ABSTRACT

Millions of videos are ubiquitously generated and shared everyday. Releasing videos would be greatly beneficial to social interactions and the community but may result in severe privacy concerns. To the best of our knowledge, most of the existing privacy preserving techniques for video data focus on detecting and blurring the sensitive regions in the video. Such simple privacy models have two major limitations: (1) they cannot quantify and bound the privacy risks, and (2) they cannot address the inferences drawn from the background knowledge on the involved objects in the videos. In this paper, we first define a novel privacy notion $\epsilon$-Object Indistinguishability for all the predefined sensitive objects (e.g., humans, vehicles) in the video, and then propose a video sanitization technique Verrx 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. We have conducted extensive experiments on three real video captured for pedestrians on the streets. The experimental results demonstrate that the generated synthetic videos lie close to the original video for retaining good utility while ensuring rigorous privacy guarantee.

KEYWORDS

Privacy, Video, Local Differential Privacy, Indistinguishability

1 INTRODUCTION

Millions of videos are ubiquitously generated and shared everyday via dedicated facilities (e.g., video surveillance devices, traffic cameras), smart phones, among others. Sharing such video data would greatly benefit human interactions and the community. For instance, surveillance cameras in offices and other business establishments capture potential threats to the corporate assets (such videos are shared for analysis in many cases [46]). Traffic videos contribute to monitoring the street traffic and traffic analysis applications such as vehicle counting and traffic flow analysis [3] as well as pedestrian behavior analysis.

However, these scenarios have often raised severe privacy concerns since individuals’ faces, bodies, identities, activities and other sensitive information can be recorded in the captured videos [8, 46]. Thousands of vehicles are involved in a traffic monitoring video (over a short period) and not every vehicle owner is willing to share the visual information (e.g., vehicle plate, make and model), the extracted traveling data (e.g., locations and driving speed) [2]. Another example is video surveillance systems 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 [46]. 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.

To this end, we first define a novel privacy notion $\epsilon$-Object Indistinguishability for all the predefined sensitive objects (e.g., humans, vehicles) in the video, and then propose a video sanitization technique Verrx 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. We have conducted extensive experiments on three real video captured for pedestrians on the streets. The experimental results demonstrate that the generated synthetic videos lie close to the original video for retaining good utility while ensuring rigorous privacy guarantee.
Contributions. With the privacy notion of \( \epsilon \)-Object Indistinguishability\(^1\), 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 \( \epsilon \)-Object Indistinguishability and good utility.

More specifically, we design a novel random response scheme (by optimizing the RAPPOR [16]) that randomly generates different objects in the video by maximizing the utility of random response [16] applied to the objects. Thus, we name our proposed technique as "Video with Randomly Responded Objects (Verro)". 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 [17] (without additional privacy leakage [14], as analyzed in Section 4). Thus, the synthetic video can be disclosed to any untrusted recipient. Finally, we summarize the major contributions of this paper as below:

- To the best of our knowledge, we define the first rigorous privacy notion for all the sensitive objects (predefined by the video owner) in the video data, which ensures that all the objects are indistinguishable in the randomized output video against arbitrary background knowledge.
- We propose a novel video sanitization technique Verro that randomly generates utility-driven synthetic videos in which any two sensitive objects are \( \epsilon \)-Object Indistinguishable.\(^1\) The video owner can also specify its privacy budget \( \epsilon \) for all the objects in its video.
- The proposed novel random response scheme (by optimizing RAPPOR [16]) in Verro that optimally picks frames of the video to randomly generate objects (while satisfying indistinguishability). The utility of the synthetic video is further improved using computer vision techniques.
- We have conducted extensive experiments on real videos to validate the performance of Verro. The experimental results demonstrate that Verro can effectively generate private synthetic videos with high utility.

The remainder of this paper is organized as follows. Section 2 introduces some preliminaries. Section 3 illustrates the first phase of Verro and analyzes the privacy guarantee. Section 4 presents the second phase of Verro (for further boosting the utility) and its privacy guarantee. Section 5 gives discussions for Verro. Section 6 demonstrates the experimental results. Section 7 and 8 present the literature and conclusions.

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.

\(^1\)As formally defined in Definition 2.1, \( \epsilon \)-Object Indistinguishability ensures a similar privacy guarantee as \( \epsilon \)-local differential privacy [4, 10, 16].
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 [4, 10, 16], 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 [10, 16, 48]. Migrating the LDP model to the objects in any video \( V \), we define the ε-Object Indistinguishability as below:

**Definition 2.1 (ε-Object Indistinguishability).** A randomization algorithm \( \mathcal{A} \) satisfies ε-Object Indistinguishability, if and only if for any two input objects \( O_i, O_j \in \mathcal{G} \) in the input video \( V \), and for any output object of \( \mathcal{A} \) in the synthetic video \( V' \) (denoted as \( y \)), we have \( \Pr[\mathcal{A}(O) = y] \leq e^\epsilon \cdot \Pr[\mathcal{A}(O) = y] \).

Notice that, similar to ε-LDP [16], 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 \( \epsilon \) decides the degree of indistinguishability (which is identical to LDP [16]).

Definition 2.1 guarantees that the randomly perturbed output of any two objects in \( V \) (both the object contents and the trajectories in all the frames) are ε-indistinguishable in \( V' \). It also ensures plausible deniability for every object [5]. 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 \( V' \). 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) [32].

### 2.3 Verro Framework

We now illustrate the major components of Verro:

1. **Preprocessing:** all the objects are detected and tracked, and background scene (for each frame) is extracted using computer vision techniques [11, 51, 52].

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 [16]) 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 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 [14]), and then Verro satisfies ε-Object Indistinguishability. Details are given in Section 4.

### 3 PHASE I: OPTIMAL OBJECT PRESENCE

As the “local data” of each object (e.g., a pedestrian or vehicle) in the video \( V \), 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.

#### 3.1 Poor Utility with Random Response (i.e., RAPPOR [16]) 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 3.1 (Object Presence Vector).** Given video \( V' \) which includes \( m \) different frames \( F_1, \ldots, F_m \) and \( n \) distinct objects \( \mathcal{O} = \{O_1, \ldots, O_n\} \), whether each object \( O_i \in [1, n] \) is present in frame \( F_k, k \in [1, m] \) or not (all \( m \) frames) can form a bit vector: \( B_i = (b_i^1, \ldots, b_i^m) \in \{0, 1\}^m \) for object \( O_i \).

It has been proven that a classic randomized response (RR) technique (e.g., RAPPOR [4, 16]) 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 3.2 and 3.3). For each object \( O_i \in \mathcal{O}, i \in [1, n] \), if object \( O_i \) exists in frame \( F_k \), we set \( b_i^k = 1 \). 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.
Then, we have (of all the objects) can be indistinguishable. Algorithm 1 \[16\] shows the details.

**Algorithm 1 Random Response for Object Presence \[16\]**

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

**Theorem 3.2.** Algorithm 1 randomly generates object presence vectors for objects with \(\epsilon\)-Object Indistinguishability.

**Proof.** \(\epsilon\)-Object Indistinguishability can be proven by following the proof of \(\epsilon\)-LDP with random response \[16\]. 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 \(O_i, O_j \in O\), for any possible output \(m\)-bit vector \(y = (y^1, \ldots, y^m)\), we have:

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

(1)

Since each bit is allocated with an equal privacy budget \(\epsilon/m\) \[16\]. For \(k \in [1, m]\), if \(b^k_i = b^k_j\) (either 0 or 1), then \(Pr(b_i^k = y^k)/Pr(b_j^k = y^k)\) always equals 1. If \(b^k_i \neq b^k_j\) and \(b^k = y^k\), thus we have:

\[
\frac{Pr(b_i^k = y^k)}{Pr(b_j^k = y^k)} = \frac{\epsilon/m}{1 + \epsilon/m} = \epsilon/m
\]

(2)

Similarly, if \(b^k_i \neq b^k_j\) and \(b^j = y^k\), we have \(Pr(b_i^k = y^k)/Pr(b_j^k = y^k)\) equals \(\epsilon/m\). Combining all \(m\) bits, we have:

\[
\frac{Pr[A(B_i) = y]}{Pr[A(B_j) = y]} \leq \epsilon^m
\]

(3)

Thus, the generated presence bit vectors satisfy \(\epsilon\)-Object Indistinguishability. This completes the proof. \(\square\)

**Poor Utility.** Although Algorithm 1 satisfies \(\epsilon\)-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 \[16\]). 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 3.2 and 3.3.

**3.2 Dimension Reduction in the Video**

Recall that the limited utility in Algorithm 1 results from the high dimensions in the video (considering each frame as a dimension). Most existing LDP techniques (e.g., RAPPOR \[16\], succinct histogram \[4\], LDPMiner \[40\], FLDP \[9\]) 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 \[40\], Johnson-Lindenstrauss transform reduces the dimension of location data \[9\]). In videos, since difference between two consecutive frames is very small, we extract the key frames \[12, 19, 31\] out of \(m\) frames from \(V\) to reduce dimension in \(V\).

**3.2.1 Key Frame Extraction.** In computer vision, many existing shot detection and key frame extraction algorithms have been proposed based on the boundary method \[19\], motion analysis \[12\], clustering \[31\], among others. Since algorithms based on clustering has been shown to generate more accurate results \[31\], we integrate it into \(V\) for dimension reduction. The basic idea is to divide the video into several groups and the frames in same group are similar. The algorithm \[38\] 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 2.

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 \(\ell\)-bit vectors. For instance, key frames \(F_1, \ldots, F_\ell\) (where \(\ell\) denotes the number of key frames, and \(\ell \ll m\) in general) are extracted from \(V\). Object \(O_i\)’s presence vector \(B_i\) can be reduced to \(B_i' = (k^1_{i}, \ldots, k^\ell_{i})\).

**Algorithm 2 Segmentation and Key Frame Extraction**

1: initialize the first segment \(S_1 = F_1\), segment index \(i = 1\)
2: equally partition \(H, S, V\) value ranges to \(h, s\) and \(v\) parts
3: for each frame \(F_k, k \in [2, m]\) do
4: for each part \(h, s, v\) in \(H, S, V\) do
5: construct the histograms \(H(\hat{h}), S(\hat{s}), V(\hat{v})\) in frame \(F_k\)
6: end for
7: SimH(F_k, S_i) = \min_{h_i} \{(H(h) \log H(\hat{h}) + S(h_i) \log S(\hat{s}))\}
8: SimS(F_k, S_i) = \min_{s_i} \{(S(h_i) \log S(\hat{s}) \log S(\hat{s}))\}
9: SimV (F_k, S_i) = \min_{v_i} \{(V(\hat{v}) \log V(\hat{v}))\}
10: if \(\alpha \cdot S(h_i) + \beta \cdot S(v_i) + \gamma \cdot S(s_i) \geq \tau\) then
11: \(S_{\ell} = S_{\ell} \cup F_k\)
12: else
13: \(i = i + 1\) and initialize a new segment \(S_{\ell}\)
14: \(S_{\ell} = S_{\ell} \cup F_k\)
15: end if
16: end for
17: for each segment \(S_{\ell}\) do
18: compute the maximum frame entropy \(\text{Entropy}(F)\):
19: \(\max \{ -\alpha \cdot \sum_{h \in H} H(h) \log H(h) - \beta \cdot \sum_{s \in S} S(s) \log S(s) - \gamma \cdot \sum_{v \in V} V(v) \log V(v)\} \}
20: extract the key frame with maximum entropy \(F_{s_i}\) in \(S_{\ell}\)
21: return all the segments and key frames
3.2.2 Random Response. After dimension reduction, random response can be implemented based on the RAPPOR framework [16] for each object. Each bit $kb^k_i$ in $\ell$-bit vector of object $O_i$ is randomly flipped into 0 or 1 using the following rules:

$$
kb^k_i = \begin{cases} 
kb^k_i, & \text{with the probability of } (1 - f) \\
1, & \text{with the probability of } f 
\end{cases} \quad (4)
$$

**Theorem 3.3.** The random response (with rules in Equation 4) $\ell \log \left(\frac{2^\ell}{f}\right)$-Object Indistinguishability.

**Proof.** Again, object indistinguishability can be proven by following the proof of LDP [16]. Specifically, the RAPPOR [16] satisfies $2h \log \left(\frac{2^\ell}{f}\right)$-LDP with the output size of the hash function in the bloom filter $h$ and the flipping probability $f$. Maximum difference sizes are 2 between the two input values. Thus, the random response (with rules in Equation 4) make $\ell$ equal to $\ell \log \left(\frac{2^\ell}{f}\right)$ since size difference in any two presence vector is at most $\ell$ (by replacing the encoded bit vectors of bloom filter as the object presence vectors in RAPPOR [16]), which satisfies $\ell \log \left(\frac{2^\ell}{f}\right)$-Object Indistinguishability.

3.3 Optimizing RAPPOR for Object Presence

Although $\ell$ is far less than $m$, the number of key frames $\ell$ 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 $\ell$ 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).

3.3.1 Optimization Problem. For each key frame $f_k$, $k \in [1, \ell]$, we define a binary variable $x_k \in \{0, 1\}$ to represent if key frame $f_k$ 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 Theorem 3.3, we have the random response satisfies $\sum_{k=1}^{\ell} x_k \log \left(\frac{2^\ell}{f}\right)$-Object Indistinguishability.

**Figure 3:** Dimension Reduction, Utility Maximization and Random Response

An example for dimension reduction, utility maximization and random response is given in Figure 3. Considering the $n$ objects $O_1, \ldots, O_n$, after dimension reduction, all the $n$ object presence vectors are reduced to $\ell$ different $\ell$-bit vectors. Our goal is to accurately retain more objects in the video, thus we aim at minimizing the distance between $V_i \in [1, n], B_i^f$ (extracted from $V'$) and $V_i \in [1, n], R_i$ (denoted as the $\ell$-bit vectors by applying random response to $V_i \in [1, n], B_i^f$).

Specifically, since $V_i \in [1, n], R_i$ are randomized bit vectors (the $k$th entry in all the vectors are $0$ if $x_k = 0$), we should measure the difference between the expectation $\forall V_i \in [1, n], E[R_i] = E[R_i^1, \ldots, R_i^\ell]$ and $B_i^f = (kb^f_1, \ldots, kb^f_\ell)$. We first learn the expectation of $R_i^k$ (the $k$th entry in $R_i$). If $x_k = 0$, then $\forall V_i \in [1, n], R_i^k = 0$ hold. Thus, we have:

$$
E(R_i^k) = x_k \cdot \left[ Pr(R_i^k = 1) \cdot 1 + Pr(R_i^k = 0) \cdot 0 \right] \quad (5)
$$

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

1. If $kb^f_i = 1$, per Equation 4, we have $E[R_i^k] = 1 \cdot \left[ (1 - f) \cdot 1 + \frac{f}{2} \cdot 0 + \frac{f}{2} \cdot 1 \right]$.
2. If $kb^f_i = 0$, we have $E[R_i^k] = 1 \cdot \left[ (1 - f) \cdot 0 + \frac{f}{2} \cdot 0 + \frac{f}{2} \cdot 1 \right]$.

Thus, the expectation can be summarized as following:

$$
\begin{align*}
E(R_i^k) &= \frac{x_k}{2}, & \text{if } x_k = 1 \text{ and } kb^f_i = 0 \\
E(R_i^k) &= 1 - \frac{x_k}{2}, & \text{if } x_k = 1 \text{ and } kb^f_i = 1 \\
E(R_i^k) &= 0, & \text{if } x_k = 0 \text{ and } kb^f_i = 0 \text{ or } 1
\end{align*} \quad (6)
$$

The objective function can be formulated as:

$$
\min : \sum_{k=1}^{\ell} |x_k \sum_{i=1}^{n} [E(R_i^k) - kb^f_i]| \quad (7)
$$

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

$$
\begin{align*}
\min : & \sum_{k=1}^{\ell} |x_k \sum_{i=1}^{n} [E(R_i^k) - kb^f_i]| \\
s.t.: & 2 \leq \sum_{k=1}^{\ell} x_k \leq \ell \\
& \forall k \in [1, \ell], x_k \in \{0, 1\}
\end{align*} \quad (8)
$$

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

$$
\begin{align*}
\min : & \sum_{k=1}^{\ell} \left( x_k \cdot \frac{n \cdot f}{2} - \sum_{i=1}^{n} kb^f_i \right) \\
&s.t.: 2 \leq \sum_{k=1}^{\ell} x_k \leq \ell \\
& \forall k \in [1, \ell], x_k \in \{0, 1\}
\end{align*} \quad (9)
$$

3.3.2 Complexity and Solver. Since $f$ and $\forall k \in [1, \ell], x_k$ are constants, $\forall k \in [1, \ell], \frac{n \cdot f}{2} - \sum_{i=1}^{n} kb^f_i$ are constants. Then, Equation 9 is a binary integer programming (BIP) problem. Although solving the BIP problems can be NP-hard [27], we can approximately solve Equation 9 using linear programming (LP) since the objective function and the constraints are linear: (1) letting the binary variable $x_k \in [1, \ell], x_k$ 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.
As discussed in Section 2, video preprocessing includes detecting each frame. Thus, it can be addressed by injecting a small amount of generic Laplace noise \( \text{Lap}(\frac{1}{\lambda}) \) into \( \sum_{k=1}^{n} \Delta b_k \), \( k \in [1, \ell] \) 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.

3.4 Privacy Guarantee

After solving the optimization problem, as shown in Figure 3, each of the picked key frames will be allocated with a privacy budget \( \epsilon = \sum_{k=1}^{n} x_k \). In the meanwhile, Verro utilizes the optimal solution \( \forall k \in [1, \ell], x_k \) to derive the optimal presence vectors \( \{\sum_{k=1}^{n} \Delta b_{k}^x \} \), denoted as \( B_1^x, \ldots, B_n^x \). Next, random response is applied to \( B_1^x, \ldots, B_n^x \) to generate output presence vectors \( R_1, \ldots, R_n \).

**Theorem 3.4.** Phase I satisfies \( \epsilon \)-Object Indistinguishability.

**Proof.** Phase I derives the presence bit vectors \( B_1^x \) and \( B_1^y \) for any two objects \( O_1 \) and \( O_2 \) after the optimization. Then, random response is applied to \( B_1^x \) and \( B_1^y \) and generate random vectors \( R_1 \) and \( R_2 \). Per Theorem 3.3, Phase I satisfies \( \epsilon \)-Object Indistinguishability where \( \epsilon = \sum_{k=1}^{n} x_k \) in \( \frac{2-f}{f} \) (note that the privacy guarantee for utility maximization has been discussed in Section 3.3.3).

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 \( V' \) will be generated in Phase II.

4 PHASE II: VIDEO GENERATION

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

4.1 Background Scene(s)

As discussed in Section 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 [11] 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., \( 3 \times 3 \)). 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 6.

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 \( V' \). Specifically, we denote all the frames in the synthetic video \( V' \) as \( \{F_1', \ldots, F_{\ell}^y \} \), and the frames in \( V' \) corresponding to the original key frames as \( \{F_1, \ldots, F_{\ell} \} \). We then discuss different cases of generating coordinates for the objects in each frame.

4.2.1 \( R_i \neq 0 \)

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 6, and most of the objects can be retained by Verro in practice.

4.2.2 \( R_i = 0 \)

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

Specifically, in each key frame of the synthetic video \( \forall k \in [1, \ell], F_k' \), the number of objects inserted into key frame \( F_k' \) is \( \sum_{c=1}^{n} R_k \) (derived in Phase I). Denoting the number of objects in the \( k \)-th key frame of \( V \) as \( c_k, k \in [1, \ell] \) where \( c_k = 0 \) if \( x_k = 0 \), we thus have:

- **Sufficient candidate coordinates**: if \( \sum_{c=1}^{n} R_k \leq c_k \), the number of required objects in \( F_k' \) is no greater than the number of candidate coordinates in \( F_k \). Then, Verro randomly picks \( \sum_{c=1}^{n} R_k \) out of \( c_k \) candidate coordinates for \( \sum_{c=1}^{n} R_k \) different objects in the background scene (frame \( F_k' \)). Please see the left example in Figure 4.

- **Insufficient candidate coordinates**: if \( \sum_{c=1}^{n} R_k > c_k \), the number of required objects in \( F_k' \) is greater than the number of candidate coordinates in \( F_k \). For instance, in the right example in Figure 4, we expand the set of candidate coordinates by adding the candidate coordinates in \( F_k \)’s neighboring frames in the same segment. Then, Verro randomly picks \( \sum_{c=1}^{n} R_k \) out of \( c_k \) candidate coordinates for \( \sum_{c=1}^{n} R_k \) different objects into the background scene (frame \( F_k' \)).

After assigning coordinates to the key frames (where \( R_k = 1 \)), 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 6. 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, there are many interpolation methods for moving object trajectories data (e.g., linear interpolation [45], nearest neighbor
interpolation [21], Lagrange interpolation [17]). 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 4.1.** VERRo (Phase I and Phase II) satisfies $\epsilon$-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 $\epsilon$-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 $\mathcal{V}^\epsilon$.

Specifically, given any output presence vector $y$ and any output trajectory $t = \{t_1, \ldots, t_m\}$ in $\mathcal{V}^\epsilon$, for simplicity of notation, we also denote the trajectories of $O_i$ and $O_j$ in $\mathcal{V}^\epsilon$ as $O_i = \{T^i_1, \ldots, T^i_m\}$ and $O_j = \{T^j_1, \ldots, T^j_m\}$, respectively.

$$
Pr[\mathcal{A}(O_i) = t] = Pr[\mathcal{A}(O_j) = t]
$$

$$
Pr[\mathcal{A}(R^i_j) = y] = Pr[\mathcal{A}(T^i_j) = t_1] \cdot \ldots \cdot Pr[\mathcal{A}(T^i_m) = t_m]
$$

On one hand, we have $Pr[\mathcal{A}(R^i_j) = y] \leq \epsilon^c$ (Phase I). On the other hand, if $\forall k \in [1, m], R^k_i = R^k_j = 1$, two objects are present in the same frame $F_k$ (and $F_{k+1}$). 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 $\forall k \in [1, m], Pr[\mathcal{A}(T^k_i) = t_k] = Pr[\mathcal{A}(T^k_j) = t_k]$. If $\forall k \in [1, m], R^k_i = R^k_j = 0$ (the coordinates are interpolated from the coordinates randomly assigned in the previous case [14]), we also have $\forall k \in [1, m], Pr[\mathcal{A}(T^k_i) = t_k] = Pr[\mathcal{A}(T^k_j) = t_k]$.

To sum up the above three cases, we have:

$$
Pr[\mathcal{A}(O_i) = t] \leq \epsilon^c
$$

where $\epsilon = \sum_{k=1}^{\ell} x_k \log \left(\frac{2^{\ell+k}}{F}\right)$, as analyzed in Theorem 3.3 and Section 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 $\mathcal{V}^\epsilon$ (with random response in Phase I and random coordinates assignment in Phase II). For instance, the trajectory (in $\mathcal{V}^\epsilon$) closest to object $O_1$’s original trajectory might be generated by object $O_5$.

**5 DISCUSSION**

**Distributed Framework:** LDP techniques [4, 16, 41, 48] 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 $\mathcal{V}^\epsilon$ 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 6-8 in our experiments). Also, such noise can be cancelled in data aggregation applications [9] (e.g., object counting, as shown in Figure 12 and 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 $\epsilon$-Object Indistinguishable while all the vehicles are $\epsilon$-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 $\epsilon$-indistinguishable. In case that the video includes only one sensitive object, VERRo can also protect such object against re-identification. In existing LDP techniques [4, 16, 41, 48], if only one user perturbs its Object and discloses it to the untrusted aggregator, the original data cannot be identified from its perturbed data. Similar to such works (e.g., RAPPOR [16], 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, 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 $\epsilon$-Object Indistinguishability can be guaranteed.
6 EXPERIMENTS
In this section, we present the performance evaluations.

6.1 Experimental Setup
We conduct our experiments on three real videos in the repository of multiple object tracking benchmark\textsuperscript{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") [35]: 23 distinct pedestrians are sensitive objects in 450 frames (static camera).
2. MOT16-03 (pedestrians on the street at night, denoted as "MOT03") [35]: 148 distinct pedestrians are sensitive objects in 1,500 frames (static camera).
3. MOT16-06 (street scene from a moving platform, denoted as "MOT06") [35]: 221 distinct pedestrians are sensitive objects in 1,194 frames (moving camera).

Table 1: Characteristics of Experimental Videos

<table>
<thead>
<tr>
<th>Video</th>
<th>Resolution</th>
<th>Frame</th>
<th>Objects</th>
<th>Camera</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOT16-01</td>
<td>1920 x 1080</td>
<td>450</td>
<td>23</td>
<td>static</td>
</tr>
<tr>
<td>MOT16-03</td>
<td>1920 x 1080</td>
<td>1,500</td>
<td>148</td>
<td>static</td>
</tr>
<tr>
<td>MOT16-06</td>
<td>640 x 480</td>
<td>1,194</td>
<td>221</td>
<td>moving</td>
</tr>
</tbody>
</table>

We implement the detecting/tracking algorithm [51, 52] 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 [11] is also utilized to extract/reconstruct the background scene(s) from the input video 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.

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.

6.2.1 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 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 ~ 80% distinct objects).

Table 2: Distinct Objects after Key Frame Extraction

<table>
<thead>
<tr>
<th>Video</th>
<th>Frame</th>
<th>Objects</th>
<th>Key Frame</th>
<th>Remaining</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOT01</td>
<td>450</td>
<td>23</td>
<td>22</td>
<td>19</td>
</tr>
<tr>
<td>MOT03</td>
<td>1,500</td>
<td>148</td>
<td>52</td>
<td>124</td>
</tr>
<tr>
<td>MOT06</td>
<td>1,194</td>
<td>221</td>
<td>48</td>
<td>191</td>
</tr>
</tbody>
</table>

Figure 5(a), 5(c) and 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 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 3.2) — such utility loss is indeed minor in our experiments. In addition, we can draw similar observations in Figure 5(c) and 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).

\textsuperscript{2}https://motchallenge.net/
6.2.2 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: \( \sum_{i=1}^{n} \sum_{k=1}^{m} P(O_i, F_k) - P(O_i, F_k^*) \), 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 5(b), 5(d) and 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, 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 \( V \). In addition, we also extract their corresponding trajectories in the synthetic video \( V^* \). Figure 6, 7 and 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 \((X, Y)\) 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 6). This is ensured by the \( \epsilon \)-indistinguishable presence bit vectors randomly generated from all the objects in VERRO.

6.3 Visual & Aggregated Results

We also randomly pick a frame from each of the three experimental videos, and present the generated background scenes and the
corresponding frames in the synthetic videos. For video MOT01, Figure 9(a) shows the input frame and the detected objects in the frame. Also, we use a background interpolation algorithm [11] to fill the missing pixels (after removing all the detected objection), as shown in Figure 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 10 and 11. Some human silhouettes still exist in the background scenes. Clearly, the silhouettes cannot be associated to any objects in the synthetic video (as shown in Figure 10(c), 10(d), 11(c) and 11(d)). This confirms the discussion for imperfect background scene in Section 5.

In the synthetic videos, we use different colors for different synthetic objects. Compare to $f = 0.1$ (shown in Figure 9(c), 10(c) and 11(c)), $f = 0.9$ would lead to more coordinates/trajectory deviation (as shown in Figure 9(d), 10(d) and 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 based on the count of objects, e.g., head counting and crowd density [24, 34]. 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 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 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 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.

6.4 Overheads
We evaluate the overheads of Verro. Table 3 presents the runtime of the two phases and the required bandwidth for sending the synthetic videos to an untrusted recipient.
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 almost identical to the original video size.

### 7 RELATED WORK

In the context of privacy preserving video publishing, many solutions have been proposed in literature (e.g., [6, 20, 42, 44, 46]). Saini et al. [42] 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 [6, 20, 44, 46] and some of them [15, 22, 42, 50] aims at preserving the privacy against other three implicit inference channels.  

Specifically, these techniques often leverage computer vision techniques [20, 30] 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 [33, 37]. Last but not least, these techniques often use naive measures for quantifying the privacy loss in videos. For instance, in [20, 36], if faces are present, then it is considered as complete privacy loss, otherwise no privacy loss is reported. Fan [18] applied Laplace noise to randomly perturb the pixels in an image to ensure differential privacy for protecting specific regions of an image. However, the quality of the image is significantly deviated in the sanitized results. Our proposed privacy notion and the Verro technique have addressed all the above limitations.

On the other hand, in the context of privacy preserving data publishing, the notion of differential privacy has emerged as a standard specification during past decade. This strong notion of privacy was first proposed by Dwork [13] to guarantee indistinguishability in the published data against an adversary armed with arbitrary background knowledge. Although differential privacy has been widely used to sanitize and release data in statistical databases [13], histograms [47], location data [39], search logs [29], and graphs [25], to the best of our knowledge, no attempt has yet been made to benefit from differential privacy in video databases. Furthermore, to fully utilize differential privacy for sanitizing videos, we have defined our privacy notion based on a recently proposed locally implemented notion of differential privacy in which individuals in the videos (i.e., as objects) can directly interact with the sanitized result to ensure trustworthiness and fine-grained privacy. The emerging local differential privacy is a promising approach for addressing privacy concerns in video publishing.

---

\[\text{Table 3: Computational and Communication Overheads}\]

<table>
<thead>
<tr>
<th>Video</th>
<th>Phase I (Sec)</th>
<th>Phase II (Sec)</th>
<th>Bandwidth (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOT01</td>
<td>0.89</td>
<td>34.78</td>
<td>9.58</td>
</tr>
<tr>
<td>MOT03</td>
<td>1.56</td>
<td>36.12</td>
<td>16.6</td>
</tr>
<tr>
<td>MOT06</td>
<td>1.57</td>
<td>43.12</td>
<td>19.4</td>
</tr>
</tbody>
</table>

\[\text{The synthetic videos generated by Verro can preserve the information of "where and when the videos are captured" while ensuring indistinguishability of objects.}\]

\[\text{(the linkage between every object and such inference channels can be broken to avoid leakage in the disclosure of the background scene).}\]
privacy (LDP) models [4, 10, 16] have been utilized in a wide variety of applications (e.g., heavy hitters or histogram construction [4, 16], social graphs [41], frequent itemset mining [49]), but cannot be directly applicable to local video perturbation. Our Verra technique complements the literature with strong privacy protection for (local) objects in the video against arbitrary background knowledge.

8 CONCLUSION

Privacy concerns arise in considerable number of real world videos (e.g., individuals might be re-identified by the video re-identifying individuals might be re-identified by the video re-identification) 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 Verra.

REFERENCES

[18] Feng Han, Ying Shan, Ryan Cekander, Harpreet S Sawhney, and Rakesh Kumar. 2006. A two-stage approach to people and vehicle detection with HOG-based SVM. In Performance Metrics for Intelligent Systems. 133–140.