TOWARDS UTILITY-DRIVEN DATA ANALYTICS WITH DIFFERENTIAL PRIVACY

BY

HAN WANG

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science
in the Graduate College of the
Illinois Institute of Technology

Approved _____
Adviser

Chicago, Illinois
May 2023

ACKNOWLEDGEMENT

I am humbled and grateful to have the opportunity to complete this thesis, and I would like to express my deepest appreciation to all those who have supported me throughout my graduate studies. First and foremost, I would like to thank my thesis advisor, Dr. Yuan Hong, for his unwavering support, guidance, and encouragement. His expertise in data security and privacy has been invaluable in shaping the direction of my research and helping me develop the skills and knowledge needed to complete this thesis. I am truly grateful for his mentorship, which has inspired me to become a better researcher and a better person. I would also like to express my gratitude to the members of my thesis committee, Dr. Pengjun Wan, Dr. Boris Glavic, and Dr. Wei Chen, for their insightful feedback and constructive criticism of my work. Their expertise and experience have contributed greatly to the quality and rigor of my research, and I am thankful for their guidance and support.

I am also grateful to all of my collaborators and labmates, Dr. Jaiddep Vaidya, Dr. Yu Kong, Dr. Shangyu Xie, Dr. Bingyu Liu, Hanbin Hong, Shuya Feng, and Jayashree Sharma for their companionship, intellectual stimulation, and help in various aspects of my research. Their dedication, passion, and enthusiasm have been a source of inspiration and motivation for me, and I am fortunate to have had the opportunity to work with such a talented and supportive group of people. Furthermore, I owe a special debt of gratitude to my family, for their unwavering love, support, and encouragement throughout my academic journey. Their sacrifices, patience, and understanding have enabled me to pursue my dreams and achieve this milestone in my life. In conclusion, I am deeply grateful to all those who have supported me along the way, and I look forward to applying the knowledge and skills I have gained to make a positive contribution to future research. Thank you all for being a part of my academic journey and for contributing to my personal and professional growth.

AUTHORSHIP STATEMENT

I, Han Wang, attest that the work in this thesis is substantially my own.

In accordance with the disciplinary norm of authorship in Computer Science, the following collaborations occurred in the thesis. (Please consult Appendix S of the IIT Faculty Handbook for further information.)

My adviser, Prof. Yuan Hong, contributed to all of my research and this thesis as a Doctorate supervisor. I would like to acknowledge the invaluable contribution of Prof. Jaideep Vaidya, Prof. Yu Kong, Prof. Li Xiong, and Prof. Zhan Qin. Their expertise and guidance have been instrumental in shaping the direction and scope of my research. Specifically, they provided insightful feedback and constructive criticism throughout the process, which helped me to refine my ideas and arguments. Without their guidance, this work would not have been possible. I am deeply grateful for their support and mentorship. Dr. Shangyu Xie helped me write the code of experiments in Chapter 3. Hanbin Hong helped me develop the methodology, analyze the results and adjust the figures in Chapter 4. Chapter 3, 4, and 5 were published and referenced in [1–3]. I have obtained permission from all other co-authors, Jaideep Vaidya, Yu Kong, Li Xiong, Zhan Qin, Shangyu Xie and Hanbin Hong, to use the material in the thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The widespread use of personal devices and dedicated recording facilities has led to the generation of massive amounts of personal information or data. Some of them are high-dimensional and unstructured data, such as video and location data. Analyzing these data can provide significant benefits in real-world scenarios, such as videos for monitoring and location data for traffic analysis. However, while providing benefits, these complicated data always raise serious privacy concerns since all of them involve personal information. To address privacy issues, existing privacy protection methods often fail to provide adequate utility in practical applications due to the complexity of high-dimensional and unstructured data. For example, most video sanitization techniques merely obscure the video by detecting and blurring sensitive regions, such as faces, vehicle plates, locations, and timestamps. Unfortunately, privacy breaches in blurred videos cannot be effectively contained, especially against unknown background knowledge. In this thesis, we propose three different differentially private frameworks to preserve the utility of video and location data (both are high-dimensional and unstructured data) while meeting the privacy requirements, under different well-known privacy settings. Specifically, to our best knowledge, we propose the first differentially private video analytics platform (VideoDP) which flexibly supports different video queries or query-based analyze with a rigorous privacy guarantee. Given the input video, VideoDP randomly generates a utility-driven private video in which adding or removing any sensitive visual element (e.g., human, and object) does not significantly affect the output video. Then, different video analyses requested by untrusted video analysts can be flexibly performed over the sanitized video with differential privacy. Secondly, we define a novel privacy notion ϵ-*Object Indistinguishability* for all the predefined sensitive objects (e.g., humans, vehicles) in the video, and then propose a video sanitization technique Verro that randomly generates utility-driven synthetic videos with indistinguishable objects. Therefore,

all the objects can be well protected in the generated utility-driven synthetic videos which can be disclosed to any untrusted video recipient. Third, we propose the first strict local differential privacy (LDP) framework for location-based service (LBS) ("L−SRR") to privately collect and analyze user locations or trajectories with ϵ-LDP guarantees. Specifically, we design a novel LDP mechanism "*staircase randomized response*" (SRR) and extend the empirical estimation to further boost the utility for a diverse set of LBS Apps (e.g., traffic density estimation, k nearest neighbors search, origin-destination analysis, and traffic-aware GPS navigation). Finally, we conduct experiments on real videos and location dataset, and the experimental results demonstrate all frameworks can have good performance.

CHAPTER 1
INTRODUCTION

1.1 Motivation

The widespread use of personal devices and dedicated recording facilities has led to the generation of massive amounts of personal information or data. Some of them are high-dimensional and unstructured data, such as video and loacation data. Analyzing such complex, unstructured and voluminous data [8] would be extremely beneficial in real world. For instance, traffic monitoring videos can be analyzed by traffic authorities, urban planning officials, and researchers [9] for learning urban traffic and pedestrian behavior. The App Waze not only navigates the routes with real-time traffic conditions but actively collects extra information (e.g., accidents, road construction, and police) from users based on their locations and shares them to other users.

However, privacy is a major concern in real-world applications where personal data is collected and analyzed for various purposes. The privacy issues arise when data is not adequately protected, and sensitive information is exposed, leading to negative consequences for individuals, which highlights the need for robust privacy protection techniques in real-world applications. Video data is a kind high-dimensional and unstructured data that stores massive amount of personal information. Directly releasing videos to the analysts may result in severe privacy concerns due to the considerable amount of sensitive information involved in videos, such as human faces, objects, identities and activities [10]. For instance, traffic monitoring cameras can capture all the vehicles which may involve the make, model and color of vehicles, moving speed and trajectories, and even the drivers' faces. Not every vehicle owner is willing to share the visual information, the extracted traveling data (e.g., locations and driving speed) [11]. Another example is video surveillance sys-

tems that are meant to monitor an area of interest with one or few of the following goals: law enforcement, personal safety, traffic control and resource planning, and security of assets [12]. While ensuring safety and deterrence, it also easily compromises the privacy of innocent individuals. Thus, privacy preserving solutions for videos have attracted significant interests recently. Location and traffic data is another kind high-dimensional and unstructured data. Location-based services (LBS) are widely deployed in mobile devices to provide useful and timely location-based information to users. All of these LBS Apps highly rely on the personal locations collected from millions of users. Such locations should be protected, e.g., per the General Data Protection Regulation (GDPR) since visited places can be sensitive (e.g., hospital) or used to re-identify users from the data (e.g., a sequence of them can be unique).

The notion of differential privacy was first proposed by Dwork [13] which provides a rigorous privacy guarantee for statistical databases [14] against arbitrary background knowledge possessed by adversaries. Differential privacy incorporates a central aggregator with access to the raw data of users and aims at bounding the risk enhancement to one's privacy when she/he contributes her/his data. Such privacy notion has been extended to sanitize and release data for different applications, such as classification [15], histograms [16], search logs [17], locations [18], trajectories [19], and data synthesis [20, 21]. Local differential privacy is a privacy notion that extends from the differential privacy and also protects users' privacy against any background knowledge. However, even if an adversary has access to the personal responses of an individual in the database, that adversary will still be unable to learn too much about the user's personal data, which provides stronger privacy protection than differential privacy. As we know either the differential privacy or local differential privacy has been widely used to sanitize and release data in many real database (e.g., statistical databases [13], search logs [22], and graphs [23] [24]), to the best of our knowledge, no attempt has yet been made to benefit from them in the high-dimensional unstructured

databases, especially the video dataset and location dataset.

For instance, Fan [25] applied Laplace noise to obfuscate pixels in an image to ensure DP for protecting specific regions of an image. However, the image quality has been significantly reduced (protection for each single pixel without composition may not work as well [26]). AdaTrace [27], a differentially private location trace synthesizer was proposed to ensure provable privacy, deterministic attack resilience, and strong utility. However, in the DP scenario setting [17], it requires an authorized data center to collect user's location. Unfortunately, in the 2011 Microsoft survey, 87% of participants reported that they care about who accesses their location information; over 78% workers of Amazon interviewed in 2014 still do not trust these LBS applications on collecting their locations and believed apps accessing to their locations can pose significant privacy threats [28]. Thus, it is highly desirable to explore private location collection by an *untrusted* server with differential privacy techniques.

1.2 Objective and Contributions

During my research, I have combined the development of practical differentially private framework with rigorous theoretical analysis, and incorporated techniques from various disciplines such as computer vision, privacy, and statistical analysis. Specifically, we proposed three different frameworks for two type high-dimensional and unstructured data in some well-known privacy settings in the following, shown in Figure 1.1.

We first propose a novel platform (namely, VideoDP) that ensures differential privacy [14] for any video analysis requested from untrusted data analysts, including queries or query-based analyses over the input video. Notice that, as the state-of-the-art privacy model, differential privacy (DP) [14] can ensure indistinguishable analysis result derived from the input data with and without any single record (protecting any

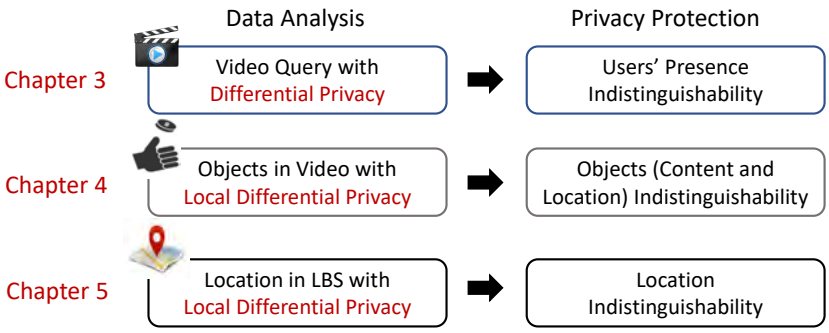


Figure 1.1. Overview of the Proposed Frameworks

record against arbitrary background knowledge). Second, I apply the concept of the emerging *local differential privacy* [29–31] (user-level indistinguishability) to video data and propose a novel platform (namely, VERRO) to generate an utility-driven synthetic video object indistinguishability (object-level indistinguishability instead of user-level indistinguishability), in which the content and trajectories of objects in videos are indistinguishable. The Compared to VideoDP, VERRO could release the synthetic video without revealing any privacy of users. At last, we peopose the first strict LDP framework (namely, "L-SRR") to support general location/trajectory aggregation and individual Location-based services to provide user-level privacy protection. We have conducted extensive experiments to validate the performance of VideoDP, VERRO and `L-SRR` on real-world datasets.

1.2.1 VideoDP. Specifically, we define a novel DP notion in which adding or removing any sensitive visual element (e.g., human or object) into the input video does not significantly affect the analysis result. Thus, the privacy risks can be strictly bounded even if the adversaries possess arbitrary background knowledge (e.g., knowing the objects or humans). To our best knowledge, this is the first work proposed to provide DP video analysis. Specifically, in VideoDP, we address the following unique challenges (different from the existing DP schemes applied to other datasets, e.g., [14, 16, 18, 32–34]).

We consider the identification of sensitive visual element as the root cause of privacy leakage in videos, and then seek for the protection that the untrusted analyst cannot *distinguish if any sensitive visual element (e.g., an object or human) is included in the video or not*, even if the adversaries have arbitrary background knowledge about the visual elements. Then, we first address the challenge on accurately detecting and tagging all the sensitive visual elements in any video (by utilizing state-of-the-art computer vision techniques [35]). For instance, given a video recorded on the street, our objective is to protect sensitive objects (e.g., vehicles) and/or humans (e.g., pedestrians) which are pre-specified by the video owner.

Given any input video for analysis, different from traditional differentially private mechanisms (e.g., injecting noise into queries or analyses), we propose a novel randomization scheme (via sampling) to generate a utility-driven private video while ensuring the defined differential privacy. Specifically, our VideoDP involves three phases. The first phase randomly samples pixels for the output video based on the visual elements and background scene in the vidoe. Since videos are extremely large scale and highly-dimensional (generally consisting of millions of pixels with very diverse RGBs [36]), it is extremely challenging to ensure good utility for video via pixel sampling (e.g., many RGBs/pixels cannot be sampled). To further improve the output utility, after executing pixel sampling in VideoDP, the second phase generates a (random) *utility-driven private video* by interpolating the RGB values of unsampled pixels and integrates such "estimated pixels" into the missing pixels. Note that the addition of interpolation into the randomizaiton algorithm still ensures the same indistinguishability (regardless of adding or removing any visual element in the input video) since the interpolation can be considered as a post-processing procedure performed on differentially private outputs [37].

In the first two phases, VideoDP generates the (probabilistic) utility-driven

private video with differential privacy. Therefore, in the third phase, different video analyses requested by untrusted data analysts (e.g., queries over the video for analytics) can be flexibly performed over the utility-driven private video, as analyzed in Chapter 3. VideoDP significantly outperforms the PINQ platform [38] in the context of video analytics with reduced perturbation and superior flexibility for different video analyses as validated in the experiments (Chapter 3).

1.2.2 VERRO. Although the VideoDP can make the query result of videos guarantee differential privacy, it cannot directly release the videos for users. To tackle such critical limitation, we define a novel privacy notion for protecting the objects in the video – "ϵ-*Object Indistinguishability*", which is extended from the emerging *differential privacy* in local setting [29–31]. Specifically, in the past decade, the notion of differential privacy has emerged essentially as the de facto privacy standard for bounding the privacy risks while sanitizing different data [13, 18, 22, 39]. Adversaries cannot infer if a certain individual is included in the input or not from the noisy aggregated result (perturbed by a trusted aggregator) regardless of their background knowledge [13]. More recently, local differential privacy (LDP) models have been proposed to privately perturb data by each individual such that the collected (random) data from different individuals can be *indistinguishable*. Inspired by the LDP models, our privacy notion also ensures *indistinguishability* for all the objects (ϵ-*Object Indistinguishability*) in the randomized output video, and thus the perturbed video can be safe to be disclosed to any untrusted video recipient.

With the privacy notion of ϵ-*Object Indistinguishability*, we propose a video sanitization technique that randomly generates a synthetic video by the video owner (i.e., the agency which captures the video) while ensuring ϵ-*Object Indistinguishability* and good utility. More specifically, we design a novel random response scheme (by optimizing the RAPPOR [29]) that randomly generates different objects in the video

by maximizing the utility of random response [29] applied to the objects. As a result, we boost the utility of VERRO in two folds: (1) for each object, optimizing its random response in different frames, and (2) interpolating the trajectories of objects in the video [40] (without additional privacy leakage [37], as analyzed in Chapter 4). Thus, the synthetic video can be disclosed to any untrusted recipient.

1.2.3 L-SRR. To mitigate privacy risks in location data, location anonymization models [41] were first proposed to achieve k-anonymity via location generalization. However, k-anonymity can only provide a weak privacy guarantee (e.g., vulnerable to the background knowledge attacks). To address such limitations, we propose the first strict LDP framework (namely, "L−SRR") to support both location aggregation and individual LBS. First, we design a novel LDP mechanism "*staircase randomized response* (SRR)" and revised the empirical estimation to privately aggregate locations with significantly improved utility and strictly satisfied ϵ-LDP. Second, different from all existing works [42–45], we design and integrate additional components (e.g., private matching [46] and private information retrieval [47]) into L−SRR to ensure ϵ-LDP for a wide variety of LBS Apps such as k nearest neighbors search [48], origin-destination analysis [49] and traffic-aware GPS navigation [5], which may collect user trajectories or perform individual services with the aggregated locations/trajectories.

The utility of L−SRR is significantly enhanced by the proposed new SRR mechanism and estimation method. Specifically, SRR perturbs input locations with *staircase probabilities* for different possible output locations. The probability of perturbing any input x in the domain $\mathcal{D}$ to each possible location $y \in \mathcal{D}$ is optimally pre-computed. Then, users can locally perturb their locations with the optimal probabilities. Different from relaxed privacy notions (e.g., PLDP [45] and geo-indistinguishability [44]), every user is still strictly protected by ϵ-LDP. At the server end (data aggregator), we extend an empirical estimation [50] to further improve the

utility for the SRR mechanism without extra privacy leakage [37].

The remainder of this thesis is organized as follows. Chapter 2 introduces the background knowledge of (local) differential privacy and related works. Chapter 3 firstly overviews the VideoDP framework, describes the privacy model, sampling mechanisms, and then presents the performance evaluation results. Chapter 4 shows the overview of VERRO framework, describes the privacy model, perturbation mechanism, and then presents the performance evaluation results. Chapter 5 shows the overview of L−SRR framework, describes the privacy model, perturbation mechanism, and then presents the performance evaluation results. Chapter 6 concludes the report and proposes future work.

CHAPTER 2
RELATED WORK

2.1 Detect and Blur

In the context of privacy preserving video publishing, many solutions have been proposed in literature (e.g., [12,51–54]). Saini et al. [51] have categorized such works in terms of the sensitive attributes obfuscated in the sanitization. These sensitive attributes include the evidence types *bodies, what*(activity), *where* (location where the video is recorded) and *when* (time when the video is recorded). In general, most of these works employ a *detect* and *blur* policy for only body attributes [12,52–54] and some of them [51, 55–57] aims at preserving the privacy against other three implicit inference channels.

Specifically, these techniques often leverage computer vision techniques [54,58] to first *detect* faces and/or other sensitive regions in the video frames and then *obscure* them. However, such *detect-and-protect* solutions have some limitations. For instance, the *detect-and-protect* techniques cannot formally quantify and bound the privacy leakage. In addition, blurred regions might still be reconstructed by deep learning methods [59, 60]. Last but not least, these techniques often use naive measures for quantifying the privacy loss in videos. For instance, in [54, 61], if faces are present, then it is considered as complete privacy loss, otherwise no privacy loss is reported.

2.2 Privacy Protection for Image/Video with Differential Privacy

Although differential privacy has been widely used to sanitize and release data in statistical databases [13, 62], histograms [16], location data [18], search logs [22], and graphs [23], to the best of our knowledge, there are only several works that have been made to benefit from differential privacy in image or video databases. Fan [25] applied Laplace noise to obfuscate pixels in an image to ensure DP for protecting

specific regions of an image. However, the image quality has been significantly reduced (protection for each single pixel without composition may not work as well [26]). Neither the privacy notion or the Laplace noise (generated with high sensitivity) can be effectively applied to all the pixels for sanitizing full videos. VideoDP can address these limitations with strong privacy protection against arbitrary prior knowledge. Cangialosi [63] provided a new event-duration differential privacy (DP) notion for video analytics queries, which protects all private information visible for less than a particular duration.

2.3 Privacy Protection for LBS

Many privacy-preserving location-based services techniques have been proposed (e.g., [41, 64]). K-anonymity was first defined to protect privacy for LBS. Dummy locations [64] and cloaking region [41] have been utilized for anonymity. However, these methods are highly vulnerable to background knowledge attacks. Another type of techniques design cryptographic protocols [65] to securely perform LBS computations. However, both computational costs and communication overheads might be very high. More recently, rigorous privacy notion differential privacy (DP) has also been applied to LBS Apps [66–69]. For instance, a synthetic data generation technique [66] was proposed to publish statistical information about commuting patterns (including locations) with DP guarantee. Moreover, a quadtree spatial decomposition technique [67] has been used to ensure DP in a database with location pattern mining capabilities. However, the DP model may not be suitable to real LBS apps in case that the users do not trust the server.

The emerging LDP models [29–31] enable private data collection by the untrusted server, which provides stronger protection than the centralized DP models. They have been utilized in a wide variety of applications (e.g., heavy hitters or histogram construction [29, 30], social graphs [24], and frequent itemset mining [70]).

There are two works directly applying randomized response and unary encoding to collect workload-aware indoor positioning data [42] and generate synthetic locations [43] but result in poor utility. Moreover, several relaxed LDP notions have been proposed to protect location privacy [44, 45]. Andrés et al. [44] relaxes the protection for locations within a radius via geo-indistinguishability. Chen et al. [45] relaxes LDP to `PLDP` which allows users to specify personalized privacy budgets for private location collection. However, they cannot ensure rigorous LDP and are also less accurate than our `SRR`.

CHAPTER 3

VIDEODP: A FLEXIBLE PLATFORM FOR VIDEO ANALYTICS WITH DIFFERENTIAL PRIVACY

In this Chapter, I present the framework VideoDP in details, which includes the introduction, preliminaries, privacy model, framework, discussion and experiments [1].

3.1 Introduction

Massive amounts of video data are ubiquitously generated everyday from many different sources such as personal cameras and smart phones, traffic monitoring and video surveillance facilities, and many other video recording devices. Analyzing such complex, unstructured and voluminous data [8] would be extremely beneficial in real world. However, directly releasing videos to the analysts may result in severe privacy concerns due to the considerable amount of sensitive information involved in videos, such as human faces, objects, identities and activities [10]. For instance, traffic monitoring cameras can capture all the vehicles which may involve the make, model and color of vehicles, moving speed and trajectories, and even the drivers' faces. Most of the existing privacy preserving video sanitization techniques (including the YouTube Blurring application [71]) obfuscate the video by *detecting* and then directly *blurring* the region of interests, e.g., faces, vehicle plates, and locations [51,72]. Unfortunately, the *privacy leakage* in the blurred videos cannot be effectively bounded, especially against unknown background knowledge. Specifically, such approaches cannot quantify and bound the privacy leakage in the outputs (e.g., limiting the probability of identifying any individual from the sanitized video [14, 73, 74]). Although all the detected sensitive information can be blurred with fully black/white boxes to address the privacy leakage, the sanitized videos may result in very low utility (see Section 3.7). To address such deficiency, we propose a novel platform (namely, VideoDP)

that ensures differential privacy [14] for any video analysis requested from untrusted data analysts, including queries or query-based analyses over the input video.

3.2 Preliminaries

In this section, we present some preliminaries required for VidepDP. First, Table 3.1 shows the frequently used notations in the following sections.

3.2.1 Video Processing. Referring to the RGB color model [36], video data includes *frame ID, pixel coordinates, red, green, blue* (we focus on visual information in this paper). Thus, we denote any pixel's frame ID as t, its coordinates as (a,b), and its RGB as a 3-dimensional vector $\theta(a, b, t) \in [0, 255]^3$ (*16,581,375 distinct RGBs in the universe*).

VE Detection The state-of-the-art computer vision algorithms can be utilized to accurately detect VEs (e.g., objects [75] and humans [76]) in videos. Specifically, all the VEs in a video (denoted as $\Upsilon_j, j \in [1, n]$) are detected using the tracking algorithm [35] in which the same human/object in different frames is assigned the same unique identifier (see Section 3.7 for details). Notice that, each VE from different angles will be considered as the same VE for protection if they can be tracked in multiple frames by the algorithm (in most cases). If they cannot be tracked in multiple frames, they are also protected separately in VideoDP. In addition, the detection/tracking accuracy can be as high as 90%+ on general videos [77] (which can be further improved by integrating multiple algorithms and repeated detection; and the accuracy is close to 100% in our experiments). These make our defined differential privacy (for VEs) strong enough for protecting the entire video.

Notice that different VEs may have different sizes, and the same VE Υ_j may also have different sizes and different RGB values (e.g., as a vehicle moves close to the camera, its size visually grows). Then, VideoDP aims at protecting all the RGBs

Table 3.1. Frequently Used Notations

Notation	Description
VE	visual element (e.g., object, human)
V, O	orignal video and output synthetic video
$\|V\|, \|O\|$	total pixel counts in V, O
m	the number of distinct RGBs in V
θ_i	the ith RGB in V where $i \in [1, m]$
n	the number of distinct VEs in V
Υ_j	the jth VE in V (all the frames), $j \in [1, n]$
$\|\Upsilon_j\|$	total number of pixels in $\|\Upsilon_j\|$
Ψ_j	set of RGBs in Υ_j with budgets
$\|\Psi_j\|$	cardinality of Ψ_j
d_j	total pixel count in Υ_j
$\widetilde{\theta}_{ij}$	the ith RGB in Ψ_j
k_j	(optimal) number of distinct RGBs in Υ_j
Ψ, $\|\Psi\|$	$\bigcup_{j=1}^{n} \Psi_j$, cardinality of Ψ
$\widetilde{\theta}_i, \theta_i$	the ith RGB in Ψ, the ith RGB in V
$\widetilde{c}_i$ (or c_i), $\widetilde{c}_i^j$	total pixel count for RGB $\widetilde{\theta}_i$ (or θ_i) in V, Υ_j
$\widetilde{x}_i$ (or x_i)	total pixel count for RGB $\widetilde{\theta}_i$ (or θ_i) in O
(a, b, t)	the pixel with coordinates (a, b) and frame t
$\theta(a, b, t)$	the RGB of pixel (a, b, t) in V
$\hat{\theta}(a, b, t)$	the RGB of pixel (a, b, t) in O
$Pr(a, b, t)$	probability that pixel (a, b, t) is sampled
$\sigma_0, \ldots, \sigma_4$	probabilities that pixel (a, b, t) has $0, 1, \ldots, 4$
	neighboring pixels after Phase I (sampling)
$\hat{\theta}_N$	simplified notation for $\hat{\theta}(a-1, b, t)$
$\hat{\theta}_S$	simplified notation for $\hat{\theta}(a+1, b, t)$
$\hat{\theta}_W$	simplified notation for $\hat{\theta}(a, b-1, t)$
$\hat{\theta}_E$	simplified notation for $\hat{\theta}(a, b+1, t)$

of different VEs in all the frames. To break down the video into pixels with RGBs, we denote the set of distinct RGBs in VE Υ_j (in all the frames) as Ψ_j where the cardinality is written as $|\Psi_j|$ (the number of distinct RGBs in Υ_j).

3.2.2 Privacy Model. To protect sensitive VEs in the video, we first consider two input videos V and V' that differ in any visual element Υ (in all the frames) as two neighboring inputs. Specifically, given a video V, after completely removing Υ in all the frames of V, we can obtain V' (or vice-versa). Note that V and V' have identical number of frames and background scene. Then, VideoDP ensures that adding any VE into any number of frames in a video or completely removing any VE from the video would not result in significant privacy risks in video analytics, assuming that the adversary possesses arbitrary background knowledge on all the VEs. W.l.o.g., denoting $V = V' \cup \Upsilon$, we have:

Definition 1 (ϵ-Differential Privacy)**.** *A randomization algorithm $\mathcal{A}$ satisfies ϵ-differential privacy if for any two input videos V and V' that differ in any visual element (e.g., object or human) Υ, and for any output $O \in range(\mathcal{A})$, we have $e^{-\epsilon} \leq \frac{Pr[\mathcal{A}(V)=O]}{Pr[\mathcal{A}(V')=O]} \leq e^{\epsilon}$.*

Definition 1 protects all the sensitive VEs in the video (which are pre-defined and accurately detected by the video owner, as discussed in Section 3.2.1). If necessary, any part of the video can be specified as a sensitive VE for protection (including the background scene), as discussed in Section 3.6 (see the "Background Scene(s) as VE" mode in VideoDP).

Moreover, given two neighboring videos V and V', a possible output $O \in range(\mathcal{A})$ may make any of $Pr[\mathcal{A}(V) = O]$ and $Pr[\mathcal{A}(V') = O]$ equal to 0. For instance, in case that the extra VE Υ is included in V but not in V', an output O

involving Υ cannot be generated from V' (simply due to $\Upsilon \cap V' = \emptyset$). At this time, for such output O, we have $Pr[\mathcal{A}(V) = O] > 0$ while $Pr[\mathcal{A}(V') = O] = 0$. In such cases, the multiplicative difference between $\frac{Pr[\mathcal{A}(V)=O]}{Pr[\mathcal{A}(V')=O]}$ and $\frac{Pr[\mathcal{A}(V')=O]}{Pr[\mathcal{A}(V)=O]}$ cannot be bounded by e^{ϵ} (due to the zero denominator). Thus, a relaxed privacy notion [32, 78] can be defined:

Definition 2 ((ϵ, δ)-Differential Privacy [32, 78])**.** *A randomization algorithm $\mathcal{A}$ satisfies (ϵ, δ)-differential privacy if for all video V, we can divide the output space range($\mathcal{A}$) into two sets Ω_1, Ω_2 such that (1) $Pr[\mathcal{A}(V) \in \Omega_1] \leq \delta$, and (2) for any of V's neighboring video V' and for all $O \in \Omega_2$: (2) $e^{-\epsilon} \leq \frac{Pr[\mathcal{A}(V)=O]}{Pr[\mathcal{A}(V')=O]} \leq e^{\epsilon}$.*

This definition guarantees that algorithm $\mathcal{A}$ achieves ϵ-DP with a high probability ($\geq 1 - \delta$) [32, 78]. The probability that generating the output with unbounded multiplicative difference for V and V' is bounded by δ.

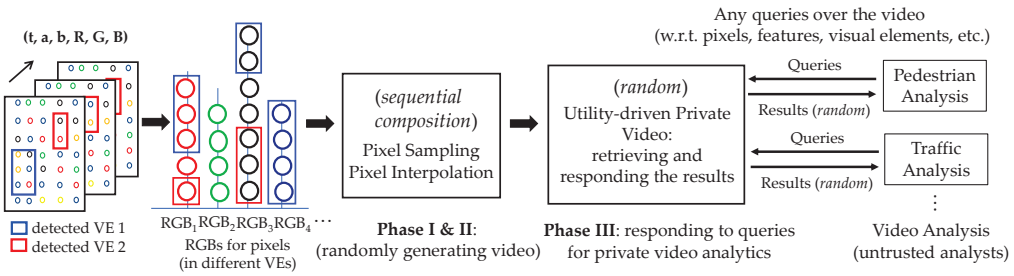


Figure 3.1. VideoDP Framework: ϵ-differential privacy for Phase I–III (which can be relaxed to (ϵ, δ)-differential privacy)

3.2.3 VideoDP Framework.

Limitations of PINQ-based Video Analytics: Privacy Integrated Queries (PINQ) [38] platform was proposed to facilitate data analytics by injecting Laplace noise into the queries. Similarly, PINQ can be simply extended to function video analytics. However, there are two major limitations of PINQ-based video analytics,

which greatly limit the usability in practice.

Sensitivity: In PINQ-based video analytics, global sensitivity [14] can be defined for some queries with small sensitivities such as "the count of vehicles in the video" (sensitivity as 1). However, for queries with large sensitivities, the query result would be overly obfuscated (see Section 3.7). For instance, in the query "the average time each object stays in the video", since an object can stay in the video for the entire video or only 1 second (a few frames), global sensitivity would be too large and difficult to define. Meanwhile, it might be also impractical to achieve (smooth) local sensitivity [79] for all different queries in the analysis due to computational overheads.

Flexibility: PINQ is inflexibly adapted for different video analyses. For each requested analysis, a specific DP scheme would be required for improving the utility of the private analysis. The algorithm (e.g., budget allocation, composition of queries [38]) has to be redesigned for any new analysis on the video.

Instead, we propose a novel flexible framework VideoDP for universally optimizing the utility of different video analysis, detailed as follows. Figure 5.1 shows that VideoDP consists of three major phases (after detecting all the sensitive VEs):

1. **Phase I**: video (including detected VEs) can be represented as pixels, which can be grouped by their RGBs (notice that, different from generating RGB histograms, each pixel still keeps its original coordinates and frame ID). Rather than injecting Laplace noise, this phase randomly samples a subset of pixels (with its original features) for each RGB, where privacy budgets are allocated for different RGBs (*sequential composition* [38]) to optimize the output utility. Phase I in VideoDP satisfies ϵ-DP, which can be relaxed to (ϵ, δ)-DP. See details in Section 3.3.

2. **Phase II**: after sampling all the pixels, the output video has numerous unsam-

pled pixels (due to privacy constraints). This phase estimates the RGBs for unsampled pixels via interpolation. We show that Phase II does not leak any additional information (still ensuring *indistinguishability*). Thus, Phase II can boost the video utility without additional privacy loss. See details in Section 3.4.

3. **Phase III**: VideoDP applies the requested queries (for video analysis, e.g., traffic and pedestrian analysis [80, 81]) to the *random* utility-driven private video and directly returns the results (which are also random) to untrusted analysts, where differential privacy is also guaranteed (as analyzed in Section 3.5).

3.3 Phase I: Mechanism of VideoDP

In this section, we present the sampling algorithms while ensuring differential privacy.

3.3.1 Pixel Sampling Mechanism. Recall that Section 3.2.3 has briefly discussed the pixel sampling. Since each sensitive VE involves a set of RGBs and the pixel sampling for distinct RGBs (in all the VEs) is expected to satisfy differential privacy, the privacy budget ϵ will be allocated for the individual pixel sampling w.r.t. distinct RGBs (which follows sequential composition [38]). Specifically, for each RGB θ_i in V, a number of x_i pixels with such RGB (out of the original c_i pixels in V) will be randomly selected (uniform distribution) to output with their original coordinates and frame ID. A privacy budget ϵ_i will be allocated for sampling pixels for RGB θ_i that is used to bound the probabilities for its differential privacy guarantee.

Since every video may involve millions of distinct RGBs, given a privacy budget ϵ for pixel sampling, it is nearly impossible to allocate an equal budget to every unique RGB (each share would be negligible). To address such challenge, we categorize all

the RGBs $i \in [1, m], \theta_i$ for pixel sampling in different cases (some of which indeed do not consume any privacy budget) and explore the optimal budget allocation as well as the differential privacy guarantee in Section 3.3.2.

3.3.2 Privacy Budget Allocation. As the privacy budget ϵ is specified for pixel sampling, our goal is to optimize the allocated budgets for RGBs towards their count distributions in the original video. Given V and V' where $V = V' \cup \Upsilon$ (w.l.o.g.) and Υ can be any VE, we have three types of RGBs:

- Case (1): RGB $\theta_i \in \Upsilon \setminus V'$ (the RGB is included in the extra visual element Υ but not V').
- Case (2): RGB $\theta_i \in V' \setminus \Upsilon$ (the RGB is included in V' but not the extra visual element Υ).
- Case (3): RGB $\theta_i \in V' \cap \Upsilon$ (the RGB is included in both V' and the extra visual element Υ).

Then, we investigate the budget and the privacy guarantee for these three cases as below.

Case (1): RGB $\theta_i \in \Upsilon \setminus V'$. Pixels in this case is the reason why we need the relax in definition, which we will discuss this in the Section 3.6. Given x_i as the output count of θ_i and c_i is the input count in V, we let $x_i = 0$ (does not output pixels with such RGB θ_i) since θ_i cannot be found in V', if generating any pixel with RGB θ_i into the output video O (in Phase I). Extending it to an randomization algorithm $\mathcal{A}$ applied to V (with n VEs $\Upsilon_1, \ldots, \Upsilon_n$), w.l.o.g., considering V as the video with an arbitrary extra VE $\Upsilon \in \{\Upsilon_1, \ldots, \Upsilon_n\}$ (compared to V'), we thus have: $\forall j \in [1, n], \Upsilon_j$, *if RGB* $\theta_i \in \Upsilon_j \setminus (V - \Upsilon_j)$, *then* $x_i = 0$ *(do not sample pixels with such RGB).*

Case (2): RGB $\theta_i \in V' \setminus \Upsilon$. Since all the pixels with such RGB θ_i in V and V' are equivalent, we can let $x_i = c_i$ (retaining all the pixels with such RGB θ_i) without violating privacy. Then, for any $x_i > 0$ (can be maximized to c_i), sampling pixels for this RGB θ_i does not consume any privacy budget.

Similarly, extending it to the randomization algorithm $\mathcal{A}$ (applied to V), w.l.o.g., considering V as the video with an arbitrary extra VE $\Upsilon \in \{\Upsilon_1, \ldots, \Upsilon_n\}$ (compared to V'), since VideoDP should protect any arbitrary VE, we thus have: $\forall j \in [1, n], \Upsilon_j$, *if any RGB* $\theta_i \in V' \setminus \Upsilon_j$, *then* $x_i = c_i$ *(retaining all the pixels with such RGB in the utility-driven private video). This does not consume any privacy budget since such RGBs do not exist in any of the VEs.*

Case (3): RGB $\theta_i \in V' \cap \Upsilon$. The pixel sampling for each RGB in this case should satisfy ϵ-differential privacy where $e^{-\epsilon} \leq \frac{Pr[\mathcal{A}(V)=O]}{Pr[\mathcal{A}(V')=O]} \leq e^{\epsilon}$ holds. Thus, we should allocate privacy budgets for different RGBs in this case. However, due to the *sequential composition* [38], we cannot allocate a budget for every RGB in this category (otherwise, given any ϵ, for a large number of distinct RGBs, each share of the budget would be too extremely small). In other words, all the RGBs in this category may have to be suppressed (not sampled in the output video). To improve the output utility, our VideoDP has the following three procedures for budget allocation in pixel sampling (Phase I):

1. Determine the RGBs selection rule (*selecting the most representative RGBs in each VE for generating the utility-driven private video*).
2. Derive an optimal number of distinct RGBs within each VE (*maximizing the utility of the VEs in the utility-driven private video*).
3. Allocate appropriate budgets for selected RGBs (*per their RGB count distribution in the original video*).

1) RGBs Selection Rule. Denoting the number of distinct RGBs in $\Upsilon_j, j \in [1, n]$ (which receive privacy budgets to output after Phase I) as k_j, the remaining RGBs in Υ_j will be suppressed (not sampled) during pixel sampling. Thus, this procedure ensures that the selected k_j RGBs in Υ_j are most representative to reconstruct the object (without compromising privacy).

An intuitive rule is to select the top frequent k_j RGBs in Υ_j. However, it might be biased to specific regions with intensive counts of similar RGBs in a VE. To address such limitation, we adopt the multi-scale analysis [82] in computer vision to partition each VE Υ_j into k_j cells and select the top frequent RGB in each cell to allocate privacy budgets (as the "representative RGBs"). Then, the sampled RGBs can be effective to reconstruct the VE in the utility-driven private video.

2) Optimal k_j in Each VE. This procedure is designed to maximize the utility of the VEs in the utility-driven private video (after bilinear interpolation [83] in Phase II). If the number of distinct RGBs in Υ_j that receive privacy budgets k_j is large, more distinct RGBs can be sampled in the VE but the budget allocated for each RGB would be extremely small; if k_j is small, the budget allocated for each RGB would be large but less distinct RGBs can be sampled. We now seek for the optimal k_j for Υ_j that can minimize the MSE between the interpolated VE (after Phase II) and the original VE.

Specifically, since every pixel in Υ_j can be sampled (with the original RGB) or unsampled (with an estimated RGB), we minimize the expectation of MSE (referring to Equation 3.2) after the Phase II bilinear interpolation [83]. The expectation of each pixel's RGB is determined by the probabilities of "sampled" (denoted as $Pr(a, b, t)$) and "unsampled but interpolated by its neighboring pixels" (4 neighbors for non-border pixels, 3 neighbors for border-but-not-corner pixels, and 2 neighbors for corner pixels, as shown in Figure 3.4). Denoting pixel (a, b, t)'s RGB in the output as

$\hat{\theta}(a,b,t)$, for simplicity of notations, we denote the RGBs of its neighboring pixels as $\hat{\theta}_N$, $\hat{\theta}_S$, $\hat{\theta}_W$ and $\hat{\theta}_E$, for pixels $(a-1,b,t)$, $(a+1,b,t)$, $(a,b-1,t)$ and $(a,b+1,t)$, respectively. For a non-border pixel (4 neighbors), the expectation of its RGB[1] can be derived as:

$$
\begin{aligned}
&E[\hat{\theta}(a,b,t)] = Pr(a,b,t) * \theta(a,b,t) + \sigma_0(a,b,t) * 0 \\
&+\frac{\sigma_1(a,b,t)[1-Pr(a,b,t)][E(\hat{\theta}_N)+E(\hat{\theta}_S)+E(\hat{\theta}_W)+E(\hat{\theta}_E)]}{4} \\
&+\frac{\sigma_2(a,b,t)[1-Pr(a,b,t)][3E(\hat{\theta}_N)+3E(\hat{\theta}_S)+3E(\hat{\theta}_W)+3E(\hat{\theta}_E)]]}{6*2} \\
&+\frac{\sigma_3(a,b,t)[1-Pr(a,b,t)][3E(\hat{\theta}_N)+3E(\hat{\theta}_S)+3E(\hat{\theta}_W)+3E(\hat{\theta}_E)]}{4*3} \\
&+\frac{\sigma_4(a,b,t)[1-Pr(a,b,t)][E(\hat{\theta}_N)+E(\hat{\theta}_S)+E(\hat{\theta}_W)+E(\hat{\theta}_E)]}{4}
\end{aligned}
\tag{3.1}
$$

where $\theta(a,b,t)$ is the original RGB (a constant) and probability of "sampled" $Pr(a,b,t)$ is determined by k_j (given V and k_j, it is deterministic if the RGB selection rule is decided previously). Probabilities $\sigma_0(a,b,t), \sigma_1(a,b,t), \sigma_2(a,b,t), \sigma_3(a,b,t)$ and $\sigma_4(a,b,t)$ are probabilities that pixel (a,b,t) has 0 neighbor, 1 neighbor, 2 neighbors, 3 neighbors and 4 neighbors after sampling (which are also constants if V, k_j and sampling mechanism are determined; note that $\sigma_0(a,b,t) + \cdots + \sigma_4(a,b,t) = 1$). In the equation, $E[\hat{\theta}_N]$, $E[\hat{\theta}_S]$, $E[\hat{\theta}_W]$ and $E[\hat{\theta}_E]$ are the RGB expectation of its four neighbors in the same tth frame (*Equation 3.1 presents the relation among the RGB expectations of the five pixels*, which are detailed in Appendix A.1.1). Similarly, we can obtain two other equations for pixels with special coordinates (border-but-not-corner or corner pixels of the frame, please see Equation A.1 and A.2 in Appendix A.1.1).

[1]Although the RGBs of all the pixels in Υ_j are random (due to the sampling in Phase I), the expectations of RGBs for its neighboring pixels in Υ_j always satisfy a condition (ensured by bilinear interpolation [83]), e.g., Equation 3.1

Thus, for each pixel in VE Υ_j (in all the frames), there exists exactly one equation out of three cases in Equation 3.1, A.1 and A.2 (latter two are in Appendix A.1.1). As k_j is determined, $\theta(a,b,t)$ and $Pr(a,b,t)$ are constants, then we can solve all the equations to obtain $\forall(a,b,t) \in \Upsilon_j, E[\hat{\theta}(a,b,t)]$. Thus, each k_j value corresponds to the solved $\forall(a,b,t) \in \Upsilon_j, E[\hat{\theta}(a,b,t)]$, and then we can efficiently derive the optimal k_j for Υ_j as:

$$\arg\min_{k_j} \frac{1}{|\Upsilon_j|} \sum_{\forall(a,b,t)\in\Upsilon_j} \left(E[\theta(a,b,t)] - E[\hat{\theta}(a,b,t)]\right)^2 \tag{3.2}$$

where $|\Upsilon_j|$ denotes the total number of pixels in Υ_j. Solving the above problem requires complexity $O(n^3 \log(n))$, which is much faster than executing pixel sampling for all the possible k_j and then comparing all the MSE results to get the optimal k_j (*since iteratively sampling all the pixels is expensive*). Details of the solver is given in Appendix A.1.2. Notice that,

- **Range for k_j.** The optimal k_j is derived from a specified range of k_j. It is unnecessary to traverse k_j to a extremely large number (otherwise, the allocated budget for each RGB would be extremely small). The larger k_j, more diverse RGBs can be allocated with a privacy budget; the smaller k_j, each RGB will be allocated with a larger privacy budget. Thus, the lower/upper bounds for k_j can be selected according the requested diversity of RGBs in the visual elements in practice ($k_j \leq 20$ can give good utility in our experiments).

- **Approximation.** As discussed before, since Υ_j in different frames may have different sizes and different sets of RGBs (though the difference can be minor), the most accurate k_j can be obtained by solving the equations for all the pixels of Υ_j in all the frames (*with complexity* $O(n^3 \log(n))$, as proven in Appendix A.1). If more efficient solvers are desirable, we can randomly select a frame (including

Υ_j) to solve the equations to obtain an approximated k_j for Υ_j by assuming the VE does not change much in the video. Another alternative solution is to solve the optimal k_j for each frame and average them (which is more efficient but less accurate).

Thus, we can repeat the above procedure for all the VEs such that the optimal $k_j, j \in [1, n]$ can be obtained to minimize the MSE of the VEs in the output video.

3) Budget Allocation. As the optimal k_j for each visual element $\Upsilon_j, j \in [1, n]$ is derived, we denote the set of RGBs in $\Upsilon_j, j \in [1, n]$ to allocate budgets as Ψ_j with the cardinality $|\Psi_j| = k_j$. Then, we have the total number of RGBs to sample in V (Case (3)) as the cardinality $|\Psi|$ of the union $\Psi = \bigcup_{j=1}^{n} \Psi_j$. We then present how to allocate privacy budget ϵ in Phase I for $|\Psi|$ different RGBs. The criterion for allocating budget is to allocate the privacy budgets based on the count distributions of RGBs in different VEs while fully utilizing the privacy budget ϵ. For each VE Υ_j, all the RGBs in Ψ_j can fully enjoy the budget ϵ (since Ψ_j includes all the RGBs that could generate visual element Υ_j in all the frames, and other RGBs would not be sampled into the visual element Υ_j).[2]

Then, we denote the ith RGB in Ψ_j as $\widetilde{\theta}_{ij}$ where $i \in [1, k_j]$, and the count of $\widetilde{\theta}_{ij}$ in Υ_j as $d_j(\widetilde{\theta}_{ij})$ and the overall pixel count in Υ_j (in all frames) as d_j. Apparently, we can allocate $\frac{d_j(\widetilde{\theta}_{ij})\epsilon}{d_j}$ to RGB $\widetilde{\theta}_j(i), i \in [1, k_j]$ and apply this criterion to all the VEs. However, if any RGB $\widetilde{\theta}$ is included in multiple VEs (the intersections among the sets $\forall j \in [1, n], \Psi_j$), $\widetilde{\theta}$ will receive privacy budgets from different VEs (and should satisfy differential privacy for all of them). At this time, its budget should be allocated as

[2]Any two VEs do not share pixels in the video since the front VE blocks a part of the back VE if they overlap in any frame. In such complex scenario, both VEs can be accurately detected in our experiments. The front VE includes all the pixels while the back VE will be all the pixels that cameras can capture.

the *minimum* one out of all (otherwise, not all the VEs in pixel sampling can be protected with ϵ-differential privacy since the budget for some VEs may exceed ϵ).

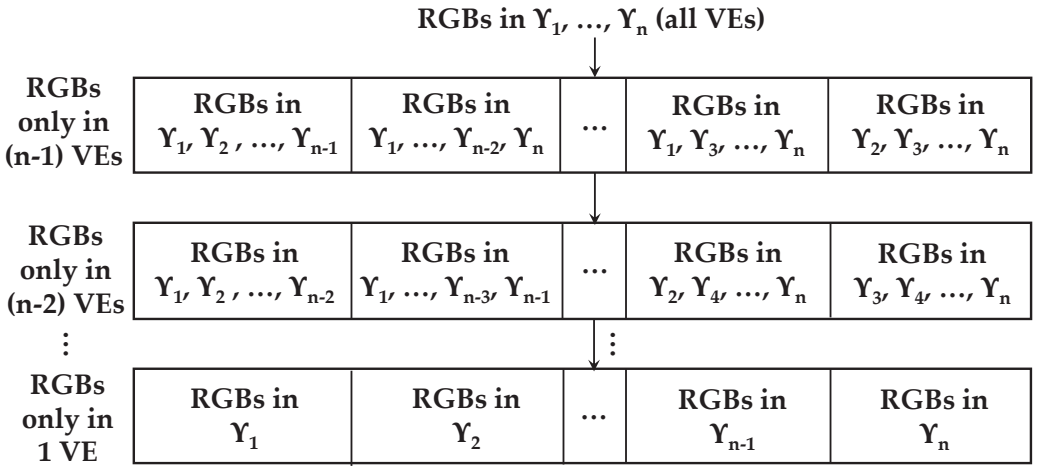


Figure 3.2. Prioritizing RGBs (for allocating budgets)

Nevertheless, if the minimum budget is adopted as above, some VEs cannot fully enjoy ϵ (the gap between $\widetilde{\theta}$'s original budget in a specific VE and its minimum budget among all the VEs would be wasted). To fully utilize the privacy budgets, we propose a *budget allocation algorithm* for all the $|\Psi|$ distinct RGBs by *prioritizing* them in the RGB set $\Psi = \bigcup_{j=1}^{n} \Psi_j$.

Specifically, we prioritize $|\Psi|$ different RGBs into n disjoint partitions: as shown in Figure 3.2 (from top to down), RGBs in the first partition are included in all the VEs, RGBs in the second partition are included in $(n-1)$ VEs, ..., RGBs in the nth partition are only included in a single VE. Then, our algorithm iteratively allocates budgets for RGBs in n partitions (*allocating budgets for all the RGBs in a partition in each iteration*).

Since all the RGBs within each VE follow *sequential composition* [38], after allocating the budgets for all the RGBs in the ℓth partition, the allocation in the $(\ell+1)$th partition will be based on the remaining budget for every VE. In the ℓth iteration (for the ℓth partition), the budget for each RGB $\widetilde{\theta}$ is allocated based on its

count distribution out of the remaining RGBs in each of the $(n-\ell+1)$ VEs (which include $\widetilde{\theta}$). Then, the *minimum* budget derived from all the VEs is allocated to $\widetilde{\theta}$.

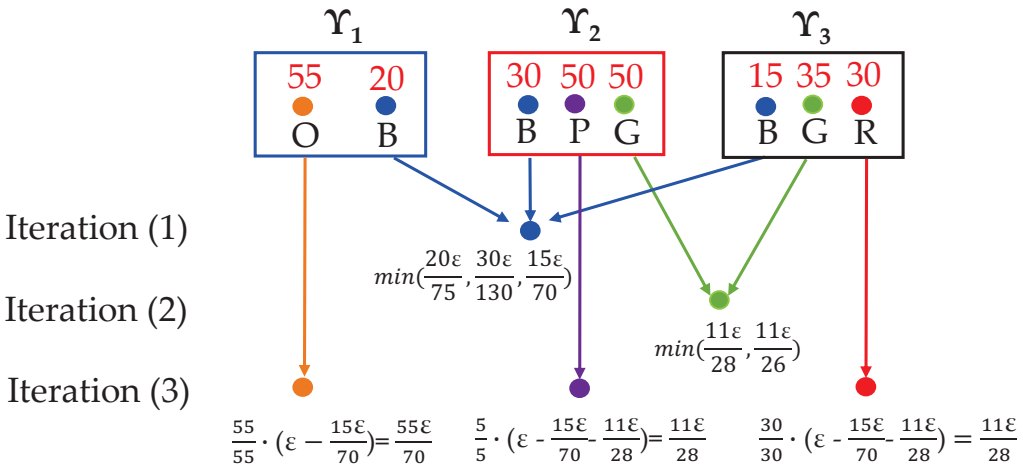


Figure 3.3. Example of Budget Allocation

Example 1. *Figure 3.3 shows three VEs (Υ_1, Υ_2 and Υ_3). Blue exists in all the VEs $\Upsilon_1, \Upsilon_2, \Upsilon_3$ with counts $20, 30, 15$. Green exists in Υ_2 and Υ_3 with counts 50 and 35. All the remaining RGBs only exist in only one VE (and the non-VE part of the video): Orange in Υ_1 with count 55, Purple in Υ_2 with count 5, and Red in Υ_3 with count 30. Thus, five RGBs are prioritized to (three partitions):{B}, {G}, and {O, P, R}.*

In the 1st iteration (partition), Blue is first allocated with a privacy budget as the $\min\{\frac{20\epsilon}{75}, \frac{30\epsilon}{130}, \frac{15\epsilon}{70}\}$ (the minimum budget from three different VEs). The remaining budget for all the VEs is $\frac{55\epsilon}{70}$. In the 2nd iteration, Green is allocated with a privacy budget $\frac{55\epsilon}{70} \cdot \min\{\frac{50}{100}, \frac{35}{65}\} = \min\{\frac{11\epsilon}{28}, \frac{11\epsilon}{26}\}$. In the 3rd iteration, Orange is allocated with budget $\frac{55}{55} \cdot (\epsilon - \frac{15\epsilon}{70}) = \frac{55\epsilon}{70}$, Purple is allocated with budget $\frac{5}{5} \cdot (\epsilon - \frac{15\epsilon}{70} - \frac{11\epsilon}{28}) = \frac{11\epsilon}{28}$, and Red is allocated with budget $\frac{30}{30} \cdot (\epsilon - \frac{15\epsilon}{70} - \frac{11\epsilon}{28}) = \frac{11\epsilon}{28}$.

Since almost all the VEs have RGBs in the last partition (every VE in real videos include numerous RGBs that are not included in other VEs), the budget can

be fully allocated for all the RGBs. Thus, *the budget sum of all the RGBs* in any VE equals ϵ, and the size of VEs does not result in additional leakage. Algorithm 6 in Appendix A.2 presents the details of budget allocation.

3.3.3 Pixel Sampling Algorithm. To illustrate the algorithm for Phase I, we again discuss the pixel sampling for three different cases of RGBs.

Recall that in Case (1), for all the RGBs $\theta_i \in \Upsilon \setminus V'$, all the pixels with such RGBs will not be sampled (ensuring that $\delta = 0$). In Case (2), for all the RGBs $\theta_i \in V' \setminus \Upsilon$, all the pixels with such RGBs will be sampled (with the original coordinates and frame). Sampling pixels for all the RGBs in Case (2) satisfy 0-DP.

In Case (3), for all the RGBs $\theta_i \in V' \cap \Upsilon$, as discussed in Section 3.3.2, we sample pixels for $|\Psi|$ distinct RGBs where $|\Psi| \leq \sum_{j=1}^{n} k_j$ (since different VEs may have common RGBs). We denote the set $\Psi = \bigcup_{j=1}^{n} \Psi_j = \{\widetilde{\theta}_1, \ldots, \widetilde{\theta}_{|\Psi|}\}$ (the set of RGBs which request privacy budgets), and its set of budgets $\{\epsilon(\widetilde{\theta}_1), \ldots, \epsilon(\widetilde{\theta}_{|\Psi|})\}$. It is straightforward to show the *sequential composition* [38] of allocated privacy budgets (by Algorithm 6) for all the RGBs:

$$\sum_{\forall \widetilde{\theta}_i \in \Psi_j} \epsilon(\widetilde{\theta}_i) = \epsilon \tag{3.3}$$

where $\widetilde{\theta}_i$ is denoted as the ith RGB in Ψ. Then, for any V and V' differing in an arbitrary VE $\Upsilon_j, j \in [1, n]$,

$$\forall \widetilde{\theta}_i \in \Psi_j, e^{-\epsilon(\widetilde{\theta}_i)} \leq \frac{Pr[\mathcal{A}(V(\widetilde{\theta}_i)) = O(\widetilde{\theta}_i)]}{Pr[\mathcal{A}(V'(\widetilde{\theta}_i)) = O(\widetilde{\theta}_i)]} \leq e^{\epsilon(\widetilde{\theta}_i)} \tag{3.4}$$

where $V(\widetilde{\theta}_i)$ and $V'(\widetilde{\theta}_i)$ are the pixels with RGB $\widetilde{\theta}_i$ in V and V'. Deriving the probability for randomly picking $\widetilde{x}_i$ out of $\widetilde{c}_i$ pixels with RGB $\widetilde{\theta}_i$ (pixel sampling using

input V and V', differing in Υ_j), we have:

$$\forall i \in [1, |\Psi|],\ Pr[\mathcal{A}(V(\widetilde{\theta}_i)) = O(\widetilde{\theta}_i)] = 1/\binom{\widetilde{c}_i}{\widetilde{x}_i}$$
$$Pr[\mathcal{A}(V'(\widetilde{\theta}_i)) = O(\widetilde{\theta}_i)] = 1/\binom{\widetilde{c}_i - \widetilde{c}_i^j}{\widetilde{x}_i}$$

$$\Longrightarrow e^{-\epsilon(\widetilde{\theta}_i)} \leq \binom{\widetilde{c}_i}{\widetilde{x}_i} \Big/ \binom{\widetilde{c}_i - \widetilde{c}_i^j}{\widetilde{x}_i} \leq e^{\epsilon(\widetilde{\theta}_i)} \tag{3.5}$$

where $\widetilde{c}_i$ and $\widetilde{x}_i$ are the input and output counts of RGB $\widetilde{\theta}_i$ while $\widetilde{c}_i^j$ denotes the count of $\widetilde{\theta}_i$ in VE Υ_j.

Thus, we can derive a maximum output count for sampling pixels for each RGB $\widetilde{\theta}_i, i \in [1, |\Psi|]$ and the maximum $\widetilde{x}_i$ can be efficiently computed as below (the only variable): $\forall i \in [1, |\Psi|]$,

$$\max\{\widetilde{x}_i | \forall j \in [1, n], \binom{\widetilde{c}_i}{\widetilde{x}_i} \Big/ \binom{\widetilde{c}_i - \widetilde{c}_i^j}{\widetilde{x}_i} \leq e^{\epsilon(\widetilde{\theta}_i)}\} \tag{3.6}$$

The maximum output count of the ith RGB $\widetilde{x}_i, i \in [1, |\Psi|]$ can be efficiently computed from Equation 3.6 (e.g., via binary search) since the left-side of the inequality is monotonic on $\widetilde{x}_i$. To sum up, Algorihtm 1 presents the details of Phase I.

Theorem 1. *The pixels sampling in VideoDP (Phase I) satisfies ϵ-differential privacy.*

Proof. We can prove the differential privacy guarantee for three cases of pixel sampling in the algorithm.

Algorithm 1 Pixel Sampling (ϵ-DP)

Input: input video V, privacy budget ϵ

Output: sampled video O (pixels)

1: detect all the visual elements $(\Upsilon_1, \ldots, \Upsilon_n)$ in V
2: **for** each $\Upsilon_j, j \in [1, n]$ **do**
3: **for** each RGB $\theta_i \in \Upsilon_j$ but $\notin (V \setminus \Upsilon_j)$ **do**
4: suppress all c_i pixels with RGB θ_i in V
5: **end for**
6: **end for**
7: **for** each RGB $\theta_i \in V \setminus \bigcup_{j=1}^{n} \Upsilon_j$ **do**
8: output all c_i pixels with RGB θ_i in V (original coordinates and frame)
9: **end for**
10: **for** each $\Upsilon_j, j \in [1, n]$ **do**
11: compute the optimal number of distinct RGBs to sample in Υ_j (minimum expectation of MSE): k_j
12: execute Algorithm 6 (in Appendix A.2) to allocate budgets for all the RGBs in $\Psi = \{\widetilde{\theta}_1, \ldots, \widetilde{\theta}_{|\Psi|}\}$
13: **end for**
14: **for** each $\widetilde{\theta}_i, i \in [1, |\Psi|]$ **do**
15: compute the maximum $\widetilde{x}_i$: $\max\{\widetilde{x}_i | \forall j \in [1, n], \binom{\widetilde{c}_i}{\widetilde{x}_i} \big/ \binom{\widetilde{c}_i - \widetilde{c}_i^j}{\widetilde{x}_i} \leq e^{\epsilon(\widetilde{\theta}_i)}\}$
16: randomly pick $\widetilde{x}_i$ pixels with RGB $\widetilde{\theta}_i$ in V to output (original coordinates and frame)
17: **end for**

In Case (1), since all the pixels with such RGBs are suppressed, $\delta = 0$ always holds with Line 2-4 in Algorithm 1. In Case (2), since $\forall \theta_i, \frac{Pr[\mathcal{A}(V(\theta_i))=O(\theta_i)]}{Pr[\mathcal{A}(V'(\theta_i))=O(\theta_i)]}$ always equals 1, Line 5-6 in Algorithm 1 does not result in privacy loss. In Line 7-12 of the algorithm (Case (3)), we have $\forall i \in [1, |\Psi|]$, $e^{-\epsilon(\widetilde{\theta}_i)} \leq \frac{Pr[\mathcal{A}(V(\widetilde{\theta}_i))=O(\widetilde{\theta}_i)]}{Pr[\mathcal{A}(V'(\widetilde{\theta}_i))=O(\widetilde{\theta}_i)]} \leq e^{\epsilon(\widetilde{\theta}_i)}$ holds.

Per the *sequential composition* of differential privacy [38], for all V and V' differing in any VE $\Upsilon_j, j \in [1, n]$, we have:

$$\prod_{\forall \widetilde{\theta}_i \in \Psi_j} \frac{Pr[\mathcal{A}(V(\widetilde{\theta}_i)) = O(\widetilde{\theta}_i)]}{Pr[\mathcal{A}(V'(\widetilde{\theta}_i)) = O(\widetilde{\theta}_i)]} \leq exp[\sum_{\forall \widetilde{\theta}_i \in \Psi_j} \epsilon(\widetilde{\theta}_i)]$$

$$\prod_{\forall \widetilde{\theta}_i \in \Psi_j} \frac{Pr[\mathcal{A}(V(\widetilde{\theta}_i)) = O(\widetilde{\theta}_i)]}{Pr[\mathcal{A}(V'(\widetilde{\theta}_i)) = O(\widetilde{\theta}_i)]} \geq exp[-\sum_{\forall \widetilde{\theta}_i \in \Psi_j} \epsilon(\widetilde{\theta}_i)]$$

$$\implies e^{-\epsilon} \leq \frac{Pr[\mathcal{A}(V) = O]}{Pr[\mathcal{A}(V') = O]} \leq e^{\epsilon} \tag{3.7}$$

Thus, this completes the proof. □

Note that composing the sampled pixels would not result in additional leakage. First, composing pixels with the same RGB is done within each individual sampling (that satisfies differential privacy with the allocated budget for the RGB). Second, composing pixels with different RGBs follows sequential composition. Thus, the sum of the allocated budgets would be the privacy bound (total leakage), and there is no additional leakage. Furthermore, in case of $V' = V \cup \Upsilon$, adding an arbitrary VE Υ to V to generate V'. Similarly, for all $\widetilde{\theta}_i$, $\widetilde{x}_i$ can also be derived from V' and V to ensure differential privacy for pixel sampling.

3.4 Phase II: Video Generation

After sampling pixels in Phase I, the suppressed pixels in Case (1) and unsampled pixels in Case (3) do not have any RGB (see Figure 3.4). Then, Phase II generates the utility-driven private video by estimating the RGBs for the missing pixels, and Phase III responds to the queries (over the private video) for video analysis.

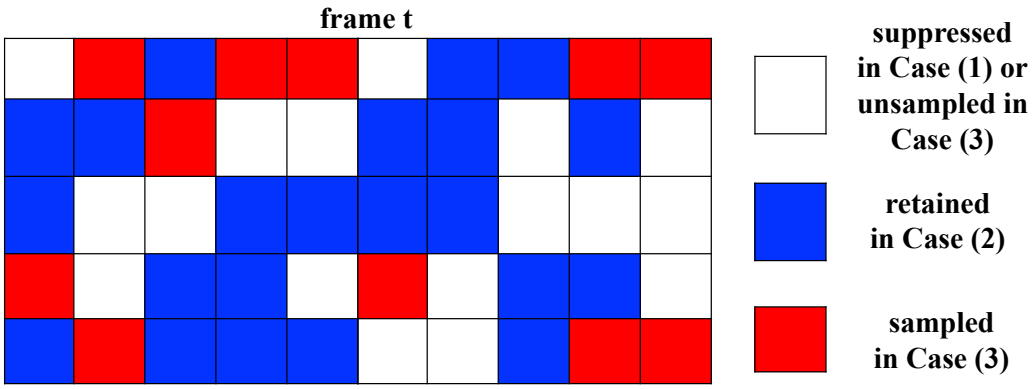


Figure 3.4. Pixels after Sampling (Phase I)

For all the coordinates with a RGB value after sampling, the RGBs of such pixels can be estimated using bilinear interpolation [83]. As discussed in Section 3.3.2, the allocated privacy budgets have been shown to optimize the utility of both sampling and bilinear interpolation, e.g., the optimal number of RGBs selected in each VE for sampling k_j tends to minimize the expectation of MSE between the utility-driven private video (after interpolation) and the original video. Thus, Phase II can directly apply bilinear interpolation. For simplicity of notations, we consider both retained pixels and sampled pixels as "sampled pixels", and both suppressed pixels and unsampled pixels as "unsampled pixels". Specifically,

First, in the output video of Phase I, pixels (not on the border) have at most 4 neighbors in each frame; the pixels on the border of each frame (not corner) have at most 3 neighbors; the pixels at the corner of each frame have at most 2 neighbors. Second, the algorithm interpolates pixels in visual elements and the remaining pixels (background), separately. For each interpolation, it traverses all the unsampled pixels in all the frames (e.g., a specific visual element). If any unsampled pixel has any sampled neighbor(s), the RGB for current unsampled pixel is estimated as the *mean* of all its sampled neighbors. Third, if any unsampled pixel's all the neighbors are also unsampled, the algorithm skips such unsampled pixel in the current traversal. The algorithm iteratively traverses all the skipped unsampled pixels. The algorithm

terminates until every unsampled pixel is assigned with an interpolated RGB. In our experiments, the interpolation terminates very quickly since the RGB of any pixel can be readily estimated as long as it has at least one neighbor which is sampled or previously interpolated. Finally, if any VE does not have a sampled pixel in any frame, the interpolation of the pixels for the visual element in such frame will be executed with the remaining pixels (background) $V \setminus \bigcup_{j=1}^{n} \Upsilon_j$.

3.5 Phase III: Video Analytics and Privacy Analysis

Similar to the framework of PINQ for data analytics [38], VideoDP can also function most of the analyses performed on videos. If breaking down any video analysis into queries, VideoDP (Phase III) directly applies the queries to the *utility-driven private video* (which is randomly generated in Phase I and II) and return the results to untrusted analysts. For any query created at the pixel, feature or visual element level [84–86], VideoDP (Phase III) could efficiently respond the results with differential privacy guarantee.

Theorem 2. *VideoDP satisfies ϵ-differential privacy.*

Proof. Recall that we have proven Phase I satisfies ϵ-differential privacy in Theorem 3. We now prove that Phase II and III do not result in additional privacy risks.

Since Phase I in VideoDP satisfies ϵ-DP, for any pair of neighboring videos V and V', we have $e^{-\epsilon} \leq \frac{Pr[\mathcal{A}(V)=O]}{Pr[\mathcal{A}(V)=O]} \leq e^{\epsilon}$. Such differential privacy satisfies ϵ-probabilistic differential privacy [32, 78], which also satisfies ϵ-indistinguishability differential privacy [13, 14] (bounding $Pr[\mathcal{A}(V) \in S]$ and $Pr[\mathcal{A}(V') \in S]$ where S is any set of possible outputs), as proven in [32, 78].

Then, after applying VideoDP to inputs V and V', the outputs of Phase I are ϵ-indistinguishable. Since the pixel interpolation (Phase II) and video queries/analysis

(Phase III) are deterministic procedures applied to the output of Phase I (which can be considered as *post-processing* differentially private results), the output $\mathbb{O}$ of Phase II and the analysis/query results of Phase III derived from V and V' are also ϵ-indistinguishable ("*Differential privacy is immune to post-processing*" was proven in [37]). Thus, VideoDP also satisfies ϵ-DP. □

The procedures and privacy guarantee in VideoDP can be interpreted as follows. Given any two videos V and V' that differ in any VE (e.g., a pedestrian), while applying a randomization algorithm (i.e., Phase I-III in VideoDP) to V and V', respectively, the possible outputs of sampling/obfuscating pixels (and post-processing) from V and V' are guaranteed to be indistinguishable. Then, the adversaries cannot identify if any VE (e.g., the pedestrian) is included in the input video or not (since "including" or "not including" such VE does not result in significant difference in the output). Such protection applies to any VE for any two neighboring videos V and V' (differing in a VE). Thus, all the sensitive VEs in any video can be protected by the randomization (obfuscating the pixels in the video).

In the meanwhile, the utility-driven private video can maintain good utility to allow useful computer vision algorithms to execute for the following reasons. First, the sampling randomly generates a subset of pixels with the original coordinates and RGBs in the output video, which are utilized for interpolating a video frame by frame. Second, the pixels in the background scene but not in the sensitive VEs are retained in the output video. Third, privacy budgets are allocated for different RGBs to maximally preserve the utility in the output. For instance, VideoDP only allocates budgets for the most representative RGBs in the VEs (given the privacy bound). Then, computer vision algorithms may still recognize some objects (*but not the specific objects due to uncertainty*) from the features extracted from the retained pixels and interpolated pixels.

3.6 Discussion

Relaxed Differential Privacy. Theorem 2 ensures that the analysis satisfies ϵ-DP. Thus, similar to PINQ [38], all the aggregation-based queries (w.r.t. more than one VE) could be protected with ϵ-DP in VideoDP. However, if querying on a specific VE (e.g., a unique red car or license plate) which is not included in one of the two neighboring videos, the protection requires another privacy bound δ to ensure (ϵ, δ)-DP (Definition 2). Such additional privacy bound is also required in other contexts (e.g., [32]). We will leave it for the future work.

Background Scene(s) as VE(s). If necessary, any part of the video can be specified as a sensitive VE for protection, including the background scene(s). In VideoDP, the failure of detection/tracking algorithms may occur (though the state-of-the-art techniques could minimize such risks [87]). To avoid such risks, we can consider the background scene as a VE (the "Background Scene(s) as VE" mode in VideoDP). Specifically, in Phase I, the same sampling algorithm will be applied to all the VEs by adding background VEs to Case (1), (2) and (3). Unique RGBs in background VE(s) will be suppressed (same as other VEs), and budgets are also allocated for non-unique RGBs in the background VE(s) using the same Algorithm 1 (same as other VEs). In Phase II and III, the same bilinear interpolation and queries are applied. Thus, DP can be ensured for all the pixels in the video. We have experimentally evaluated the performance of such strong protection in Section 3.7.

Defense against Correlations. Videos include a large number of sequential frames, if protecting specific VEs in only one frame, the correlations in sequential frames may also leak information to adversaries [88, 89]. Our VideoDP can address such vulnerabilities since all the VEs in all the frames are protected using our privacy notion – adding or removing any VE in any number of frames would not result in

significant risks. From this perspective, the privacy notion is defined for the entire period of the video, rather than a specific time. Thus, possible privacy leakage resulted from correlations among multiple frames can be tackled.

System Usability. Similar to many smartphone Apps with face/object detection, the detection/tracking algorithms for different types of VEs can be simply integrated into VideoDP (in the preprocessing), and upgraded with newer algorithms when necessary. Thus, both the video owner and the video analyst are not required to be experts of computer vision. The video owner only needs to specify that what types of visual elements (e.g., humans) should be protected. Then, the pre-processed video can be sampled and interpolated (Phase I and II). After that, the utility-driven private video will be generated and stored by the trusted server for external video analyses (Phase III). The video analysts only need to submit the video name and query (e.g.,$<$*VEH, total vehicle#*$>$), which may include additional parameters. The trusted server will respond the query result with differential privacy.

3.7 Experiments

In VideoDP, we implement the VE detection/tracking algorithm in [35] throughout the entire video. It first detects all the VEs in each frame, and then utilizes the tensorflow training database to tag all the humans/objects, which are considered as sensitive VEs. Each detected VE can be tracked with the same ID if their overlap in multiple frames has exceeded a threshold. This method ensures a high detection/-tracking accuracy [35]. We conduct our experiments on three video datasets in which different VEs (with different sizes) are protected. Table 4.1 shows the characteristics of the videos.

1. MOT [81]: 15 videos with different scenes. Sensitive VEs in these videos are pedestrians and vehicles. We denote this dataset as "MOT".

Table 3.2. Characteristics of Experimental Datasets

Datasets	Avg. Resolution	Video #	Avg. Frame #
MOT	1920×1080	15	846
UAD	740×480	24	180
BVD	2464×2056	5	1200

2. UAD [90]: the UCSD anomaly detection dataset includes crowded pedestrians as sensitive VEs. 24 different videos are captured at 2 different scenes. We denote this dataset as "UAD".

3. BVD [80, 91]: the Boxy [91] vehicle detection dataset includes over 200,000 sequential images at 5 different scenes such as sunny, rainy, and nighttime drive. We take such sequential images (as videos) and a "highway" video [80]. We denote them as "BVD".

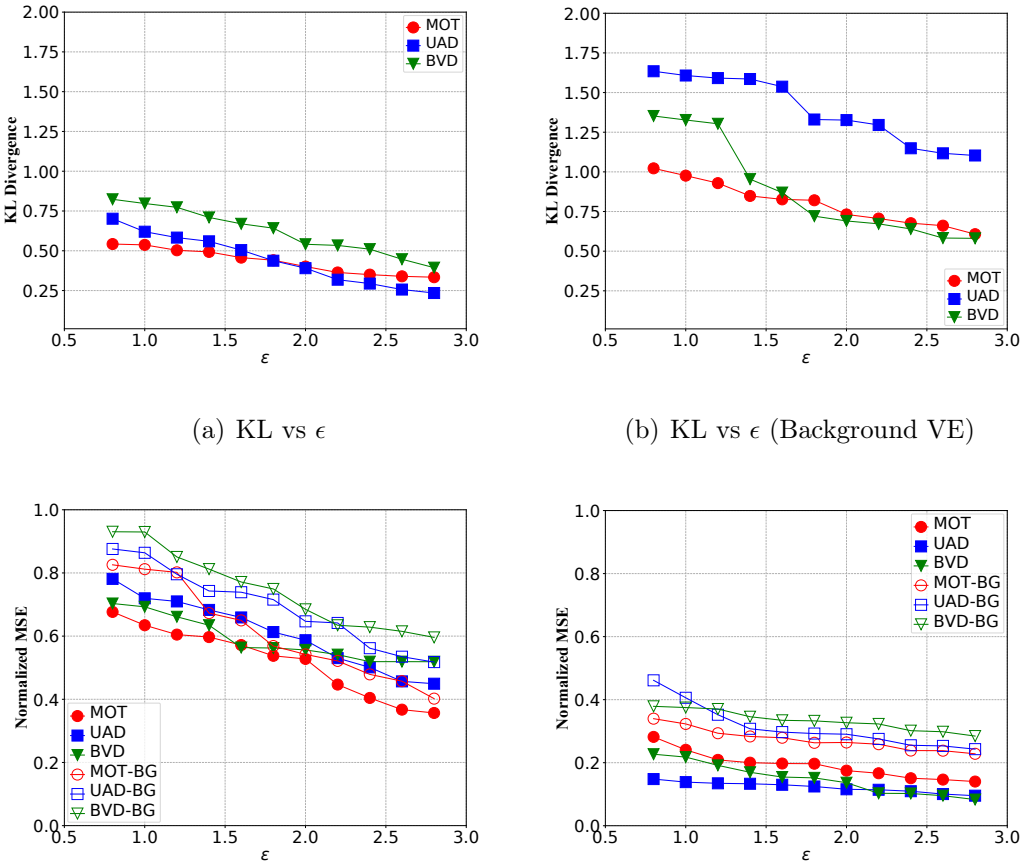
All the programs were implemented in Python 3.6.4 with OpenCV 3.4.0 library [92] and tested on an HP PC with Intel Core i7-7700 CPU 3.60GHz and 32G RAM.

3.7.1 Evaluating Utility-driven Private Video.

Pixel Level Evaluation. We consider the RGB color model [36] by breaking down the videos into pixels with RGBs at different coordinates (a, b) and frame t, and then measure the differences between input V and output O. Specifically, we evaluate two types of utility: (1) the difference between the count distributions of all the RGBs in V and O, and (2) the difference between RGB values of all the pixels in V and O.

First, considering the distributions of all the RGB counts $\forall c_i$ and $\forall x_i$ in the input/output, we can measure the utility loss using their KL divergence. If the

distribution of RGBs lie closes in the input and output, the performance of pixel interpolation (estimating RGBs for unsampled pixels based on the RGBs of sampled pixels) can be greatly improved [83]. For other measures, e.g., L_1 norm, the output counts of different RGBs might be biased towards certain RGBs with high counts such that the interpolated RGBs might be significantly deviated.



(a) KL vs ϵ

(b) KL vs ϵ (Background VE)

(c) MSE vs ϵ (after Phase I)

(d) MSE vs ϵ (after Phase II)

Figure 3.5. Pixel Level Utility Evaluation (three video datasets MOT, UAD and BVD) – BG refers to "Background Scene(s) as VE(s)"

Second, after interpolating all the pixels in the Phase II of VideoDP, we measure the difference between all the pixel RGBs in V and O using the expectation of mean squared error (MSE). The 3-dimensional RGBs are generally converted to gray for measuring the MSE [93], which are normalized to values in $[0, 1]$.

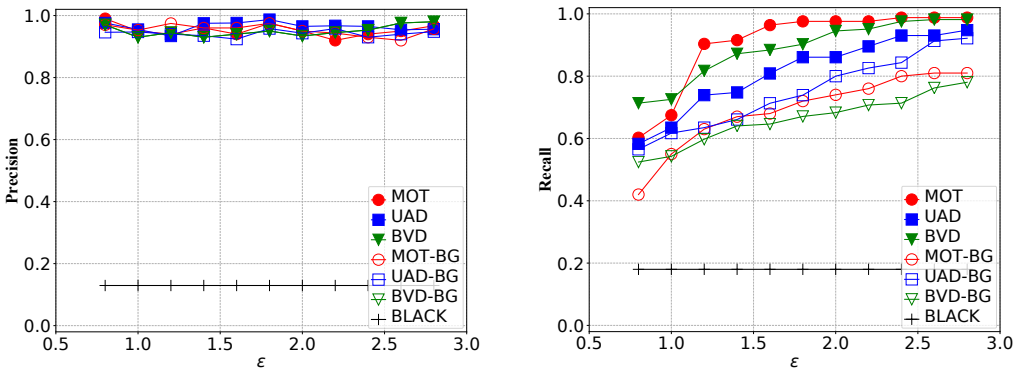
Note that we will demonstrate the average of KL-divergence and MSE values in different datasets, each of which includes multiple videos. Specifically, we conducted two groups of experiments to test how ϵ influences the utility (ϵ=0.8,. . . , 2.8). As discussed earlier, if necessary, VideoDP can define any part of the video including the background scenes (pixel-level protection) as sensitive VEs. Then, we conduct experiments for both cases (background scene(s) as sensitive VE(s) or not). Figure 3.5(a) and 3.5(b) present the KL divergence values for two cases, respectively. In all the datasets, the results monotonically decrease while ϵ increases, and the results of "background scene(s) as VE(s)" are larger than "background scene(s) not as VE(s)" since more pixels can be preserved within background in the latter case.

In addition, we also evaluated the MSE of the output videos (after Phase I, and after Phase II). Figure 3.5(c) and 3.5(d) show that the MSE (of the entire video) declines as ϵ increases. This matches the fact that larger ϵ (with weaker privacy protection) trades off less utility. Also, the MSE has been greatly reduced after Phase II (comparing the results in Figure 3.5(c) and 3.5(d)), which greatly improves the query accuracy for video analyses. We can also observe that the MSEs of "background scene(s) as VE(s)" are larger since pixels in the background scenes are sampled rather than fully retained.

Video Utility Evaluation. Detection and tracking accuracy (e.g., precision and recall) is an important measure for utility evaluation. Considering the results obtained from three original video datasets (MOT, UAD and BVD) as the benchmarks, we test the *precision* and *recall* of detecting and tracking VEs in different outputs. Precision returns the percent of true VEs out of all the detected/tracked results in the videos. Recall returns the percent of detected/tracked true VEs out of all the true VEs (the benchmarking results).

Specifically, we compare VideoDP with the method of blacking the detected

VEs in the entire video (denoted as "Black") in which the contours of VEs are detected and pixels within the contours are assigned the black RGB ("000000"). Since the classifiers in common detection algorithms (e.g., HOG [76], SIFT [94] and CNN [87]) primarily rely on the features rather than the contours, the detection accuracy is quite close to 0. Then, we use the recent contour detection algorithm [95] in the experiments instead, which can maintain a relatively good detection accuracy (i.e., around 80%). However, the accuracy of tracking black contours across multiple frames is still quite low (less than 20% of precision and recall in all the three video datasets MOT, UAD and BVD) since the tracking algorithm cannot distinguish the VEs in multiple frames (which are similar contours with black pixels). Figure 3.6 demonstrates the precision and recall on a varying privacy budget ϵ (vs the low accuracy of "Black"). The precision can always be high (close to 1), and the recall grows quickly as ϵ increases (since a larger ϵ can generate more accurate random videos for analysis).



(a) Precision vs ϵ (b) Recall vs ϵ

Figure 3.6. Visual Elements Detection and Tracking

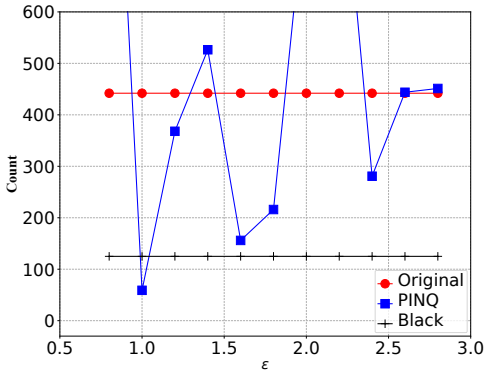
Based on the detection/tracking, we also empirically evaluate the utility of queries over the VEs by benchmarking with Black and PINQ [38] in which the sensitivity might be extremely large (e.g., queries involving the frames). The example queries are set as "the number of frames with more than 15 pedestrians in each video of

MOT, 10 pedestrians in each video of UAD, and 10 vehicles in each video of BVD, respectively" where the results are averaged in each video dataset (similar performance can be derived from other similar queries).

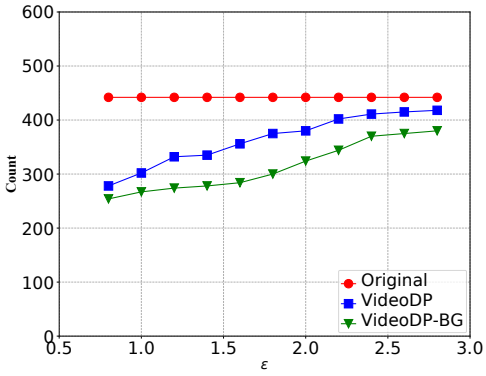
Figure 3.7 demonstrates the average counts of frames with 15+ pedestrians in the MOT videos, 10+ vehicles in the BVD videos, and 10+ pedestrians in the UAD videos, including the PINQ results, Black results, VideoDP results and original results, respectively (different privacy budgets ϵ for PINQ and VideoDP). We can observe that VideoDP returns more accurate results (also random) than PINQ, and more accurate results than Black in general (only except the very small ϵ cases). Also, in the Black results, the accuracy of counting the contours is highly reduced in the videos in which VEs frequently overlap or there are more than one type of VEs (e.g., both pedestrians and vehicles are included in some videos in the MOT dataset).

3.7.2 Case Study: Video Queries/Analysis. The videos randomly generated in VideoDP can function a wide variety of analyses (aggregation-based queries), such as head counting, crowd density and traffic flow analysis [80, 81, 96]. We empirically evaluate some representative queries for such analysis by benchmarking with the PINQ platform [38] in specific videos (e.g., results in different frame) since different VEs cannot be accurately tracked by the Black method these applications. We choose three empirical videos ("MOT16-04" "MOT16-14" and "highway" [80]) from the MOT and BVD datasets, with pedestrians, vehicles, and both. Then, the three videos are denoted as "PED" "VEH" and "PV", respectively. Note that all the queries satisfy ϵ-DP in the following case study.

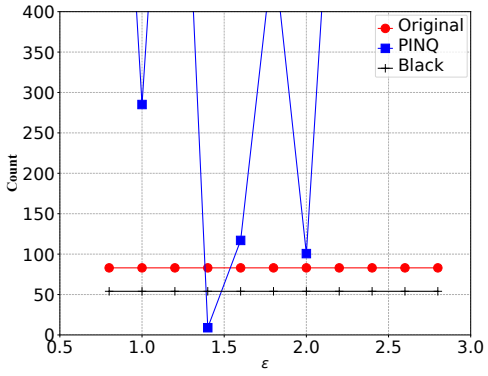
(1) VE Stay Time (Large Sensitivity). Besides the queries on counting, VideoDP can also privately return query results based on detected/tracked VEs in different applications. For instance, a query returns "how long each pedestrian/vehicle stays in the video" (namely, stay time) which can be measured by the number of frames



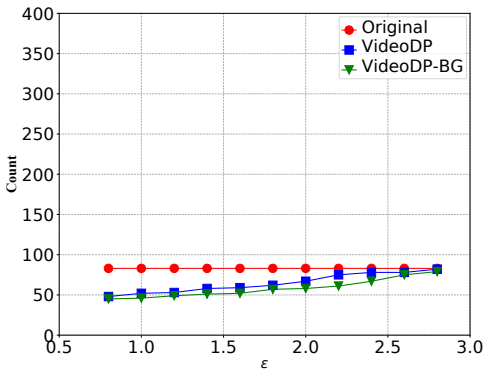
(a) PINQ and Black (MOT)



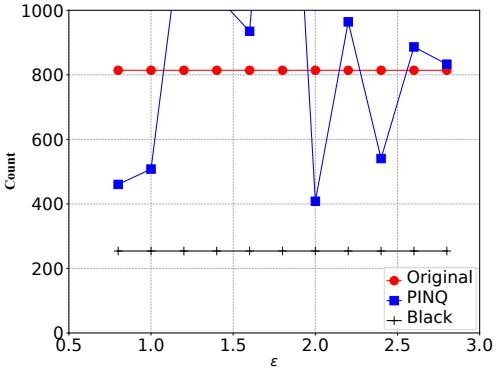
(b) VideoDP (MOT)



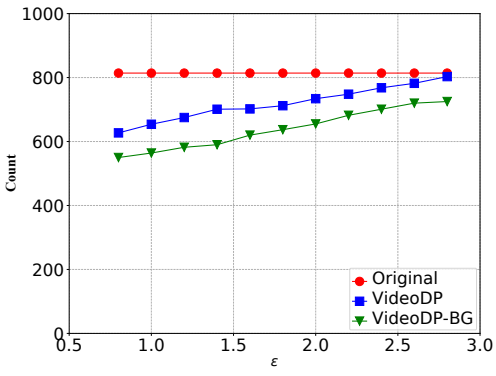
(c) PINQ and Black (UAD)



(d) VideoDP (UAD)



(e) PINQ and Black (BVD)

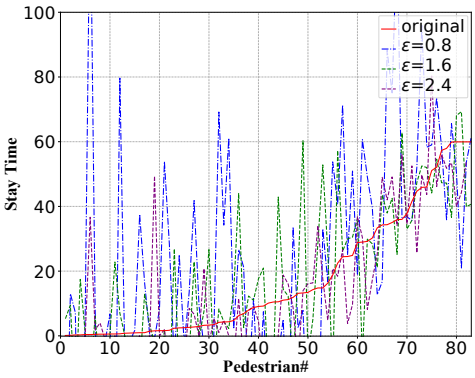


(f) VideoDP (BVD)

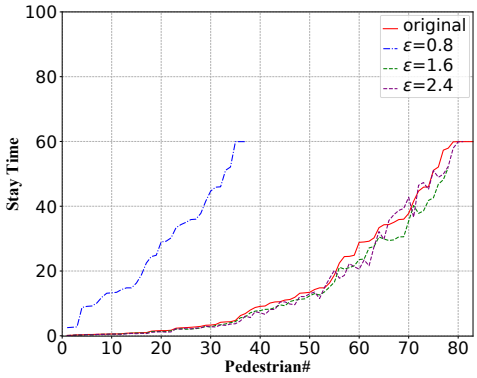
Figure 3.7. Average Frame Counts with 15+ VEs in MOT Videos, 10+ VEs in UAD Videos, and 10+ VEs in BVD Videos

involving each VE. Then, pedestrians/vehicles are detected/tracked in all the frames, and then query results can be computed and returned for private analysis. Since too many groups of fine-grained empirical results may mess up the plots (e.g., in Figure 3.8), we only show three groups of results for $\epsilon = 0.8, 1.6$ and 2.4, which represent small, medium and large ϵ, respectively. Other groups of results lie between them.

- **Pedestrians**. In PED, 83 pedestrians are walking on the street. How long each pedestrian stays in the video can be utilized to learn the human behavior. Figure 3.8 presents the original results, PINQ results and VideoDP results for the PED. The 83 pedestrians in the PED (marked on the x axis), and the stay time is ranked from short to long (see the red curve in two subfigures). In PINQ (Figure 3.11(c)), the stay times of all the pedestrians are overly obfuscated even if ϵ is large since sensitivity Δ should be set as 60 (for even longer videos, Δ should be larger). Nevertheless, VideoDP significantly outperforms PINQ. As shown in Figure 3.11(d), in case of $\epsilon = 0.8$ (small privacy budget), approximately 40 distinct pedestrians are detected in the result. Although not all the pedestrians are sampled in VideoDP, the distribution of all the stay times (of the pedestrians) still lies close to the original result. The results of $\epsilon = 0.8$ show less pedestrians in the x-axis than other two groups of results ($\epsilon = 1.6$ and 2.4) since many pedestrian cannot be detected for small ϵ. As ϵ increases to 1.6, VideoDP results are close to the original results (however, PINQ results are still fluctuated).
- **Vehicles**. In the VEH, there are 115 distinct vehicles driving on the highway. We define the two-way moving directions as "upstream" and "downstream". Figure 3.9 demonstrates the length of time the vehicles stay in the video (upstream and downstream), which can be utilized to estimate the moving speed of vehicles, queue length estimation, etc. We can draw similar observations for the stay times of vehicles for both moving downstream and upstream directions in

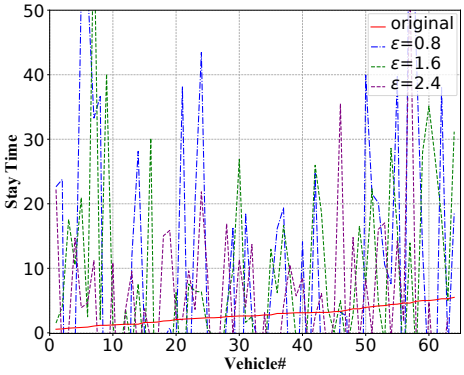


(a) Pedestrian (PINQ)

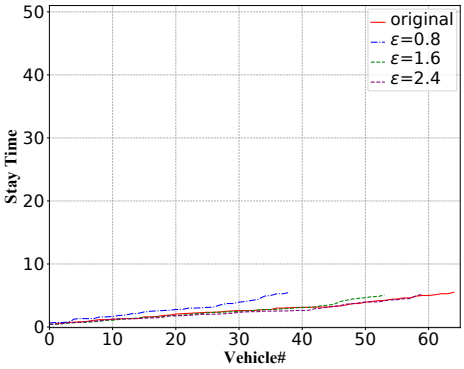


(b) Pedestrian (VideoDP)

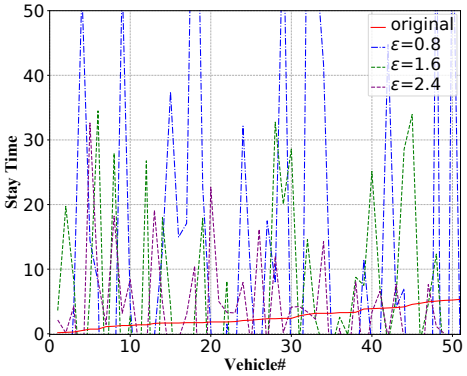
Figure 3.8. Pedestrian Stay Time in PED



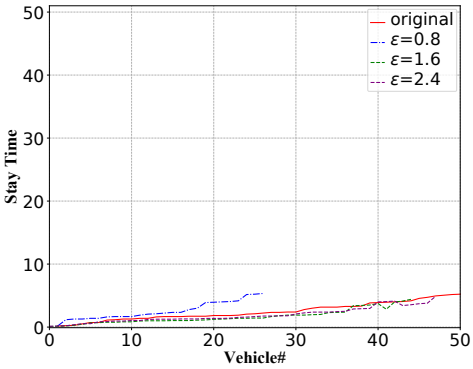
(a) Upstream (PINQ)



(b) Upstream (VideoDP)

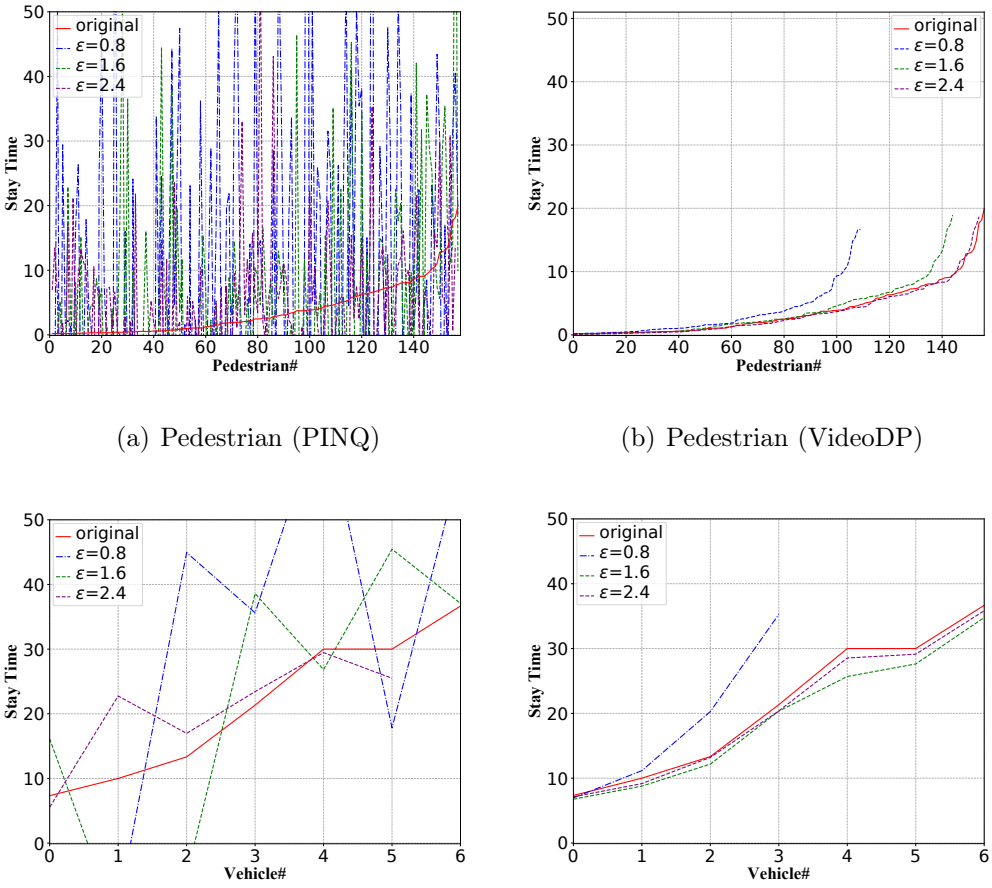


(c) Downstream (PINQ)



(d) Downstream (VideoDP)

Figure 3.9. Vehicle Stay Time in VEH



(a) Pedestrian (PINQ)

(b) Pedestrian (VideoDP)

(c) Vehicle (PINQ)

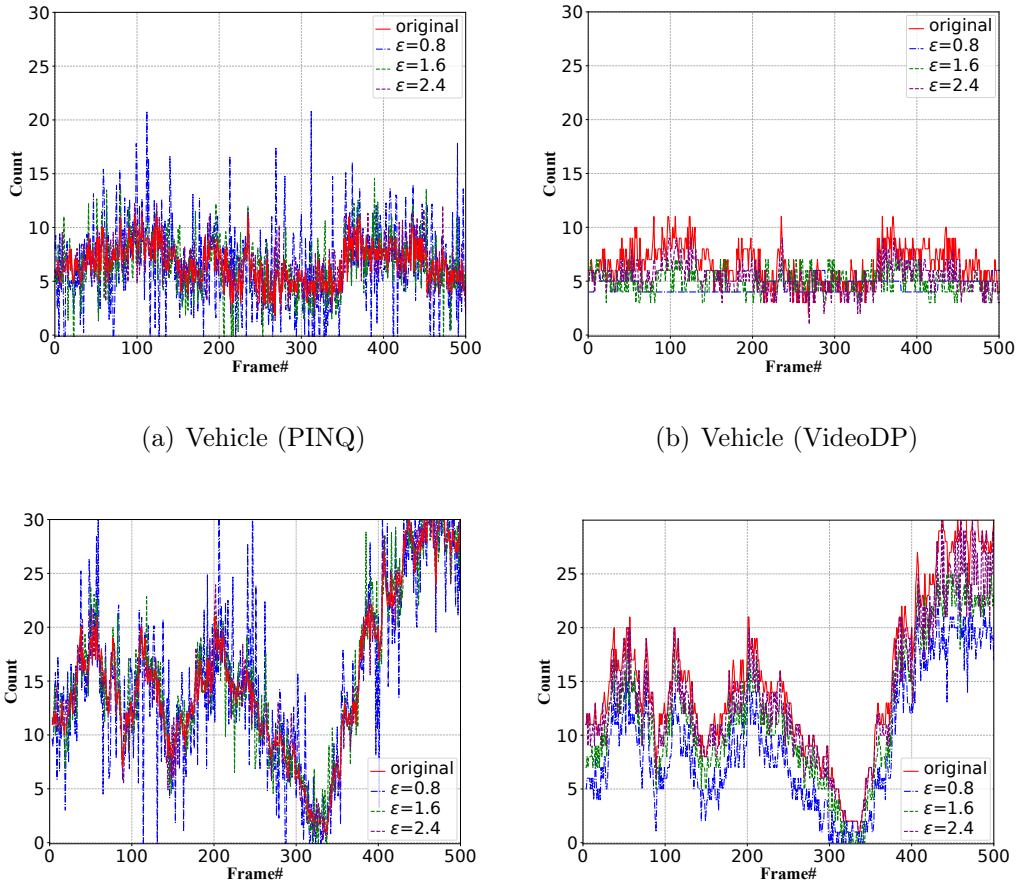
(d) Vehicle (VideoDP)

Figure 3.10. Pedestrian and Vehicle Stay Time in PV

the VEH. VideoDP also significantly outperforms PINQ.

- **Pedestrians and Vehicles**. In the PV, there are 157 distinct pedestrians and 7 vehicles. Figure 3.10 demonstrates the length of time the pedestrians and vehicles stay in the video. It presents similar trends.

(2) VE Density (Small Sensitivity). We also conduct empirical studies to compare VideoDP and PINQ on queries with a smaller sensitivity. For instance, the vehicle density query returns the vehicle count in each frame of the video (sensitivity $\Delta = 1$), which can also facilitate the analyst to learn the traffic flow. Figure 3.11 shows the count of vehicles in each frame of VEH and pedestrians of PV, including



(a) Vehicle (PINQ)

(b) Vehicle (VideoDP)

(c) Pedestrian (PINQ)

(d) Pedestrian (VideoDP)

Figure 3.11. VE Count in Each Frame

the original results, PINQ results and VideoDP results (where $\epsilon = 0.8, 1.6$ and 2.4). Note that every vehicle only appears in a few frames of the video in VEH (see Figure 3.11(a) and 3.11(b)). The noisy results are both acceptable in PINQ ($\Delta = 1$) and VideoDP. However, the counts of vehicles are more fluctuated in PINQ as ϵ is small. We can draw similar observations in Figure 3.11(c) and 3.11(d).

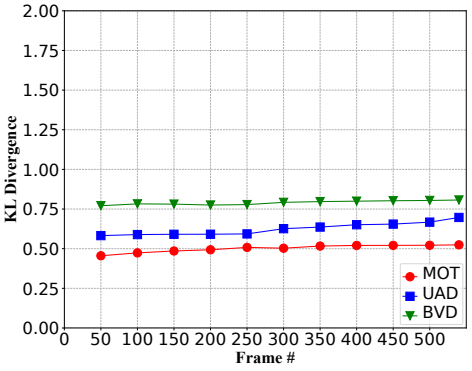
3.7.3 Deep Learning Attack. We also perform the CNN based attacks [60] to demonstrate VideoDP's protection against deep learning, though VideoDP does not directly reveal videos/frames to analysts (DP algorithms reveal the query results in general). Assuming that the adversary has known everything about specific VE(s),

and tries to re-identify such VE(s) in the output videos (notice that, for each output video of VideoDP which is random, we generate 20 videos). We compare the performance of VideoDP with the Mosaic blurring method against the CNN attack [60]. Mosaic blurring considers each square of pixels (a.k.a., "pixel box") as the mosaic window, computes the average color of every pixel in each square, and sets the entire square as that color. In the experiments, we set privacy budget ϵ as 0.8, 1.2, 1.6 and 2.4 in VideoDP, and the sizes of pixel boxes as 2×2, 4×4, 8×8 and 16×16. Table 3.3 shows the average accuracy of successfully identifying such VE(s) from the random outputs of VideoDP and the videos sanitized by Mosaic blurring. For all different ϵ, the VE(s) cannot be identified with high confidence, compared to Mosaic blurring.

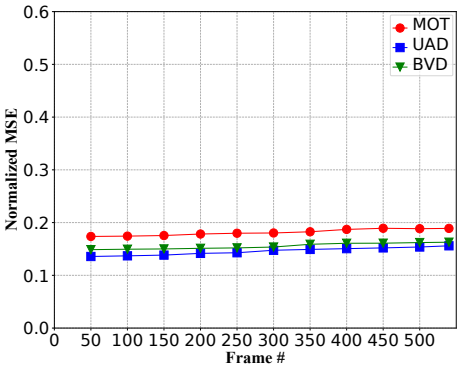
Table 3.3. Accuracy of the CNN Attack (%).

Video	Mosaic (Pixel Box Sizes for Blurring)			
Datasets	2×2	4×4	8×8	16×16
MOT	97.45	91.33	89.75	64.75
UAD	99.23	95.47	92.25	76.25
BVD	85.78	75.62	63.14	44.39
Video	VideoDP (ϵ)			
Datasets	0.8	1.2	1.6	2.4
MOT	3.62	5.96	12.96	16.47
UAD	4.93	11.27	16.22	19.78
BVD	5.34	7.54	18.77	21.41

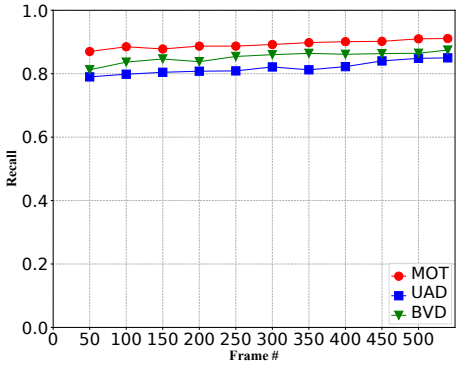
3.7.4 Scalability. We also evaluate how video length affects the performance of VideoDP (frames in UAD videos are repeated to synthesize longer videos). First, in



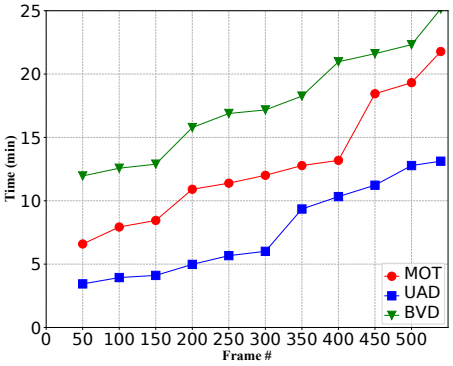
(a) KL vs. Video Length



(b) MSE vs. Video Length



(c) Recall vs. Video Length



(d) Runtime vs. Video Length

Figure 3.12. Performance vs. Video Length ($\epsilon = 1.6$)

Figure 3.12(a) and 3.12(b) ($\epsilon = 1.6$), the KL and MSE values slightly change as the length of three sets of videos increases. Second, as the number of frames increases, the detection accuracy (recall) slightly increases for all the three videos (see Figure 3.12(c)). Third, we have also evaluated the runtime of VideoDP. Figure 3.12(d) shows a linear runtime trend on the video length, which provides sufficient efficiency for randomly generating longer high-resolution videos (e.g., 1920×1080). For longer videos, we can split the input video into multiple fragments (e.g., 1 minute per fragment). Then, we can still apply VideoDP to efficiently sanitize all the fragments which are integrated later. In many videos (e.g., traffic monitoring videos), VEs move rapidly and appear in the video for a few seconds. Then, fragmentation, generation and in-

tegration would not affect the privacy. Finally, in each video, all the VEs may have different sizes. While VEs are moving, the size of the same VE may also vary in different frames. VideoDP protects all the VEs (including all the pixels of each VE in all the frames). VideoDP generates good utility for all the videos with different VE sizes.

3.8 Conclusion

In this paper, to our best knowledge, we take the first step to study the problem of video analysis with *differential privacy guarantee*. Specifically, we have proposed a new sampling based differentially private mechanism to generate utility-driven private videos for any private analysis. The proposed VideoDP has also provided a flexible platform for untrusted analysts to privately conduct any kind of query/analysis over the randomly generated utility-driven private video. We have proven the differnetial privacy guarantee, and conducted extensive experiments to validate the performance of VideoDP by benchmarking the results with the PINQ-based video analyses. The experimental results have demonstrated superior utility in different analyses.

CHAPTER 4

PUBLISHING VIDEO DATA WITH INDISTINGUISHABLE OBJECTS

In this Chapter, I present the framework VERRO which can release synthetic videos instead of the query-based video platform in details, which includes the introduction, preliminaries, privacy model, framework, discussion and experiments [2].

4.1 Introduction

Recall that, in today's society, millions of videos are generated and shared ubiquitously every day through various means such as dedicated facilities, traffic cameras, and smartphones. The availability of such video data provides an unprecedented opportunity for enhancing human interactions and benefiting the community. However, it also raises serious privacy concerns, as sharing these videos can potentially expose personal information of individuals, which can lead to various privacy violations.

To address the issue of privacy in video data, several approaches have been proposed. One such approach is the development of privacy-preserving video query analytic platforms, such as VideoDP. The goal of such platform is to provide privacy protection for video queries, which can be a useful tool for retrieving specific information from video datasets. However, it is important to note that video queries may not be suitable for all video analysis tasks, as the information they can provide by queries is very limited.

For example, when analyzing a video dataset to detect anomalies or unusual events, a video query may only return the number of unusual events in the video, but may not provide any additional information about the nature of these events. In such cases, releasing the video directly may be more appropriate. However, directly releasing the video may involve privacy concerns related to the objects, such as pedestrians and vehicles, within the video.

To address the limitations of privacy-preserving video query analytic platforms, such as VideoDP, a new framework called VERRO has been proposed. Recall that videos differ from many other data (e.g., statistical databases [13], location data [18], search logs [22]) - a local video may include numerous objects corresponding to multiple different individuals, e.g., many pedestrians are recorded in a single video, and many vehicles (w.r.t. different drivers) are recorded in the same video. A video includes the "local data" of many individuals (e.g., humans) which will be shared to the untrusted recipients via the video owner. VERRO ensures ϵ-object indistinguishability for any video and generates synthetic videos for any untrusted recipients. thus enhancing privacy protection in video data. Thus, the primary difference between ϵ-*Object Indistinguishability* and the original definition of LDP [29–31] is that the

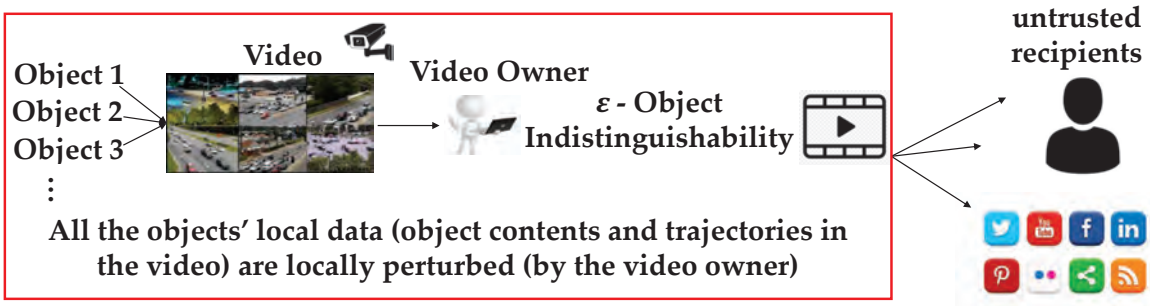


Figure 4.1. VERRO: Ensuring *Object Indistinguishability* in the Video Data Sanitization

In conclusion, choosing the appropriate privacy-preserving video analysis technique will depend on the specific task and the characteristics of the video data. While video queries can be a useful tool for retrieving specific information from video datasets, they may not be suitable for all video analysis tasks. To address privacy concerns related to video data, privacy-preserving video query analytic platforms and synthetic video generation are both required.

4.2 Preliminaries

In this section, we first describe the adversary model, then define our privacy notion, and finally provide a general overview of our proposed approach VERRO.

4.2.1 Adversary Model. Denote a video by $\mathcal{V}$ which is captured by a video owner (e.g., a hospital or a company equipped with CCTV surveillance, an agency which captures the video on the street). Video $\mathcal{V}$ (all the frames) includes a set of n sensitive objects $\mathbb{O} = \{O_1, O_2, \cdots, O_n\}$ (e.g., humans, vehicles). Assume that the video owner would like to share $\mathcal{V}$ to an external party for analysis (viz. the adversary). To ensure privacy, instead of directly sending $\mathcal{V}$, our proposed VERRO (randomly) generates a synthetic video $\mathcal{V}^*$ which is close to $\mathcal{V}$, such that:

- Each sensitive object in all the frames satisfies ϵ-*Object Indistinguishability* – the adversary cannot distinguish any two objects from the output synthetic video $\mathcal{V}^*$ with arbitrary background knowledge.
- The synthetic video $\mathcal{V}^*$ retains good utility (close to $\mathcal{V}$).

In VERRO, we assume that the adversaries can possess arbitrary background knowledge on each object at the scene (e.g., the object content, the trajectories, at-scene times, gathering groups of objects). To retain the output utility, VERRO does not change the background scene(s) of the video, but the privacy model can break the linkage between each object and the background scene(s) via *indistinguishability*.

With privacy guarantee for all the objects (making them indistinguishable), VERRO regularly generates synthetic videos for videos including sensitive objects w.r.t. multiple individuals (e.g., pedestrians, vehicles). Even though the video includes only one sensitive object, the adversary still cannot re-identify the object (as discussed in Section 3.6). In addition, VERRO only addresses the visual privacy con-

cern udio

is nc

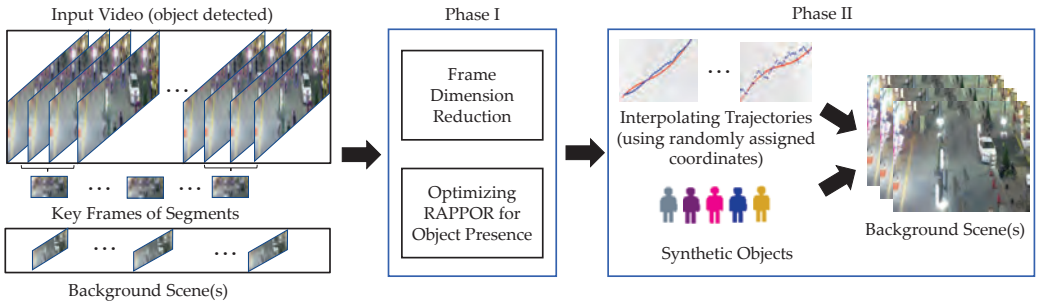


Figure 4.2. VERRO for Utility-Driven Synthetic Video Generation with Object Indistinguishability

4.2.2 Privacy Notion. Traditional Privacy Model: video $\mathcal{V}$ includes multiple sensitive objects $O_1, \ldots, O_n$, each of which can be detected and marked using the same object ID in all the frames. Specifically, we can detect objects (e.g., pedestrians, vehicles) using the tracking algorithm [35, 97]. It first detects all the sensitive objects in each frame with the existing detection algorithm (e.g., HOG for human [98], SVM for vehicles [99]). Each detected object can be tracked with the same ID if their overlap in multiple frames has exceeded a threshold (it has a high *detection/tracking accuracy [35]*).

The traditional privacy models are defined to blur all the detected objects [52–54, 71, 72]. An alternative solution could be replacing the detected objects with "synthetic objects" [100, 101]. Each object can be replaced by a unique synthetic object: for instance, a red synthetic human and a purple synthetic human can be used to represent two different pedestrians in all the frames involving them. Then, the inferences and re-identification visually from the objects can be greatly mitigated using such traditional privacy models.

ϵ-Object Indistinguishability for Sensitive Objects: Recall that only replacing the objects with synthetic objects in the video cannot address the re-

identification based on the adversaries' background knowledge (as discussed in Chapter 1). Thus, we need to ensure *indistinguishability* for not only objects themselves (can be achieved by synthetic objects) but also their moving trajectories [102] in the video.

To this end, inspired from the indistinguishability provided by the ϵ-LDP, we define a novel privacy notion ϵ-*Object Indistinguishability* by considering each object's trajectory in the video (coordinates at different frames) as its "local data". Specifically, in the standard LDP definition [29–31], there are a set of users, each of which has its own data. After each user locally perturbs its data, the obfuscated output can be directly disclosed to any untrusted recipient/aggregator, where the randomized data collected from any two different users are indistinguishable [29,31,103]. Migrating the LDP model to the objects in any video $\mathcal{V}$, we define the ϵ-*Object Indistinguishability* as below:

Definition 3 (ϵ-Object Indistinguishability)**.** *A randomization algorithm $\mathcal{A}$ satisfies* ϵ-Object Indistinguishability*, if and only if for any two input objects $O_i, O_j \in \mathbb{O}$ in the input video $\mathcal{V}$, and for any output object of $\mathcal{A}$ in the synthetic video $\mathcal{V}^*$ (denoted as y), we have $Pr[\mathcal{A}(O_i) = y] \leq e^{\epsilon} \cdot Pr[\mathcal{A}(O_j) = y]$.*

Notice that, similar to ϵ-LDP [29], ϵ-*Object Indistinguishability* also focuses on the indistinguishability of randomizing any two objects, rather than the indistinguishability of randomizing any two neighboring inputs (whether any object is included or not included in the input) in traditional differential privacy setting [13]. Privacy budget ϵ decides the degree of indistinguishability (which is identical to LDP [29]).

Definition 3 guarantees that the randomly perturbed output of any two objects in $\mathcal{V}$ (both the object contents and the trajectories in all the frames) are ϵ-

indistinguishable in $\mathcal{V}^*$. It also ensures *plausible deniability* for every object [104]. Since ϵ-*Object Indistinguishability* also requires all the objects to be visually indistinguishable (object contents), VERRO randomly assigns synthetic objects (e.g., the same shape but different colors) to replace the original distinct objects while generating the synthetic video $\mathcal{V}^*$. The synthetic objects are generated and placed by considering the distance of the object to the camera (e.g., the size of the synthetic object is larger if it is closer to the camera) [105].

4.2.3 VERRO Framework. We now illustrate the major components of VERRO:

1. **Preprocesssing**: all the objects are detected and tracked, and background scene (for each frame) is extracted using computer vision techniques [35,97,106].

2. **Phase I**: for each object, its presence or absence in different frames/segments of the video are randomly generated (by random response) to be indistinguishable. Before executing random response, VERRO reduces the frame dimension in the video by detecting the key frames in the video. Then, the utility can be improved by allocating optimal budgets for different dimensions. Furthermore, we also formulate a utility maximizing random response problem (optimizing RAPPOR [29]) to retain the optimal object presence information after Phase I. Note that this phase satisfies ϵ-*Object Indistinguishability*: all the objects' presence in all the frames are indistinguishable. Details are given in Section 4.2.3.

3. **Phase II**: with the randomly generated presence/absence information for each object, VERRO generates the synthetic video by inserting the synthetic objects into the video (background scene(s)). Specifically, the coordinates (where to insert the synthetic objects) are assigned, and computer vision techniques are applied to interpolate object moving trajectories between two assigned coordinates in the synthetic video. We also shown that Phase II does not leak

any additional information (as a post-processing step [37]), and then VERRO satisfies ϵ-*Object Indistinguishability*. Details are given in Section 4.4.

As the "local data" of each object (e.g., a pedestrian or vehicle) in the video $\mathcal{V}$, the object trajectory includes its presence or absence information in each frame and the coordinates in the frame (if present). In this section, we illustrate the Phase I of VERRO that first generates indistinguishable object presence.

4.3 Phase I: Optimal Object Presence

4.3.1 Poor Utility with Random Response (i.e., RAPPOR [29]) for Object Presence. We first define a bit vector for each object to indicate if such object is included in different frames or not:

Definition 4 (Object Presence Vector)**.** *Given video $\mathcal{V}$ which includes m different frames $F_1, \ldots, F_m$ and n distinct objects $\mathbb{O} = \{O_1, ..., O_n\}$, whether each object $O_i, i \in [1, n]$ is present in frame $F_k, k \in [1, m]$ or not (all m frames) can form a bit vector: $B_i = (b_i^1, .., b_i^m) \in \{0, 1\}^m$ for object O_i.*

It has been proven that a classic randomized response (RR) technique (e.g., RAPPOR [29,30]) can be adapted to ensure ϵ-LDP for locally randomizing bit vectors. Similarly, a naive solution of ensuring ϵ-*Object Indistinguishability* for the object presence vectors is to directly the random response mechanism (*we will discuss how to optimize the utility in Section 4.3.2 and 4.3.3*). For each object $O_i \in \mathbb{O}, i \in [1, n]$, if object O_i exists in frame F_k, we set $b_i^k = 1, k \in [1, m]$. Otherwise, $b_i^k = 0$ holds in the vector B_i. Then, we flip one bit in vector $B_i, i \in [1, n]$ with a certain probability to report the true value. Then, all the perturbed bits in the object presence vector B_i can be combined as the output object presence vector for object O_i. Thus, the vectors $B_1, \ldots, B_n$ (of all the objects) can be indistinguishable. Algorithm 2 [29] shows the details.

Algorithm 2 Random Response for Object Presence [29]

1: detect all the objects $\mathbb{O} = \{O_1, \ldots, O_n\}$ in $\mathcal{V}$
2: **for** each $O_i, i \in [1, n]$ **do**
3: collect the object presence vector $B_i = (b_i^1, .., b_i^m)$ in $\mathcal{V}$
4: **for** each frame $F_k, k \in [1, m]$ **do**
5: equally allocate budget ϵ/m to frame F_k
6: random response for bit b_i^k with the probability $\frac{e^{\epsilon/m}}{1+e^{\epsilon/m}}$
7: **end for**
8: $B_i \leftarrow (b_i^1, ..., b_i^m)$
9: **end for**
10: Return $\forall i \in [1, n], B_i$

Theorem 3. *Algorithm 2 randomly generates object presence vectors for objects with ϵ-Object Indistinguishability.*

Proof. ϵ-Object Indistinguishability can be proven by following the proof of ϵ-LDP with random response [29]. Given the object presence vectors $B_i = \{b_i^1, \ldots, b_i^m\}$ and $B_j = \{b_j^1, \ldots, b_j^m\}$ of any two objects $O_i, O_j \in \mathbb{O}$, for any possible output m-bit vector $y = (y^1, \ldots, y^m)$, we have:

$$\frac{Pr[\mathcal{A}(B_i) = y]}{Pr[\mathcal{A}(B_j) = y]} = \frac{Pr(b_i^1 = y^1)}{Pr(b_j^1 = y^1)} \cdots \frac{Pr(b_i^m = y^m)}{Pr(b_j^m = y^m)} \tag{4.1}$$

Since each bit is allocated with an equal privacy budget ϵ/m, the flipping probability would be $\frac{e^{\epsilon/m}}{1+e^{\epsilon/m}}$ [29]. For $k \in [1, m]$, if $b_i^k = b_j^k$ (either 0 or 1), then $\frac{Pr(b_i^k = y^k)}{Pr(b_j^k = y^k)}$ always equals 1. If $b_i^k \neq b_j^k$ and $b_i^k = y_k$, thus we have:

$$\frac{Pr(b_i^k = y^k)}{Pr(b_j^k = y^k)} = \frac{e^{\frac{\epsilon}{m}}}{1 + e^{\frac{\epsilon}{m}}} \cdot (1 + e^{\frac{\epsilon}{m}}) = e^{\frac{\epsilon}{m}} \tag{4.2}$$

Similarly, if $b_i^k \neq b_j^k$ and $b_j^k = y_k$, we have $\frac{Pr(b_i^k=y^k)}{Pr(b_j^k=y^k)} = e^{-\epsilon/m}$. Then, we have $\forall k \in [1, m]$, $\frac{Pr(b_i^k=y^k)}{Pr(b_j^k=y^k)} \leq e^{\epsilon/m}$ (equals one of 1, $e^{\epsilon/m}$ and $e^{-\epsilon/m}$). Combining all m bits, we have:

$$\frac{Pr[\mathcal{A}(B_i) = y]}{Pr[\mathcal{A}(B_j) = y]} \leq e^{\epsilon} \tag{4.3}$$

Thus, the generated presence bit vectors satisfy ϵ-*Object Indistinguishability*. This completes the proof. □

Poor Utility. Although Algorithm 2 satisfies ϵ-*Object Indistinguishability*, the utility of synthetic video would be extremely low since the total number of frames in a video m can be thousands or more, and then the allocated budget for each frame would be negligible. It destroys the utility of random response (i.e., RAPPOR [29]). For instance, a vehicle occurs in 100 frames out of a 1000-frame video, then the privacy budget for each frame is $\epsilon/1000$, which makes the flipping probability close to 0.5. Then, each of the 1000 frames would have 50% probability to include the vehicle (and other vehicles), then the objects in the video are too random (extremely low utility at this time). Thus, we explore an alternative solution for the video data in Section 4.3.2 and 4.3.3.

4.3.2 Dimension Reduction in the Video. Recall that the limited utility in Algorithm 2 results from the high dimensions in the video (considering each frame as a dimension). Most existing LDP techniques (e.g., RAPPOR [29], succinct histogram [30], LDPMiner [107], PLDP [45]) have reduced the dimension (e.g., bloom filter reduces the bits dimension for RAPPOR, top k frequent items reduces the dimension of items in LDPMiner [107], Johnson-Lindenstrauss transform reduces the dimension of location data [45]). In videos, since difference between two consecutive frames is very small, we extract the key frames [108–110] out of m frames from $\mathcal{V}$ to reduce

dimension in VERRO.

Key Frame Extraction. In computer vision, many existing shot detection and key frame extraction algorithms have been proposed based on the boundary method [108], motion analysis [109], clustering [110], among others. Since algorithms based on clustering has been shown to generate more accurate results [110], we integrate it into VERRO for dimension reduction. The basic idea is to divide the video into several groups and the frames in same group are similar. The algorithm [111] first transforms each pixel RGB value to construct the HSV (hue, saturation, value) histogram for each frame, and then calculates the pixel distribution in terms of hue, saturation, value, respectively. Each cluster is initialized with a new frame, and expanded by adding new consecutive frames which are similar to the existing frames (measured by the HSV histograms). After the clustering, each cluster includes a group of consecutive frames, which can be considered as a segment of the video. Finally, a key frame can be extracted from each cluster/segment. The details are illustrated in Algorithm 3.

As a result, the key frame can be utilized to represent every segment. Then, the m-bit object presence vectors (for all the objects) can be reduced to ℓ-bit vectors. For instance, key frames $\mathcal{F}_1, \ldots, \mathcal{F}_\ell$ (where ℓ denotes the number of key frames, and $\ell \ll m$ in general) are extracted from $\mathcal{V}$. Object O_i's presence vector B_i can be reduced to $B_i' = (kb_i^1, \cdots, kb_i^\ell)$.

Random Response. After dimension reduction, random response can be implemented based on the RAPPOR framework [29] for each object. Each bit kb_i^k in ℓ-bit vector of object O_i is randomly flipped into 0 or 1 using the following rules:

Algorithm 3 Segmentation and Key Frame Extraction

initialize the first segment $S_1 = F_1$, segment index $i = 1$
equally partition H, S, V value ranges to h, s and v parts
for each frame $F_k, k \in [2, m]$ **do**
  for each part $\hat{h}, \hat{s}, \hat{v}$ in H, S, V **do**
    construct the histograms $H(\hat{h})$, $S(\hat{s})$, $V(\hat{v})$ in frame F_k
  end for
  $Sim_H(F_k, S_i) = \sum_{\hat{h}=1}^{h} \min\{H(\hat{h}), S_i[H(\hat{h})]\}$
  $Sim_S(F_k, S_i) = \sum_{\hat{s}=1}^{s} \min\{S(\hat{s}), S_i[S(\hat{s})]\}$
  $Sim_V(F_k, S_i) = \sum_{\hat{v}=1}^{v} \min\{V(\hat{v}), S_i[V(\hat{v})]\}$
  $\{\alpha, \beta, \gamma$: weights for H, S, V; similarity threshold: $\tau\}$
  if $(\alpha \cdot SI_H + \beta \cdot SI_V + \gamma \cdot SI_S) \geq \tau$ **then**
    $S_i \leftarrow S_i \cup F_k$
  else
    $i = i + 1$ and initialize a new segment S_i
    $S_i \leftarrow S_i \cup F_k$
  end if
end for
for each segment S_i **do**
  compute the maximum frame entropy $Entropy(F)$:
  max $\{-\alpha \cdot \sum_{\hat{h}=1}^{h}[H(\hat{h}) \log H(\hat{h})] - \beta \cdot \sum_{\hat{s}=1}^{s}[S(\hat{s}) \log S(\hat{s})] - \gamma \cdot \sum_{\hat{v}=1}^{v}[V(\hat{v}) \log V(\hat{v})]\}$
  extract the key frame with maximum entropy $\mathcal{F}_i$ in S_i
end for
return all the segments and key frames

$$kb_i^k = \begin{cases} kb_i^k, & \text{with the probability of } (1-f) \\ 1, & \text{with the probability of } \frac{f}{2} \\ 0, & \text{with the probability of } \frac{f}{2} \end{cases} \tag{4.4}$$

Theorem 4. *The random response (with rules in Equation 5.1)* $\ell \log(\frac{2-f}{f})$*-Object Indistinguishability.*

Proof. Again, object indistinguishability can be proven by following the proof of LDP [29]. Specifically, the RAPPOR [29] satisfies $2h \log(\frac{2-f}{f})$-LDP with the output size of the hash function in the bloom filter h and the flipping probability f. Maximum difference sizes are $2h$ between two input values. Thus, the random response (with rules in Equation 5.1) make ϵ equal to $\ell \log(\frac{2-f}{f})$ since size difference in any two presence vector is at most ℓ (by replacing the encoded bit vectors of bloom filter as the object presence vectors in RAPPOR [29]), which satisfies $\ell \log(\frac{2-f}{f})$*-Object Indistinguishability.* □

4.3.3 Optimizing RAPPOR for Object Presence. Although ℓ is far less than m, the number of key frames ℓ may still be large depending on the background scene(s), activity motion and light density. To solve this, we can further reduce the dimension by choosing a subset of key frames out of ℓ key frames to allocate the privacy budget. Indeed, determining whether each key frame is picked for allocating the privacy budget or not can be formulated as an optimization problem (*maximizing the utility of generating the synthetic video using the random object presence vectors in Phase II*).

Optimization Problem. For each key frame $\mathcal{F}_k, k \in [1, \ell]$, we define a binary variable $x_k \in \{0, 1\}, k \in [1, \ell]$ to represent if key frame $\mathcal{F}_k$ is picked for budget allocation or not. Then, the total number of picked key frames is referred as $\sum_{k=1}^{\ell} x_k$. Per the Theorem 4, we have the random response satisfies $\sum_{k=1}^{\ell} x_k \log(\frac{2-f}{f})$*-Object Indistinguishability.*

An example for dimension reduction, utility maximization and random response is given in Figure 4.3. Considering the n objects $O_1, \ldots, O_n$, after dimension

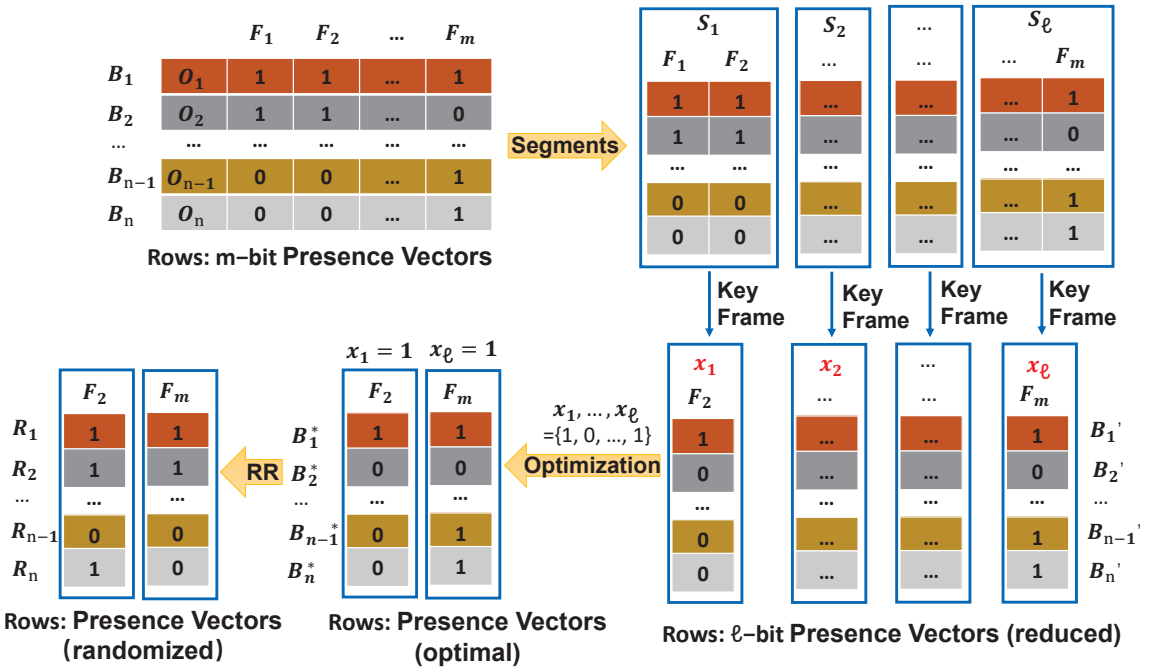


Figure 4.3. Dimension Reduction, Utility Maximization and Random Response

reduction, all the n object presence vectors are reduced to n different ℓ-bit vectors. Our goal is to accurately retain more objects in the video, thus we aim at minimizing the distance between $\forall i \in [1,n], B'_i$ (extracted from $\mathcal{V}$) and $\forall i \in [1,n], R_i$ (denoted as the ℓ-bit vectors by applying random response to $\forall i \in [1,n], B'_i$).

Specifically, since $\forall i \in [1,n], R_i$ are randomized bit vectors (the kth entry in all the vectors are 0 if $x_k = 0$), we should measure the difference between the expectation $\forall i \in [1,n], E(R_i) = E[(R_i^1, \ldots, R_i^\ell)]$ and $B'_i = (kb_i^1, \ldots, kb_i^\ell)$. We first learn the expectation of R_i^k (the kth entry in R_i). If $x_k = 0$, then $\forall i \in [1,n], R_i^k = 0$ hold. Thus, we have:

$$E(R_i^k) = x_k \cdot [Pr(R_i^k = 1) \cdot 1 + Pr(R_i^k = 0) \cdot 0] \tag{4.5}$$

There are two cases for R_i^k (in case of $x_k = 1$):

1. If $kb_i^k = 1$, per Equation 5.1, we have $E[R_i^k] = 1 \cdot [(1-f) \cdot 1 + \frac{f}{2} \cdot 0 + \frac{f}{2} \cdot 1]$.

2. If $kb_i^k = 0$, we have $E[R_i^k] = 1 \cdot [(1-f) \cdot 0 + \frac{f}{2} \cdot 0 + \frac{f}{2} \cdot 1]$.

Thus, the expectation can be summarized as following:

$$\begin{cases} E(R_i^k) = \frac{f}{2}, & \text{if } x_k = 1 \text{ and } kb_i^k = 0 \\ E(R_i^k) = 1 - \frac{f}{2}, & \text{if } x_k = 1 \text{ and } kb_i^k = 1 \\ E(R_i^k) = 0, & \text{if } x_k = 0 \text{ and } kb_i^k = 0 \text{ or } 1 \end{cases} \tag{4.6}$$

The objective function can be formulated as:

$$\min : \sum_{k=1}^{\ell} \sum_{i=1}^{n} |E(R_i^k) - kb_i^k| \tag{4.7}$$

Furthermore, for accurately interpolating the objects in different frames in Phase II, the number of key frames picked for each object should be no less than 2. Therefore, we formulate the optimization problem as below:

$$\begin{aligned} &\min : \sum_{k=1}^{\ell} \sum_{i=1}^{n} |E(R_i^k) - kb_i^k| \\ &s.t. \begin{cases} \forall i \in [1, n], 2 \leq \sum_{k=1}^{\ell} R_i^k \leq \ell, \\ \forall k \in [1, \ell], x_k \in \{0, 1\} \end{cases} \end{aligned} \tag{4.8}$$

Detailing expectation $E(R_i^k)$ with the flipping probability, the optimization problem can be converted to:

$$\min : \sum_{k=1}^{\ell} \sum_{i=1}^{n} |x_k \cdot |kb_i^k - \frac{f}{2}| - kb_i^k| \\ s.t. \begin{cases} \forall i \in [1,n], 2 \leq \sum_{k=1}^{\ell} R_i^k \leq \ell, \\ \forall k \in [1,\ell], x_k \in \{0,1\} \end{cases} \tag{4.9}$$

Complexity and Solver. Since f and $\forall k \in [1,\ell], \forall i \in [1,n], kb_i^k$ are constants, $\forall k \in [1,\ell], |kb_i^k - \frac{f}{2}|$ are constants. Then, Equation 4.9 is a binary integer programming (BIP) problem. Although solving the BIP problems can be NP-hard [112], we can approximately solve Equation 4.9 using linear programming (LP) since the objective function and the constraints are *linear*: (1) letting the binary variable $\forall k \in [1,\ell], x_k$ be continuous in $[0,1]$, (2) solving the problem using standard LP solvers (e.g., the Simplex algorithm), and (3) in the optimal solution of the LP problem, $\forall k \in [1,\ell]$, if $x_k \in [0,0.5)$, we assign $x_k = 0$; if $x_k \in [0.5,1]$ we assign $x_k = 1$ as the approximated optimal solution of the BIP problem.

Addressing Possible Privacy Leakage in Optimization. Compared to randomly picking a number of key frames for budget allocation, computing the optimal frames for budget allocation may result in some minor privacy leakage since the total number of objects in the kth key frame $\sum_{i=1}^{n} kb_i^k, k \in [1,\ell]$ (which is used in the optimization) might be different. Such privacy leakage is generally minor due to a small sensitivity Δ of the object count in each frame (e.g., $\Delta = 1$ for protecting the presence/absence of each object in every frame). Thus, it can be addressed by injecting a small amount of generic Laplace noise $Lap(\frac{\Delta}{\epsilon'})$ into $\sum_{i=1}^{n} kb_i^k, k \in [1,\ell]$ before formulating the optimization problem. Although adding such small amount of noise may slightly deviate the optimality, this could guarantee end-to-end indistinguishability (differential privacy). Since such privacy guarantee is well studied in literature [13], we do not discuss it in this paper due to space limitation.

4.3.4 Privacy Guarantee. After solving the optimization problem, as shown in Figure 4.3, each of the picked key frames will be allocated with a privacy budget $\epsilon/\sum_{k=1}^{\ell} x_k$. In the meanwhile, VERRO utilizes the optimal solution $\forall k \in [1, \ell], x_k\ n$ to derive the optimal presence vectors ($\sum_{k=1}^{\ell} x_k$-bit), denoted as $B_1^*, \ldots, B_n^*$. Next, random response is applied to $B_1^*, \ldots, B_n^*$ to generate output presence vectors $R_1, \ldots, R_n$.

Theorem 5. *Phase I satisfies ϵ-Object Indistinguishability.*

Proof. Phase I derives the presence bit vectors B_i^* and B_j^* for any two objects O_i and O_j after the optimization. Then, random response is applied to B_i^* and B_j^* and generate random vectors R_i and R_j. Per Theorem 4, Phase I satisfies ϵ-*Object Indistinguishability* where $\epsilon = \sum_{k=1}^{\ell} x_k \ln \frac{2-f}{f}$ (note that the privacy guarantee for utility maximization has been discussed in Section 4.3.3). □

It is worth noting that the presence of objects in the remaining $(m - \sum_{k=1}^{\ell} x_k)$ frames and the coordinates of the objects in all m frames in the synthetic video $\mathcal{V}^*$ will be generated in Phase II.

4.4 Phase II: Video Generation

In this section, we illustrate the details of Phase II.

4.4.1 Background Scene(s). As discussed in Section 4.2, video preprocessing includes detecting/tracking objects and background scene(s) extraction. While removing objects from digital images (e.g., each frame of a video), the pixels within the objects are missing in the frame and need to be reconstructed for the background scene(s). In VERRO, we utilize an efficient algorithm [106] to fill the blank area by considering both texture and structure.

First, the quality of the output image/frame highly depends on the order of filling different parts of the blank areas. The algorithm provides a filling strategy

by prioritizing them using the combination of the continuation of strong edges and high-confidence surrounded pixels. The priority is computed for every border patch, with distinct patches for each pixel on the boundary of the blank areas. Then, we always start filling at the border pixels with the highest priority.

Second, while filling the pixel p, the algorithm places it at the centroid of a patch with certain size (e.g., 3×3). Then, we traverse all the background pixels, and the centroid pixel of the most similar patch from the source background region will be filled in p, where the similarity is measured by the sum of squared errors. Some reconstructed background scenes are demonstrated in Section 3.7.

4.4.2 Randomly Generating Object Coordinates. Phase I generates indistinguishable presence information (in different frames) for all the objects. Next, we need to insert synthetic objects into the background scene (each frame) to generate the synthetic video $\mathcal{V}^*$. Specifically, we denote all the frames in the synthetic video $\mathcal{V}^*$ as $\{F_1^*, \ldots, F_m^*\}$, and the frames in $\mathcal{V}^*$ corresponding to the original key frames as $\{\mathcal{F}_1^*, \ldots, \mathcal{F}_\ell^*\}$. We then discuss different cases of generating coordinates for the objects in each frame.

(1) $R_i = \emptyset$**.** If all the entries in any object presence vector are 0, such random vector output R_i would result in object loss (the synthetic video will lose one object), and it is unnecessary to identify the coordinates for them in this case. We have evaluated such utility loss in Section 3.7, and most of the objects can be retained by VERRO in practice.

(2) $R_i \neq \emptyset$**.** If there exist at least one non zero entry in R_i, then an object will be inserted to the synthetic video $\mathcal{V}^*$. A critical and challenging question is that where to insert the object. We employ the coordinates of all the objects in the original video $\mathcal{V}$ as "Candidate Coordinates" to generate the coordinates in each frame of the

synthetic video.

Specifically, in each key frame of the synthetic video $\forall k \in [1, \ell], \mathcal{F}_k^*$, the number of objects inserted into key frame $\mathcal{F}_k^*$ is $\sum_{i=1}^{n} R_i^k$ (derived in Phase I). Denoting the number of objects in the kth key frame of $\mathcal{V}$ as $c_k, k \in [1, \ell]$ where $c_k = 0$ if $x_k = 0$, we thus have:

- *Sufficient candidate coordinates*: if $\sum_{i=1}^{n} R_i^k \leq c_k$, the number of required objects in $\mathcal{F}_k^*$ is no greater than the number of candidate coordinates in $\mathcal{F}_k$. Then, VERRO randomly picks $\sum_{i=1}^{n} R_i^k$ out of c_k candidate coordinates for $\sum_{i=1}^{n} R_i^k$ different objects in the background scene (frame $\mathcal{F}_k^*$). Please see the left example in Figure 4.4.

- *Insufficient candidate coordinates*: if $\sum_{i=1}^{n} R_i^k > c_k$, the number of required objects in $\mathcal{F}_k^*$ is greater than the number of candidate coordinates in $\mathcal{F}_k$. For instance, in the right example in Figure 4.4, we expand the set of candidate coordinates by adding the candidate coordinates in $\mathcal{F}_k$'s neighboring frames in the same segment. Then, VERRO randomly picks $\sum_{i=1}^{n} R_i^k$ out of c'_k candidate coordinates (c'_k is expanded from c_k where $c_k < \sum_{i=1}^{n} R_i^k \leq c'_k$) to insert $\sum_{i=1}^{n} R_i^k$ different objects into the background scene (frame $\mathcal{F}_k^*$).

After assigning coordinates to the key frames (where $R_i^k = 1$), we obtain at least 1 frame with the corresponding coordinates for any O_i (if the corresponding object is retained in the synthetic video) – the retained object has been assigned with coordinates in at least two frames in almost all the cases in our experiments in Section 3.7. With such randomly assigned coordinates in some key frames, we can interpolate the coordinates in other frames (out of m frames in total) between such key frames. For instance, given coordinates in two key frames F_1 and F_{10} for object O_i, then its coordinates between F_1 and F_{10} can be estimated. In literature,

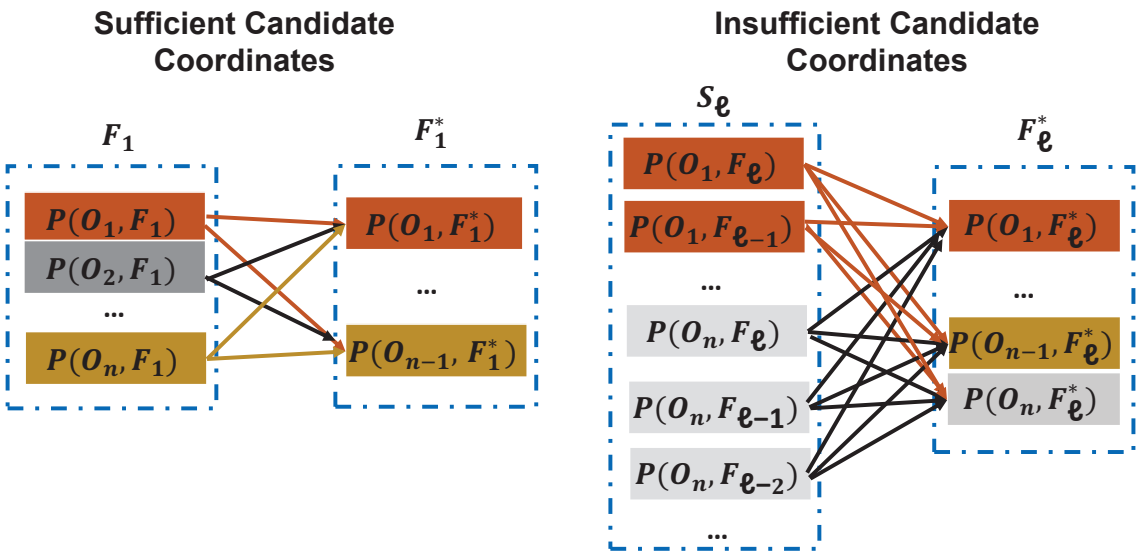


Figure 4.4. Random Coordinates Assignment (before Interpolation)

there are many interpolation methods for moving object trajectories data (e.g., linear interpolation [113], nearest neighbor interpolation [114], Lagrange interpolation [40]). In VERRO, we adopt the Lagrange interpolation to estimate such trajectories.

Finally, after interpolation, we define the first frame in which any object first occurs as "head" and the frame where such object last occurs as "end" in the interpolated trajectory. The head and end generally involve such object on the border of the frame. Thus, the interpolation terminates as each object's head and end are identified on the border of the frame (*objects do not occur in all the frames in general*).

Theorem 6. VERRO *(Phase I and Phase II) satisfies ϵ-Object Indistinguishability.*

Proof. Given any two objects O_i and O_j, their randomly generated presence vectors R_i and R_j are proven to be ϵ-*Object Indistinguishable* (after Phase I). We now examine the randomly assigned coordinates in the key frames and two full interpolated trajectories in the synthetic video $\mathcal{V}^*$.

Specifically, given any output presence vector y and any output trajectory $t = \{t_1, \ldots, t_m\}$ in $\mathcal{V}^*$, for simplicity of notation, we also denote the trajectories of O_i and O_j in $\mathcal{V}^*$ as $O_i = \{T_i^1, \ldots, T_i^m\}$ and $O_j = \{T_j^1, \ldots, T_j^m\}$, respectively.

$$\begin{aligned}&\frac{Pr[\mathcal{A}(O_i) = t]}{Pr[\mathcal{A}(O_j) = t]}\\ =&\frac{Pr[\mathcal{A}(B_i') = y]}{Pr[\mathcal{A}(B_j') = y]} \cdot \frac{Pr[\mathcal{A}(T_i^1) = t_1]}{Pr[\mathcal{A}(T_j^1) = t_1]} \cdots \frac{Pr[\mathcal{A}(T_i^m) = t_m]}{Pr[\mathcal{A}(T_j^m) = t_m]}\end{aligned}$$

On one hand, we have $\frac{Pr[\mathcal{A}(B_i')=y]}{Pr[\mathcal{A}(B_i')=y]]} \leq e^{\epsilon}$ (Phase I). On the other hand, if $\forall k \in [1, m], R_i^k = R_j^k = 1$, two objects are present in the same frame F_k (and F_k^*). In this case, since the same randomization is applied to O_i and O_j to pick the coordinates from the same set of candidates, we have $\forall k \in [1, m], Pr[\mathcal{A}(T_i^k) = t_k] = Pr[\mathcal{A}(T_j^k) = t_k]$. If $\forall k \in [1, m], R_i^k = R_j^k = 0$ (the coordinates are interpolated from the coordinates randomly assigned in the previous case [37]), we also have $\forall k \in [1, m], Pr[\mathcal{A}(T_i^k) = t_k] = Pr[\mathcal{A}(T_j^k) = t_k]$.

To sum up the above three cases, we have:

$$\frac{Pr[\mathcal{A}(O_i) = t]}{Pr[\mathcal{A}(O_j) = t]} \leq e^{\epsilon} \tag{4.10}$$

where $\epsilon = \sum_{k=1}^{\ell} x_k \log(\frac{2-f}{f})$, as analyzed in Theorem 4 and Section 4.3.3. This completes the proof. □

Therefore, we claim that any object in the input $\mathcal{V}$ can possibly generate any object in the synthetic video $\mathcal{V}^*$ (with random response in Phase I and random coordinates assignment in Phase II). For instance, the trajectory (in $\mathcal{V}^*$) closest to object O_1's original trajectory might be generated by object O_3.

4.5 Discussion

Distributed Framework: LDP techniques [24, 29, 30, 103] are deployed in distributed setting where each user perturbs its local data to share. Our object-based privacy model ensures indistinguishability at the object level where all the "distributed" local data can be perturbed by a "local agent" (aka. video owner) and shared as $\mathcal{V}^*$ to untrusted recipients.

Different video owners can also share their perturbed videos to any untrusted recipient (all the objects in each video are still well protected). Note that VERRO does not ensure video level indistinguishability (all the videos are indistinguishable). We will investigate the utility of the video level indistinguishability in practice and explore the LDP solutions in the future.

Noise Cancellation: in VERRO, objects and their trajectories are generated in the sanitized video. Thus, the individual noises resulted from random response for all the objects may not be directly canceled in the output video. Indeed, after random response and random coordinates assignment, there exists trajectories in the sanitized video which are close to the original trajectories (as shown in Figure 4.6-4.8 in our experiments). Also, such noise can be cancelled in data aggregation applications [45] (e.g., object counting, as shown in Figure 4.12 and 4.13).

Multiple Object Types: if any video includes multiple types of objects (e.g., pedestrians and vehicles), VERRO can generate the synthetic video for different types of objects, respectively. For instance, it first randomly generates pedestrians, and then randomly generates the vehicles. All the pedestrians are ϵ-*Object Indistinguishable* while all the vehicles are ϵ-*Object Indistinguishable*, assuming that it does not leak additional information across different object types (as all the objects have been replaced with random synthetic objects in the same type).

Protection for One-Object Video: VERRO can generate synthetic videos in which all the objects are ϵ-indistinguishable. In case that the video includes only one sensitive object, VERRO can also protect such object against re-identification. In existing LDP techniques [24, 29, 30, 103], if only one user perturbs its Object data and discloses it to the untrusted aggregator, the original data cannot be identified from its perturbed data. Similar to such works (e.g., RAPPOR [29], the objects and the trajectories cannot be identified from the perturbed presence in the synthetic video even if the adversary has arbitrary background knowledge on the presence of individuals at specific times.

Imperfect Background Scene(s): as discussed in Section 4.4, background scene(s) is extracted from the original video. The reconstructed scene may not be as perfect as the original frame (e.g., human/vehicle silhouette or duplicated/blurred region may occur). Thus, imperfect background scene(s) may leak some privacy about "there exists some object in the silhouette or blurred regions in the original video". However, adversaries cannot infer that "who is in that region or which object is in that region" since all the objects are indistinguishable from end to end.

System Deployment: the proposed VERRO can be implemented as an application, and deployed as a component to generate utility-driven synthetic videos by processing the videos captured by each camera (e.g., in the surveillance system, integrated with the traffic monitoring facilities, in smart phones or other mobile devices) where ϵ-*Object Indistinguishability* can be guaranteed.

4.6 Experiments

In this section, we present the performance evaluations.

4.6.1 Experimental Setup. We conduct our experiments on three real videos in the repository of multiple object tracking benchmark[2]. To benchmark the results, we

choose three pedestrian videos, two videos are captured by static cameras while the third video is recorded by a moving camera (where multiple background scenes are extracted):

1. MOT16-01 (people walking around a large square, denoted as "MOT01") [81]: 23 distinct pedestrians are sensitive objects in 450 frames (static camera).
2. MOT16-03 (pedestrians on the street at night, denoted as "MOT03") [81]: 148 distinct pedestrians are sensitive objects in 1,500 frames (static camera).
3. MOT16-06 (street scene from a moving platform, denoted as "MOT06") [81]: 221 distinct pedestrians are sensitive objects in 1,194 frames (moving camera).

Table 4.1. Characteristics of Experimental Videos

Video	Resolution	Frame #	Objects	Camera
MOT16-01	1920×1080	450	23	static
MOT16-03	1920×1080	1,500	148	static
MOT16-06	640×480	1,194	221	moving

We implement the detecting/tracking algorithm [35, 97] to identify all the objects (pedestrians). Objects are detected in each frame, and the same object is marked with the same ID in the entire video. Computer vision technique [106] is also utilized to extract/reconstruct the background scene(s) from the input video $\mathcal{V}$. All the programs are implemented in Python 3.6.4 with the OpenCV 3.4.0 library and tested on an HP PC with Intel Core i7-7700 CPU 3.60GHz and 32G RAM.

[2]https://motchallenge.net/

4.6.2 Generic Utility Evaluation. We first evaluate the utility of our synthetic videos. The proposed VERRO is a two-phase LDP approach. In Phase I, it randomly generates the object presence in all the frames of the synthetic video ("1" or "0"). In Phase II, we interpolate the trajectories. Thus, we evaluate two different types of utility: (1) the retained utility after Phase I (Random Response), and (2) the utility of synthetic video after Phase II.

Utility for Phase I. Phase I generates "presence bit vectors" for all the objects with frame dimension reduction, optimization ("OPT") and random response ("RR"). Some objects might not be included in the key frames, and/or might not be generated in the random response. Then, such objects cannot be generated in the synthetic video (all the entries in the corresponding vectors are 0) since they cannot be interpolated without any object presence in Phase I (also treated as noise). Thus, we evaluate the count of distinct objects (pedestrians) in Phase I.

First, Table 4.2 shows some results after detecting key frames for frame dimension reduction. In video MOT01, there are 22 key frames, and 19 out of 23 objects are present in the key frames. In video MOT03, 52 key frames are extracted, and 124 out of 148 objects are present in such key frames. In video MOT06, 191 out of 221 objects are captured in the identified 48 key frames. We can observe that frame dimension reduction results in less utility loss (retaining $\sim 80\%$ distinct objects).

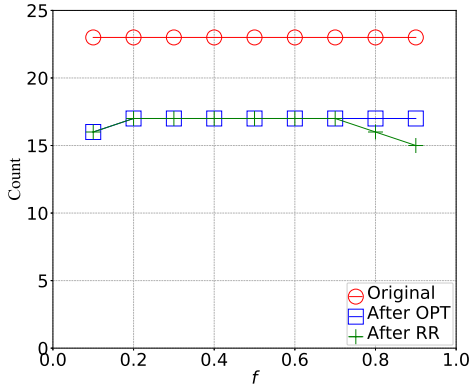
Table 4.2. Distinct Objects after Key Frame Extraction

Video	Frame #	Objects #	Key Frame #	Remaining #
MOT01	450	23	22	19
MOT03	1,500	148	52	124
MOT06	1,194	221	48	191

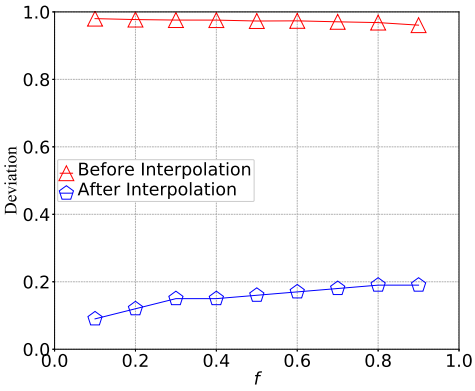
Figure 4.5(a), 4.5(c) and 4.5(e) present the count of distinct objects in original video, after optimization ("OPT"), and random response ("RR"). We set the flipping probability f from 0.1 to 0.9 for random response. In Figure 4.5(a), approximately 17 distinct objects can be retained in 10 key frames (optimized). f only slightly affects the optimization: the count of distinct objects increases a little bit as f grows. To evaluate how f affects the random response, we can observe that one or two objects are not randomly generated in RR as f grows to a large flipping probability (e.g., 0.8). This matches the fact that higher f results in worse utility in random response (Theorem 3) – such utility loss is indeed minor in our experiments. In addition, we can draw similar observations in Figure 4.5(c) and 4.5(e) where the utility loss of random response is even less for videos MOT03 and MOT06. Thus, Phase I retains a high percent of distinct objects via their random presence vectors, which means less side effect introduced by RR (this facilitates the interpolation in Phase II for boosting utility).

Utility for Phase II. Since the synthetic video generated in Phase II includes the synthetic objects at the same scene, the corresponding synthetic object of each original object (e.g., pedestrian) may have different coordinates in the same frame. All the coordinates in different frames may form a trajectory in the synthetic video. Thus, we also measure the deviation for the trajectories of all the objects in the original video and synthetic video: $\sum_{i=1}^{n}\sum_{k=1}^{m}\frac{P(O_i,F_k)-P(O_i,F_k^*)}{P(O_i,F_k)}$, where $P(O_i,F_k)$ and $P(O_i,F_k^*)$ are the center coordinates of object O_i in the kth frame of the input video and the synthetic video.

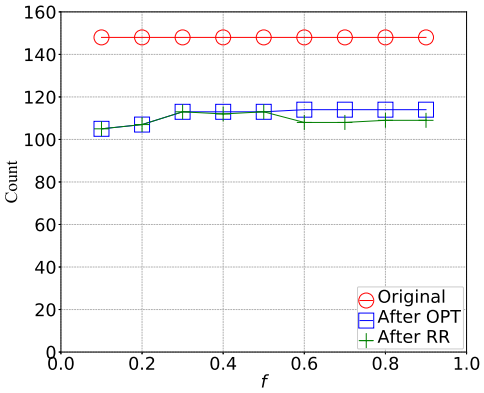
In Figure 4.5(b), 4.5(d) and 4.5(f), we can observe that the deviation before Phase II is higher than 0.9, since each object is only generated in a few frames. The deviation of trajectories increases as the flipping probability f gets larger since more flips occur more frequently (e.g., "0" to "1", or vice-versa). In such three figures,



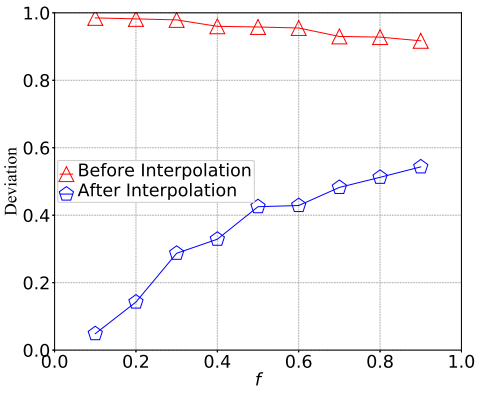
(a) Count of Distinct Ojects



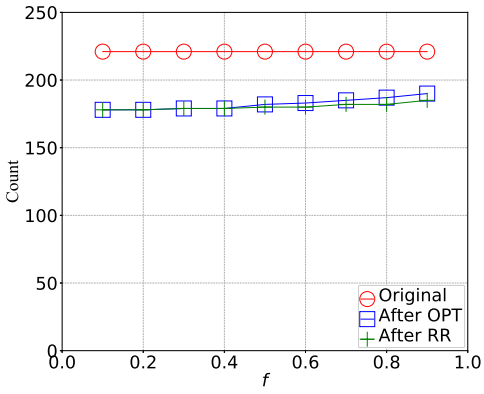
(b) Deviation of Object Trajectories



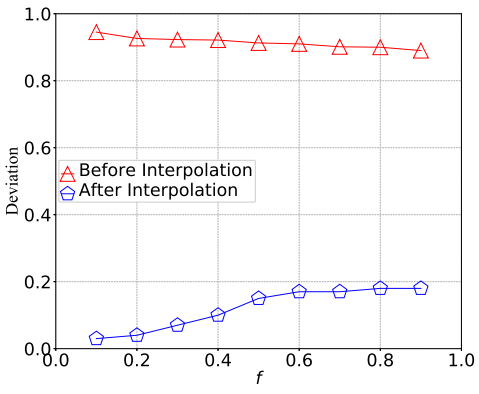
(c) Count of Distinct Ojects



(d) Deviation of Object Trajectories

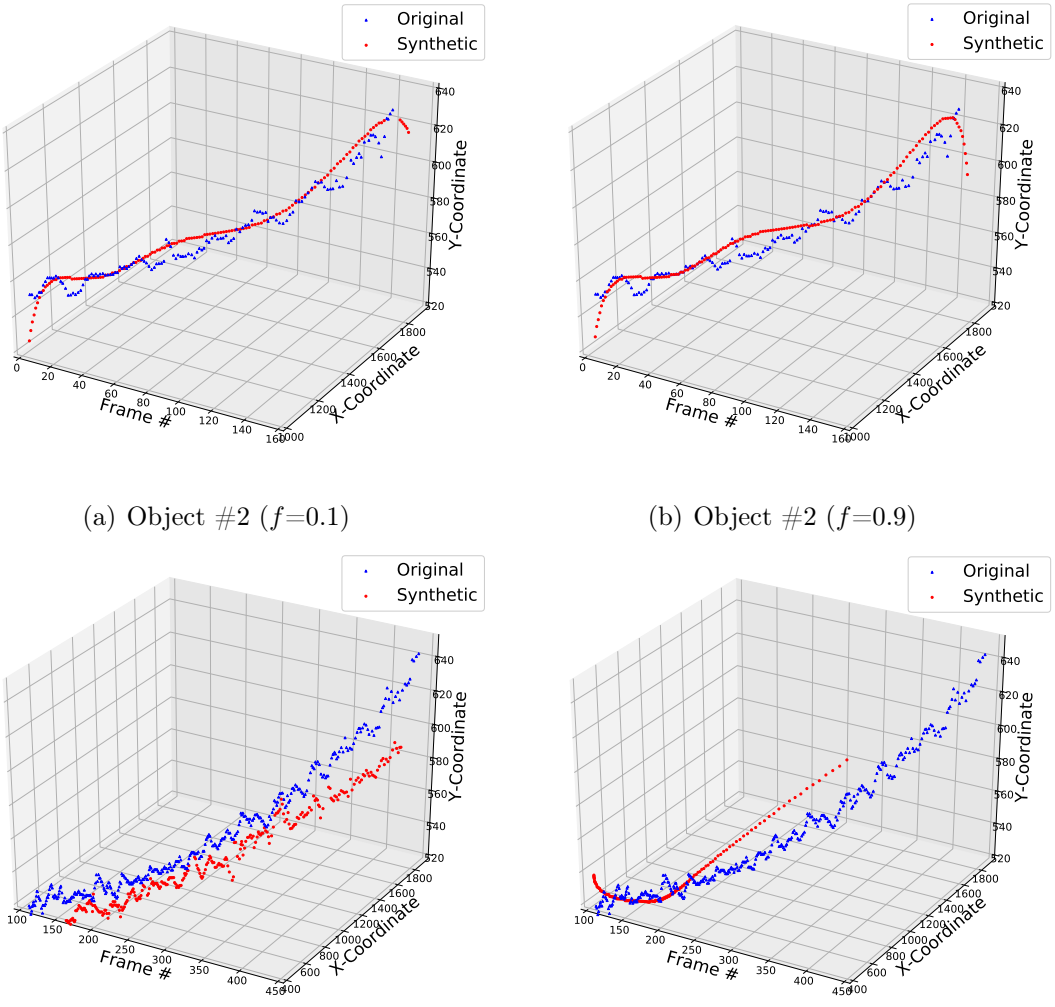


(e) Count of Distinct Ojects



(f) Deviation of Object Trajectories

Figure 4.5. Utility Evaluation of Phase I & II of MOT01 (MOT03 and MOT06)



(a) Object #2 (f=0.1)

(b) Object #2 (f=0.9)

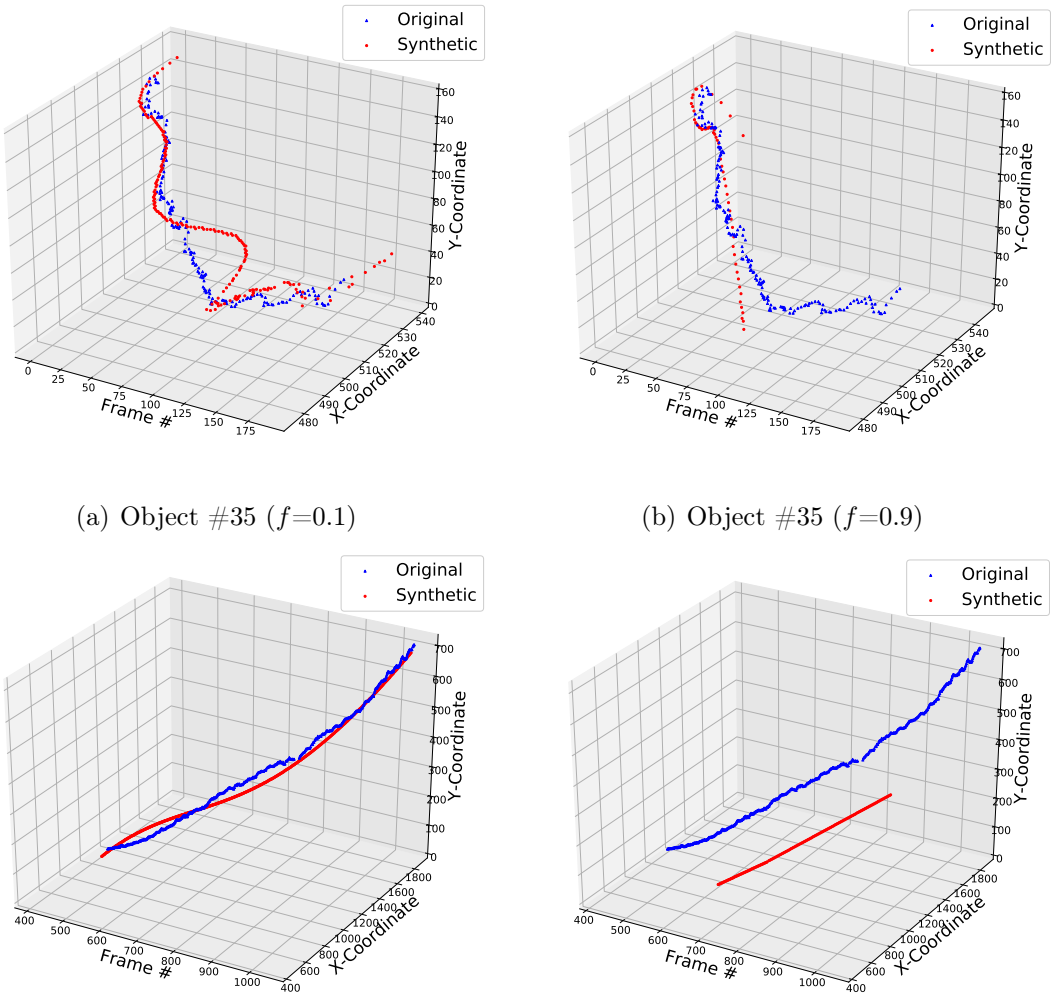
(c) Object #9 (f=0.1)

(d) Object #9 (f=0.9)

Figure 4.6. Trajectories of Two Randomly Selected Objects in MOT01

after Phase II, the deviation can be significantly reduced (e.g., in $[0.1, 0.2]$ for video MOT01, in $[0.02, 0.2]$ for video MOT06).

More specifically, we randomly select two objects (e.g., pedestrians) from each of the three videos, and extract their trajectories in the original video $\mathcal{V}$. In addition, we also extract their corresponding trajectories in the synthetic video $\mathcal{V}^*$. Figure 4.6, 4.7 and 4.8 demonstrate the trajectories of those objects in the input videos and synthetic videos, where 3-dimensional axes refer to the frame ID and coordinates



(a) Object #35 (f=0.1)

(b) Object #35 (f=0.9)

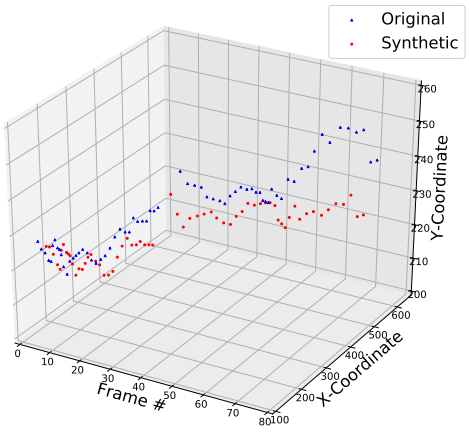
(c) Object #105 (f=0.1)

(d) Object #105 (f=0.9)

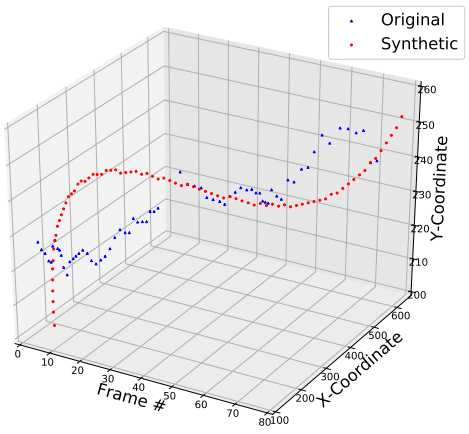
Figure 4.7. Trajectories of Two Randomly Selected Objects in MOT03

(X, Y) in videos. As $f = 0.1$, the trajectories of the objects lie closer to the original ones (compared to $f = 0.9$). It is worth noting that any object (pedestrian) in the original video can generate the corresponding trajectory of any object (e.g., the plotted trajectories corresponding to Object #2 and Object #9 in Figure 4.6). This is ensured by the ϵ-indistinguishable presence bit vectors randomly generated from all the objects in VERRO.

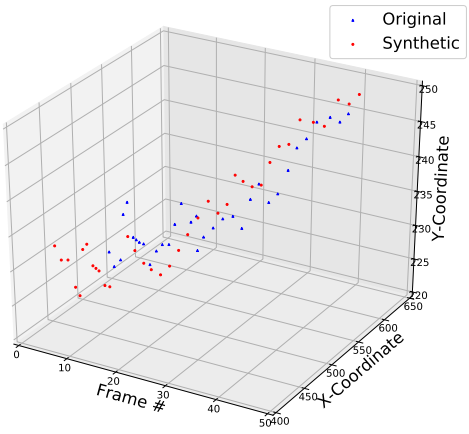
4.6.3 Visual & Aggregated Results. We also randomly pick a frame from each of



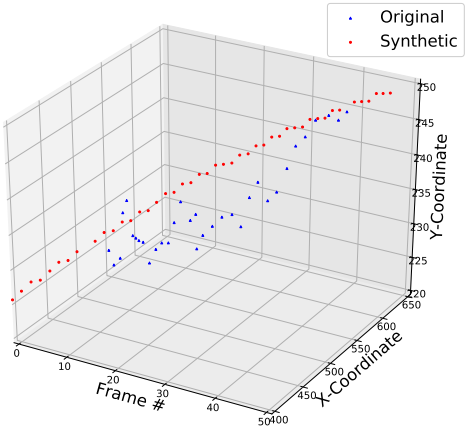
(a) Object #5 (f=0.1)



(b) Object #5 (f=0.9)



(c) Object #165 (f=0.1)



(d) Object #165 (f=0.9)

Figure 4.8. Trajectories of Two Randomly Selected Objects in MOT06

the three experimental videos, and present the generated background scenes and the corresponding frames in the synthetic videos. For video MOT01, Figure 4.9(a) shows the input frame and the detected objects in the frame. Also, we use a background interpolation algorithm [106] to fill the missing pixels (after removing all the detected objection), as shown in Figure 4.9(b). Similarly, a randomly picked frame (with the detected objects) and the generated background scenes in MOT03 and MOT06 are given in the first two subfigures of Figure 4.10 and 4.11. Some human silhouettes still exist in the background scenes. Clearly, the silhouettes cannot be associated to

(a) Frame 8 (b) Background Scene

(c) Synthetic Frame (f=0.1) (d) Synthetic Frame (f=0.9)

Figure 4.9. Representative Frames in MOT01 and the Generated Synthetic Video

any objects in the synthetic video (as shown in Figure 4.10(c), 4.10(d), 4.11(c) and 4.11(d)). This confirms the discussion for imperfect background scene in Section 3.6.

In the synthetic videos, we use different colors for different synthetic objects. Compare to $f = 0.1$ (shown in Figure 4.9(c), 4.10(c) and 4.11(c)), $f = 0.9$ would lead to more coordinates/trajectory deviation (as shown in Figure 4.9(d), 4.10(d) and 4.11(d)). However, accurate count of objects (pedestrians) can be retained in the synthetic frames even if the flipping probability f is specified as 0.9 (small privacy bound). Thus, we can still use such synthetic videos to function specific application

(a) Frame 134 (b) Background Scene

(c) Synthetic Frame (f=0.1) (d) Synthetic Frame (f=0.9)

Figure 4.10. Representative Frames in MOT03 and the Generated Synthetic Video

based on the count of objects, e.g., head counting and crowd density [34, 96]. To confirm such observation, we also detect and count all the pedestrians in each frame of the synthetic videos ($f = 0.1$ and $f = 0.9$).

Figure 4.12 shows the pedestrian counts in the (optimized) key frames (after Phase I). The aggregated result lies very close to the original result when f is small. When f goes larger, the aggregated result is slightly more fluctuated, and more objects are generated in the frames. Figure 4.13 demonstrates the aggregated counts of pedestrians in each frame (after Phase II). Note that many objects (with the coordinates outside the frames; not between the "head" and "end") are suppressed in Phase

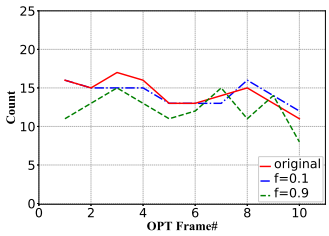
(a) Frame 216

(b) Background Scene

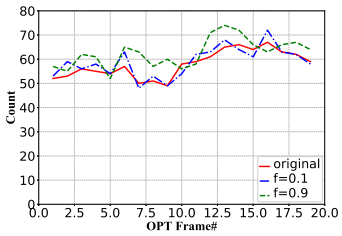
(c) Synthetic Frame (f=0.1)

(d) Synthetic Frame (f=0.9)

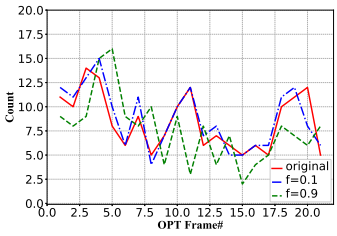
Figure 4.11. Representative Frames in MOT06 and the Generated Synthetic Video



(a) MOT01



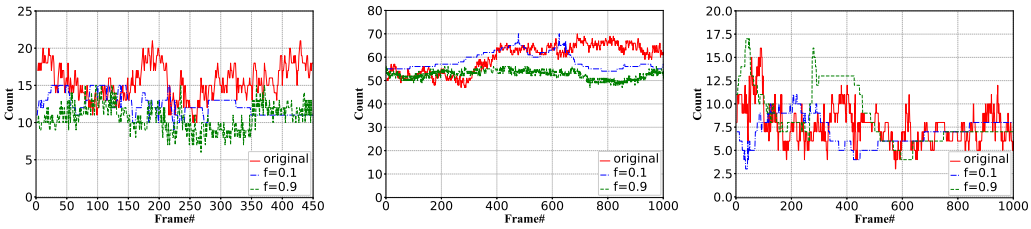
(b) MOT03



(c) MOT06

Figure 4.12. Object Counts in the Optimized Key Frames (by each frame)

II, making the object counts in different frames more accurate. Note that if multiple cameras capture more videos (e.g., surveillance or traffic monitoring cameras for the smart city) for joint analysis, the noise can be further cancelled in the applications.



(a) MOT01 (b) MOT03 (c) MOT06

Figure 4.13. Object Counts in the Synthetic Videos (by each frame)

4.6.4 Overheads. We evaluate the overheads of VERRO. Table 4.3 presents the runtime of the two phases and the required bandwidth for sending the synthetic videos to an untrusted recipient.

Table 4.3. Computational and Communication Overheads

Video	Phase I (Sec)	Phase II (Sec)	Bandwidth (MB)
MOT01	0.89	34.78	9.58
MOT03	1.56	36.12	16.6
MOT06	1.57	43.12	19.4

The computational cost increases as the count of distinct objects increases (MOT01 has the least pedestrians while MOT06 has the most pedestrians). The results reflect a *sublinear* increase trend, which enables VERRO to be scaled to generate synthetic videos for longer videos (with more frames). In addition, although MOT06 has a lower resolution (less pixels) than MOT01 and MOT03, it is captured by a moving camera. Since more background scenes have to be interpolated, it requires longer runtime (but still efficient). Note that the runtime for object detecting and background scene(s) generation (1-2 minutes in our experiments) can be considered as computational costs for preprocessing.

Finally, the communication overhead for sharing three synthetic videos is al-

most identical to the original video size.

4.7 Conclusion

Privacy concerns arise in considerable number of real world videos (e.g., individuals might be re-identified by the video recipients with their background knowledge). To the best of our knowledge, we take the first cut to pursue indistinguishability for objects in the video by defining a novel privacy notion ϵ-*Object Indistinguishability*. We propose a two-phase video sanitization technique VERRO that locally perturbs all the objects in the video and generates a utility-driven synthetic video with indistinguishable objects, which can be directly shared to any untrusted recipient.

In the synthetic videos, not only the object contents (e.g., different humans, vehicle make/model/color), but also their moving trajectories in the video (e.g., a series of coordinates) can be effectively protected since every synthetic object and its trajectory can be possibly generated from any object in the original video. Experiments performed on real videos have validated the effectiveness and efficiency of VERRO.

CHAPTER 5

L−SRR: LOCAL DIFFERENTIAL PRIVACY FOR LOCATION-BASED SERVICES WITH STAIRCASE RANDOMIZED RESPONSE

In the Chapter 4, I perturb the coordinates of objects in the video with the existing LDP mechanism to achieve object-level indistinguishability. In this Chapter, I will propose a novel distance-based LDP mechanism for general locations of location-based services to achieve user-level indistinguishable while preserving the utility. I will present the framework L−SRR in details, which includes the introduction, preliminaries, privacy model, framework, discussion and experiments [3].

5.1 Introduction

Location-based services (LBS) are widely deployed in mobile devices to provide useful and timely location-based information to users. For instance, WeatherBug provides weather information based on users' regions; Google Map not only navigates the routes with real-time traffic conditions but also responds to queries such as nearby restaurants or gas stations; Waze is similar to Google Map but actively collects extra information (e.g., accidents, road construction, and police) from users and shares them to other users. All of these LBS applications highly rely on the personal locations collected from millions of users. Such locations should be protected since visited places can be sensitive (e.g., hospital) or used to re-identify users from the data (e.g., a sequence of them can be unique). As a rigorous privacy model against arbitrary prior knowledge known to the adversaries, differential privacy (DP) has been extensively studied to address location privacy risks (e.g., [27]). It ensures that adding or removing any user's location or trajectory still generates indistinguishable results. However, recall that 87% of participants reported that they care about who accesses their location information in the 2011 Microsoft survey; over 78% workers of Amazon interviewed in 2014 still do not trust these apps on collecting their locations and be-

lieved apps accessing to their locations can pose significant privacy threats [28]. Thus, it is highly desirable to explore private location collection by an *untrusted* server.

Recently, *local differential privacy* (LDP) techniques [29,31,103,115] have been successfully deployed in industry (e.g., Google [29], Apple [116], and Microsoft [117]) to privately aggregate locally perturbed data. It provides stronger privacy against attackers with arbitrary background knowledge (not only the downstream analysts but also the data aggregator can be *untrusted*). To date, existing LDP schemes such as RAPPOR and generalized randomized response have been extended to privately aggregate different types of data, e.g., set-valued data [117], numerical data [118], video [2], and graphs [24]. In my previous work VERRO, I extended the existing LDP framework to video data [2]. However, existing LDP schemes are not very effective on private location data collection and analysis due to either limited utility or relaxed privacy protection. To our best knowledge, only [42, 43] applied existing LDP schemes to locations but the utility is still poor. Moreover, PLDP [45] relaxed LDP to personalized LDP (not every user can be protected with ϵ-LDP) in the location collection for spatial density estimation.

Furthermore, some other privacy-enhancing techniques [44,45] privately collect locations for LBS that provides services to individual users (e.g., GPS navigation [5], and nearest point-of-interest (POI) search [46]) *without a trusted server*. For instance, geo-indistinguishability [44] adds Laplace noise to the user's location for ensuring privacy in LBS. However, it cannot strictly satisfy LDP (the locations are indistinguishable only within a radius), and the Laplace mechanism has been shown to be worse than randomized response for local perturbation [119].

To address such limitations, we propose the first strict LDP framework (namely, "L−SRR") to support a variety of LBS applications. First, we design a novel LDP mechanism "*staircase randomized response* (SRR)" and revise the empirical estima-

tion to privately aggregate locations with significantly improved utility and strictly satisfied ϵ-LDP. Second, different from all existing works [42–45], we design additional components (e.g., private matching [46], and private information retrieval [47]) into L−SRR to ensure ϵ-LDP for a variety of LBS applications such as k nearest neighbors search [48], origin-destination analysis [49], and traffic-aware GPS navigation [5], which may collect user trajectories or perform individual services with the aggregated locations/trajectories.

5.2 Preliminaries

5.2.1 LBS Applications. We first categorize two different types of LBS Apps:

Location-Input LBS: LBS App collects a single location from each user, and the untrusted server privately analyzes the aggregated data, e.g., identifying the top crowded areas [120], and spatial density estimation [45]. In some LBS Apps, the clients may query the analysis results from the server (e.g., location-based advertising [121], and k nearest point of interests (POIs) for each user [48]).

Trajectory-Input LBS: LBS App collects multiple sequential locations (trajectory) from each user, and the untrusted server privately analyzes the aggregated data, e.g., aggregating users' origin-destination (OD) pairs to learn the traffic flow [49, 122]. Similarly, users may query the analysis results computed by the server, e.g., users query the real-time traffic for the GPS navigation [122].

5.2.2 Privacy Model. Users in L−SRR will locally randomize their location(s) [30] with algorithm $\mathcal{A}$ and send the noisy results to the untrusted server. After local perturbation, all the input locations can be indistinguishable [29]. The privacy notion is formally defined as below:

Definition 5 (ϵ-LDP)**.** *A randomization algorithm $\mathcal{A}$ satisfies ϵ-Local Differential*

Privacy, if and only if for any pair of input locations $x, x' \in \mathcal{D}$, and for any perturbed output $x_r \in range(A)$, sent to the untrusted server, we have: $Pr[A(x) = x_r] \leq e^{\epsilon}$.

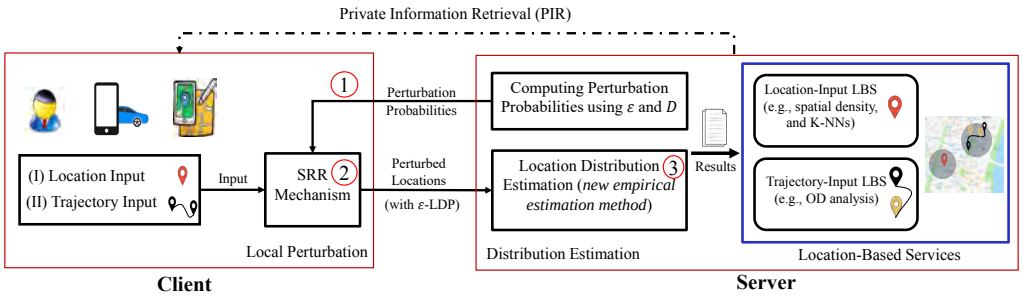


Figure 5.1. The L−SRR framework

5.2.3 L-SRR Framework. As shown in Figure 5.1, we design three major components in L−SRR: perturbation (by client), analysis (by server), and private retrieval (by both client and server only when the user needs to privately query the analysis results, e.g., traffic-aware GPS navigation):

1. **Perturbation (client)**: Each user's location data (location or trajectory) is locally perturbed by the client with ϵ-LDP. DRR optimizes the utility after hierarchically encoding the location domain $\mathcal{D}$. Encoding and optimal perturbation probabilities are pre-computed by the server (only based on ϵ and $\mathcal{D}$) to ensure ϵ-LDP. See details in Section 5.3.2.

2. **Analysis (server)**: Before perturbations, the server share the pre-computed perturbation probabilities with all the clients. After receiving the perturbed user locations, the untrusted server estimates the *location distribution* with a revised empirical estimation method. Then, the server loads such results into specific LBS (along with the required components) to privately derive the analysis result. See details in Section 3.3.

3. **Private Retrieval (only for LBS with client queries)**: Each user privately queries his/her result (e.g., nearby traffic) from the analysis results at the server side with a private information retrieval (PIR) protocol [123]. Server does not know which result is delivered to which user, and each user does not know other users' results either.

User Requirements. L−SRR can be deployed as a privacy preserving API in each LBS App. Users only need to periodically update the privacy bound ϵ and the location domain $\mathcal{D}$ with the server. In each LBS, users only need to locally perturb their location(s) with the pre-computed perturbation probabilities, and send the result to the server. The integrated PIR [47] also requires very minor computation and communication overheads.

LDP Protection. Similar to existing LDP models [29, 103], L−SRR ensures strong privacy against inferences on users' local data based on arbitrary background knowledge, which is orthogonal to mitigating other types of risks (e.g., encryption [124] and defenses against side-channel attacks [125]). Thus, L−SRR can be integrated with them to further improve security and privacy if necessary.

5.3 L-SRR for Location-Input LBS

In this section, we design the SRR mechanism to privately collect a location from each user for analysis (standard LDP setting [45, 103]).

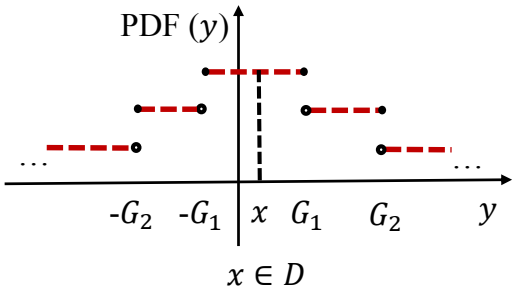
5.3.1 Staircase Randomized Response. We first review a family of LDP mechanisms. Randomized Response (RR) based schemes, such as generalized randomized response (GRR) [126] and unary encoding (UE) [103], satisfy ϵ-LDP. For instance, in GRR, given the domain size $d = |\mathcal{D}|$, privacy bound ϵ, and input $x \in \mathcal{D}$, the true value has a higher probability to be sampled (output y). The following perturbation probabilities $q(y|x)$ ensure ϵ-LDP.

$$\texttt{GRR}: \; q(y|x) = \begin{cases} \frac{e^{\epsilon}}{d+e^{\epsilon}-1}, & \text{if } y = x \\ \frac{1}{d+e^{\epsilon}-1}, & \text{otherwise} \end{cases} \tag{5.1}$$

Also, Hadamard Response (HR) [50] has a subset domain for each value x and a higher probability for values in the subset to be sampled. Then, the remaining values in the domain are sampled with a smaller probability. However, *only two different perturbation probabilities* are defined in the existing LDP mechanisms (e.g., GRR [126], UE [103], and HR [50]), not sufficiently fine-grained to optimize the utility (since the perturbation probabilities simply treat all the other output locations in the domain equally).

Thus, we propose a novel Staircase Randomized Response (SRR) mechanism for locations and LBS. Intuitively, if the probabilities for locations that are closer to the input location x can be higher, it is more possible for users that the query results of the LBS are the same. To this end, SRR will first consider the location distances to the input location x. Then, a set of fine-grained probabilities should be pre-computed for all the possible output locations $y \in \mathcal{D}$.

When pre-computing these probabilities, there are several issues in practice. For instance, for each input location x, if we compute the probability $q(y|x)$ for each possible output $y \in \mathcal{D}$, the number of probabilities is the domain size d. Then, $\forall x \in \mathcal{D}$, there are d probabilities for each location x and $d \times d$ different probabilities for all the locations in the domain. Thus, there are d^2 unknown probabilities to be determined, which makes it time-consuming to derive the optimal probabilities [127] and not extensible if the domain is updated. Second, general objective function (e.g., the variance) to optimize the perturbation probabilities is dependent on the unknown true frequencies. To address this, output locations can be partitioned into different groups in terms of their distances to x (*the probabilities of all the output locations in*



(b) SRR mechanism (for L−SRR)

Figure 5.2. Probability density function (PDF) for GRR and SRR

The probability density functions (PDFs) of GRR (w.l.o.g.) and SRR are illustrated in Figure 5.2. It is worth noting that the Figure 5.2 is the 1-D representation of the 2-D discrete locations in the domain. In GRR, the probability that outputs the true value (the point in Figure 5.2(a)) is higher than other values. On the contrary, since SRR discretizes the perturbation probabilities for all the grouped possible output locations, the PDF of SRR has a similar shape to the staircase mechanism in differential privacy [128], which also has a staircase PDF for different groups to satisfy ϵ-DP. Motivated by that, we name our new randomization mechanism as the "Staircase Randomized Response" (SRR) in local differential privacy. We formally define the perturbation probabilities from input x to all the output locations as follows.

Given the domain $\mathcal{D}$, for any input $x \in \mathcal{D}$, all the possible output locations can be partitioned into m groups $G_1(x), ..., G_m(x)$ based on their distances to x.[3] Notice that, the partitioning $G_j(x)$ is dependent on the input location x. For each input location x, all its m location groups and the perturbation probabilities (for perturbing x to any output location y) will be efficiently computed as:

[3]W.l.o.g., the distances from x to locations in $G_j(x)$ are farther if j is larger. The closest group is $G_1(x)$ whereas the farthest group is $G_m(x)$.

$$\texttt{SRR} : \forall x \in \mathcal{D}, q(y|x) = \begin{cases} \alpha_1(x), & \text{if } y \in G_1(x) \\ \vdots \quad \vdots & \vdots \\ \alpha_m(x), & \text{if } y \in G_m(x) \end{cases} \tag{5.2}$$

where $\alpha_1(x), ..., \alpha_m(x)$ are the distance-based perturbation probabilities for locations in m different groups perturbed from $x \in \mathcal{D}$, and the gap between the perturbation probabilities in every adjacent groups is the same ("Staircase PDF") in $\alpha_1(x), ..., \alpha_m(x)$.

Also, the sum of all the perturbation probabilities for each input location x should satisfy: $\sum_{j\in[1,m]} \sum_{y\in G_j(x)} q(y|x) = 1$. The details for computing the probabilities will be given in Section 5.3.3. SRR generates more accurate locally perturbed locations than the state-of-the-art LDP mechanisms with only two perturbation probabilities (e.g., GRR [126] and HR [50]), as validated in Section 3.7.

5.3.2 Data Encoding and Domain Partitioning. Hierarchical Location Encoding: To encode the location data, we use a hierarchical encoding scheme based on the Bing Map Tiles System [129], which recursively partitions geo-coordinates into 4 blocks, and indexes all the locations to reach the desired resolution [27]. Then, the locations are encoded into bit strings by hierarchically concatenating the indices of all the levels for every specific location. Figure 5.3 illustrates an example for the encoding. Specifically, starting from the root node, at each level h, the 4 children of each node (four sub-blocks) can be encoded by $00, 01, 10, 11$ (2-bit), and thus form 4^h blocks for indexing locations. Then, we can derive the encoded bit string by concatenating the bits from the first level to the leaf node level. For all the locations on the earth, h can be as large as 23 (46 bits for a location) to index each 4.7m×4.7m region. As a result, all the locations can be encoded with the same length of bits if the same precision (h) is applied to all the locations.

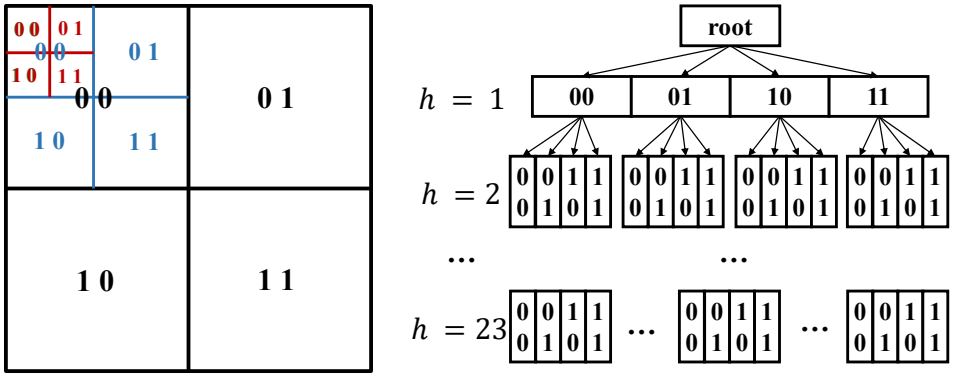


Figure 5.3. Hierarchical encoding for locations

Example 2 (Encoding for "New York"). *The coordinates of the center of "New York" are* $(40.730610, -73.935242)$. *Given* $h = 23$, *the location is encoded as "e1147b6afff" (hex of the bit string).*

Location Groups: With hierarchical encoding for the location domain $\mathcal{D}$, the distance between any two locations $x, x' \in \mathcal{D}$ can be directly measured by the longest common prefixes (LCP) of their encoded bit strings. Then, given a location x and any of its output groups $G_j(x), j \in [1, m]$, we define the LCP of the group.

Definition 6 (Group LCP). *Given an input location* x *and any of its groups* $G_j(x), j \in [1, m]$, *the group LCP (aka. GLCP) is the shortest LCP between the input location* x *and* $\forall y \in G_j(x)$. *The length of GLCP for group* $G_j(x)$ *is denoted as* $\beta_j(x)$.

Thus, the distance between the input location x and each location group $G_j(x), j \in [1, m]$ can be measured by the length of its GLCP $\beta_j(x)$: the larger, the closer. Then, we can partition all the output locations into groups using the GLCP lengths. In each group $G_j(x)$, all the locations share a prefix with at least $\beta_j(x)$ bits with location x (applying such rule for partitioning could reduce the complexity of partitioning to $O(d)$ though not optimal). For the group with a longer GLCP shared with the input location x, higher probabilities will be assigned to them (for perturbing

x).[4]

Location Partitioning: We next partition the locations into m groups for each input $x \in \mathcal{D}$, and assign the same perturbation probability to all the locations in the same group. Specifically, for m groups, we define a GLCP length vector $\{\beta_1(x), \ldots, \beta_m(x)\}$. All the encoded locations in group $G_j(x), 1 \leq j \leq m$ share *at*

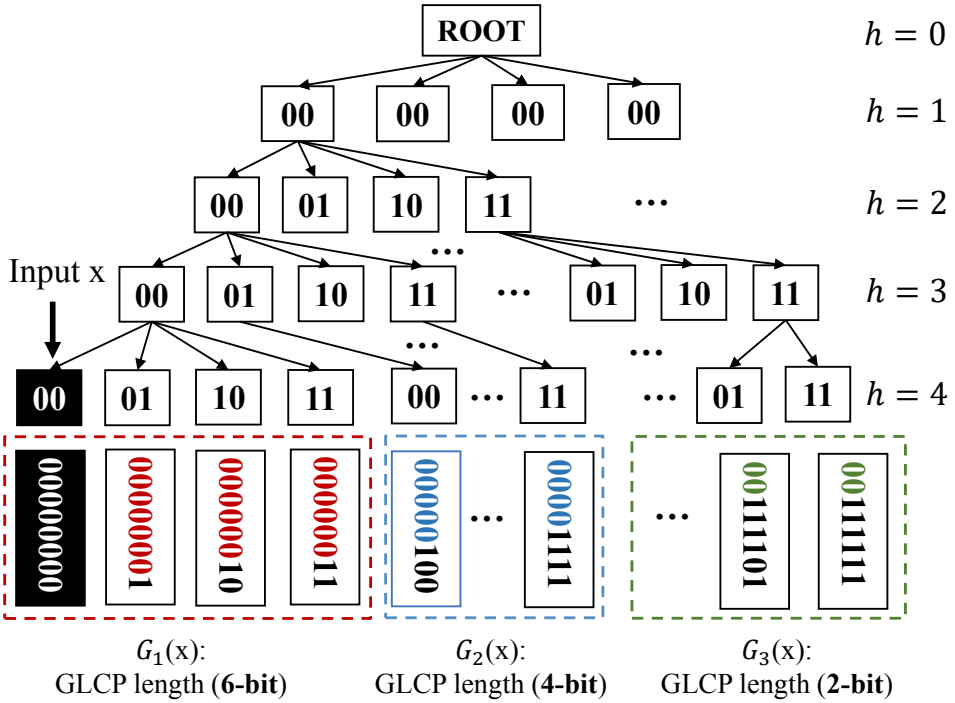


Figure 5.4. Example of location domain partitioning

Figure 5.4 shows an example for partitioning the location domain. Given the input location x, all the locations are partitioned into three groups with the GLCP lengths $\{\beta_1(x) = 6, \beta_2(x) = 4, \beta_3(x) = 2\}$ where $m = 3$. In $G_1(x)$, $G_2(x)$ and $G_3(x)$, all the locations share at least 6-bit, 4-bit and 2-bit prefix with x, respectively. Thus,

[4]In SRR, every input location x will be only perturbed to another location y in the domain $\mathcal{D}$ (rather than an arbitrary location on the map).

given any GLCP length vector $\beta_1(x), \ldots, \beta_m(x)$, the m groups $G_1(x), \ldots, G_m(x)$ for the input location x can be efficiently generated with complexity $O(d)$. Then, denoting the LCP between input x and output y as $LCP(x, y)$, the optimal $\{\beta_1(x), \ldots, \beta_m(x)\}$ and the m groups that maximizes $\sum_{\forall y \in \mathcal{D}} LCP(x, y)$ can be derived.

More specifically, if m is not large, we can traverse all the GLCP lengths $\{\beta_1(x), \ldots, \beta_m(x)\}$ where $\beta_1(x) > \beta_2(x) > \cdots > \beta_m(x)$ to find the optimal result. Otherwise, the server can apply a meta-heuristic algorithm (e.g., simulated annealing [130]) to derive a near-optimal $\{\beta_1(x), \ldots, \beta_m(x)\}$ for partitioning. Next, the location domain $\mathcal{D}$ can be efficiently partitioned by the optimal $\{\beta_1(x), \ldots, \beta_m(x)\}$. First, locations sharing a $\beta_1(x)$-bit or longer prefix with x will be assigned to $G_1(x)$; second, the locations sharing a prefix (length between $\beta_2(x)$-bit and $(\beta_1(x) - 1)$-bit) with x will be assigned to $G_2(x)$; repeat the above until $G_m(x)$ is formed.

Offline Computation: Since the optimization and partitioning are solely based on the domain $\mathcal{D}$, they can be executed offline and periodically updated with $\mathcal{D}$ by the server in `L-SRR`. In general, the location domain is stored in the server of companies and released as public knowledge for users (e.g., Google Maps) and these companies will take about several days to update the domains since these companies have to verify locations before making the changes available to the public. Then, for each $x \in \mathcal{D}$, the perturbation probabilities for all the m output location groups $\alpha_1(x), \ldots, \alpha_m(x)$ can also be derived offline (see Section 5.3.3). This is consistent with other LDP schemes [29, 103, 117].

5.3.3 Optimal Perturbation Probabilities. Recall that the possible output locations can be partitioned into m groups based on their distances to input and the PDF similar to the staircase mechanism [128] in differential privacy. We define the perturbation probabilities from input x to all the output locations as follows. Given any two output locations y and y' in any two neighboring groups $y \in G_j(x)$ and

$y' \in G_{j+1}(x)$, we have probability $q(y|x) = q(y'|x) + \Delta(x)$ where the step $\Delta(x) \in [0, 1)$ is the constant probability difference for any two neighboring groups of input x. Compared to the staircase mechanism in differential privacy which aims to the unbounded domain (entire real line or the set of all integers) and these probabilities are geometric sequence to maintain ϵ-DP, for the bounded location domain, the probabilities in the L−SRR follow a linear sequence. Note that the perturbation probability from the given input location x to output location y decreases as y moves to further groups (larger j).

Denoting $\alpha_{max}(x)$ and $\alpha_{min}(x)$ as the max and min probabilities in $\alpha_1(x), ..., \alpha_m(x)$, we have $\alpha_{max}(x) = \alpha_1(x)$ and $\alpha_{min}(x) = \alpha_m(x)$. In SRR, for all the input locations $x \in \mathcal{D}$, we specify a constant $c \geq 1$ as the ratio $\frac{\alpha_{max}(x)}{\alpha_{min}(x)}$. Thus, we have:

$$\Delta(x) = \frac{\alpha_{max}(x) - \alpha_{min}(x)}{m - 1} = \frac{\alpha_{min}(x) \cdot (c - 1)}{m - 1} \tag{5.3}$$

For each $x \in \mathcal{D}$, the sum of the perturbation probabilities of all the output locations is 1. Given the differences of perturbation probabilities for output locations in different groups in Equation 5.3 and the number of output locations in each group, all the perturbation probabilities can be derived, including $\alpha_{max}(x)$ and $\alpha_{min}(x)$:

$$\begin{aligned} \alpha_{min}(x) =& \frac{m - 1}{(m - 1)d \cdot c - (c - 1)\sum_{j=2}^{m}[(j - 1) \cdot |G_j(x)|]} \\ \alpha_{max}(x) =& \alpha_{min}(x) \cdot c \end{aligned} \tag{5.4}$$

where d is the location domain size and $|G_j(x)|$ is the size of group $G_j(x)$. Notice that, different $\alpha_1(x), \ldots, \alpha_m(x)$ will be derived for different input location x since the group sizes $\forall j \in [1, m], |G_j(x)|$ might be different for different x. Thus, the privacy upper bound ϵ can be computed (for any two input locations $x, x' \in \mathcal{D}$).

Theorem 7. *Staircase randomized response (SRR) satisfies ϵ-local differential privacy, where*

$$\epsilon = \max_{x,x' \in \mathcal{D}} \log(c \cdot \frac{(m-1)d \cdot c - (c-1)\sum_{j=2}^{m-1}[(j-1) \cdot |G_j(x)|]}{(m-1)d \cdot c - (c-1)\sum_{j=2}^{m-1}[(j-1) \cdot |G_j(x')|]})$$

Proof. See details in the Appendix A.4.2. □

For each input location $x \in \mathcal{D}$, the groups $G_1(x), \ldots, G_m(x)$ are constants if m and $\mathcal{D}$ are specified (as discussed in Section 5.3.2). Thus, given the value of c, we can derive a constant ϵ as a strict privacy upper bound for the LDP guarantee.

Selecting c for ϵ-LDP. Since ϵ is positively correlated to c, for any desired ϵ-LDP, the required c can be uniquely calculated using ϵ, $\mathcal{D}$ and m (see the relationship between ϵ and c in Figure 5.5(a)). Then, all the perturbation probabilities $\alpha_1(x), \ldots, \alpha_m(x)$ for all the input locations $x \in \mathcal{D}$ can be derived and made available to the users.

Optimal m with Mutual Information. In practice, both the server and clients do not know the data distribution before collecting them. Hence, it is critical to learn that the optimal m is also *independent of input data* and ensure good utility for all possible location data distributions in the SRR mechanism. To this end, we will optimize m for location domain partitioning with the mutual information [131, 132] between the input x and output y, which can measure the mutual dependence between them. As mutual information varies for different distributions, the maximum mutual information can cover all the cases (since the mutual dependence of any case would not violate such dependence [118]). Thus, the optimal m can be derived by the upper bound of mutual information for all the distributions [118, 133]. Specifically,

the mutual information between x and y is expressed by the difference between the differential entropy and conditional differential entropy of x and y [118]:

$$I(X,Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) \tag{5.5}$$

where $H(\cdot)$ is the entropy function. X and Y are the input and output random variables representing the input and output, respectively. Since no prior knowledge on the input data, it considers the distribution of y as uniform distribution U to maximize the mutual information (the output y is the random sampling result) [134]. $H(U)$ is an upper bound for any possible input distribution [131]. Thus, we have:

$$I(X,Y) \leq H(U) - H(Y|X) \tag{5.6}$$

where $H(U) = \log d$. The conditional differential entropy $H(Y|X)$ can be computed as below:

$$\begin{aligned} H(Y|X) &= -[\sum_{j=1}^{m} |G_j(x)| \cdot \alpha_j(x) \cdot \log \alpha_j(x)] \\ &\geq -d \cdot \alpha_{min}(x) \log \alpha_{max}(x) \end{aligned}$$

Thus, $H(Y|X)$ is lower bounded by $-d \cdot \alpha_{min}(x) \log \alpha_{max}(x)$ for $\alpha_1(x), \ldots, \alpha_m(x)$. Finally, the upper bound of mutual information can be expressed with the number of groups m:

$$I(X,Y) \leq \log d - H(Y|X) \leq \log d + d \cdot \alpha_{min}(x) \log \alpha_{max}(x)$$

We then explore the optimal m based on the mutual information metric. Since the smaller mutual information between two variables indicates more independence

between them, and the mutual information on m for LDP is convex (as proven in the Appendix A.4.1), the optimal m can be computed by making the derivation of the upper bound to 0 which is equal to minimize the mutual information bound.

Lemma 1. *The optimal m to minimize the mutual information bound is*

$$m = \frac{2 \cdot (c \cdot d - e^{1+\log c})}{(c-1) \cdot d} \tag{5.7}$$

Proof. The mutual information bound is $\log d + d \cdot \frac{m-1}{(m-1)\cdot c \cdot d - R} \cdot \log \frac{c(m-1)}{(m-1)\cdot c \cdot d - R}$ where $R = (\sum_{j=2}^{m}\{(j-1)\cdot|G_j|\}) \cdot (c-1)$ is a part of $\alpha_{min}(x)$ (see Equation 5.4). We can see that R is also determined by m. If $|G_1| \neq |G_2| \neq \cdots \neq |G_m|$, R non-differentiable (discrete). To solve this, we consider the worst case: assuming group size d and R is replaced with $R_{max} = (\sum_{j=2}^{m}\{(j-1)\cdot d\}) \cdot (c-1)$ (relaxed). The mutual information bound can be derived as below:

$$\begin{aligned} & [\frac{m-1}{(m-1)\cdot c \cdot d - R_{max}} \cdot \log \frac{c(m-1)}{(m-1)\cdot c \cdot d - R_{max}}]' \\ = \ & (\log \frac{m-1}{(m-1)\cdot c \cdot d - R_{max}} + \log c + 1) \cdot \frac{(m-1)\cdot R'_{max} - R_{max}}{[(m-1)\cdot c \cdot d - R_{max}]^2} \end{aligned}$$

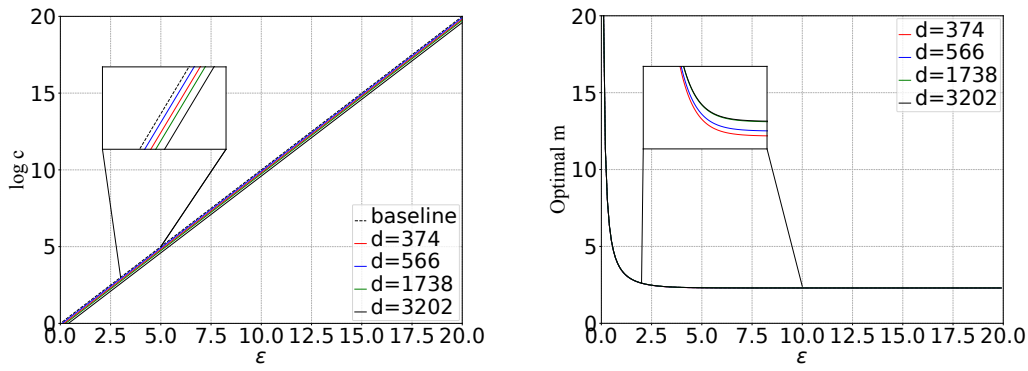
Due to $R_{max} = (\sum_{j=2}^{m}(j-1)\cdot d) \cdot (c-1)$, we have:

$$R_{max} = (c-1)\cdot d \cdot \frac{m^2-m}{2}, \quad R'_{max} = (c-1)\cdot d \cdot (m - \frac{1}{2}) \tag{5.8}$$

Then, we replace the derivative of mutual information with R_{max} and R'_{max}. Since $(m-1)\cdot(R'_{max}) < R_{max}$, the second part of the derivative cannot be 0. Thus, m is optimal when $\log \frac{m-1}{(m-1)\cdot c \cdot d - R_{max}} + \log c + 1 = 0$, and we have $m = \frac{2\cdot(c\cdot d - e^{\log c + 1})}{(c-1)\cdot d}$. □

Specifying ϵ for LDP. In our setting, there are three parameters ϵ, c and m. With the given privacy requirement ϵ, the server can calculate the m and the corresponding c with Lemma 1 and Theorem 7 to make the privacy meet the requirement

ϵ. Specifically, we can set a value c and get the corresponding m with Lemma 1. Since the location domain can be partitioned into m groups and $\forall j \in [2, m-1], |G_j(x)|$ are fixed for all x, we can then calculate the privacy bound by Theorem 7 to see if it meets the privacy requirement ϵ. Per Theorem 7, the ϵ is positively correlated to c with the fixed m and partition groups. Thus, there should be many values c that make the privacy requirement satisfy ϵ. For example, if the c value equals to 5 to meet the privacy requirement $\epsilon = 6$, the value less than 5 would make ϵ smaller which also meets the privacy requirement. However, to fully utilize the privacy that can make the utility maximize, it should only take the maximum of c with the fixed domain. Figure 5.5 shows the numeric results for $c = \frac{\alpha_{max}(x)}{\alpha_{min}(x)}, \forall x \in \mathcal{D}$ and the optimal m versus a varying $\epsilon \in [0.01, 20]$ (given four different domains in our experimental datasets). The plots confirm that ϵ is positively correlated to c (given any domain $\mathcal{D}$), and c is extremely close to e^{ϵ} (slightly smaller). In the experiment, with the given ϵ value and the domain $\mathcal{D}$, we search the maximum value c to satisfy the ϵ-LDP by the binary search method. In Figure 5.5(b), the optimal m is mainly determined by ϵ. The optimal m (rounded to its floor or ceiling) is a small integer, e.g., 2-6 for all the four different domains.



(a) $\log c$ vs ϵ (baseline curve $\epsilon = \log c$) (b) Optimal m vs ϵ

Figure 5.5. $\log c$ and optimal m vs ϵ with various domain size d; domain size d is $374, 566, 1738$, and 3202 in datasets Portcabs [4], Geolife [5], Gowalla [6], and Foursquare [7], respectively

5.3.4 Perturbation Algorithm. For each location $x \in \mathcal{D}$, the server partitions m groups $G_1(x), \ldots, G_m(x)$ and derives the perturbation probabilities for all the output locations in m groups $\alpha_1(x), \ldots, \alpha_m(x)$. After receiving such information from the server, each client perturbs its location x by sampling the output location y. Plsease see details in Algorithm 4.

Algorithm 4 Staircase Randomized Response

Input: user location x, privacy budget ϵ, and domain $\mathcal{D}$

Output: perturbed location y

server pre-computes the optimal m and $\beta_j(x), j \in [1, m]$
for each location $x \in \mathcal{D}$ **do**
  for each group $j \in [1, m]$ **do**
    for each location $z \in \mathcal{D}$ **do**
      if $length(LCP(z, x)) \geq \beta_j(x)$ **then**
        $G_j(x) \leftarrow z$; $\mathcal{D} \leftarrow \mathcal{D} \setminus z$
      end if
    end for
  end for
  for each $j \in [1, m]$ **do**
    compute the perturbation probability $\alpha_j(x)$ for locations in $G_j(x)$
  end for
end for
client samples an output location y from all the locations in $G_1(x), \cdots, G_m(x)$ (per Equation 5.2) and submit it to the server

5.3.5 Distribution Estimation. Similar to other LDP mechanisms, the expectation of the aggregated random location counts would be biased [103]. Given samples from unknown data distribution p, estimating the distribution $\tilde{p}$ of p has been extensively studied [50, 118]. In `L-SRR`, we extend the empirical estimation method with

two perturbation probabilities [50] to estimate the location distribution from the perturbed locations using staircase perturbation probabilities. In our experiment, we also compare the performance of [50] (named HR) with L−SRR.

In the GRR, the estimation counts of location x is only related to the sampled counts of location x. Then, users try to send more information by the perturbation mechanism to have more accurate estimation results. Specifically, in the empirical estimation, for each $x \in \mathcal{D}$, the server creates a candidate location set C_x for input x to estimate the item distribution $\tilde{p}$ from the observed noisy distribution p. Each set C_x which contains $\frac{d}{2}$ locations is a subset of the domain[5]. The server will estimate the $p(x)$ by the C_x. In L−SRR, the server generates a candidate location set C_x for each x with a Hadamard matrix (a square matrix with either $+1$ or -1 entries and mutually orthogonal rows). Given $\mathcal{H}_1 = 1$, for any $\mathcal{H}_K$, we have:

$$\mathcal{H}_K = \begin{pmatrix} \mathcal{H}_{K/2} & \mathcal{H}_{K/2} \\ \mathcal{H}_{K/2} & -\mathcal{H}_{K/2} \end{pmatrix} \tag{5.9}$$

The server then applies a recursion algorithm [50] to generate such Hadamard matrix with size $K \times K$ (denoting it as $\mathcal{H}_K \in \{-1, +1\}^{K \times K}$) where $K = 2^{\lceil \log_2(d+1) \rceil}$ and d is the domain size [50]. Then, each row of $\mathcal{H}_K$ except the 1st row (the 1st row includes only "1" and $\mathcal{H}_K$ includes $d+1$ rows) can be mapped into a unique location in domain $\mathcal{D}$. Specifically, given location $x \in \mathcal{D}$, its candidate set will be derived using the $(i+1)$th row in $\mathcal{H}_K$ where i is the index of x in $\mathcal{D}$. Then, $\forall x \in \mathcal{D}$, we can generate the candidate set C_x for each user's input x as the locations related to the column indices with a "+1" in the mapping row of matrix $\mathcal{H}_K$ [50]. We denote the candidate set of all the locations in $\mathcal{D}$ as $\mathcal{H}_K \circ \mathcal{D}$.

[5]We follow the generation of C_x in [50].

Let $p(C_x)$ be the probability for sampling $y \in C_x$. Then, we can derive $p(C_x)$ with the output y in the corresponding candidate set in case of inputs x and x' (x differs from x' and C_x also differs from $C_{x'}$). Thus, we have $\forall x \in \mathcal{D}$, $p(C_x) = p(x)\sum_{y\in C_x} q(y|x) + \sum_{x'_i \neq x} p(x') \cdot [\sum_{y\in C_x \setminus C_{x'}} q(y|x') + \sum_{y\in C_x \cap C_{x'}} q(y|x')]$, where $p(x)$ is the distribution of x (to be estimated).

All the perturbation probabilities $q(y|x)$ are known in Equation 5.2. Thus, for each $x \in \mathcal{D}$, there exists one equation as above. Given d independent linear equations (due to random coefficients), the d variables $\forall x \in \mathcal{D}, p(x)$ can always be solvable. Specifically, $\forall x \in \mathcal{D}, p(C_x)$ are the observed distribution of all the locations from the aggregated noisy data. Each user sends its perturbed location to the server, which derives the total frequency of all the locations in the pre-computed candidate set of location x. Then, the above d equations can be constructed for estimating the distribution of all the locations $\forall x \in \mathcal{D}, p(x)$. We apply the lower-upper (LU) decomposition algorithm [135, 136] to solve these independent linear equations. Moreover, if the domain D is too large, we can make the heuristic decision using the sampled counts of x' in place of the true count of x' [137]. Algorithm 5 presents the details.

Algorithm 5 Location Distribution Estimation

Input: perturbed locations $y_1, ..., y_n$

Output: estimated location distribution $\tilde{p}(x)$

1: server generates the candidate location set $\mathcal{H}_K \circ \mathcal{D}$ for all the locations in $\mathcal{D}$

2: **for** each $x \in \mathcal{D}$ **do**

calculate the $p(C_x)$ with $y_1, ..., y_n$: $p(C_x) := \sum_{j=1}^{n} \frac{\mathbb{I}\{y_j \in C_x\}}{n}$ construct a linear equation for x with $p(C_x)$ and perturbation probabilities

3: **end for**

4: solve linear equations with the LU decomposition to derive $\forall x \in \mathcal{D}, p(x)$

5: return the estimated location distribution $\forall x \in \mathcal{D}, \widetilde{p}(x) = p(x)$

5.3.6 Private Retrieval for Client Queries. Recall that the client may need to query the estimated location distribution with its true location, e.g., k nearest users [48] (see Section 5.6.3), and traffic-aware GPS navigation [122] (see Section 5.4.2). In L−SRR, users can retrieve the results from the server using the Private Information Retrieval (PIR) protocol [47,123,138] (when needed), which enables any user to privately retrieve information from a database server without letting the server know which record has been retrieved. In the PIR, the database server has an n-bit string $V = \{v_1, ..., v_n\}$, and the client would like to know v_i. The client first sends an encrypted request $E(i)$ for the i-th value to the server, where $E(\cdot)$ denotes encryption function. The server also responds with an encrypted value $r(v_i, E(i))$ (e.g., by quadratic residuosity). Finally, the client can retrieve the record v_i privately based on the server's encrypted response.

Most of the off-the-shelf PIR algorithms can work as a post-processing component (e.g., [47] takes only a few seconds in our experiments). Moreover, the local perturbation and distribution estimation require only ~ 0.014 second for the client and a few seconds for the server (see Section 5.6.5). Thus, the system performance of L−SRR would be very efficient for real-time LBS deployment.

5.3.7 Privacy and Utility Analysis.

Privacy Analysis: ϵ-LDP has been proven for the SRR mechanism in Theorem 7. The server cannot distinguish users' true locations from the noisy data. Moreover, as post-processing procedures applied on the results of LDP scheme, the empirical estimation and PIR (if needed) do not leak any extra information [37].

Error Bounds: Error bounds for the estimation methods in LDP schemes can be derived to understand the expectation of the randomized noise. Then, we derive the error bounds (based on the expectation of the L_1 and L_2-distance) for the

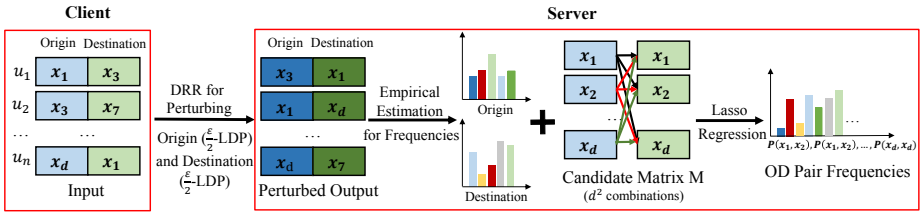


Figure 5.6. Extending SRR to collect and aggregate origin-destination pairs with ϵ-LDP

estimated distribution of all the locations $\tilde{p}$ deviated from the true distribution p.

Theorem 8. *In SRR,* $\mathbb{E}[L_1(\tilde{p}, p)] \leq \frac{2d}{\sqrt{n}\cdot(2\gamma - d\cdot\mu)}$, *where* $\gamma = \min\{\sum_{y\in C_x} q(y|x), x \in \mathcal{D}\}$ *and* $\mu = \max\{\alpha_{min}(x), x \in \mathcal{D}\}$.

Theorem 9. *in SRR,* $\mathbb{E}[L_2(\tilde{p}, p)] \leq \frac{2\sqrt{d}}{\sqrt{n}(2\gamma - d\cdot\mu)}$, *where* $\gamma = \min\{\sum_{y\in C_x} q(y|x), x \in \mathcal{D}\}$ *and* $\mu = \max\{\alpha_{min}(x), x \in \mathcal{D}\}$.

Proof details are given in the Appendix A.4.2. Both error bounds decline if increasing the privacy bound ϵ or the number of users n (thus the error bound would be minor in practice due to a large number of users). Notice that, the expected L_1-distance for the GRR is upper bounded by $\frac{d}{\epsilon}\sqrt{\frac{2(d-1)}{n\pi}}$ [134], which can be $\sqrt{d}$ times of the SRR error bound in the worst case.

5.4 L-SRR for Trajectory-Input LBS

In this section, we extend SRR to support trajectory-input LBS using two example applications: (1) collecting the origin and destination (OD) of users for OD analysis [49], and (2) collecting a sequence of user locations for traffic-aware GPS navigation [5].

5.4.1 Origin-Destination Analysis. OD analysis aggregates a pair of origin-destination from each user to estimate the traffic flow [49]. In this case, the LDP notion (Definition 5) should be extended to protect each user's OD pair.

Definition 7 (ϵ-Local Differential Privacy)**.** *A randomization algorithm* $\mathcal{A}$ *satisfies* ϵ*-LDP, if for any two different location pairs* $(x_o, x_d), (x'_o, x'_d) \in \mathcal{D} \times \mathcal{D}$, *and for*

any output location pair $(y_o, y_d) \in range(\mathcal{A})$ *sent to the untrusted server, we have* $Pr[\mathcal{A}(x_o, x_d) = (y_o, y_d)] \leq e^{\epsilon} \cdot Pr[\mathcal{A}(x'_o, x'_d) = (y_o, y_d)]$.

The LDP scheme for OD analysis should preserve the sequential correlation from the origin to the destination (OD pair). Thus, the domain has been greatly expanded to d^2 OD pairs in $\mathcal{D} \times \mathcal{D}$. To avoid the bad utility resulted from a large domain, we extend the Lasso regression [139] to a novel *private matching method* to preserve the OD sequence.[6] Then, we integrate the private matching into `L-SRR` to ensure accurate OD distribution with ϵ-LDP.

Specifically, users perturb their two locations with privacy budget $\frac{\epsilon}{2}$ for each. The server receives a large number of noisy samples of all users from specific distributions for origins and destinations, respectively. The server may estimate the distribution from the noisy sample space using the linear regression $\vec{y} = \mathrm{M} * \vec{w}$, where matrix M includes the predictor variables, vector $\vec{y}$ includes the response variables, and vector $\vec{w}$ includes the regression coefficients. The predictor variables in M consist of all the combinations of trajectories from each origin to each destination (d^2 pairs), which could be known to the server and client beforehand. Moreover, the response variables $\vec{y}$ can be estimated from the `SRR` perturbed values. Notice that, the frequencies of most combinations $(x_o, x_d) \in \mathcal{D} \times \mathcal{D}$ are very small or even equal to zero in LBS. Thus, Lasso regression [139] can effectively solve such sparse linear regression by encoding the predictor variables M for all the OD pairs.

As shown in Figure 5.6, we have two steps in `L-SRR`: (1) perturbing the origin and destination separately by each client, and (2) estimating the joint distribution of OD pairs using Lasso regression by the server. Each client first applies `SRR` to perturb the origin and destination with privacy budget $\frac{\epsilon}{2}$ each. Then, the server estimates

[6]Lasso regression was used to generate the synthetic high-dimensional dataset with LDP and preserve the correlation across dimensions [139].

the distribution of origin and destination to generate the vector $\vec{y}$. Meanwhile, the server encodes the overall candidate set of OD pairs M based on the location domain $\mathcal{D}$. Finally, the server fits a Lasso regression model to the vector $\vec{y}$ and the candidate matrix M to learn $\vec{w}$. Therefore, the non-zero coefficients in w will be considered as the frequencies for the candidate OD pairs.

Privacy Bound. Although the origin and destination are correlated, each user sends these two perturbed locations sequentially. The sequential composition of releasing two locations would only result in the total leakage (ϵ-LDP) even if they are highly correlated [37]. The Lasso regression is performed on the two sets of perturbed data (one set of origins and another set of destinations) as post-processing to retain the correlation, which would not consume privacy budget [37]. Thus, the OD analysis still satisfies ϵ-LDP.

5.4.2 Traffic-Aware GPS Navigation. In this App, users may seek the route with shortest time by avoiding congested roads. At that moment, users may update and send multiple locations to the server in sequence. Meanwhile, each user will privately retrieve the real-time nearby traffic from the server to help update the route in case of traffic congestion.

Specifically, the route recommendation algorithm can be deployed in the client to compute the best route with the shortest traveling time on an offline map (integrated with the real-time traffic information from the server) [5]. For any route, the total traveling time t can be predicted with the historical dataset.[7] Also, each user can send the current location x_i to the server again and learn the current traffic density. Then, the client may recompute the best route and update the estimated traveling time. Intuitively, if the suggested route does not have any traffic, it is unnecessary

[7]These historical datasets could be obtained from public traces and check-in datasets, or datasets generated from LBS applications.

to update the user's location to learn the real-time traffic density (this would avoid consuming more privacy budget). Thus, we follow this idea to extend our SRR. In L−SRR, the client will identify these "location updates" (similar to [140]). Let $\mathbb{T}$ denote a trajectory and $Agg(x_o, x_i), x_i \in \mathbb{T}$ represent the actual traveling time from the origin x_o to current location x_i. In the meanwhile, the GPS can predict the piece-wise traveling times between the origin x_o and any location $x_i \in \mathbb{T}$ before the arrival. It is worth noting that the time is treated as the condition for the update (as above). It can be extended to update the location with other criterion in specific applications (e.g., distance, and checkpoints).

Denoting such predicted time as $Agg^p(x_o, x_i), x_i \in \mathbb{T}$, the client will examine the difference between their actual traveling time $Agg^t(x_o, x_i)$ and the predicted time $Agg^p(x_o, x_i)$ at different locations $x_i \in \mathbb{T}$. If the client finds that the actual traveling time $Agg^t(x_o, x_i)$ is significantly more than predicted one $Agg^p(x_o, x_i)$, e.g., delayed time exceeds a threshold: $Agg^t(x_o, x_i) - Agg^p(x_o, x_i) > \theta$, there is likely a traffic congestion. Then, the client requests a "location update" to privately upload the perturbed location to the server, and privately retrieve the current traffic density. Moreover, the server will periodically estimate the traffic density using all the perturbed locations collected from the clients in the past time window (e.g., 5 minutes for each time window). Once a location update is requested by any client, the server privately delivers the traffic density to the client via the PIR protocol.

Privacy Bound. Since every perturbed location is individually aggregated (based on individual locations) rather than as a combination, such data collection can be done for all the locations separately and simply follows sequential composition [127]. Thus, SRR for such trajectory-input LBS satisfies $\lambda\epsilon$-LDP where λ is the number of requested location updates from the origin to the destination. We have empirically evaluated that λ is small in practice (e.g., 2 or 3). Finally, PIR may

result in side-channel leakage (e.g., who requested the location update may be in the congested areas). If necessary, this can be simply mitigated by an anonymizer (e.g., shuffler [141]), which also further amplifies the LDP protection [141].

5.5 Discussion

Relaxed LDP. Some recent works [44,127,137] relaxed the LDP by considering the input variants. For instance, `ID-LDP` [127] relaxes the LDP with different ϵ for different inputs; geo-indistinguishability (`GI`) makes every pair of locations indistinguishable, but the "level" of indistinguishability depends on their distance (locations that are far apart are more distinguishable than locations that are close together); `CLDP` [137] provides distance discriminative privacy, and relaxes the protection for different pairs of inputs. Different from `L-SRR`, all of them cannot strictly satisfy ϵ-LDP. To validate their limitations on rigorous LDP guarantee, we present some numeric analysis with the same setting (by converting them to ϵ-LDP). *`PLDP` [45] is experimentally compared in Section 3.7 since it focuses on LBS.*

First, we generate a synthetic dataset including items with uniformly distributed frequencies (the distance between inputs can also be directly measured). For `ID-LDP`, we randomly assign the privacy bound from $\{0.5\epsilon, 0.8\epsilon, \epsilon\}$ to each distinct item. Since $\{0.5\epsilon, 0.8\epsilon, \epsilon\}$-ID-LDP satisfies $\min\{\{\epsilon\}, 2 \times \{0.5\epsilon\}\}$-LDP, it can guarantee ϵ-LDP for all the items. For `GI`, we sample the output y with the Laplace-based PDF centered at input x. For `CLDP`, we adopt the conversion between ϵ and α [137]. Table 5.1 shows the L_1-distance of the outputs on different ϵ. The utility of `L-SRR` significantly outperforms all the relaxed LDP with the same LDP guarantees.

Generalization. `L-SRR` can be potentially extended to other data types if the distances between values/items can be measured (e.g., numerical data). In such contexts, the data items can also be partitioned and staircase perturbation

Table 5.1. Average L_1-distance

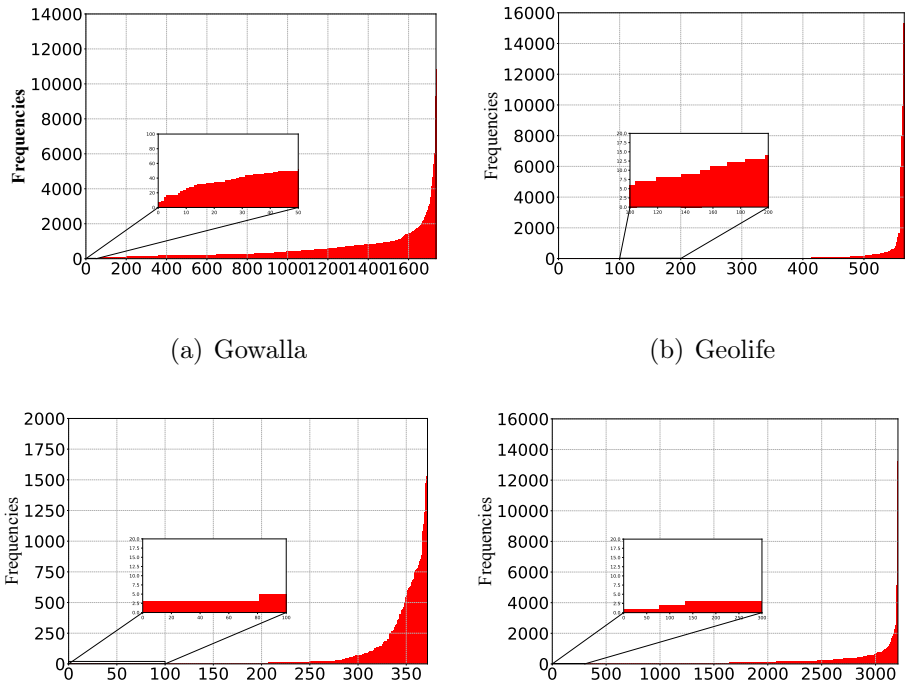
Privacy Bound ϵ	0.5	1	2	3	4
ID-LDP	2.14	1.97	1.64	1.45	1.18
GI	2.21	1.84	1.75	1.43	1.21
CLDP	0.93	0.90	0.84	0.72	0.70
L-SRR	0.65	0.62	0.51	0.44	0.36

probabilities can be derived and allocated to values/items in different groups. We will evaluate its performance in other domains and benchmark with the corresponding LDP schemes (e.g., Piecewise [142]) in the future.

Encoding and Precision. The precision of the encoded locations can be tuned by the level of the bit string hierarchy. Although larger h more accurately encodes locations, the domain size will grow and thus the perturbation probability (for the true location) may decline for the same privacy. Thus, larger h does not necessarily make the staircase perturbation scheme more accurate (thus we use the standard $h = 23$ as Bing Map). In the experiment, every location can only be possibly flipped to other locations in the domain not every pixel on the map. There are two benefits for such encoding and design: (1) locations will not be perturbed to an unrealistic location (e.g., in the ocean), and (2) it is more efficient to compute the perturbation probabilities offline (due to reduced domain size).

Larger and Worldwide Domain. In this paper, we evaluated our scheme within each city (four datasets) by following the same settings as other LBS since each experimental dataset is collected within a city. If all the locations on the planet are considered, the domain size would be much larger and the utility might be degraded

since the error bound is related to the domain size d.



(a) Gowalla (b) Geolife

(c) Portocabs (d) Foursquare

Figure 5.7. Location frequencies in experimental datasets

System Deployment. L−SRR can be deployed as an application or integrated with the existing LBS applications in the server and clients (e.g., mobile devices). Given the privacy bound ϵ and a location domain $\mathcal{D}$, the server will precompute the required c, the optimal m, the GLCP for group partitioning $\forall x \in \mathcal{D}, \beta_1(x), \ldots, \beta_m(x)$, and the perturbation probabilities $\forall x \in \mathcal{D}, \alpha_1(x), \ldots, \alpha_m(x)$ for SRR, and then share them to all the clients. In L−SRR, the location domain is updated periodically by the server rather than per users' requests. It would not cause any privacy leakage, and it is very efficient to update the domain. If a user is at a location not in the domain before the update, the client will approximate it to the nearest location in the domain. Each client only needs to perturb their locations based on the stored md perturbation probabilities $q(y|x)$, and then directly send the output to the server. Even if the client may privately retrieve the analysis result related to

his/her location from the server, the PIR protocol can be efficiently executed without many overheads. Thus, the clients do not need to be equipped with strong computing capabilities (mobile devices suffice). Each client should download an offline map if required in certain LBS applications, e.g., traffic-aware GPS navigation.

Provable Privacy for PIR and LDP. The PIR protocol is applied as the post-processing to the query results that guarantees ϵ-LDP. The index (w.r.t. the domain) can be public knowledge and shared to users. The PIR protocol does not cause any additional information leakage since the query results are retrieved based on the encrypted location by employing the provably secure cryptographic technique. From the viewpoint of the server, the PIR request might be originated from any user. Therefore, the probability to identify every user as the querying user is exactly $\frac{1}{n}$ (for all the users). Thus, it does not cause additional leakage from such random guess either (after the private data collection with ϵ-LDP).

Complex Applications. The staircase randomized response can generate more accurate location distribution than existing LDP mechanisms. As a key building block of LBS applications, such high accurate location frequency/distribution estimation by the proposed `SRR` mechanism could universally support different LBS applications, including complex LBS such as traffic-aware GPS navigation. In our experiments, we simulate the route recommendation by the GPS, which shows better performance of `SRR` (see the details in Appendix A.5). In practice, as the LBS application becomes more complicated (e.g., more data collection), `SRR` would outperform the state-of-the-art LDP schemes more.

5.6 Experiments

5.6.1 Experimental Setting. We conduct our experiments on four real-world location datasets.

- *Gowalla Dataset* [6] collects $6,442,890$ check-ins records of $196,591$ users in Austin, USA via the social network app Gowalla between 02/2009 and 10/2010.
- *Geolife Dataset* [5] collects 17,621 GPS trajectories of 182 users in Beijing between 04/2007 and 08/2012.
- *Portocabs Dataset* [4] collects the GPS trajectories of 441 taxis in Porto between 07/2013 and 06/2014.
- *Foursquare Dataset* [7] collects $90,048,627$ check-in locations of $2,733,324$ users in New York City, USA.

Since each of the four datasets is collected from locations within a city, we focus on a large geographical region covering a $40 \times 30\text{km}^2$ area for each dataset. Only the reported locations in this area are considered as the domain. Since the encoded bit strings for all the locations in each dataset share a 20-bit common prefix, the last 26 bits (out of 46 bits for $h = 23$) could sufficiently index all the locations with high accuracy for all the 4.7m×4.7m regions (removing the common prefixes does not affect the accuracy due to fixed domain size and groups). All the experiments were performed on the NSF Chameleon Cluster with Intel(R) Xeon(R)Gold 6126 2.60GHz CPUs and 192G RAM [143]. Docker is used to start containers to emulate the server/clients with system and network setup.

Dataset Characteristics: Table 5.2 presents the number of locations and users in four datasets. The total user number can vary from $30,000$ to 1M. As we know, infrequent locations in the LDP can cause more utility loss than frequent locations [103]. So, we use four dataset that have different densities of users. Figure 5.7 presents the original frequencies of all the locations in four datasets.

5.6.2 Distribution Estimation (Location-Input). We first evaluate the utility of `L-SRR` for the distribution estimation while benchmarking with the state-of-the-art

Table 5.2. Characteristics of datasets (after pre-processing)

Dataset	**Location #**	**User #**
Gowalla	1,738	1,120,147
Geolife	566	104,488
Portcabs	374	34,438
Foursquare	3,202	701,528

LDP schemes, including Generalized Randomized Response [126] (`GRR`), Optimal Local Hash with hierarchy structure [144] (`OLH-H`), the Location Data Aggregation [45] (`PLDP`), and the Hadamard Response (`HR`) [50]. We follow the original perturbation and estimation method in each benchmark. Here, we choose the OLH mechanism since it has better utility than unary encoding (`UE`), and choose the existing location LDP framework `PLDP` instead of existing location framework in [42, 43] since the `PLDP` is an optimized framework that boosts the utility. For fair comparisons, in `OLH-H`, we randomly sample a hierarchical level for each location. Then, we adapt the constrained inference [145] to adjust the frequencies of parent and leaf nodes for consistency. In `PLDP`, we assign the same protection region level for all the users as other LDP schemes to satisfy the strict ϵ-LDP.

The server derives the spatial density for many LBS applications, e.g., urban traffic density [146], and crowd density for events [120]. In most existing LDP settings, the ϵ is in the range between 0.5 to 10 for privacy protection. Too large ϵ value can't protect user's location well. Similar to that, we set ϵ between 1 and 8 with a step of 0.5 (covering both strong and weak privacy regime).

Figure 5.8 shows the average L_1-distance and KL-divergence between the true

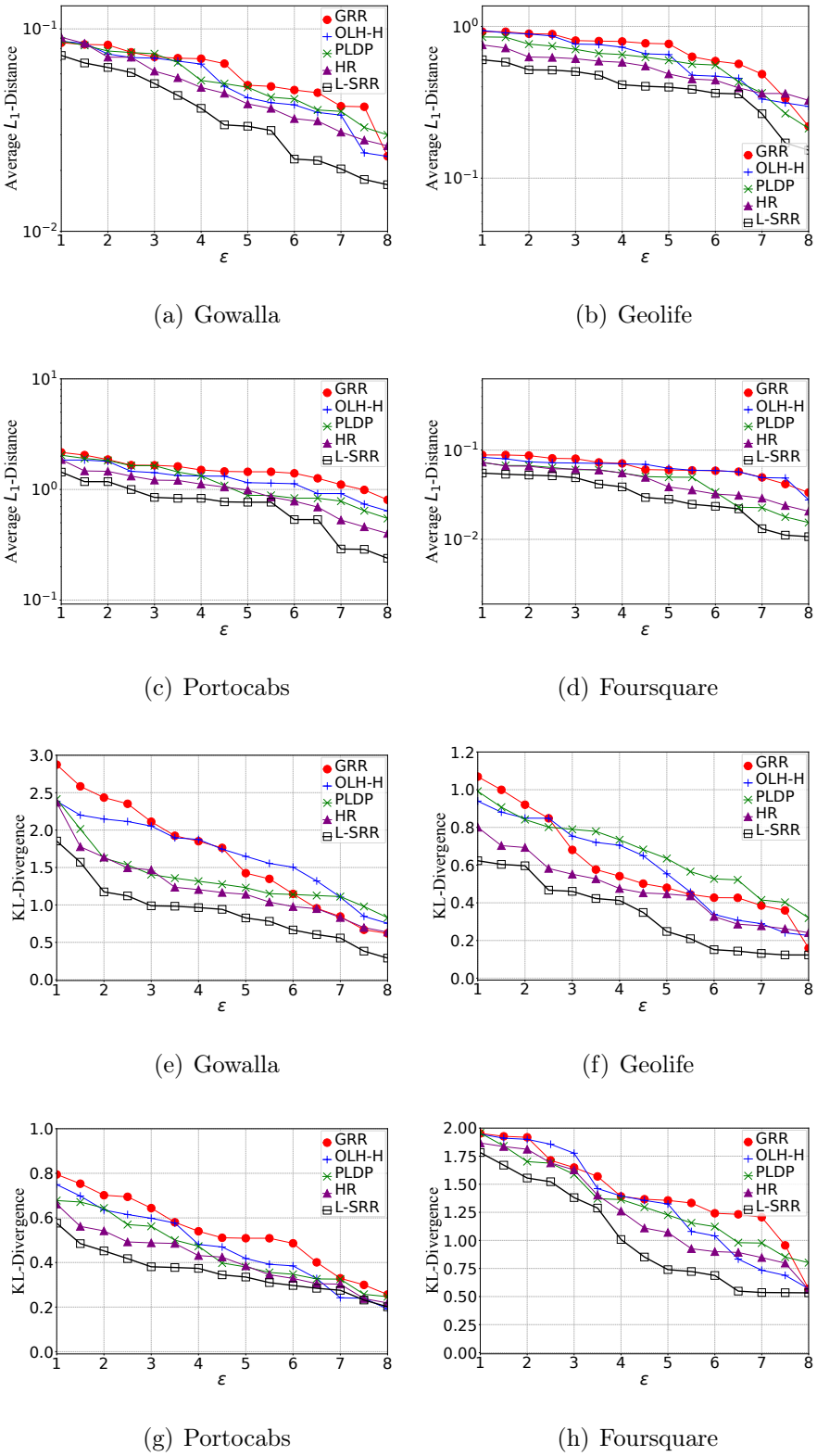
and estimated distributions of all the locations. Both L_1-distance and KL-divergence decrease as ϵ increases. Especially for the GRR, the error dramatically decreases (e.g., Figure 5.8(e)) since the probability grows exponentially. However, L−SRR still greatly outperforms other LDP schemes on all the four datasets.

5.6.3 Case Study I: k-NN Query (Location-Input). We first evaluate the performance of SRR in specific applications on recommendations based on the location distribution. k-nearest neighbors (k-NNs) is a typical application in which queries can be made for the nearest point-of-interests or users. We next show the results for querying the k-NN users [48], which can be extended from the distribution estimation. The k-NN lists for any user (with a location) are the other k closest users, measured by the MSE of their coordinates. Then, given the estimated location distribution, the server can directly derive each location's list of k-NNs.

k-NN Lists Computed by Server. Figure 5.9 shows the normalized MSE between the true and estimated coordinates of all the users' k-NN lists. The normalized MSE also decreases while ϵ increases. In Figure 5.9(a), 5.9(b), 5.9(c), and 5.9(d), L−SRR outperforms GRR, OLH−H, PLDP, and HR, which is consistent with the previous results.

We also present the *precision* and *recall* of all the users' estimated k-NN lists in Table 5.3 and Table 5.4. Again, L−SRR can produce more accurate k-NN lists than all the other LDP schemes. Note that ϵ might be relatively large for very high accuracy (e.g., $\epsilon = 5$ similar to the privacy setting by Apple [116]). If involving more users in the practical LBS App, ϵ can be much smaller for such very high accuracy.

5.6.4 Case Study II: Trajectory-Input LBS. We next evaluate the performance of L−SRR on collecting trajectories for two example LBS applications: (1) origin and destination (OD) analysis which estimates the OD pairs frequencies with the Lasso regression, and (2) traffic-aware GPS navigation.



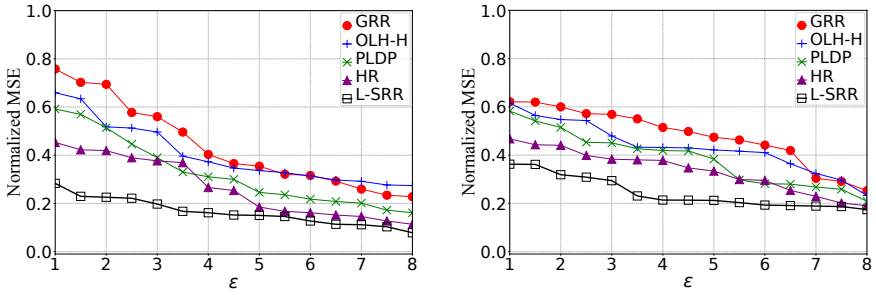
(a) Gowalla (b) Geolife

(c) Portocabs (d) Foursquare

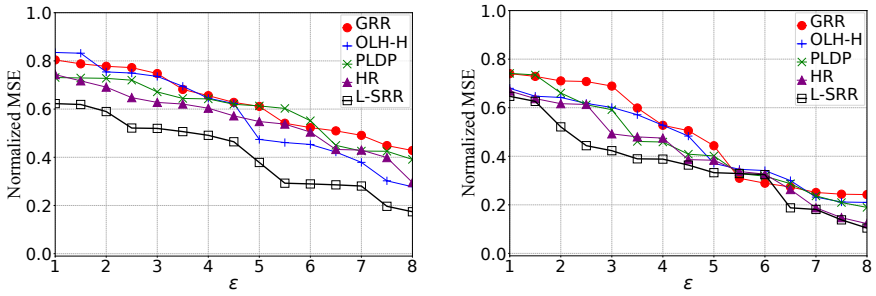
(e) Gowalla (f) Geolife

(g) Portocabs (h) Foursquare

Figure 5.8. Average L_1-distance and KL-divergence for the distribution estimation on four datasets using different LDP schemes



(a) Gowalla (b) Geolife

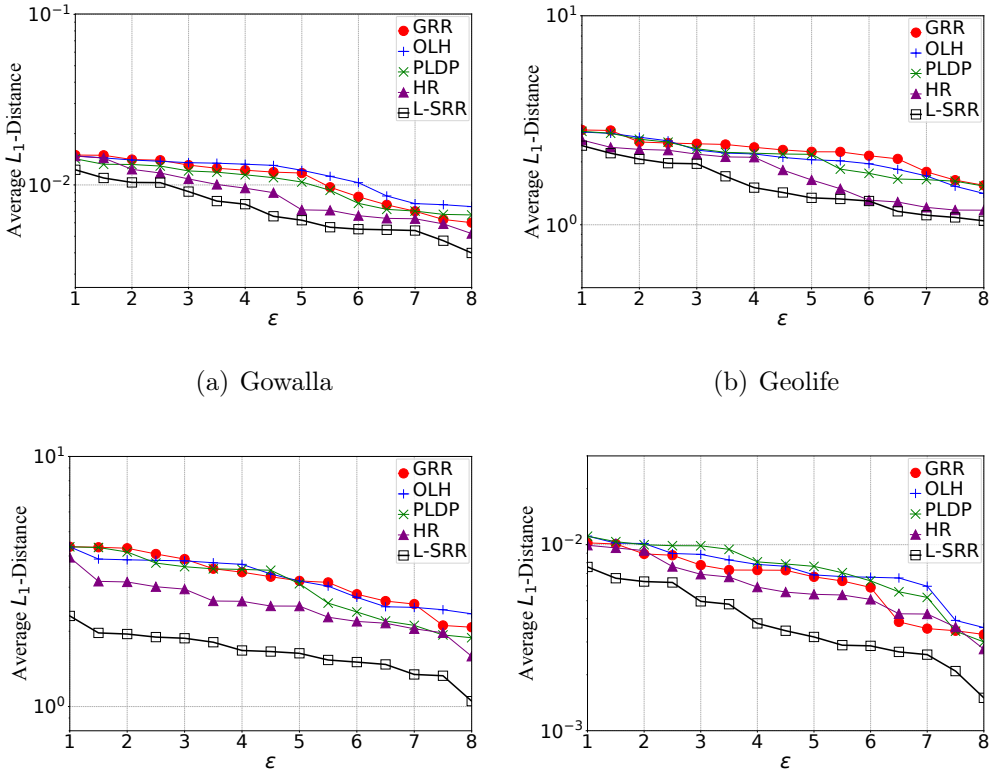


(c) Portocabs (d) Foursquare

Figure 5.9. MSE of all the locations' k-NN lists on four datasets using different LDP schemes ($k = 25$)

OD Analysis. The true number of distinct OD pairs in four datasets are $2,315$, 876, $1,034$, and $5,634$, respectively. We apply the same Lasso regression algorithm to all the LDP schemes. Figure 5.10 presents the average L_1-distance between the true and estimated OD pair distribution. As ϵ increases, L_1-distance decreases. `L-SRR` again shows the smallest L_1-distance of `L-SRR` in all the experiments. Moreover, we also observe that the L_1-distance is smaller than LBS with single-location input (see Figure 5.8).

Traffic-Aware GPS Navigation. To test the performance of the traffic-aware GPS navigation, we make the simulation the experiment of recommendation for the fastest route. We can also draw the conclusion that `L-SRR` outperforms other LDP schemes (see the detailed results and discussions in Appendix A.5).



(a) Gowalla (b) Geolife

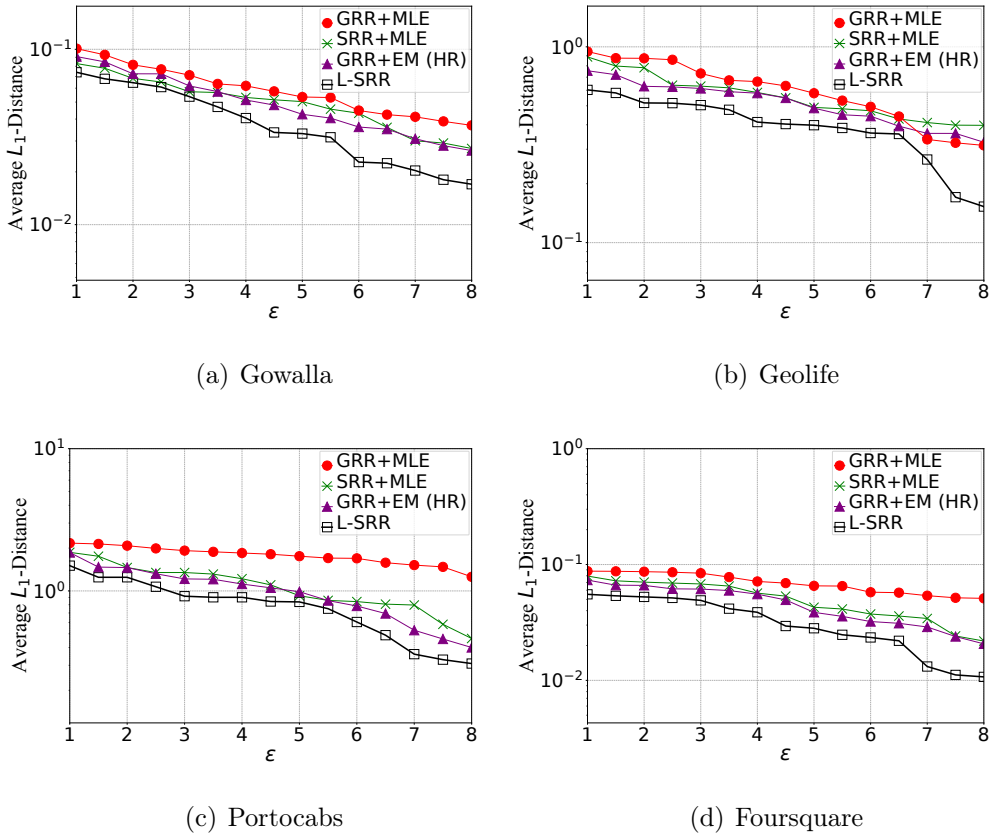
(c) Portocabs (d) Foursquare

Figure 5.10. Average L_1-distance for the OD pair frequency on four datasets using different LDP schemes

5.6.5 Ablation Study and Runtime.

Ablation Study. We compare the results with different combinations of perturbation mechanisms (GRR, HR and SRR) and estimation methods. Since the standard estimation method cannot be applied to SRR (more than two perturbation probabilities), we apply the maximum likelihood estimation (MLE) instead. Moreover, the GRR with empirical estimation (EM) is a special case of Hadamard response (HR): $|C_x| = 1$. Figure 5.11 shows that SRR and the revised EM (L−SRR) perform the best. Even with the MLE, SRR is better than GRR in most cases. Also, the revised EM can further boost the utility of SRR (compared to SRR and MLE).

Runtime. Since users only need to perturb their locations, the user-side



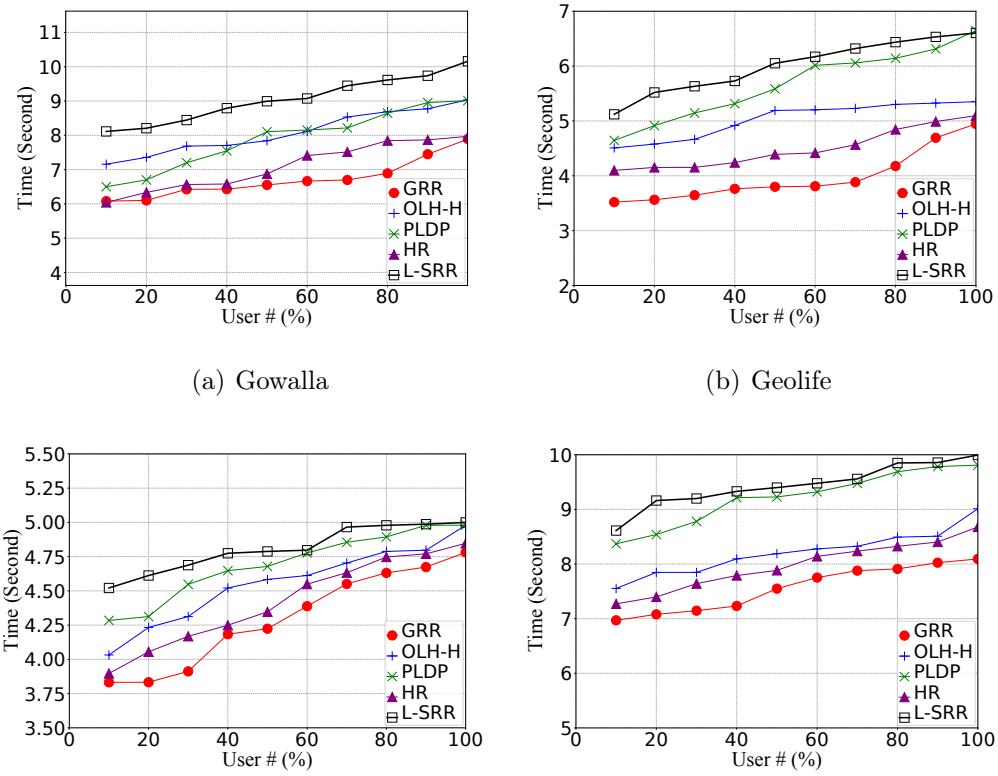
(a) Gowalla (b) Geolife

(c) Portocabs (d) Foursquare

Figure 5.11. Average L_1-distance for frequency estimation using different combinations of perturbation and estimation methods

runtime is negligible. It takes only 0.014 second for each user on average in the experiments, and thus we only report the server-side runtime in Figure 5.12. We test 10% to 100% of each dataset with a step of 10%. Similar to `GRR`, `OLH-H`, `PLDP` and `HR`, the runtime of `L-SRR` only slightly increases as the number of users reaches ∼1M (e.g., 9 seconds for Gowalla dataset), which is acceptable.

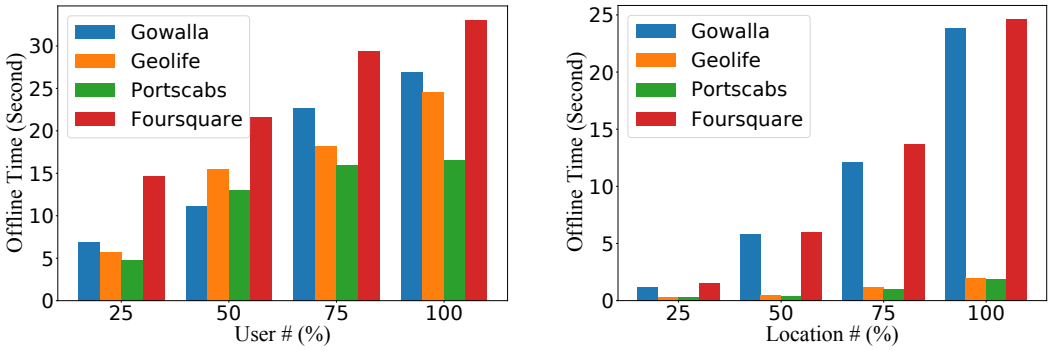
Notice that, group partitioning dominates the offline costs $O(d^2)$ for `L-SRR`. Thus, we also present such offline partitioning time w.r.t. the number of users and the number of locations, as shown in Figure 5.13. We uniformly extract $25\%, 50\%, 75\%$, and 100% of users and locations from each dataset as the test datasets. As shown in Figure 5.13(a), the offline time (including the the preprocessing time to get the sub-dataset) increases as the number of users increases due to the growth of distinct



(a) Gowalla (b) Geolife

(c) Portocabs (d) Foursquare

Figure 5.12. Runtime for the server (vs. the number of users)



(a) Runtime vs.# of users (b) Runtime vs. # of location

Figure 5.13. Offline runtime

locations. Since the group partitioning that is related to the domain size dominates the offline costs. In Figure 5.13(b), we also see that the offline runtime (excluding the preprocessing time) grows on the number of locations, and the offline time is around 30 seconds at most. However, the offline execution is needed when the location domain

is updated. Recall that the domain is only updated periodically (e.g., every day). Thus, such offline costs are efficient for real-world deployments.

5.7 Conclusion

Severe privacy risks arise in LBS applications due to sensitive location collection. To address the deficiency on privately collecting locations with LDP guarantees and high utility, we propose a novel LDP mechanism "Staircase Randomized Response" (SRR) and extend the empirical estimation for SRR to significantly improve the accuracy of the LDP model for LBS applications. In addition, we have also extended SRR to privately collect trajectories with ϵ-LDP. We have conducted extensive experiments on real datasets to show that L−SRR drastically outperforms other LDP schemes.

Table 5.3. Precision and recall for the derived k-NNs of all the users (k=25)

Dataset	ϵ	GRR		OLH-H		PLDP		HR		L-SRR	
		Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Gowalla	1	31.9%	47.6%	27.1%	35.6%	38.1%	46.4%	53.2%	63.1%	**60.7%**	**69.4%**
	3	55.5%	54.1%	30.6%	38.9%	50.1%	56.9%	66.8%	74.7%	**68.7%**	**77.4%**
	5	63.5%	66.2%	51.0%	57.2%	67.6%	74.3%	68.3%	75.8%	**73.6%**	**78.1%**
	7	78.4%	81.2%	68.8%	73.3%	73.9%	79.5%	75.4%	80.9%	**80.1%**	**81.9%**
	9	86.1%	87.2%	69.2%	74.4%	80.3%	85.3%	82.1%	84.1%	**87.7%**	**89.3%**
Geolife	1	17.8%	26.4%	30.1%	34.2%	33.4%	34.2%	30.8%	35.3%	**35.2%**	**39.9%**
	3	35.0%	43.6%	42.4%	49.1%	48.7%	54.5%	50.4%	53.1%	**51.6%**	**58.7%**
	5	53.4%	60.3%	60.5%	65.9%	69.9%	74.9%	67.1%	68.8%	**78.3%**	**83.3%**
	7	78.6%	82.9%	73.1%	76.5%	77.0%	80.7%	76.4%	78.1%	**85.9%**	**88.9%**
	9	91.4%	93.0%	89.8%	90.8%	90.2%	93.8%	90.8%	92.2%	**92.7%**	**94.2%**

Table 5.4. Precision and recall for the derived k-NNs of all the users (k=25) (– continued from Table 5.3)

Dataset	ϵ	GRR		OLH-H		PLDP		HR		L-SRR	
		Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Portocabs	1	41.9%	50.7%	30.4%	40.4%	48.8%	58.4%	51.2%	58.4%	**56.2%**	**64.1%**
	3	63.8%	72.6%	43.6%	50.6%	55.6%	63.3%	57.7%	63.1%	**68.3%**	**75.8%**
	5	70.5%	78.2%	61.9%	66.9%	70.6%	76.1%	59.7%	65.0%	**77.4%**	**83.8%**
	7	87.8%	93.3%	66.0%	69.4%	76.2%	81.3%	84.9%	88.2%	**92.7%**	**98.1%**
	9	93.4%	98.7%	86.5%	89.5%	86.7%	89.2%	91.6%	93.3%	**95.9%**	**98.9%**
Foursquare	1	32.2%	40.9%	42.2%	52.1%	46.6%	56.1%	52.2%	60.7%	**55.7%**	**65.3%**
	3	58.8%	65.2%	50.1%	57.4%	50.1%	58.0%	59.1%	67.3%	**67.1%**	**75.3%**
	5	80.7%	84.6%	64.6%	68.6%	68.1%	75.7%	80.6%	86.9%	**83.9%**	**87.2%**
	7	87.1%	89.7%	65.4%	69.1%	68.7%	76.3%	85.4%	88.9%	**87.2%**	**91.3%**
	9	88.1%	92.3%	76.1%	77.4%	82.6%	86.9%	86.1%	89.1%	**91.3%**	**94.6%**

CHAPTER 6
CONCLUSION & FUTURE WORK

6.1 Conclusion

In conclusion, the prevalence of portable devices that generate vast amounts of high-dimensional and unstructured data has raised concerns about privacy. To address this issue, this dissertation proposes three frameworks for safeguarding the privacy of such data in various domains.

Specifically, the first work focuses on video analysis with differential privacy guarantee and proposes a new sampling-based mechanism called VideoDP that generates utility-driven private videos for any private analysis. The proposed mechanism provides a flexible platform for untrusted analysts to conduct private queries and analyses with superior utility, as demonstrated through extensive experiments.

The second paper addresses privacy concerns in real-world videos and proposes a two-phase video sanitization technique called VERRO that perturbs all objects' content and coordinates in the video to generate a synthetic video with indistinguishable objects. This approach helps to prevent individuals from being re-identified with background knowledge and has been validated through experiments on real videos.

The third work focuses on location-based services (LBS) applications and proposes a novel LDP mechanism called Staircase Randomized Response (SRR) to privately collect locations and trajectories with LDP guarantees and high utility. The paper extends SRR to improve the accuracy of the LDP model for LBS applications and demonstrates the superiority of L−SRR over other LDP schemes through extensive experiments on real datasets.

Overall, these framework demonstrate the importance of privacy protection in various domains and propose innovative solutions to address privacy concerns while

maintaining high utility. These solutions have significant implications for a wide range of industries and applications, and can inspire further research and innovation in the field of privacy protection.

6.2 Future Work

6.2.1 Trustworthy AI. Deep learning is a subset of machine learning that involves training artificial neural networks with many layers. Deep learning algorithms can learn to recognize patterns and features in data, making them particularly useful for tasks such as image recognition, natural language processing, and speech recognition. These algorithms are often used in applications such as self-driving cars, facial recognition systems, and language translation software. To get an accurate deep learning model, one of the key points is to train the model on massive datasets. It requires that datasets are clean, well-labeled, and representative. The privacy problem in deep learning arises when sensitive or personal information is present in the datasets (e.g., medical records or financial information) used to train the models. There is a risk that this information could be used to identify individuals or disclose their private information.

There are several ways in which deep learning can pose privacy risks. One is through the use of training data that contains sensitive information. This information can be inadvertently learned by the deep learning model and used to make predictions or inferences about individuals, potentially leading to unintended consequences. For example, Reza et al. [147] proposed membership inference attacks against machine learning models. Specifically, the pre-trained models have been publicly available and model parameters are fully exposed. With the model parameter, the adversaries can train an attack model by the shadow models which use the same machine learning platform for training the target model and the same format training dataset (disjoint with the training dataset of the target model). Another issue is the possibility of

adversarial attacks on deep learning models. These attacks involve deliberately manipulating the input data in order to cause the model to make incorrect predictions or reveal sensitive information. For example, an attacker could add noise to an image/video in a way that causes a deep-learning model to misidentify the contents of the image/video [148–152]. In summary, the privacy of deep learning is an important consideration in the development and deployment of machine learning models. It is important to design models and train data with privacy in mind and to use appropriate privacy-preserving techniques to ensure that individual privacy is protected.

Nowadays, differential privacy is widely applied to solve the privacy problems in machine learning [62, 153–156], and it has a certain defense effect against membership inference attacks for the training dataset [157]. Moreover, differential privacy is also used to different layers of DNN models to propose defense methods against the state-of-the-art adversarial perturbations of the DNN models, which guarantees the robustness of models [158]. However, there are still some issues to be addressed since there are different machine learning tasks and they use different metrics as loss functions. Thus, we can investigate the relationship between the defense or robustness of deep learning and differential privacy and explore the utility-optimized privacy-preserving deep learning model aiming at specific machine learning tasks and loss functions (universal solution) [62]. In addition, we will also investigate cryptographic techniques [159–166] for preserving the privacy of deep learning systems by studying the trade-off between user privacy protection and system efficiency.

6.2.2 Security and Privacy for Biomedical Data. Biomedical data is always analyzed to understand how living systems function. For instance, biomedical data science can develop new analysis technologies with machine learning in order to predict disease and provide disease diagnosis at lower human costs. However, there are several issues to be addressed when we process biomedical data with machine learning

models.

First, as we know, some additive noise may be generated by the respiratory and body movements of patients in medical images (e.g., MR images). These noise may be Gaussian, Rician, and Speckle noise. It is highly possible for radiologists to misdiagnose these medical images with additive noises. With the fact that the prediction and diagnosis result should be more accurate if these medical images are more "clean", we would like to pre-process these images and eliminate the noise before diagnosing these images with deep learning models. However, the traditional methods to pre-process the noise still need a lot of human work to identify and exclude such poor images prior to any algorithmic analysis. Thus, we can propose a smooth classifier that can predict the noised medical image correctly. In the concept of certified robustness against adversarial examples (noisy image for misclassification), we can also add the noise into the classifier and use the smoothed classifier to do the prediction. The smoothed classifier outputs the statistic prediction probability of the input over the noises and guarantees the prediction accuracy with the noisy input image. Based on this, for a general problem by classifying medical images to classes in Y, given an arbitrary base classifier f, it can be converted to a "smoothed" classifier g by adding isotropic noise (e.g., Gaussian noise) to the input x: $g(x) = argmax_{c \in Y} P(f(x + \epsilon) = c)$. The smooth classifier then makes the correct prediction $P(f(x + \epsilon + \delta) = c_A) \geq max_{c \neq c_A} P(f(x + \epsilon + \delta) = c)$ where c_A means the correct class of the medical image and δ means the noise of medical image. Thus, our mission is to find a suitable noise ϵ for the classifier.

Second, there is privacy and security problem for medical image collection. In many recent machine learning work with medical images, there is only a small training dataset since they can't have enough patients' medical images which involve in the privacy of patients. However, the training dataset size affects the results significantly

and too small training dataset can't guarantee the accuracy. If each hospital or agent would like to share their patients' record, it would make the training dataset size larger. However, not each hospital can obtain the authorization from patient to release the image directly. Thus, we propose a platform that trains the DNN models locally by differentially private federated learning without sharing records, which can guarantee the accuracy of the trained model while protecting the medical privacy of users.

APPENDIX A

PROOF, ALGORITHM, ADDITIONAL RESULTS

A.1 Optimal k_j for VE Υ_j

A.1.1 Equations for Different Pixels. If pixel (a, b, t) is a non-border pixel, we have Equation 3.1 to represent the relation between the RGB expectation of any pixel (a, b, t) and the RGB expectation of its four neighbors (denoted as $\hat{\theta}_N, \hat{\theta}_S, \hat{\theta}_W$ and $\hat{\theta}_E$). We now briefly discuss how to derive such relation.

First, if pixel (a, b, t) is sampled, then the RGB expectation equals $Pr(a, b, t) * \theta(a, b, t)$ where $Pr(a, b, t)$ is the probability of sampling (a, b, t) and $\theta(a, b, t)$ denotes its RGB in the original video V.

Second, if pixel (a, b, t) is not sampled, then it will be interpolated based on the RGBs of its neighbors. There are five subcases (denoting the probabilities that (a, b, t) has 0, 1, 2, 3 and 4 neighbors before interpolation as $\sigma_0(a, b, t), \sigma_1(a, b, t)$, $\sigma_2(a, b, t)$, $\sigma_3(a, b, t), \sigma_4(a, b, t)$):

1. 0 neighbor: all its neighbors are not sampled in Phase I. Then, the probability share is $\sigma_0(a, b, t) * 0$.

2. 1 neighbor: 3 of its neighbors are not sampled in Phase I. Then, the probability share is:

$$\sigma_1(a, b, t) * [1 - Pr(a, b, t)] * \frac{E(\hat{\theta}_N) + E(\hat{\theta}_S) + E(\hat{\theta}_W) + E(\hat{\theta}_E)}{4}$$

   where all 4 neighbors can be used for interpolation.

3. 2 neighbors: 2 of its neighbors not sampled in Phase I. Then, the probability share is:

$$\sigma_2(a, b, t) * [1 - Pr(a, b, t)] * \frac{3E(\hat{\theta}_N) + 3E(\hat{\theta}_S) + 3E(\hat{\theta}_W) + 3E(\hat{\theta}_E)}{6 * 2}$$

where 6 different combinations of two neighbors can be used for interpolation and the interpolated RGB is the average of two neighbors' RGBs.

4. 3 neighbors: 1 of its neighbors is not sampled in Phase I. Then, the probability share is:

$$\sigma_3(a,b,t) * [1 - Pr(a,b,t)] * \frac{3E(\hat{\theta}_N) + 3E(\hat{\theta}_S) + 3E(\hat{\theta}_W) + 3E(\hat{\theta}_E)}{4 * 2}$$

where 4 different combinations of two neighbors can be used for interpolation and the interpolated RGB is the average of three neighbors' RGBs.

5. 4 neighbors: no neighbor is suppressed in sampling. Then, the probability share is:

$$\sigma_4(a,b,t) * [1 - Pr(a,b,t)] * \frac{E(\hat{\theta}_N) + E(\hat{\theta}_S) + E(\hat{\theta}_W) + E(\hat{\theta}_E)}{4}$$

where only 1 combination of 4 neighbors can be used for interpolation and the interpolated RGB is the average of 4 neighbors' RGBs.

Similarly, if pixel (a,b,t) is on the border but not at the corner (w.l.o.g., the left border), then we have:

$$\begin{aligned} E[\hat{\theta}(a,b,t)] &= Pr(a,b,t) * \theta(a,b,t) + \sigma_0(a,b,t) * 0 \\ &+ \frac{\sigma_1(a,b,t) * [1 - Pr(a,b,t)] * [E(\hat{\theta}_N) + E(\hat{\theta}_S) + E(\hat{\theta}_E)]}{3} \\ &+ \frac{\sigma_2(a,b,t) * [1 - Pr(a,b,t)] * [2E(\hat{\theta}_N) + 2E(\hat{\theta}_S) + 2E(\hat{\theta}_E)]]}{3 * 2} \\ &+ \frac{\sigma_3(a,b,t) * [1 - Pr(a,b,t)] * [E(\hat{\theta}_N) + E(\hat{\theta}_S) + E(\hat{\theta}_E)]}{3} \end{aligned} \tag{A.1}$$

If pixel (a, b, t) is located at the corner of the tth frame (w.l.o.g., the upper-left corner), then we have:

$$\begin{aligned} E[\hat{\theta}(a,b,t)] &= Pr(a,b,t) * \theta(a,b,t) + \sigma_0(a,b,t) * 0 \\ &+ \frac{\sigma_1(a,b,t) * [1 - Pr(a,b,t)] * [E(\hat{\theta}_S) + E(\hat{\theta}_E)]}{2} \\ &+ \frac{\sigma_2(a,b,t) * [1 - Pr(a,b,t)] * [E(\hat{\theta}_S) + E(\hat{\theta}_E)]]}{2} \end{aligned} \tag{A.2}$$

A.1.2 Solving Algorithm. The optimal number of distinct RGBs k_j (to allocate privacy budget) is computed based on minimizing the MSE expectation of visual element Υ_j (averaged by the number of pixels). Thus, we solve the following optimization (which is equivalent to Equation 3.2):

$$\arg\min_{k_j} \sum_{\forall (a,b,t) \in \Upsilon_j} \left(E[\theta(a,b,t)] - E[\hat{\theta}(a,b,t)] \right)^2$$

$$\begin{cases} E[\hat{\theta}(1,1)] = Pr(1,1)) * \theta(1,1) + \frac{[1-Pr(1,1)]*(\sigma_1(1,1)+\sigma_2(1,1))*(E[\hat{\theta}(1,2)]+E[\hat{\theta}(2,1)])}{2} \\ E[\hat{\theta}(1,2)] = Pr(1,2) * \theta(1,2) + \frac{[1-Pr(1,2)]*(\sigma_1(1,2)+\sigma_2(1,2)+\sigma_3(1,2))*(E[\hat{\theta}(1,1)]+E[\hat{\theta}(2,2)]+E[\hat{\theta}(1,3)])}{3} \\ \quad \vdots \qquad\qquad \vdots \qquad\qquad \vdots \\ \forall a \in (1, A), \forall b \in (1, B) \\ E(\hat{\theta}(a,b)) = Pr(a,b) * \theta(a,b) + \frac{[1-Pr(a,b)]*(\sigma_1(a,b)+\cdots+\sigma_4(a,b))*(E[\hat{\theta}(a-1,b)]+\cdots+E[\hat{\theta}(a,b+1)])}{4} \\ \quad \vdots \qquad\qquad \vdots \qquad\qquad \vdots \\ E(\hat{\theta}(A,B)) = Pr(A,B) * \theta(A,B) + \frac{[1-Pr(A,B)]*(\sigma_1(A,B)+\sigma_2(A,B))*(E[\hat{\theta}(A-1,b)]+E[\hat{\theta}(A,B-1)]}{2} \end{cases}$$

Note that the above equations can be simply extended to all the pixels in Υ_j in all the frames (incorporating frame t). We use the inverse matrix to solve these equations where the coefficients of all the above equations can be represented

as a $|\Upsilon_j| \times |\Upsilon_j|$ matrix (denoted as M). To ensure that the inverse matrix can solve the equations, M should have a full rank $|\Upsilon_j|$. In case that M is not a full rank matrix (indeed, the rank of M is very high since $\forall(a,b,t) \in \Upsilon_j, \sigma_1(a,b,t), \sigma_2(a,b,t), \sigma_3(a,b,t), \sigma_4(a,b,t)$ are somewhat random), we can add a tiny random noise to the non-zero entries in M (in which the deviation is negligible).

Specifically, denoting the expectation of the sth pixel in Υ_j as $E[\hat{\theta}(s)]$ where $s \in [1, AB]$. Then, we have

$$E[\hat{\theta}(s)] = \frac{1}{|M|} * \sum_{i=1}^{AB}[(-1)^{i+s} * M_{is}^{(AB-1)} * b_i] \tag{A.3}$$

where $|M|$ is the determinant of M, $M_{is}^{(AB-1)}$ denotes the sth cofactor (corresponding the sth pixel; including $(AB-1) \times (AB-1)$ entries) and b_i is the ith constant in the equation (in last column of M). Thus, $M_{is}^{(AB-1)}$ can be recursively represented:

$$M_{is}^{(AB-1)} = \sum_{i=1}^{AB}[(-1)^{i+s} * \mathbb{R}_i * M_{is}^{(AB-2)}] \tag{A.4}$$

where $M^{(AB-2)}$ represents the cofactor matrix of M^{AB-1} and $\mathbb{R}_i$ is a random constant (for ensuring full rank for M) which is close to $-\frac{[1-Pr(a,b,t)](\sigma_1(a,b,t)+\sigma_2(a,b,t))}{2}$ for corner pixels, $-\frac{[1-Pr(a,b,t)][\sigma_1(a,b,t)+\sigma_2(a,b,t)+\sigma_3(a,b,t)]}{3}$ for border pixels, and $-\frac{[1-Pr(a,b,t)][\sigma_1(a,b,t)+\sigma_2(a,b,t)+\sigma_3(a,b,t)+\sigma_4(a,b,t)]}{4}$ for non-border pixels. Then, Equation A.4 can be:

$$M_{is}^{(AB-1)} = \sum_{i=1}^{AB}[(-1)^{i+s} * (\prod_{i=1}^{AB-3} \mathbb{R}_i) * M_{is}^{(2)}] \tag{A.5}$$

Since each row of M only has at most 5 non-zero entries (corresponding to the variables of the current pixel and its four/three/two neighbors), we have:

$$E[\hat{\theta}(s)] \approx -\frac{5^{AB-3} * AB}{|M|} * \max_{\forall i \in [1,AB]} \{|\mathbb{R}_i|^{AB-3} * M_{is}^{(2)} * b_i\} \tag{A.6}$$

Thus, we have the MSE expectation in VE Υ_j:

$$\sum_{i=1}^{AB} [\theta(a,b,t) + \frac{5^{AB-3} * AB}{|M|} * \max_{\forall i \in [1,AB]} \{|\mathbb{R}_i|^{AB-3} * M_{is}^{(2)} * b_i\}]^2$$

For each k_j, the corresponding MSE expectation can be computed using the above equation. Then, the optimal k_j can be obtained by traversing k_j in any range. In addition, it is straightforward to prove that the complexity of the inverse matrix based solver is $O(n^3 \log(n))$. Note that we assume that the optimal k_j is computed for minimum MSE based on the first traversal in the interpolation of each visual element. The deviation is very minor since most pixels are interpolated in the first traversal in our experiments. Moreover, the optimal k_j (derived as above) is also experimentally validated (see Figure A.3(d)).

A.2 Budget Allocation Algorithm

A.3 Additional Results

(a) Original (b) $\epsilon = 0.8$ (I) (c) $\epsilon = 1.6$ (I) (d) $\epsilon = 0.8$ (II) (e) $\epsilon = 1.6$ (II)

Figure A.1. Representative Frames in the Random Output Video of PED (available for differentially private queries/analysis)

While evaluating the utility of the sanitized videos in three datasets using two utility measures (KL divergence and MSE), we also fix ϵ and traverse different k for all the visual elements (assigning the same $k \in [4, 30]$). Since optimal k may be

Algorithm 6 Budget Allocation

Input: n **sets of RGBs** $\Psi_j = \{i \in [1, k_j], \widetilde{\theta}_{ij}\}$**, privacy budget** ϵ **for Phase I**

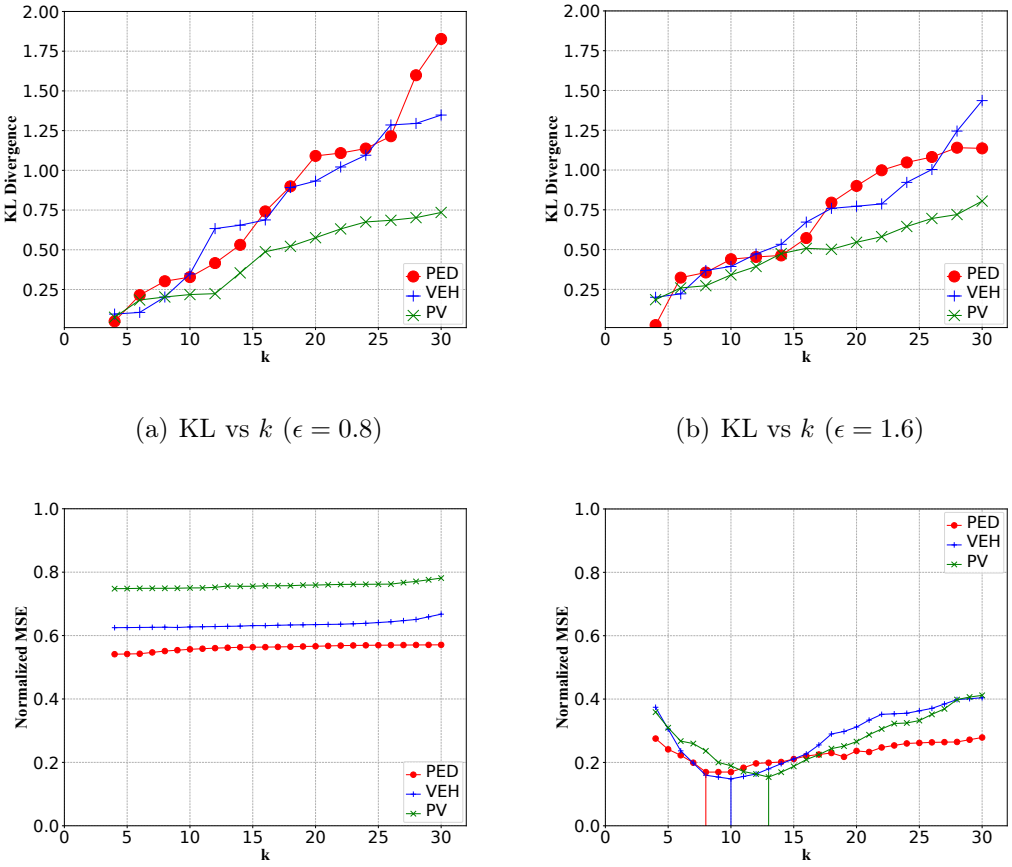
Output: privacy budget for each unique RGB

initialize the set of unique RGBs: $\Psi \leftarrow \bigcup_{j=1}^{n} \Psi_j$
for each $j \in [1, n]$ **do**
  initialize the overall budget for set $\Psi_j : \epsilon(\Psi_j) \leftarrow \epsilon$
end for
for each$\ell \in [1, n]$ **do**
  for each $\widetilde{\theta} \in \Psi$ **do**
    if $count(\widetilde{\theta} \in \{\Psi_1, \ldots, \Psi_n\}) = (n - \ell + 1)$ **then**
      initialize budget for RGB $\widetilde{\theta}$: $\epsilon(\widetilde{\theta})$
      $\epsilon(\widetilde{\theta}) \leftarrow \min_{\forall j \in [1,(n-\ell+1)]} [\frac{d_j(\widetilde{\theta})}{d_j} * \epsilon(\Psi_j)]$
      for each $j \in [1, (n - \ell + 1)], \Psi_j$ **do**
        update budget: $\epsilon(\Psi_j) \leftarrow \epsilon(\Psi_j) - \epsilon(\widetilde{\theta})$
        update total pixel count: $d_j \leftarrow d_j - d_j(\widetilde{\theta})$
      end for
    end if
    return budget $\epsilon(\widetilde{\theta})$ for RGB $\widetilde{\theta}$
    $\Psi \leftarrow \Psi \setminus \widetilde{\theta}$
  end for
end for

different, we use specific videos to see how it affects result. Figure A.3(a) and A.3(b) present the KL divergence values for all the sampled pixels (where privacy budget ϵ is fixed as 0.8 and 1.6, respectively). We can observe that the KL value increases as k increases (if the same number of distinct RGBs in all the visual elements are selected to assign privacy budgets). This is true for the following reason: smaller k samples pixels with less diverse RGBs, but it can allocate a larger privacy budget to

(a) Original (b) $\epsilon = 0.8$ (I) (c) $\epsilon = 1.6$ (I) (d) $\epsilon = 0.8$ (II) (e) $\epsilon = 1.6$ (II)

Figure A.2. Representative Frames in the Random Output Video of VEH (available for differentially private queries/analysis)



(a) KL vs k ($\epsilon = 0.8$)

(b) KL vs k ($\epsilon = 1.6$)

(c) MSE vs k_j (after Phase I)

(d) MSE vs k_j (after Phase II)

Figure A.3. Pixel Level Utility Evaluation with k

each RGB. Then, the generated results can have better count distributions for all the sampled RGBs.

We also examine the *optimal number of selected RGBs to assign privacy bud-*

gets k_j in visual elements. We select the visual element with most pixels in all the videos (PED , VEH and PV). Since the optimal values are derived based on MSEs, we plot the normalized MSEs for *all the pixels in the visual element* for two videos in Figure A.3(c) (after Phase I) and Figure A.3(d) (after Phase II), respectively. The normalized MSE does not change much (after Phase I) as k increases since the MSE expectation is optimized for Phase II. Instead, Figure A.3(d) clearly shows that k_j goes *optimal* in the range (which equals the optimal k_j after solving Equation 3.2 detailed in Appendix A.1) in both videos for all possible values in the specified range. As k_j increases, the normalized MSE of the VE first decreases and then increases. This reflects that the best k_j with respect to the optimal MSE is neither too small nor too large for different VE in all the three videos.

Finally, we present some representative frames of the PED and VEH to show the effectiveness of pixel sampling (Phase I) and utility-driven private video generation (Phase II) in VideoDP. Specifically, we randomly select a frame in video PED and VEH. Figure A.1 and A.2 demonstrate such frames in the input videos and the output videos (after Phase I and II). Figure A.1(b), A.1(c),A.2(b) and A.2(c) demonstrate that more pixels are sampled as private budget ϵ is larger ($\epsilon = 1.6$). Although the portion of the total sampled pixels is not high (after Phase I), the pixel interpolation (Phase II) can reconstruct the video with good quality as shown in Figure A.1(d), A.1(e), A.2(d) and A.2(e). We can observe that the pedestrian/vehicles are randomly generated in the frame (which are not directly revealed to the analysts). More pedestrians/vehicles can be detected as $\epsilon = 1.6$. Then, disclosing the any query/analysis result on such (random) video to analysts satisfies differential privacy.

A.4 Proof

A.4.1 Proof of Convex Property w.r.t. m.

Proof. With the mutual information bound function H, we can take its second order derivative in m as follows:

$$\begin{aligned}\frac{\partial^2 H}{\partial m^2} =& [\frac{1}{c \cdot d - \frac{m \cdot (c-1) \cdot d}{2}} \cdot \log \frac{c}{c - \frac{m \cdot (c-1) \cdot d}{2}}]'' \\ =& \frac{(c-1)^2 d^2}{4(c \cdot d - \frac{(c-1)d}{2} \cdot m)^3} \cdot (2 \log \frac{c}{c \cdot d - \frac{(c-1)d}{2} \cdot m} + 3)\end{aligned}$$

When the first order derivative is equal to zero, we have $m = \frac{2 \cdot (c \cdot d - e^{1+\log c})}{(c-1) \cdot d}$. It is very straightforward to prove that the second order derivative is greater than zero since $(cd - \frac{(c-1)d}{2} m) > 0$ and $2 \log \frac{c}{cd - \frac{(c-1)d}{2} m} + 3 > 0$. Therefore, it is a convex function, and we can derive its minimum value by the derivative. □

A.4.2 Privacy and Utility Analysis.

A.4.2.1 Proof of Theorem 7 (Privacy Analysis).

Proof. For any pair of input locations x, $x' \in \mathcal{D}$ and output y, the maximum perturbation probability $q(y|x)$ for sampling location y based on input x is $\alpha_{max}(x)$ when y is in the same group with x (the first group $G_1(x)$); the minimum perturbation probability $q(y|x')$ for sampling location y based on input x' is $\alpha_{min}(x')$ when y in the furthest group for x' (the last group $G_m(x')$). Thus, the SRR mechanism in L−SRR satisfies $\epsilon = \max_{x,x' \in \mathcal{D}} \log(c \cdot \frac{(m-1)d \cdot c - (c-1) \sum_{j=2}^{m-1} [(j-1) \cdot |G_j(x)|]}{(m-1)d \cdot c - (c-1) \sum_{j=2}^{m-1} [(j-1) \cdot |G_j(x')|]})$-LDP in all the cases, where ϵ is a strict constant privacy bound derived by c and domain $\mathcal{D}$. □

A.4.2.2 Proof of Theorem 9 (L_2 Error Bound).

Proof. With the estimation formula, we have $p(C_x) = p(x)\cdot\sum_{y\in C_x} q(y|x)+\sum_{x'\neq x} p(x')\cdot [\sum_{y\in C_x\setminus C_{x'}} q(y|x') + \sum_{y\in C_x\cap C_{x'}} q(y|x')]$. With the property of Hadamard matrix [50], the size of the set difference between any two location candidate sets is $\frac{d}{4}$, and the size of intersection between any two candidate sets of locations is also $\frac{d}{4}$. We can integrate these into the equation. Then, we have:

$$\begin{aligned} p(C_x) &\geq p(x)\cdot[\sum_{y\in C_x} q(y|x)] + \sum_{x'\neq x} p(x')\cdot\frac{d\cdot\min\{q(y|x')\}}{2} \\ &= p(x)\cdot[\sum_{y\in C_x} q(y|x)] + [1-p(x)]\cdot\frac{d\cdot\alpha_{min}(x)}{2} \\ \implies p(x) &\leq \frac{q(C_x)-\frac{d\cdot\alpha_{min}(x)}{2}}{\sum_{y\in C_x}[q(y|x)-\frac{d\cdot\alpha_{min}(x)}{2}]} \end{aligned}$$

Then, we can have the L_2^2-distance as below:

$$\mathbb{E}[L_2^2(\tilde{p},p)] \leq \ (\frac{1}{\sum_{y\in C_x} q(y|x)-\frac{d\cdot\mu}{2}})^2\cdot\mathbb{E}[L_2^2(\tilde{p}(C),p(C)]$$

where $\mu = \max\{\alpha_{min}(x)\}$. Since $\mathbb{E}[\tilde{p}(C_x)] = \mathbb{E}[\frac{\mathbb{I}\{y_j\in C_x}{n}\}]$ $= p(C_x)$, we have:

$$\mathbb{E}[L_2^2(\tilde{p}(C),p(C))] = \mathbb{E}[\sum_{x\in\mathcal{D}}(\tilde{p}(C_x)-p(C_x))^2] = \sum_{x\in\mathcal{D}} Var(\tilde{p}(C_x))$$

Moreover, each y is independently sampled and $\tilde{p}(C_x) = p(C_x)$ is the mean of n independent multinomial distributions.

$$\sum_{x\in\mathcal{D}} Var(\tilde{p}(C_x)) \leq \sum_{x\in\mathcal{D}}\frac{1}{n}\cdot\max\{p(C_x)\} \leq \frac{d}{n}$$

Thus, we have $\mathbb{E}[L_2(\tilde{p},p)] \leq (\frac{1}{\sum_{y\in C_x} q(y|x)-\frac{d\cdot\mu}{2}})\cdot\sqrt{\frac{d}{n}}$. □

A.4.2.3 Proof of Theorem 8 (L_1 Error Bound).

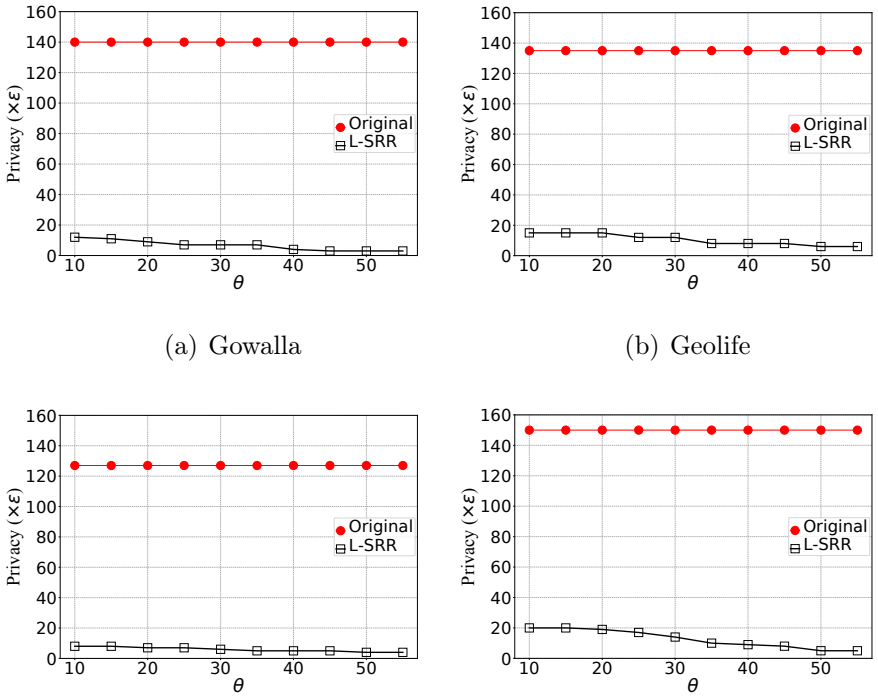
Proof. Since $\forall i, a_i > 0$, $n \cdot \sum_{i=1}^{n}(a_n)^2 \geq [\sum_{i=1}^{n}(a_n)]^2$ holds, we have $d \cdot L_2^2(\tilde{p}, p) \geq [L_1(\tilde{p}, p)]^2$. Then. we can derive:

$$(\mathbb{E}[L_1(\tilde{p}, p)])^2 \leq \frac{d^2}{n \cdot (\gamma - \frac{d \cdot \mu}{2})^2}$$

Thus, $\mathbb{E}[L_1(\tilde{p}, p)] \leq \frac{2d}{\sqrt{n} \cdot (2\gamma - d \cdot \mu)}$ completes the proof. □

A.5 Additional Experiments

A.5.1 Traffic-Aware GPS Navigation. We simulate many trajectories and



(a) Gowalla (b) Geolife

(c) Portocabs (d) Foursquare

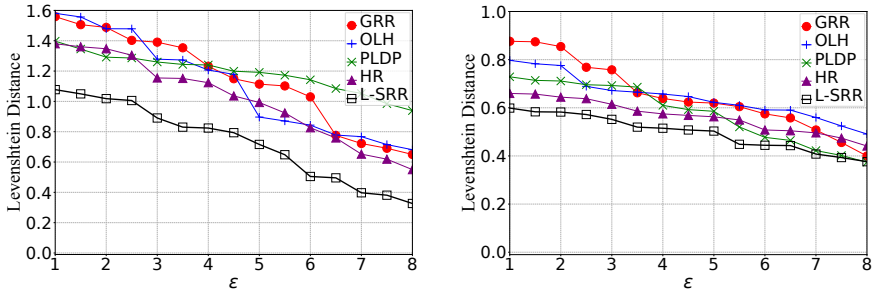
Figure A.4. The total privacy bound of L−SRR for traffic-aware GPS navigation by collecting trajectories

predict the time $Agg^p(x_1,x_2)$ between any two locations on the trajectory using the Markov Chain [167]. Specifically, we generate multiple routes for each OD pair (at

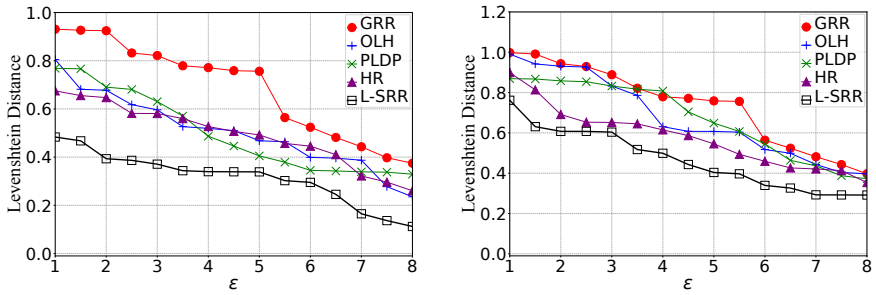
client). For each route, we compute the predicted time t based on historical datasets for any two locations. In our experiment, we use the data collected earlier as the historical data (e.g., Geolife dataset collected in 2009 as the historical data, and collected in 2010 as the test data). Furthermore, for locations on each route, such LBS calculates the frequencies of users near the location within a range (e.g., 4.7m). If the frequency exceeds a threshold (e.g., 50 users), a 3-second delay time will be added [168]. Finally, given the traffic density, it may update the route to avoid heavy traffic. With L−SRR, the route recalculation occurs if $Agg^t(x_o, x_i) - Agg^p(x_o, x_i) > \theta$ holds, where $i \in \mathbb{T}$ and θ is the delay threshold (e.g., 30 seconds). If yes, the client will submit its perturbed location, and privately retrieve the traffic density at the current position to recalculate the fastest route [168].

In the experiments, we first evaluate how the delay time threshold θ affects the total privacy guarantee. The maximum numbers of locations on the trajectories for four datasets are 140, 135, 127, and 150, respectively. In Figure A.4, we set θ between 10 seconds and 55 seconds with a step of 5 seconds. As θ increases, the total privacy bound ϵ decreases with a decreasing number of location updates. As $\theta = 60$ (updating the location once delay exceeds 1 minute), the privacy bound is around 3ϵ, which is very small for trajectories.

Second, to measure the route deviation, we apply Levenshtein distance to measure the accuracy between true route and the route recommended by L−SRR. It measures the difference by calculating the minimum number of location edits (insertions, deletions, or substitutions) required to change one route to the other. Figure A.5 shows the relative Levenshtein distance over the total size of the true routes (vs the total privacy bound ϵ). L−SRR again outperforms other LDP schemes. In addition, we also measure the deviation of the total trip time. Figure A.6 shows that the trip time deviation decreases as the privacy bound ϵ increases for all the LDP



(a) Gowalla (b) Geolife

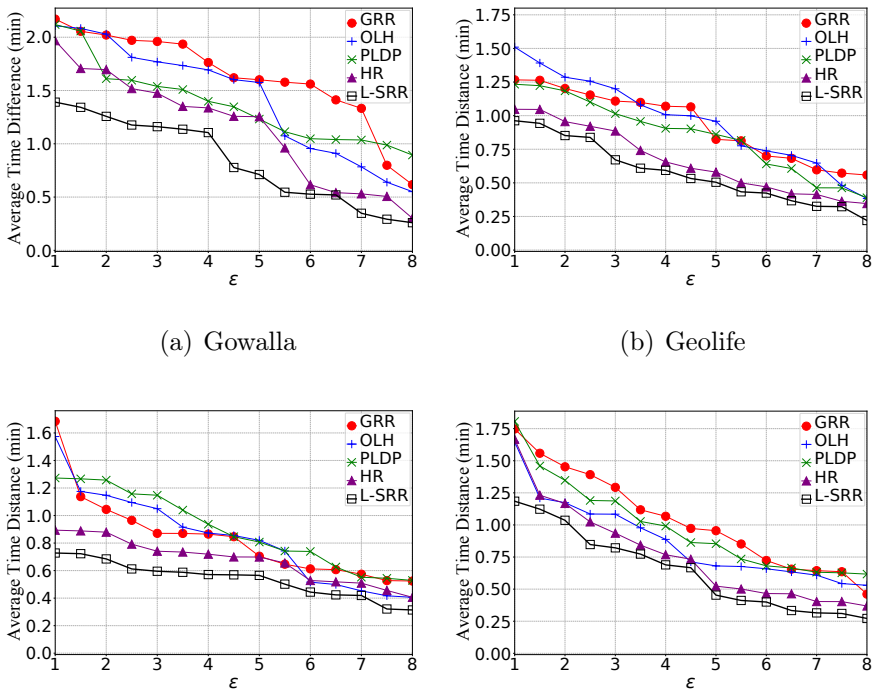


(c) Portocabs (d) Foursquare

Figure A.5. Relative levenshtein distance of trajectories in the traffic-aware GPS navigation ($\theta = 40$ seconds)

schemes, and L−SRR results in the least trip time deviation.

A.5.2 SRR and Differential Privacy. We discuss the utility of centralized differential privacy. A generic solution is to add the Laplace noise to the frequency of each location (after aggregation). Thus, ϵ should be equally allocated for d locations. Table A.1 presents the L_1-distance for the distribution on four datasets using Laplace mechanism. The results show that the L_1-distance gets smaller as ϵ becomes larger. Compared to the LDP mechanism, for Gowalla and Foursquare, the distance with SRR has smaller distance. For Geolife and Portocabs, the distance with SRR has similar distance in case of a smaller domain. Note that the privacy guarantees of LDP and DP are indeed incomparable even for the same ϵ (since the trust model and indistinguishability are defined in different ways).



(a) Gowalla (b) Geolife

(c) Portocabs (d) Foursquare

Figure A.6. Average trip time deviation in the traffic-aware GPS navigation ($\theta = 40$ seconds)

Table A.1. Average L_1-distance for the location distribution on four datasets using Laplace mechanism for centralized DP

Dataset	$\epsilon = 1$	$\epsilon = 3$	$\epsilon = 5$	$\epsilon = 7$
Gowalla	0.18	0.16	0.15	0.13
Geolife	0.29	0.11	0.08	0.04
Portocabs	0.43	0.26	0.15	0.07
Foursquare	13.87	4.69	2.78	1.91

BIBLIOGRAPHY

[1] H. Wang, S. Xie, and Y. Hong, "Videodp: A flexible platform for video analytics with differential privacy," *Proceedings on Privacy Enhancing Technologies*, vol. 2020, no. 4, pp. 277–296, 2020. [Online]. Available: DOI 10.2478/popets-2020-0073

[2] H. Wang, Y. Kong, Y. Hong, and J. Vaidya, "Publishing video data with indistinguishable objects," in *Advances in database technology: proceedings. International Conference on Extending Database Technology*, 2020, pp. 323–334. [Online]. Available: 10.5441/002/edbt.2020.29

[3] H. Wang, H. Hong, L. Xiong, Z. Qin, and Y. Hong, "L-srr: Local differential privacy for location-based services with staircase randomized response," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 2809–2823. [Online]. Available: https://doi.org/10.1145/3548606.3560636

[4] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi–passenger demand using streaming data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1393–1402, 2013.

[5] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *Proceedings of the 18th international conference on World wide web*, 2009, pp. 791–800.

[6] (04/2022). [Online]. Available: http://snap.stanford.edu/data/loc-gowalla.html

[7] D. Yang, B. Qu, J. Yang, and P. Cudre-Mauroux, "Revisiting user mobility and social relationships in lbsns: a hypergraph embedding approach," in *The world wide web conference*, 2019, pp. 2147–2157.

[8] P. Sridharan and S. Raman, "Characteristics of video data for signal analysis," in *Proceedings of Third International Conference on Signal Processing (ICSP'96)*, vol. 2. IEEE, 1996, pp. 1254–1257.

[9] B. Abreu, L. Botelho, and et.al, "Video-based multi-agent traffic surveillance system," in *Intelligent Vehicles Symposium*, 2000, pp. 457–462.

[10] (2012). [Online]. Available: H.R. 6671 (112th): Video Privacy Protection Act

[11] (2019). [Online]. Available: https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm

[12] M. Upmanyu, A. M. Namboodiri, K. Srinathan, and C. V. Jawahar, "Efficient privacy preserving video surveillance," in *Computer Vision*, 2009, pp. 1639–1646.

[13] C. Dwork, "Differential privacy," *Encyclopedia of Cryptography and Security*, pp. 338–340, 2011.

[14] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," *Journal of Privacy and Confidentiality*, vol. 7, no. 3, pp. 17–51, 2016.

[15] J. Vaidya, B. Shafiq, A. Basu, and Y. Hong, "Differentially private naive bayes classification," in *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, vol. 1. IEEE, 2013, pp. 571–576.

[16] J. Wang, S. Liu, Y. Li, H. Cao, and M. Liu, "Differentially private spatial decompositions for geospatial point data," *China Communications*, vol. 13, no. 4, pp. 97–107, 2016.

[17] Y. Hong, J. Vaidya, H. Lu, P. Karras, and S. Goel, "Collaborative search log sanitization: Toward differential privacy and boosted utility," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 5, pp. 504–518, 2014.

[18] W. Qardaji, W. Yang, and N. Li, "Differentially private grids for geospatial data," in *ICDE*, 2013, pp. 757–768.

[19] B. Liu, S. Xie, H. Wang, Y. Hong, X. Ban, and M. Mohammady, "Vtdp: Privately sanitizing fine-grained vehicle trajectory data with boosted utility," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 6, pp. 2643–2657, 2019.

[20] D. Leoni, "Non-interactive differential privacy: a survey," in *Proceedings of the First International Workshop on Open Data*. ACM, 2012, pp. 40–52.

[21] R. Bild, K. A. Kuhn, and F. Prasser, "Safepub: A truthful data anonymization algorithm with strong privacy guarantees," *Proceedings on Privacy Enhancing Technologies*, vol. 2018, no. 1, pp. 67–87, 2018.

[22] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas, "Releasing search queries and clicks privately," in *WWW*, 2009, pp. 171–180.

[23] M. Hay, C. Li, G. Miklau, and D. Jensen, "Accurate estimation of the degree distribution of private networks," in *Data Mining*, 2009, pp. 169–178.

[24] Z. Qin, T. Yu, Y. Yang, I. Khalil, X. Xiao, and K. Ren, "Generating synthetic decentralized social graphs with local differential privacy," in *CCS*. ACM, 2017, pp. 425–438.

[25] L. Fan, "Image pixelization with differential privacy," in *DBSec*, 2018, pp. 148–162.

[26] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, "Certified robustness to adversarial examples with differential privacy," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 656–672.

[27] M. E. Gursoy, L. Liu, S. Truex, L. Yu, and W. Wei, "Utility-aware synthesis of differentially private and attack-resilient location traces," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 196–211.

[28] K. Chatzikokolakis, E. ElSalamouny, C. Palamidessi, and A. Pazii, "Methods for location privacy: A comparative overview," *Foundations and Trends® in Privacy and Security*, vol. 1, no. 4, pp. 199–257, 2017.

[29] Ú. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *CCS*, 2014, pp. 1054–1067.

[30] R. Bassily and A. Smith, "Local, private, efficient protocols for succinct histograms," in *Symposium on Theory of Computing*, 2015, pp. 127–135.

[31] G. Cormode, S. Jha, T. Kulkarni, N. Li, D. Srivastava, and T. Wang, "Privacy at scale: Local differential privacy in practice," in *Management of Data*, 2018, pp. 1655–1658.

[32] M. Gotz, A. Machanavajjhala, G. Wang, X. Xiao, and J. Gehrke, "Publishing search logs—a comparative study of privacy guarantees," *IEEE transactions on knowledge and data engineering*, vol. 24, no. 3, pp. 520–532, 2011.

[33] Y. Hong, J. Vaidya, H. Lu, and M. Wu, "Differentially private search log sanitization with optimal output utility," in *Proceedings of the 15th International Conference on Extending Database Technology*, 2012, pp. 50–61.

[34] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *FOCS*, 2007, pp. 94–103.

[35] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *ICIP*, 2017, pp. 3645–3649.

[36] T. Acharya and A. K. Ray, *Image processing: principles and applications*. John Wiley & Sons, 2005.

[37] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science*, pp. 211–407, 2014.

[38] F. D. McSherry, "Privacy integrated queries: an extensible platform for privacy-preserving data analysis," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. ACM, 2009, pp. 19–30.

[39] Y. Cao, M. Yoshikawa, Y. Xiao, and L. Xiong, "Quantifying differential privacy under temporal correlations," in *ICDE*, 2017, pp. 821–832.

[40] F. Fahroo and I. M. Ross, "Direct trajectory optimization by a chebyshev pseudospectral method," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 1, pp. 160–166, 2002.

[41] P. P. B. Bamba, L. Liu and T. Wang, "Supporting anonymous location queries in mobile environments with privacygrid," in *Proceedings of the 17th international conference on World Wide Web*, 2008, pp. 237–246.

[42] J. W. Kim and B. Jang, "Workload-aware indoor positioning data collection via local differential privacy," *IEEE Communications Letters*, vol. 23, no. 8, pp. 1352–1356, 2019.

[43] X. Zhao, Y. Li, Y. Yuan, X. Bi, and G. Wang, "Ldpart: effective location-record data publication via local differential privacy," *IEEE Access*, vol. 7, pp. 31 435–31 445, 2019.

[44] K. C. M. Andrés, N. Bordenabe and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," in *CCS*, 2013, pp. 901–914.

[45] R. Chen, H. Li, A. K. Qin, S. P. Kasiviswanathan, and H. Jin, "Private spatial data aggregation in the local setting," in *ICDE*, 2016, pp. 289–300.

[46] C. Li, B. Palanisamy, and J. Joshi, "Differentially private trajectory analysis for points-of-interest recommendation," in *2017 IEEE International Congress on Big Data (BigData Congress)*. IEEE, 2017, pp. 49–56.

[47] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, "Private queries in location based services: anonymizers are not necessary," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 2008, pp. 121–132.

[48] X. Yi, R. Paulet, E. Bertino, and V. Varadharajan, "Practical approximate k nearest neighbor queries with location and query privacy," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 6, pp. 1546–1559, 2016.

[49] M. G. Bell, "The estimation of an origin-destination matrix from traffic counts," *Transportation Science*, vol. 17, no. 2, pp. 198–217, 1983.

[50] Z. S. A. Jayadev and H. Zhang, "Hadamard response: Estimating distributions privately, efficiently, and with little communication," in *Proceedings of Machine Learning Research*. PMLR, 2019, pp. 1120–1129.

[51] M. Saini, P. K. Atrey, S. Mehrotra, and M. Kankanhalli, "W3-privacy: understanding what, when, and where inference channels in multi-camera surveillance video," *Multimedia Tools and Applications*, vol. 68, no. 1, pp. 135–158, 2014. [Online]. Available: https://doi.org/10.1007/s11042-012-1207-9

[52] M. Boyle, C. Edwards, and S. Greenberg, "The effects of filtered video on awareness and privacy," in *CSCW*, 2000, pp. 1–10.

[53] A. Senior, S. Pankanti, A. Hampapur, L. Brown, Ying-Li Tian, A. Ekin, J. Connell, Chiao Fe Shu, and M. Lu, "Enabling video privacy through computer vision," *Security Privacy*, pp. 50–57, 2005.

[54] D. A. Fidaleo, H.-A. Nguyen, and M. Trivedi, "The networked sensor tapestry (nest): a privacy enhanced software architecture for interactive analysis of data in video-sensor networks," in *Video surveillance & sensor networks Workshop*, 2004, pp. 46–53.

[55] A. Erdelyi, T. Barat, P. Valet, T. Winkler, and B. Rinner, "Adaptive cartooning for privacy protection in camera networks," in *AVSS*, 2014, pp. 44–49.

[56] S. Gao, J. Ma, W. Shi, G. Zhan, and C. Sun, "Trpf: A trajectory privacy-preserving framework for participatory sensing," *Information Forensics and Security*, vol. 8, no. 6, pp. 874–887, 2013.

[57] T. Winkler and B. Rinner, "Sensor-level security and privacy protection by embedding video content analysis," in *DSP*, 2013, pp. 1–6.

[58] T. Koshimizu, T. Toriyama, and N. Babaguchi, "Factors on the sense of privacy in video surveillance," in *Continuous archival and retrival of personal experences*, 2006, pp. 35–44.

[59] S. J. Oh, R. Benenson, M. Fritz, and B. Schiele, "Faceless person recognition: Privacy implications in social media," in *European Conference on Computer Vision*, 2016, pp. 19–35.

[60] R. McPherson, R. Shokri, and V. Shmatikov, "Defeating image obfuscation with deep learning," *arXiv preprint arXiv:1609.00408*, 2016.

[61] S. Moncrieff, S. Venkatesh, and G. West, "Dynamic privacy assessment in a smart house environment using multimodal sensing," *TOMM*, p. 10, 2008.

[62] M. Mohammady, S. Xie, Y. Hong, M. Zhang, L. Wang, M. Pourzandi, and M. Debbabi, "R2dp: A universal and automated approach to optimizing the randomization mechanisms of differential privacy for utility metrics with no known optimal distributions," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 677–696.

[63] F. Cangialosi, N. Agarwal, V. Arun, S. Narayana, A. Sarwate, and R. Netravali, "Privid: Practical,{Privacy-Preserving} video analytics queries," in *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, 2022, pp. 209–228.

[64] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux, "Quantifying location privacy," in *2011 IEEE symposium on security and privacy*. IEEE, 2011, pp. 247–262.

[65] D. Riboni, L. Pareschi, and C. Bettini, "Privacy in georeferenced context-aware services: A survey," in *Privacy in location-based applications*. Springer, 2009, pp. 151–172.

[66] A. Machanavajjhala, D. Kifer, J. Abowd, J. Gehrke, and L. Vilhuber, "Privacy: Theory meets practice on the map," in *2008 IEEE 24th international conference on data engineering*. IEEE, 2008, pp. 277–286.

[67] S.-S. Ho and S. Ruan, "Differential privacy for location pattern mining," in *Proceedings of the 4th ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS*, 2011, pp. 17–24.

[68] L. Yu, L. Liu, and C. Pu, "Dynamic differential location privacy with personalized error bounds." in *NDSS*, 2017.

[69] X. He, G. Cormode, A. Machanavajjhala, C. M. Procopiuc, and D. Srivastava, "Dpt: differentially private trajectory synthesis using hierarchical reference systems," *Proceedings of the VLDB Endowment*, vol. 8, no. 11, pp. 1154–1165, 2015.

[70] T. Wang, N. Li, and S. Jha, "Locally differentially private frequent itemset mining," in *SP*, 2018, pp. 127–143.

[71] (2012). [Online]. Available: YouTube Official Blog 2012

[72] S. Hill, Z. Zhou, L. Saul, and H. Shacham, "On the (in) effectiveness of mosaicing and blurring as tools for document redaction," *Privacy Enhancing Technologies*, no. 4, pp. 403–417, 2016.

[73] L. Sweeney, "Achieving k-anonymity privacy protection using generalization and suppression," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 571–588, 2002.

[74] N. Li, W. Qardaji, and D. Su, "On sampling, anonymization, and differential privacy or, k-anonymization meets differential privacy," in *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*. ACM, 2012, pp. 32–33.

[75] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[76] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *international Conference on computer vision & Pattern Recognition (CVPR'05)*, vol. 1. IEEE Computer Society, 2005, pp. 886–893.

[77] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Occlusion-aware r-cnn: detecting pedestrians in a crowd," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 637–653.

[78] A. Machanavajjhala, D. Kifer, J. Abowd, J. Gehrke, and L. Vilhuber, "Privacy: Theory meets practice on the map," in *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*. IEEE Computer Society, 2008, pp. 277–286.

[79] K. Nissim, S. Raskhodnikova, and A. Smith, "Smooth sensitivity and sampling in private data analysis," in *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, 2007, pp. 75–84.

[80] (04/2022). [Online]. Available: https://drive.google.com/file/d/1hYa5s7fjvQc1S1wRY6GcRqOPwL0Hy_aE/view

[81] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "Mot16: A benchmark for multi-object tracking," *CoRR*, 2016.

[82] Y. Yang, J. Liu, and M. Shah, "Video scene understanding using multi-scale analysis," in *2009 IEEE 12th International Conference on Computer Vision*. IEEE, 2009, pp. 1669–1676.

[83] D. Doma, *Comparison of Different Image Interpolation Algorithms*. West Virginia University Libraries, 2008.

[84] O. Chapelle, P. Haffner, and V. N. Vapnik, "Support vector machines for histogram-based image classification," *IEEE transactions on Neural Networks*, vol. 10, no. 5, pp. 1055–1064, 1999.

[85] Z.-Q. Hong, "Algebraic feature extraction of image for recognition," *Pattern recognition*, vol. 24, no. 3, pp. 211–219, 1991.

[86] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *2005 IEEE international workshop on visual surveillance and performance evaluation of tracking and surveillance*. IEEE, 2005, pp. 65–72.

[87] Z. He, J. Zhang, M. Kan, S. Shan, and X. Chen, "Robust fec-cnn: A high accuracy facial landmark detection system," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 98–104.

[88] S. Song, Y. Wang, and K. Chaudhuri, "Pufferfish privacy mechanisms for correlated data," in *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM, 2017, pp. 1291–1306.

[89] Y. Cao, M. Yoshikawa, Y. Xiao, and L. Xiong, "Quantifying differential privacy under temporal correlations," in *ICDE*, 2017, pp. 821–832.

[90] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," in *2008 IEEE conference on computer vision and pattern recognition*. IEEE, 2008, pp. 1–7.

[91] (04/2022). [Online]. Available: https://boxy-dataset.com/boxy/

[92] (04/2022). [Online]. Available: https://opencv.org/.

[93] A. Hore and D. Ziou, "Image quality metrics: Psnr vs. ssim," in *2010 20th international conference on pattern recognition*. IEEE, 2010, pp. 2366–2369.

[94] P. Piccinini, A. Prati, and R. Cucchiara, "Real-time object detection and localization with sift-based clustering," *Image and Vision Computing*, vol. 30, no. 8, pp. 573–587, 2012.

[95] J. Yang, B. Price, S. Cohen, H. Lee, and M.-H. Yang, "Object contour detection with a fully convolutional encoder-decoder network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 193–202.

[96] M. Handte, M. U. Iqbal, and et. al, "Crowd density estimation for public transport vehicles," in *EDBT/ICDT Workshops*, 2014, pp. 315–322.

[97] N. Wojke and A. Bewley, "Deep cosine metric learning for person re-identification," in *WACV*, 2018, pp. 748–756.

[98] Q. Zhu, M.-C. Yeh, K.-T. Cheng, and S. Avidan, "Fast human detection using a cascade of histograms of oriented gradients," in *CVPR*, 2006, pp. 1491–1498.

[99] F. Han, Y. Shan, R. Cekander, H. S. Sawhney, and R. Kumar, "A two-stage approach to people and vehicle detection with hog-based svm," in *Performance Metrics for Intelligent Systems*, 2006, pp. 133–140.

[100] K. Karsch, V. Hedau, D. A. Forsyth, and D. Hoiem, "Rendering synthetic objects into legacy photographs," *Trans. Graph.*, p. 157, 2011.

[101] A. Toshev, A. Makadia, and K. Daniilidis, "Shape-based object recognition in videos using 3d synthetic object models," in *CVPR*, 2009.

[102] R. Yarovoy, F. Bonchi, L. V. Lakshmanan, and W. H. Wang, "Anonymizing moving objects: How to hide a mob in a crowd?" in *Extending Database Technology: Advances in Database Technology*, 2009, pp. 72–83.

[103] T. Wang, J. Blocki, N. Li, and S. Jha, "Locally differentially private protocols for frequency estimation," in *USENIX*, 2017, pp. 729–745.

[104] V. Bindschaedler, R. Shokri, and C. A. Gunter, "Plausible deniability for privacy-preserving data synthesis," *VLDB*, pp. 481–492, 2017.

[105] X. Li, K. Wang, Y. Tian, L. Yan, F. Deng, and F. Wang, "The paralleleye dataset: A large collection of virtual images for traffic vision research," *Intelligent Transportation Systems*, pp. 2072–2084, 2019.

[106] A. Criminisi, P. Pérez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Transactions on image processing*, vol. 13, no. 9, pp. 1200–1212, 2004.

[107] Z. Qin, Y. Yang, T. Yu, I. Khalil, X. Xiao, and K. Ren, "Heavy hitter estimation over set-valued data with local differential privacy," in *CCS*. ACM, 2016, pp. 192–203.

[108] H. Feng, W. Fang, S. Liu, and Y. Fang, "A new general framework for shot boundary detection and key-frame extraction," in *Multimedia information retrieval*, 2005, pp. 121–126.

[109] A. Divakaran, R. Radhakrishnan, and K. A. Peker, "Motion activity-based extraction of key-frames from video shots," in *ICIP*, 2002, pp. 932–935.

[110] S. K. Kuanar, R. Panda, and A. S. Chowdhury, "Video key frame extraction through dynamic delaunay clustering with a structural constraint," *Visual Communication and Image Representation*, pp. 1212–1227, 2013.

[111] L. Pan, X.-J. Wu, and Y.-Y. You, "Video shot segmentation and key frame extraction based on clustering," *Infrared and Laser Engineering*, vol. 34, no. 3, p. 341, 2005.

[112] R. Kannan and C. L. Monma, "On the computational complexity of integer programming problems," in *Optimization and Operations Research*, 1978, pp. 161–172.

[113] G. Scaglia, A. Rosales, L. Quintero, V. Mut, and R. Agarwal, "A linear-interpolation-based controller design for trajectory tracking of mobile robots," *Control Engineering Practice*, vol. 18, no. 3, pp. 318–329, 2010.

[114] E. Frentzos, K. Gratsias, N. Pelekis, and Y. Theodoridis, "Algorithms for nearest neighbor search on moving object trajectories," *Geoinformatica*, vol. 11, no. 2, pp. 159–193, 2007.

[115] H. H. Arcolezi, J.-F. Couchot, B. A. Bouna, and X. Xiao, "Improving the utility of locally differentially private protocols for longitudinal and multidimensional frequency estimates," *arXiv preprint arXiv:2111.04636*, 2021.

[116] (04/2022). [Online]. Available: https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pdf

[117] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," *arXiv preprint arXiv:1712.01524*, 2017.

[118] Z. Li, T. Wang, M. Lopuhaä-Zwakenberg, N. Li, and B. Škoric, "Estimating numerical distributions under local differential privacy," in *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, 2020, pp. 621–635.

[119] Y. Wang, X. Wu, and D. Hu, "Using randomized response for differential privacy preserving data collection." in *EDBT/ICDT Workshops*, vol. 1558, 2016, pp. 0090–6778.

[120] P. Shankar, V. Ganapathy, and L. Iftode, "Privately querying location-based services with sybilquery," in *Proceedings of the 11th international conference on Ubiquitous computing*, 2009, pp. 31–40.

[121] T. H. Dao, S. R. Jeong, and H. Ahn, "A novel recommendation model of location-based advertising: Context-aware collaborative filtering using ga approach," *Expert Systems with Applications*, vol. 39, no. 3, pp. 3731–3739, 2012.

[122] S. Shang, H. Lu, T. B. Pedersen, and X. Xie, "Finding traffic-aware fastest paths in spatial networks," in *International Symposium on Spatial and Temporal Databases*. Springer, 2013, pp. 128–145.

[123] D. Asonov and J. C. Freytag, "Almost optimal private information retrieval," in *International Workshop on PETs*, 2002, pp. 209–223.

[124] L. Li, R. Lu, and C. Huang, "Eplq: Efficient privacy-preserving location-based query over outsourced encrypted data," *IEEE Internet of Things Journal*, vol. 3, no. 2, pp. 206–218, 2015.

[125] S. Chen, X. Zhang, M. K. Reiter, and Y. Zhang, "Detecting privileged side-channel attacks in shielded execution with déjá vu," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, 2017, pp. 7–18.

[126] S. Wang, L. Huang, P. Wang, H. Deng, H. Xu, and W. Yang, "Private weighted histogram aggregation in crowdsourcing," in *International Conference on Wireless Algorithms, Systems, and Applications*. Springer, 2016, pp. 250–261.

[127] X. Gu, M. Li, L. Xiong, and Y. Cao, "Providing input-discriminative protection for local differential privacy," in *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 2020, pp. 505–516.

[128] Q. Geng, P. Kairouz, S. Oh, and P. Viswanath, "The staircase mechanism in differential privacy," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 7, pp. 1176–1184, 2015.

[129] (04/2022). [Online]. Available: https://docs.microsoft.com/en-us/bingmaps/articles/bing-maps-tile-system

[130] P. J. Van Laarhoven and E. H. Aarts, "Simulated annealing," in *Simulated annealing: Theory and applications*. Springer, 1987, pp. 7–15.

[131] A. McGregor, I. Mironov, T. Pitassi, O. Reingold, K. Talwar, and S. Vadhan, "The limits of two-party differential privacy," in *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE, 2010, pp. 81–90.

[132] S. Wang, L. Huang, P. Wang, Y. Nie, H. Xu, W. Yang, X.-Y. Li, and C. Qiao, "Mutual information optimally local private discrete distribution estimation," *arXiv preprint arXiv:1607.08025*, 2016.

[133] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. IEEE, 2013, pp. 429–438.

[134] T. Murakami and Y. Kawamoto, "Utility-optimized local differential privacy mechanisms for distribution estimation," in *28th {USENIX} Security Symposium ({USENIX} Security 19)*, 2019, pp. 1877–1894.

[135] C. Camarero, "Simple, fast and practicable algorithms for cholesky, lu and qr decomposition using fast rectangular matrix multiplication," *arXiv preprint arXiv:1812.02056*, 2018.

[136] C. Ozcan and B. Sen, "Investigation of the performance of lu decomposition method using cuda," *Procedia Technology*, vol. 1, pp. 50–54, 2012.

[137] M. E. Gursoy, A. Tamersoy, S. Truex, W. Wei, and L. Liu, "Secure and utility-aware data collection with condensed local differential privacy," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[138] P. Indyk and D. Woodruff, "Polylogarithmic private approximations and efficient matching," in *Theory of Cryptography Conference*. Springer, 2006, pp. 245–264.

[139] X. Ren, C.-M. Yu, W. Yu, S. Yang, X. Yang, J. A. McCann, and S. Y. Philip, "LoPub: High-dimensional crowdsourced data publication with local differential privacy," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 9, pp. 2151–2166, 2018.

[140] M. Joseph, A. Roth, J. Ullman, and B. Waggoner, "Local differential privacy for evolving data," *arXiv preprint arXiv:1802.07128*, 2018.

[141] Ú. Erlingsson, V. Feldman, I. Mironov, A. Raghunathan, K. Talwar, and A. Thakurta, "Amplification by shuffling: From local to central differential privacy via anonymity," in *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2019, pp. 2468–2479.

[142] N. Wang, X. Xiao, Y. Yang, T. D. Hoang, H. Shin, J. Shin, and G. Yu, "Privtrie: Effective frequent term discovery under local differential privacy," in *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 2018, pp. 821–832.

[143] K. Keahey, J. Anderson, Z. Zhen, P. Riteau, P. Ruth, D. Stanzione, M. Cevik, J. Colleran, H. S. Gunawi, C. Hammock *et al.*, "Lessons learned from the chameleon testbed," in *2020 {USENIX} Annual Technical Conference ({USENIX}{ATC} 20)*, 2020, pp. 219–233.

[144] T. Wang, B. Ding, J. Zhou, C. Hong, Z. Huang, N. Li, and S. Jha, "Answering multi-dimensional analytical queries under local differential privacy," in *Proceedings of the 2019 International Conference on Management of Data*, 2019, pp. 159–176.

[145] M. Hay, V. Rastogi, G. Miklau, and D. Suciu, "Boosting the accuracy of differentially-private histograms through consistency," *arXiv preprint arXiv:0904.0942*, 2009.

[146] C. Ozkurt and F. Camci, "Automatic traffic density estimation and vehicle classification for traffic surveillance systems using neural networks," *Mathematical and Computational Applications*, vol. 14, no. 3, pp. 187–196, 2009.

[147] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE symposium on security and privacy (SP)*. IEEE, 2017, pp. 3–18.

[148] S. Xie, H. Wang, Y. Kong, and Y. Hong, "Universal 3-dimensional perturbations for black-box attacks on video recognition systems," in *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022, pp. 1390–1407.

[149] S. Xie, Y. Yan, and Y. Hong, "Stealthy 3d poisoning attack on video recognition models," *IEEE Transactions on Dependable and Secure Computing*, 2022.

[150] H. Hong, B. Wang, and Y. Hong, "Unicr: Universally approximated certified robustness via randomized smoothing," in *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part V*, ser. Lecture Notes in Computer Science, S. Avidan, G. J. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds., vol. 13665. Springer, 2022, pp. 86–103. [Online]. Available: https://doi.org/10.1007/978-3-031-20065-6_6

[151] H. Hong and Y. Hong, "Certifiable black-box attack: Ensuring provably successful attack for adversarial examples," *CoRR*, vol. abs/2304.04343, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2304.04343

[152] ——, "Certified adversarial robustness via anisotropic randomized smoothing," *CoRR*, vol. abs/2207.05327, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2207.05327

[153] H. Wang, J. Sharma, S. Feng, K. Shu, and Y. Hong, "A model-agnostic approach to differentially private topic mining," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 1835–1845.

[154] S. Xie and Y. Hong, "Differentially private instance encoding against privacy attacks," in *In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Student Research Workshop*, 2022.

[155] M. Mohammady, H. Wang, L. Wang, M. Zhang, Y. Jarraya, S. Majumdar, M. Pourzandi, M. Debbabi, and Y. Hong, "DPOAD: differentially private outsourcing of anomaly detection through iterative sensitivity learning," *CoRR*, vol. abs/2206.13046, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2206.13046

[156] L. Ou, Z. Qin, S. Liao, Y. Hong, and X. Jia, "Releasing correlated trajectories: Towards high utility and optimal differential privacy," *IEEE Trans. Dependable Secur. Comput.*, vol. 17, no. 5, pp. 1109–1123, 2020. [Online]. Available: https://doi.org/10.1109/TDSC.2018.2853105

[157] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[158] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, "Certified robustness to adversarial examples with differential privacy," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 656–672.

[159] M. Mohammady, L. Wang, Y. Hong, H. Louafi, M. Pourzandi, and M. Debbabi, "Preserving both privacy and utility in network trace anonymization," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, D. Lie, M. Mannan, M. Backes, and X. Wang, Eds. ACM, 2018, pp. 459–474. [Online]. Available: https://doi.org/10.1145/3243734.3243809

[160] B. Liu, S. Xie, and Y. Hong, "PANDA: privacy-aware double auction for divisible resources without a mediator," in *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020*, A. E. F. Seghrouchni, G. Sukthankar, B. An, and N. Yorke-Smith, Eds. International Foundation for Autonomous

Agents and Multiagent Systems, 2020, pp. 1904–1906. [Online]. Available: https://dl.acm.org/doi/10.5555/3398761.3399022

[161] S. Xie, H. Wang, Y. Hong, and M. Thai, "Privacy preserving distributed energy trading," in *40th IEEE International Conference on Distributed Computing Systems, ICDCS 2020, Singapore, November 29 - December 1, 2020*. IEEE, 2020, pp. 322–332. [Online]. Available: https://doi.org/10.1109/ICDCS47774.2020.00078

[162] S. Xie, Y. Hong, and P. Wan, "Pairing: Privately balancing multiparty real-time supply and demand on the power grid," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 1114–1127, 2020. [Online]. Available: https://doi.org/10.1109/TIFS.2019.2933732

[163] S. Xie, B. Liu, and Y. Hong, "Privacy-preserving cloud-based DNN inference," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*. IEEE, 2021, pp. 2675–2679. [Online]. Available: https://doi.org/10.1109/ICASSP39728.2021.9413820

[164] S. Xie, M. Mohammady, H. Wang, L. Wang, J. Vaidya, and Y. Hong, "A generalized framework for preserving both privacy and utility in data outsourcing (extended abstract)," in *38th IEEE International Conference on Data Engineering, ICDE 2022, Kuala Lumpur, Malaysia, May 9-12, 2022*. IEEE, 2022, pp. 1549–1550. [Online]. Available: https://doi.org/10.1109/ICDE53745.2022.00151

[165] B. Liu, R. Wang, Z. Ba, S. Zhou, C. Ding, and Y. Hong, "Poster: Cryptographic inferences for video deep neural networks," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, H. Yin, A. Stavrou, C. Cremers, and E. Shi, Eds. ACM, 2022, pp. 3395–3397. [Online]. Available: https://doi.org/10.1145/3548606.3563543

[166] S. Xie, M. Mohammady, H. Wang, L. Wang, J. Vaidya, and Y. Hong, "A generalized framework for preserving both privacy and utility in data outsourcing," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 1–15, 2023. [Online]. Available: https://doi.org/10.1109/TKDE.2021.3078099

[167] D. Woodard, G. Nogin, P. Koch, D. Racz, M. Goldszmidt, and E. Horvitz, "Predicting travel time reliability using mobile phone gps data," *Transportation Research Part C: Emerging Technologies*, vol. 75, pp. 30–44, 2017.

[168] U. Demiryurek, F. Banaei-Kashani, C. Shahabi, and A. Ranganathan, "Online computation of fastest path in time-dependent spatial networks," in *International Symposium on Spatial and Temporal Databases*. Springer, 2011, pp. 92–111.