cascaded hash value (16 bytes). Excluding PHY- and MAC-layer headers, the length of our VD message is thus only 72 bytes—even can be piggybacked into a DSRC beacon whose size is as large as nearly 300 bytes [17]. Vehicles also communicate with the system, where original videos are never transmitted unless specifically solicited. Instead, VPs are only transmitted, whose sizes are negligibly small compared with those of videos as shown below.

Storage. Given a 1-min dashcam video, its VP consists of 60 VDs and one Bloom filter bit-array (256 bytes in our context). Each video also has its secret number of 8 bytes for untraceable rewarding. Thus, the total storage

Platform	Blur time	I/O time	Frame rate
Rasp. Pi 3 (1.2 GHz)	50.19 ms	49.32 ms	10 fps
iMac 2008 (2.4 GHz)	10.72 ms	41.78 ms	18 fps
iMac 2014 (4.0 GHz)	10.18 ms	20.44 ms	30 fps

Table 1: Frame rates of realtime license plate blurring.

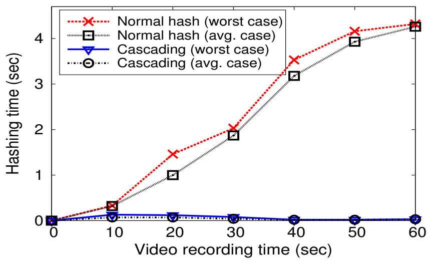


Figure 8: Hash generation times.

overhead incurred by a single VP is (60 $\times$ 72 bytes + 256 bytes + 8 bytes) = 4584 bytes. Given that the average size of 1-min video is 50 Mbytes, the storage overhead is less than 0.01% of original videos.

Computation. Two key operations that require computation are: car plate blurring (discussed shortly) and VD generation. To broadcast a new VD on time, a hash of the currently recording video should be produced within one second. Our test (Fig. 8), with Raspberry Pi (1.2 GHz CPU) as a dashcam for a 50-Mbyte 1-min video, shows that normal hashing times increase with recording time, and miss the deadline before 20 second of the video, reaching to 4.32 second at the end. On the other hand, our cascaded hashing results in constant-time hash generation with the worst-case of 0.13 second.

6.2 The Privacy of ViewMap

6.2.1 Visual Anonymity

Realtime license plate blurring. We have implemented the realtime license plate blurring using OpenCV [18] on Raspberry Pi as a ViewMap-enabled dashcam. The key task of license plate blurring is to localize plates in an image, a procedural part of popular car plate recognition algorithms [19, 20], whose realtime versions have been implemented for various mobile platforms such as iOS [21] and Android [22]. The license plate blurring procedure is as follows: (i) take the realtime video frame from camera module (**I/O time**), (ii) localize regions that contain license plates in the image via various parameters (e.g., area, aspect ratio)[7] and blur those areas (**Blur time**), (iii) write the plate blurred frame to the video file (**I/O time**). Our implementation on Raspberry Pi (1.2 GHz CPU) results in realtime processing with a frame rate of 10 fps (frame per second). Table 1 shows the time taken in each step when running our plate blurring implementation on Raspberry Pi as well as other platforms. We point out that the current prototype leaves more room for

[7]We use parameters tailored for South Korean license plates.

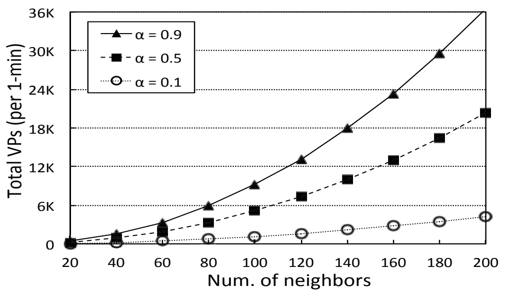


Figure 9: Volume of VP creation.

improvement, such as image processing using GPUs and multi-threading for blur and I/O operations. Still, our resulting videos at 10 fps, albeit frame rates not as high as normal movies (24-30 fps), are practically usable for the purpose of dashcams or video surveillance systems.

6.2.2 Location Privacy

The system can become a tracker using collected location traces in the VP database. We analyze the degree of privacy protection in the VP dataset against tracking.

Guard VPs against tracking. In order to create path confusion, each vehicle, among its m neighbor VPs, randomly selects $\lceil \alpha \times m \rceil$ neighbor VPs ($0 < \alpha \leq 1$) and generates guard VPs for them. Having a larger value of α creates higher degree of path confusion, but incurs excessively high volume of VPs in a dense environment (Fig. 9). Our design choice is to keep α small, but enough to preserve a high degree of privacy. The intuition behind is that driving time usually spans at least several minutes (10-min driving reported in [23]) so we exploit time factor t to have $P_t = \left[1 - \{1-(1-\alpha)^m\}^m\right]^t$ below 0.01, where P_t is the probability that there is any vehicle not covered by others' guard VP until time t. In this work, we use $\alpha = 0.1$, which makes P_t below 0.01 with 5-min driving.

Tracking process. The tracker's process can be formalized through target tracking algorithms [24, 25]. In our VP-based traces, the tracker's prediction only happens at the end of the currently tracking VP to link to the next VP. We assume a strong adversary with perfect knowledge of the initial position of target vehicle u such that $p(u,0) = 1$, where $p(i,t)$ denotes the attacker's belief (probability) that location sample $l(i,t)$ of time t belongs to the vehicle currently tracked. At each VP start-time t, predicted position $l_u(t)$ of target u is given based on the last sample $l(j,t-1)$ of the previous VP. We derive the tracker's belief $p(i,t)$ of target u in $l(i,t)$ within possible distance from $l_u(t)$ based on [23], which follows a probability model of distance deviation from the prediction and this model ensures $\sum_i p(i,t) = 1$ for any time t.

Location entropy. To measure the degree of privacy under tracking, we use location entropy, defined as $H_t = \sum_i p(i,t) \log p(i,t)$, a quantitative measure of the tracker's

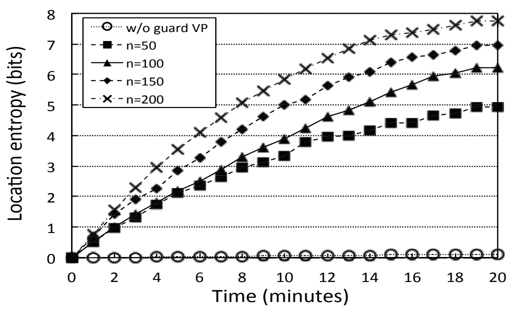


Figure 10: Location entropy.

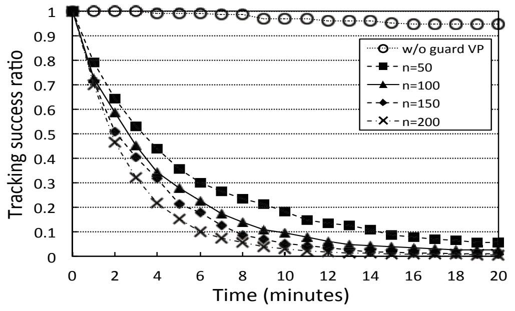


Figure 11: Tracking success ratio.

uncertainty [11, 23]. Here, H_t measures the uncertainty of correctly tracking a vehicle over time t. Lower values of H_t indicate more certainty or lower privacy.

Tracking success ratio. To give more intuitive privacy results, we also assess a tracking success ratio S_t, that measures the chance that the tracker's belief, when tracking target u over time t, is indeed true. Thus, S_t is equivalent to $p(u,t)$ of actual target u, since $\sum_i p(i,t) = 1$ at any time t. Note S_t is unknown to the tracker, who becomes unsure which $l(i,t)$ belongs to target u over time.

Privacy experiments. We collect a dataset of VPs created from 50-200 vehicles[8] traveling in the area of $4{\times}4\text{km}^2$ using *ns*-3 simulator. Fig 10 shows the average entropy over time. We also plot the entropy without guard VPs in the lowest density case ($n = 50$) as reference. Before ten minutes of driving, vehicles reach three bits of location entropy in the sparse case of $n = 50$. In other words, the tracker of a certain vehicle may suspect 8 different locations,[9] but without knowing the exact location. Fig 11 plots the average tracking success ratio over time. We see that, in the sparse case of $n = 50$, the tracking success ratio decreases to 0.2 before ten minutes and further drops below 0.1 before fifteen minutes. In the case without guard VPs, on the other hand, the tracking success ratio still remains above 0.9 even after twenty minutes. This result shows: (i) the privacy risk from anonymous location data in its raw form; and (ii) the privacy protection via guard VPs in the VP database.

[8]We present large-scale evaluations later in Section 8.

[9]X bits of entropy corresponds roughly to 2^X equally likely locations.

6.3 The Security of ViewMap

6.3.1 Attacks with a Large Number of Fake VPs

Attackers may submit fake VPs simply cheating locations and times. Such VPs are immediately excluded from a viewmap G unless linked to any legitimate 'member' VPs of G. Attackers must have physically positioned themselves in space and time on G to obtain legitimate VPs of G. This is a highly restrictive condition for attackers as they cannot predict the future investigation on G.

Let us nonetheless consider the case of attackers with such legitimate VPs on G. Under anonymity, the attackers can easily generate and inject a large number of fake VPs now all disguised as members of G. We further assume that the attackers share their fake VPs to increase their trust scores. To examine how ViewMap performs on such attacks, we provide our analysis on trust scores and our experiment results below.

Upper bounds of trust scores. Let $D_L(v)$ be the the set of VPs that are reachable from VP v by following at least L links. Similarly, we refer to the set of VPs that are distant from all VPs in a given group $\mathcal{G}$ by at least L links as $D_L(\mathcal{G})$ (i.e., $D_L(\mathcal{G}) = \cap_{v\in\mathcal{G}} D_L(v)$). By adopting Theorems 1 and 2 in [26], we obtain the upper bound for the sum of trust scores over $D_L(\mathcal{G})$ as follows.

Lemma 1. *Given a set T of trusted VPs, the sum of the trust scores over all VPs in $D_L(T)$ is at most α^L. That is, $\sum_{v\in D_L(T)} P_v \leq \alpha^L$ where P_v is the trust score of v.*

Due to the above lemma, a VP not closer to a given trusted VP than L links cannot have a trust score greater than α^L. It implies that a trust score decreases by at most the ratio α to the minimum distance to a given trusted VP.

We next provide the upper bound of the sum of trusted scores over all fake VPs in the following lemma.

Lemma 2. *Let A and F_A denote a set of attackers and the set of fake VPs, generated by attackers in A to try a colluding attack, respectively. The sum of trust scores over F_A is upper bounded as*

$$\sum_{v\in F_A} P_v \leq \frac{\alpha}{1-\alpha} \sum_{v\in A} \frac{|O_v \cap F_A|}{|O_v|} P_v \qquad (1)$$

where O_v represents the neighbors of v.

The proof is given in the Appendix.

The above lemma suggests that attackers may achieve high trust scores for fake VPs if attackers obtain high trust scores for their legitimate VPs. Considering Lemma 1, attackers can increase the chance of success in attacks if they position their legitimate VPs close to a given trusted VP. Furthermore, it shows that attackers will connect as many links to fake VPs as possible to increase the term $\frac{|O_v \cap F_A|}{|O_v|}$ in Inequation (1). Thus, the best strategy for an

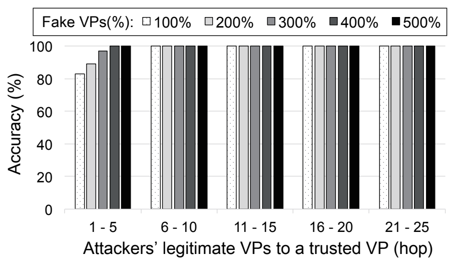


Figure 12: Verification result 1.

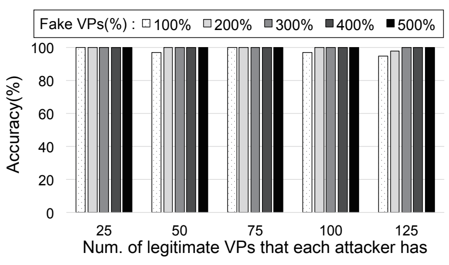


Figure 13: Verification result 2.

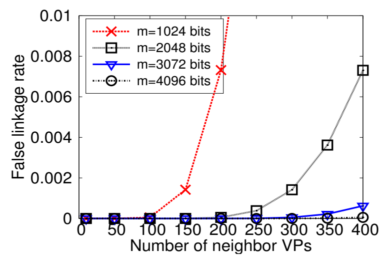


Figure 14: False linkage rate.

attacker is to connect every fake VP with its legitimate VP directly (while not practically possible in our viewmap), and the intuition leads us to the following corollary.

Corollary 1. *Assuming that attackers in collusion generate and inject n fake VPs, the upper bound of trust score of a fake VP positioned at the investigate site of a viewmap is* $\frac{1}{n}\frac{\alpha}{1-\alpha}\sum_{v\in A}\frac{|O_v \cap F_A|}{|O_v|}P_v$.

The above corollary suggests that injecting a large number of fake VPs is somewhat counterintuitively disadvantageous to attackers since their trust scores are distributed over all fake VPs—the more fake VPs, the lower their trust scores. Nevertheless, attackers have to create many fake VPs to spread them over a wide area so that some can reach the investigation site (publicly unknown), which in turn decreases the chances of their success in attacks.

Verification experiments. We run experiments on synthetic geometric graphs, as viewmaps with 1000 legitimate VPs. We use $5-15\%$ "human" colluding attackers injecting fake VPs that outnumber legitimate VPs up by 500% (i.e., 5000 fake VPs). Fig. 12 shows the results where accuracy indicates the cases that we correctly identify legitimate VPs during 1000 runs of each test. We see that, only the attackers whose legitimate VPs are very close to the trusted VP have the chance, albeit slight, of successful attacks (Lemma 2) while injecting many fake VPs rather decreases their chances (Corollary 1). In the experiments, accuracy is nearly 99% except the case of attackers in vicinity of the trusted VP (83% at worst). We however argue that such a case is rare since attackers cannot predict the future, e.g., location/time of an incident. We also consider another type of attacks where attackers prepare a lot of dummy videos beforehand and use them to obtain many legitimate VPs for a single viewmap. Fig. 13 shows that, under such a condition, accuracy is still high above 95%. This is because the trust scores of attackers are upperbounded by their topological positions within viewmap's linkage structure (out of their control) rather than their quantity.

6.3.2 False Linkage Attacks

ViewMap uses Bloom filters to validate VP linkage due to its compact size. One key problem with Bloom filters is false positives, leading to false linkage. ViewMap requires a low false linkage rate to prevent incorrect links. Given our two-way neighborship checking, the probability of false linkage is calculated as follows: $p = \left(1-\left[1-\frac{1}{m}\right]^{2nk}\right)^{2k}$ where m is a bit-array size, n and k are the number of neighbor VPs and the number of hash functions, respectively. Fig. 14 shows the false linkage rate using an optimal number of hash functions $k=(m/n)\ln 2$. Considering the maximum possible number of vehicles encountered in 1 minute, we choose to use m=2048 bits for our implementation. This has a false linkage rate of 0.1% with 300 neighbor VPs.

Attackers may fabricate their VPs' (Bloom filter) bit-arrays with all 1, claiming that they are neighbors of all other VPs. However, the validation of locations/times between VPs as well as the two-way neighbor checking prevent such attacks. Attackers may also try to poison neighbor VPs' (Bloom filter) bit-arrays to all 1, by sending out extremely many dummy VDs each associated with different VP while driving. Such a poisoning attack can be mitigated by limiting the number of neighbor VPs at each vehicle.[10]

7 Experiments on Real Roads

7.1 Measurement Framework

Our field measurement aims to provide answers to the following questions:

- Does our VP linkage reflect a line-of-sight (LOS) characteristic in reality?
- What are the implications of such LOS properties on (two-way) linked VPs and their videos?

Testbed implementation. Our DSRC testbed consists of on-board units (deployed in a rear window), Raspberry Pis with camera module (as dashcams), and Galaxy S5 phone (as a bridging device) as depicted in Fig. 18. We make Raspberry Pis as ViewMap-enabled dashcams to generate VDs of currently recording videos, which connects (via a Ethernet cable) to the DSRC OBU for VD broadcast, also connects (via Bluetooth) to the Galaxy phone installed with a Tor client [27] to anonymously upload its past VPs to our server.

[10]We set the maximum number of neighbor VPs accepted at each vehicle as 250 (neighbor vehicles) at this moment.

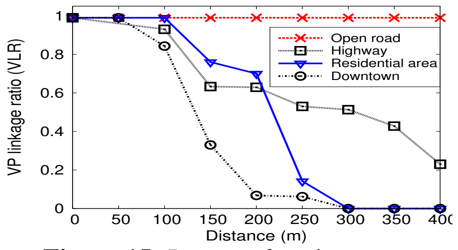


Figure 15: Impact of environment.

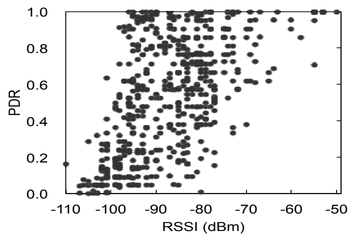


Figure 16: RSSI vs. PDR.

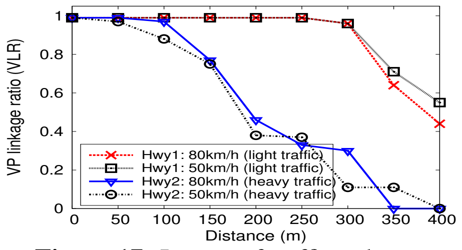


Figure 17: Impact of traffic volume.

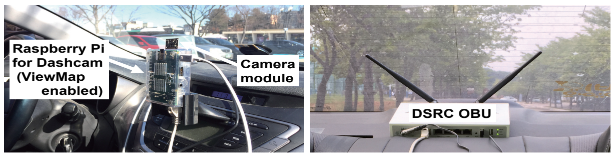


Figure 18: Pictures of our in-vehicle setup.

Experiment settings. We conduct our experiments for three weeks in the metropolitan area of Seoul, Korea. We run experiments using two vehicles equipped with our ViewMap-enabled DSRC testbeds under various environments: university campus, downtown, residential area, and highways. Each vehicle, continuously recording the real view using a Raspberry Pi's camera (with realtime processing of car plate blurring), sends out VD broadcast messages every second. The transmission power is set to 14 dBm as recommended in [17]. In our experiments, we make vehicles only upload to our server their actual VPs in order to facilitate analyzing the relationship between linked VPs and their videos.

7.2 Measurement Results

7.2.1 Clear Manifestation of Line-of-Sight Links

Fig. 15 shows the measurement result of VP linkage ratio (VLR) on various environments. In the figure, VLRs of the open road environment are consistently very high ($>$ 99%), indicating that *the range of VP linkage extends up to 400m if no obstacle is on the way*. Whereas, we see that VLRs vary and decrease with distance in the other scenarios. During the experiments, we observe that such unlinkage occurs mostly when the vehicles are blocked by buildings, or sometimes blocked by heavy vehicle traffic.

To inspect other factors that may affect VP linkage, we analyze the reception of VDs at both vehicles. Fig. 16 shows a scatter plot of average PDR vs. RSSI obtained from this experiment. It is generally true that higher RSSI results in better PDRs. However, when RSSI values fall in a range of -100 dBm to -80 dBm, we observe the fluctuating PDRs,[11] making it a less likely impacting factor to VP linkage. We also run experiments with different vehicle speed to see whether vehicle mobility such as Doppler effect affects VP linkage (Fig. 17). We see that, in each highway scenario, VLRs are insensitive to velocity for

[11]This observation conforms to the previous results reported in [17].

Scenario	Condition	VP linkage	On Video
Open road	LOS	100%	100%
Building 1	NLOS	0%	0%
Intersection 1	LOS	100%	93%
Intersection 2	NLOS	9%	0%
Overpass 1	LOS	84%	77%
Overpass 2	NLOS	0%	0%
Traffic	LOS/NLOS	61%	52%
Vehicle array	NLOS	13%	0%
Pedestrians	LOS	100%	100%
Tunnels	NLOS	0%	0%
Building 2	LOS/NLOS	39%	18%
Double-deck bridge	NLOS	0%	0%
House	LOS/NLOS	56%	51%
Parking structure	NLOS	3%	0%

Table 2: Summary of our measurement results.

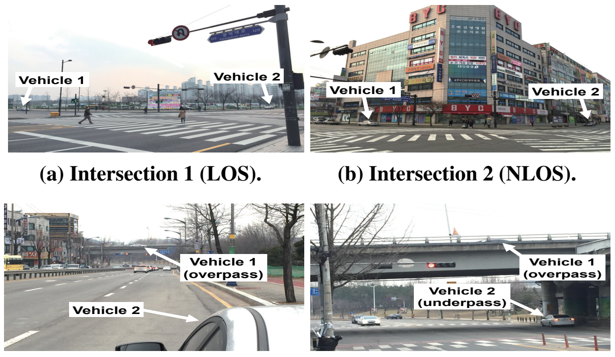
any given separation distance. We rather observe that traffic density on the road affects VP linkage. In our highway experiments, we obtain high(/low) VLRs when the traffic volume is heavy(/light). This result suggests that blockage by heavy vehicle traffic is also likely a impacting factor.

From this set of experiments, we observe that *distance, RSSI, and vehicle mobility have little impact on VLRs, rather line-of-sight condition appears a dominating factor to VP linkage*.

7.2.2 Strong Correlation between Linked VPs and Contents of Their Videos

To further examine our observation, we conduct a set of semi-controlled experiments. We carefully select various locations where two vehicles are situated in line-of-sight (LOS) or non line-of-sight (NLOS) or mixed conditions. We measure VLRs to analyze how such conditions affect VP linkage. Table 2 presents the measurement summary from those various scenarios (Pictures of some of our senarios are given in Fig. 19). As shown in the table, our measurements demonstrate that obstacles, especially artificial structures (e.g., buildings and bridges) in vehicular environments cause significant impact on VP linkage.

Given LOS-based VP linkage, we investigate the relationship between linked VPs and their videos. We review the videos taken by the vehicles in the experiments, and discover that, *either vehicle appears on the other's video only when two VPs (of two vehicles) are linked.* The results are reported in Table 2, where 'On Video' indicates



(a) Intersection 1 (LOS). **(b) Intersection 2 (NLOS).**

(c) Overpass 1 (LOS). **(d) Overpass 2 (NLOS).**

Figure 19: Pictures of our experiments.

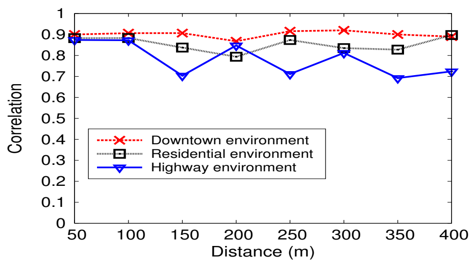


Figure 20: Correlation of VP links and video contents.

the case that either video of two time-aligned VPs captured the other vehicle at least for a moment. We see clear dependency between VP linkage and video contents.

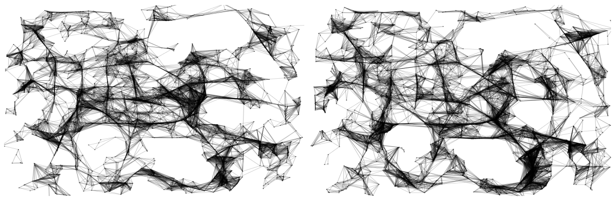
To further validate our analysis, we quantify the degree of association between two events, i.e., linkage between two VPs and visibility on their videos, using the Pearson correlation coefficient [28] as a function of the separation distances across all the data collected from the experiments. The result is shown in Fig. 20, and exhibits a strong degree of association. The correlation varies from 0.7 to 0.9, indicating that VP linkage is indeed associated with the shared “view”. This also suggests that when a vehicle involved in a traffic accident has its own VP$_u$, then videos of the neighbor VPs that are linked to VP$_s$ highly likely captured the accident.

8 Evaluation

To evaluate ViewMap in a large-scale environment, we run simulations using traffic traces.

Simulation setting. We use the network simulator *ns*-3 [29] based on traffic traces of 1000 vehicles obtained from SUMO [30]. We extract a street map (8×8km^2) of the city of Seoul via OpenStreetMap [31]. Output of each simulation is a collection of VPs (each for 1-min). Given those VPs, we construct viewmaps, each of which corresponds to a single 1-min during simulation time.

Simulation results. We first examine the features of such traffic-derived viewmaps. To give a feel for how they actually look, Fig. 21 depicts viewmaps[12] built from VPs of our simulations where all vehicles move at an average speed of 50km/h (Fig. 21a) and an average speed of 70km/h (Fig. 21b), respectively. Fig. 22c shows the average contact interval between vehicles. We see that contact intervals are not very short in general. This result suggests that vehicles have sufficient time to establish VP links with nearby vehicles as long as they are in line-of-sight condition. Fig. 22f quantifies the viewmaps. We see the some VPs, albeit a few (<3%), are isolated from the viewmaps. We point out that building a viewmap does not require every VP in the area to join. We rather expect that such a case is normal when ViewMap is deployed.



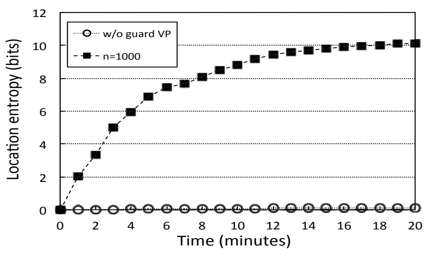
(a) Vehicle speed: 50km/h **(b)** Vehicle speed: 70km/h

Figure 21: Viewmaps from traffic traces.

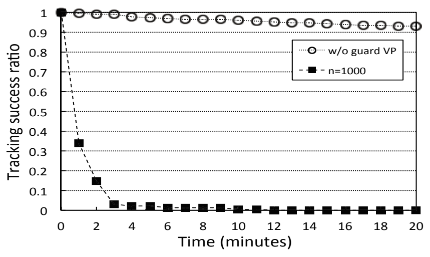
We assess the privacy of the collected VP dataset against tracking as in Section 6. We here present the results of the mix speed scenario as the other cases show similar trends. Fig. 22a shows the average entropy over time. Before ten minutes of driving, vehicles reach eight bits of location entropy, implying that the tracker of a certain vehicle may suspect 256 different locations not knowing exactly where to locate it. Fig. 22b plots the average tracking success ratio over time. It decreases to 0.1 before three minutes and further drops nearly 0.01 before ten minutes. Whereas, it still remains above 0.9 even after twenty minutes without guard VPs. This conforms to the privacy results in a sparser environment given in Section 6 and demonstrates privacy protection in the VP database.

To evaluate the verification performance of the traffic-derived viewmaps, we create fake VPs in the same way as described in Section 6. The results are averaged over 1000 runs of each test set. Fig. 22d plots the verification accuracy in face of fake VPs that outnumber the legitimate ones in the viewmaps. The results confirm the high accuracy—100% in most cases and 82% at worst in the special case of attackers in vicinity of the trusted VP. This is consistent with our analysis in Section 6. We again note that such a case is rare and is a highly restrictive condition for attackers as they cannot predict the future investigation. We also plot the verification accuracy under concentration attacks in Fig. 22e, where each attacker, in a given viewmap, possess as many as 125 legitimate but dummy VPs and create a large number of fake VPs. The result shows that accuracy is still high above 95%. As

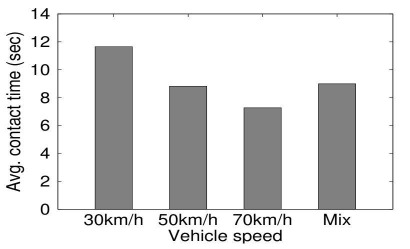
[12]The shape of the viewmap reflects the actual road network of a certain area of Seoul that we use for simulations.



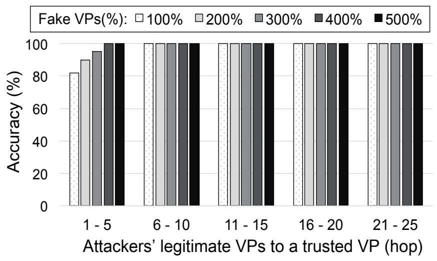
(a) Location entropy



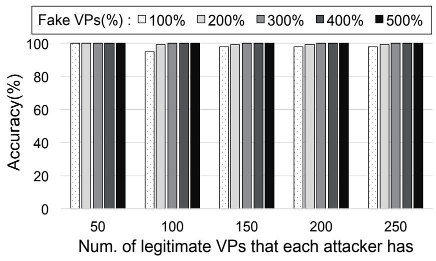
(b) Tracking success ratio



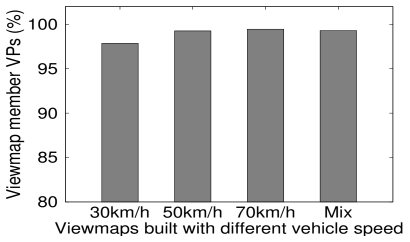
(c) Average contact time between vehicles



(d) Accuracy vs. attackers' positions



(e) Accuracy over concentration attacks



(f) Percentage of viewmap member VPs

Figure 22: Simulation results from different scenarios.

explained, the trust scores of attackers are upperbounded by their topological positions within viewmap's linkage structure (out of their control) rather than their quantity.

9 Related Work

Dashcam video sharing has received little attention so far in the research communities. Recent survey studies [6, 7] find that people with greater privacy concerns have lower reciprocal altruism and justice motive, but have higher monetary motive. These findings have significant implications on the design of dashcam video sharing systems, and ViewMap takes them as design elements.

There have been many proposals on location privacy in a mobile environment, and they focus on hiding users' true positions in the context of location-based services (LBS). *k*-anonymity [32] hides a user's location by using a large region that encloses a group of *k* users in space, instead of a single GPS coordinate. This however decreases spatial accuracy. *CliqueCloak* [33] uses a smaller region but waits for *k* different users in the same region. This delay compromises temporal accuracy. Mix-zones [13] makes users' paths indistinguishable if they coincide in space and time. However, such space-time intersections are not common, especially with frequent and high-accuracy location reports as in our ViewMap. Path confusion [23] resolves the mix-zones' space-time problem by incorporating a delay in revealing users' locations. This delay also compromises temporal accuracy in location data. *CacheCloak* [11] eliminates such delay via users' predicted location reports. Their goal is to hide user identities, but users' claimed positions are not verified.

Existing location verification techniques aim to localize users' positions or to determine the legitimacy of users' location claims. They perform location verification in various ways, from using ultrasound communication [34], to multilateration via infrastructure such as base stations [35, 36] or sensor nodes [37], to directional antenna [38]. Because their objective is to verify users' locations, users are more authenticated rather than anonymized—location privacy is not guaranteed.

10 Conclusion

This paper introduces ViewMap, an automated public service system that enables sharing of dashcam videos while preserving user privacy. The key insight is: (i) to leverage line-of-sight properties of DSRC to link between videos, in their compact form of view profiles (VPs), that share the same sight while driving; and (ii) to exploit inter-vehicle communications to create path confusion for protection against tracking. We use such LOS-based VP links to build a viewmap around a given incident, and identify videos that are worth reviewing. We demonstrate the feasibility of ViewMap via real road experiments. Our evaluation of ViewMap shows high degree of user privacy (tracking success ratio $< 0.1\%$) and high verification accuracy ($> 95\%$). In a broader scope, our solution explores to share private, location-dependent data without resorting to existing infrastructure such as 3G/4G networks where user identities may be exposed.

Acknowledgment

We sincerely thank our shepherd Dr. Ranveer Chandra and the anonymous reviewers for their valuable feedback. This work was supported by Samsung Research Funding Center for Future Technology under Project Number SRFC-IT1402-01.

References

[1] Top Rider. 2015. Research on Vehicle Blackbox. (2015). https://www.top-rider.com:447/news/articleView.html?idxno=19493.

[2] Call made for dashcam rewards. BT. 24 February 2015. http://home.bt.com/lifestyle/motoring/motoring-news/call-made-for-dashcam-rewards-11363964238054.

[3] Police appeal for public to send in dashcam footage of dangerous drivers. Dailymail. 5 January 2014. http://www.dailymail.co.uk/news/article-2534042/.

[4] 2016. "Dashcams P1210466" by Fernost - Own work. Licensed under Public Domain via Commons. (2016).

[5] Shadow Cops in Korea. Daegu Police Department. https://www.dgpolice.go.kr.

[6] Sangkeun Park, Joohyun Kim, Rabeb Mizouni, and Uichin Lee. Motives and concerns of dashcam video sharing. In *Proc. ACM CHI*, 2016.

[7] Kelly Freund. When cameras are rolling: Privacy implications of body-mounted cameras on police. In *Columbia Journal of Law & Social Problems*, 2015.

[8] Tan Zhang, Aakanksha Chowdhery, Paramvir (Victor) Bahl, Kyle Jamieson, and Suman Banerjee. The design and implementation of a wireless video surveillance system. In *Proc. ACM MOBICOM*, 2015.

[9] Marco Gruteser and Xuan Liu. Protecting privacy in continuous location tracking applications. In *IEEE Security and Privacy Magazine*, 2004.

[10] Julien Freudiger, Reza Shokri, and Jean-Pierre Hubaux. Evaluating the privacy risk of location-based services. In *Proc. International Conference on Financial Cryptography and Data Security*, 2011.

[11] Joseph Meyerowitz and Romit Roy Choudhury. Hiding stars with fireworks: Location privacy through camouflage. In *Proc. ACM MOBICOM*, 2009.

[12] Google Directions API. https://developers.google.com/maps/documentation/directions/.

[13] A.R. Beresford and F. Stajano. Location privacy in pervasive computing. In *IEEE Pervasive Computing*, 2003.

[14] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *USENIX Security Symposium*, 2004.

[15] Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. Combating web spam with trustrank. In *Proc. VLDB*, 2004.

[16] David Chaum. Blind signatures for untraceable payments. In *Springer-Verlag*, 1988.

[17] Fan Bai, Daniel D. Stancil, and Hariharan Krishnan. Toward understanding characteristics of dedicated short range communications (DSRC) from a perspective of vehicular network engineers. In *Proc. ACM MOBICOM*, 2010.

[18] OpenCV. http://opencv.org/.

[19] Shyang-Lih Chang, Li-Shien Chen, Yun-Chung Chung, and Sei-Wan Chen. Automatic license plate recognition. 2004.

[20] Duan Tran, Tran Le Hong Du, Tran Vinh Phuoc, and Nguyen Viet Hoang. Building an automatic vehicle license plate recognition system. In *Proc. International Conference in Computer Science*, 2005.

[21] Real-time video processing algorithm for instant license plate recognition in iOS Apps. http://rnd.azoft.com/instant-license-plate-recognition-in-ios-apps/.

[22] Android automatic license plate recognition. https://github.com/SandroMachado/openalpr-android.

[23] Baik Hoh and Marco Gruteser. Preserving privacy in gps traces via uncertainty-aware path cloaking. In *Proc. ACM CCS*, 2007.

[24] Donald B. Reid. An algorithm for tracking multiple targets. 1979.

[25] Marco Gruteser and Baik Hoh. On the anonymity of periodic location samples. In *Proc. Conference on Security in Pervasive Computing*, 2005.

[26] Junghoo Cho and Uri Schonfeld. Rankmass crawler: A crawler with high pagerank coverage guarantee. In *VLDB*, pages 375–386, 2007.

[27] Tor on Android. https://www.torproject.org/docs/android.html.

[28] Stefan Rolewicz. Functional analysis and control theory: Linear systems. In *volume 29 of East European Series. Springer*, 1987.

[29] ns-3. https://www.nsnam.org/.

[30] SUMO. http://sourceforge.net/projects/sumo/.

[31] OpenStreetMap. https://www.openstreetmap.org.

[32] Latanya Sweeney. K-anonymity: A model for protecting privacy. In *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 2002.

[33] B. Gedik and L. Liu. Location privacy in mobile systems: A personalized anonymization model. In *Proc. IEEE ICDCS*, 2005.

[34] Naveen Sastry, Umesh Shankar, and David Wagner. Secure verification of location claims. In *Proc. ACM Workshop on Wireless Security*, 2003.

[35] Srdjan Čapkun, Kasper Bonne Rasmussen, Mario Cagalj, and Mani Srivastava. Secure location verification with hidden and mobile base stations. In *IEEE Transactions on Mobile Computing*, 2008.

[36] Jerry T. Chiang, Jason J. Haas, and Yih-Chun Hu. Secure and precise location verification using distance bounding and simultaneous multilateration. In *Proc. ACM Conference on Wireless Network Security*, 2009.

[37] Srdjan Čapkun and Jean pierre Hubaux. Secure positioning of wireless devices with application to sensor networks. In *Proc. IEEE INFOCOM*, 2005.

[38] Ziwei Ren, Wenfan Li, and Qing Yang. Location verification for vanets routing. In *Proc. WiMob*, 2009.

A Detailed Description of Untraceable Rewarding

The system S posts VP identifier R_u (of reviewed video u) marked as 'request for reward', and corresponding user A obtains the virtual cash that is made untraceable using blind signatures via an anonymous channel as follows:

$$\begin{aligned}
A \longrightarrow S:&\quad \mathrm{VP}_u, Q_u.\\
S \longrightarrow A:&\quad n\\
A \longrightarrow S:&\quad B(H(m_u^1), r_u^1), ..., B(H(m_u^n), r_u^n)\\
S \longrightarrow A:&\quad \{B(H(m_u^1), r_u^1)\}_{K_S^-}, ..., \{B(H(m_u^n), r_u^b)\}_{K_S^-}
\end{aligned}$$

where Q_u is a secret number for video u ($R_u = H(Q_u)$) chosen by A when generating VP_u. Upon validation of VP_u and Q_u against u, the system notifies A of the amount of virtual cash (in a predefined monetary unit), n, for video u. Then, A generates n pairs of random messages and their blinding secrets $(m_u^1, r_u^1), ..., (m_u^n, r_u^n)$, and sends the system S their blinded version $B(H(m_u^1), r_u^1), ..., B(H(m_u^n), r_u^n)$ where $B(\cdot)$ is the blinding operation. The system S returns the messages signed with its private key K_S^- without knowing their 'blinded' contents. Then, A unblinds each of the signed messages as follows:

$$U(\{B(H(m_u^i), r_u^i)\}_{K_S^-}) = \{H(m_u^i)\}_{K_S^-}$$

where $U(\cdot)$ is the unblinding operation that requires the blinding secret r_u^i (only known to A). This unblinded signature-message pair $(\{H(m_u^i)\}_{K_S^-}, m_u^i)$ results in one unit of virtual cash. When A presents it for payment, anyone can verify (i) its authenticity via S' signature and (ii) its freshness via double-spending checking on m_u^i, but fails to associate A with his/her video u. Even the system S cannot derive the link between A's virtual cash $(\{H(m_u^i)\}_{K_S^-}, m_u^i)$ and video u's blinded message $B(H(m_u^i), r_u^i)$ without the blinding secret r_u^i, hence untraceable.

B Proof of Lemma 2

Proof. Suppose that v is a fake VP in F_A. Then, the static trust score in for v must be zero because v is clearly not legitimate. By the recursive definition of TrustRank for an individual VP v, we obtain

$$P_v = \alpha \sum_{u \in O_v} \frac{P_u}{|O_u|}. \tag{2}$$

Let E_A be a subset of F_A in which each VP is directly linked by an attacker in A. With an exception for the VPs in E_A, summing Equation (2) for all fake VPs in F_A is the same as we add $\frac{P_u}{|O_u|}$ repeatedly $|O_u|$ times for every $u \in F_A$. Note that for $v \in E_A$, Equation (2) is shown as $P_v = \alpha \sum_{u \in O_v \cap F_A} \frac{P_u}{|O_u|} + \alpha \sum_{u \in O_v \cap A} \frac{P_u}{|O_u|}$. Thus, we get

$$\begin{aligned}
\sum_{v \in F_A} P_v &= \alpha \sum_{v \in F_A} \sum_{u \in O_v} \frac{P_u}{|O_u|}\\
&= \alpha \left(\sum_{v \in F_A} P_v - \sum_{v \in E_A} \sum_{u \in A} \frac{P_v}{|O_v|} + \sum_{v \in A} \sum_{u \in E_A} \frac{P_v}{|O_v|} \right)\\
&\leq \alpha \sum_{v \in F_A} P_v + \alpha \sum_{v \in A} \frac{|O_v \cap E_A|}{|O_v|} P_v
\end{aligned}$$

Since $O_v \cap E_A$ is identical to $O_v \cap F_A$ with $v \in A$, it is simple to draw the upper bound of $\sum_{v \in F_A} P_v$ presented in Inequation (1). □