Privid: Practical, Privacy-Preserving Video Analytics Queries

Frank Cangialosi, *MIT CSAIL;* Neil Agarwal, *Princeton University;*
Venkat Arun, *MIT CSAIL;* Junchen Jiang, *University of Chicago;*
Srinivas Narayana and Anand Sarwate, *Rutgers University;*
Ravi Netravali, *Princeton University*

https://www.usenix.org/conference/nsdi22/presentation/cangialosi

This paper is included in the Proceedings of the 19th USENIX Symposium on Networked Systems Design and Implementation.

April 4–6, 2022 • Renton, WA, USA

978-1-939133-27-4

Privid: Practical, Privacy-Preserving Video Analytics Queries

Frank Cangialosi🦫, Neil Agarwal🐅, Venkat Arun🦫, Junchen Jiang🐦, Srinivas Narayana🛡, Anand Sarwate🛡, Ravi Netravali🐅
🦫 MIT CSAIL 🐅 Princeton University 🐦 University of Chicago 🛡 Rutgers University
privid@csail.mit.edu

Abstract

Analytics on video recorded by cameras in public areas have the potential to fuel many exciting applications, but also pose the risk of intruding on individuals' privacy. Unfortunately, existing solutions fail to practically resolve this tension between utility and privacy, relying on perfect detection of all private information in each video frame—an elusive requirement. This paper presents: (1) a new notion of differential privacy (DP) for video analytics, (ρ, K, ϵ)-event-duration privacy, which protects all private information visible for less than a particular duration, rather than relying on perfect detections of that information, and (2) a practical system called Privid that enforces duration-based privacy even with the (untrusted) analyst-provided deep neural networks that are commonplace for video analytics today. Across a variety of videos and queries, we show that Privid increases error by 1-21% relative to a non-private system.

1 Introduction

High-resolution video cameras are now pervasive in public settings [1,3–5,10], with deployments throughout city streets, in our doctor's offices and schools, and in the places we shop, eat, or work. Traditionally, these cameras were monitored manually, if at all, and used for security purposes, such as providing evidence for a crime or locating a missing person. However, steady advances in computer vision [32,51,53,55,65] have made it possible to automate video-content analytics (both live and retrospective) at a massive scale across entire networks of cameras. While these trends enable a variety of important applications [2,11,13,14] and fuel much work in the systems community [26,30,40,43,44,47,48,54,73], they also enable privacy intrusions at an unprecedented level [7,64].

As a concrete example, consider the operator for a network of city-owned cameras. Different organizations (i.e., "analysts") want access to the camera feeds for a range of needs: (1) health officials want to measure the fraction of people wearing masks and following COVID-19 social distancing orders [38], (2) the transportation department wants to monitor the density and flow of vehicles, bikes, and pedestrians to determine where to add sidewalks and bike lanes [21], and (3) businesses are willing to pay the city to understand shopping behaviors for better planning of promotions [19].

Unfortunately, freely sharing the video with these parties may enable them to violate the privacy of individuals in the scene by tracking where they are, and when. For example, the "local business" may actually be a bank or insurance company that wants to track individuals' private lives for their risk models, while well-known companies [17] or government agencies may succumb to mission creep [18,20]. Further, any organizations with good intentions could have employees with malicious intent who wish to spy on a friend or co-worker [15,16].

There is an *inherent tension between utility and privacy.* In this paper, we ask: is it possible to enable these (untrusted) organizations to use the collected video for analytics, while also guaranteeing citizens that their privacy will be protected? Currently, the answer is no. As a consequence, many cities have outright banned analytics on public videos, even for law enforcement purposes [9,12].

While a wide variety of solutions have been proposed (§3), ranging from computer vision (CV)-based obfuscation [23,60,68,70] (e.g., blurring faces) to differential privacy (DP)-based methods [66,67], they all use some variant of the same strategy: find *all* private information in the video, then hide it. Unfortunately, the first step alone can be unrealistic in practice (§3.1); it requires: (1) an explicit specification of all private information that could be used to identify an individual (e.g., their backpack), and then (2) the ability to spatially *locate* all of that information in *every* frame of the video—a near impossible task even with state-of-the-art CV algorithms [6]. Further, if these approaches cannot find some private information, they fundamentally cannot *know* that they missed it. Taken together, they can provide, at best, a conditional and brittle privacy guarantee such as the following: if an individual is only identifiable by their face, and their face is detectable in every frame of the video by the implementation's specific CV model in the specific conditions of this video, then their privacy will be protected.

This paper takes a pragmatic stance and aims to provide a definitively achievable privacy guarantee that captures the aspiration of prior approaches (i.e., individuals cannot be identified in any frame or tracked across frames) despite the limitations that plague them. To do this, we leverage two key observations: (1) a large body of video analytics queries are aggregations [47,49], and (2) they typically aggregate over durations of video (e.g., hours or days) that far exceed the duration of any one individual in the scene (e.g., seconds or minutes) [47]. Building on these observations, we make three contributions by jointly designing a new notion of duration-based privacy for video analytics, a system implementation to realize it, and a series of optimizations to improve utility.

Duration-based differential privacy. To remove the dependence on spatially locating all private information in each video frame, we reframe the approach to privacy to instead focus on the temporal aspect of private information in video data, i.e., *how long* something is visible to a camera. More specifically, building on the differential privacy (DP) framework [37], we propose a new notion of privacy for video, (ρ, K, ϵ)-event-duration privacy (formalized in §4.1): *anything* visible to a camera less than K times for less than ρ seconds each time ("(ρ, K)-bounded") is protected with ϵ-DP. The video owner expresses their privacy policy using (ρ, K), which we argue is powerful enough to capture many practical privacy goals. For example, if they choose $\rho =$ 5min, anyone visible for less than 5 minutes is protected with ϵ-DP, which in turn prevents tracking them. We discuss other policies in §4.2.

This notion of privacy has three benefits. First, it decouples the definition of privacy from its enforcement. The enforcement mechanism does not need to make any decisions about what is private or find private information to protect it; everything (private or not) captured by the bound is protected. Second, a (ρ, K) bound that captures a set of individuals implicitly captures and thus protects any information visible for the same (or less) time without specifying it (e.g., an individual's backpack, or even their gait). Third, protecting all individuals in a video scene requires only their maximum duration, and estimating this value is far more robust to the imperfections of CV algorithms than precisely locating those individuals and their associated objects in each frame. For example, even if a CV algorithm misses individuals in some frames (or entirely), it can still capture a representative sample and piece together trajectories well enough to estimate their duration (§4.2).

Privid: a differentially-private video analytics system. Realizing (ρ, K, ϵ)-privacy (or more generally, any DP mechanism) in today's video analytics pipelines faces several challenges. In traditional database settings, implementing DP requires adding random noise proportional to the *sensitivity* of a query, i.e., the maximum amount that any one piece of private information could impact the query output. However, bounding the sensitivity is difficult in video analytics pipelines because (1) pipelines typically operate as bring-your-own-query-implementation to support the wide-ranging applications described earlier [22, 25, 26, 28, 29, 39, 41], and (2) these implementations involve video processing algorithms that increasingly rely on deep neural networks (DNNs), which are notoriously hard to inspect or vet (and thus, trust).

To bound the sensitivity necessary for (ρ, K, ϵ)-privacy while supporting "black-box" analyst-provided query implementations (including DNNs), Privid only accepts analyst queries structured in the following *split-process-aggregate* format (§5.2): (i) videos are split into temporally-contiguous chunks, (ii) each chunk of video is processed by an arbitrary analyst-provided processing program to produce an (untrusted) table, (iii) values in the table are aggregated (e.g. averaged) to compute a result, and (iv) noise is added to the result before release. The key in this pipeline is step (ii): we treat the analyst-provided program as an arbitrary Turing machine with restricted inputs (a single chunk of video frames and some metadata) and restricted outputs (rows of a table). As a result, only one chunk can contribute to the value of each row, and we know which chunk generated each row. If an individual is (ρ, K)-bounded, the number of chunks they appear in is bounded, and thus the number of rows their presence can affect is bounded as well. With a bound on the number of rows, we can apply classic differential privacy techniques (§5.5).

Optimizations for improved utility. To further enhance utility, Privid provides two video-specific optimizations to lower the required noise while preserving an equivalent level of privacy: (i) the ability to mask regions of the video frame, (ii) the ability to split frames spatially into different regions, and aggregate results from these regions. These optimizations result in limiting the portion of the aggregate result that any individual's presence can impact, enabling a "tighter" (ρ, K) bound and in turn a higher quality query result.

Evaluation. We evaluate Privid using a variety of public videos and a diverse range of queries inspired by recent work in this space. Privid increases error by 1-21% relative to a non-private system, while satisfying an instantiation of (ρ, K, ϵ)-privacy that protects all individuals in the video. We discuss ethics in §9. Source code and datasets for Privid are available at https://github.com/fcangialosi/privid.

2 Problem Statement

2.1 Video Analytics Background

Video analytics pipelines are employed to answer high-level questions about segments of video captured from one or more cameras and across a variety of time ranges. Example questions include "how many people entered store X each hour?" or "which roads suffered from the most accidents in 2020?" (see §7.2 and Table 3 for more specific examples). A question is expressed as a *query*, which encompasses all of the computation necessary to answer that question.[1] For example, to answer the question "what is the average speed of red cars traveling along road Y?", the "query" would include an object detection algorithm to recognize cars, an object tracking algorithm to group them into trajectories, an algorithm for computing speed from a trajectory, and logic to filter only the red cars and average their speeds.

2.2 Problem Definition

Video analytics pipelines broadly involve four logical roles (though any combination may pertain to the same entity):

- **Individuals**, whose behavior and activity are observed by the camera.
- **Video Owner (VO)**, who operates the camera and thus owns the video data it captures.
- **Analyst**, who wishes to run queries over the video.
- **Compute Provider**, who executes the analyst's query.

In this work, we are concerned with the dilemma of a VO. The VO would like to enable a variety of (untrusted) analysts to answer questions about its videos (such as those in §2.1), as long as the results do not infringe on the privacy of the individuals who appear in the videos. Informally, privacy

[1] Our definition is distinct from related work, which defines a query as returning intermediate results (e.g., bounding boxes) rather than the final answer to the high-level question.

“leakage” occurs when an analyst can learn something about a specific individual that they did not know before executing a query. To practically achieve these properties, a system must meet three concrete goals:

1. **Formal notion of privacy**. The system’s privacy policies should formally describe the type and amount of privacy that could be lost through a query. Given a privacy policy, the system should be able to provide a *guarantee* that it will be enforced, regardless of properties of the data or query implementation.
2. **Maximize utility for analysts.** The system should support queries whose final *result* does not infringe on the privacy of any individuals. Further, if accuracy loss is introduced to achieve privacy for a given query, it should be possible to bound that loss (relative to running the same query over the original video, without any privacy preserving mechanisms). Without such a bound, analysts would be unable to rely on any provided results.
3. **“Bring Your Own Model”**. Computer vision models are at the heart of modern video processing. However, there is not one or even a discrete set of models for all tasks and videos. Even the same task may require different models, parameters, or post-processing steps when applied to different videos. In many cases, analysts will want to use models that they trained themselves, especially when training involves proprietary data. Thus, a system must allow analysts to provide their own video-processing models.

It is important to note that the class of analytics queries we seek to enable are distinct from *security-oriented* queries (e.g., finding a stolen car or missing child), which *require* identification of a particular individual, and are thus directly at odds with individual privacy. In contrast, analytics queries involve searching for patterns and trends in large amounts of data; intermediate steps may operate over the data of specific individuals, but they do not distinguish individuals in their final aggregated result (§2.1).

2.3 Threat Model

The VO employs a privacy-preserving system to handle queries about a set of cameras it manages; the system retains full control over the video data, analysts can only interact with it via the query interface. The VO does not trust the analysts (or their query implementation code). Any number of analysts may be malicious and may collude to violate the privacy of the same individual. However, analysts trust the VO to be honest. Analysts are also willing to share their query implementation (so that the VO can execute it). The VO views this code as an untrusted blackbox which it cannot vet.

Analysts pose queries adaptively (i.e., the full set of queries is not known ahead of time, and analysts may utilize the results of prior queries when posing a new one). A single query may operate over video from multiple cameras. We assume the VO has sufficient computing resources to execute the query, either via resources that they own, or through the secure use of third-party resources [62].

The system releases some per-camera metadata publicly (§8.1), including a sample video clip. The resulting leak is interpretable and can be minimized by the VO. The system protects all other information with a formal guarantee of (ρ, K, ϵ)-privacy (Def 4.3).

3 Limitations of Related Work

Before presenting our solution, we consider prior privacy-preserving mechanisms (both for video and in general). Unfortunately, each fails to satisfy at least one of the goals in §2.2.

3.1 Denaturing

The predominant approach to privacy preservation with video data is *denaturing* [23, 34, 60, 68, 70, 72], whereby systems aim to obscure (e.g., via blurring [23] or blocking [68] as in Fig. 1) any private information in the video before releasing it for analysis. In principle, if nothing private is left in the video, then privacy concerns are eliminated.

The fundamental issue is that denaturing approaches require *perfectly* accurate and comprehensive knowledge of the spatial locations of private information in *every frame* of a video. Any private object that goes undetected, even in just a single frame, will not be obscured and thus directly leads to a leakage of private information.

To detect private information, one must first semantically define *what* is private, i.e., what is the full set of information linked, directly or indirectly, to the privacy of each individual? While some information is obviously linked (e.g., an individual’s face), it is difficult to determine *all* such information for all individuals in all scenarios. For instance, a malicious analyst may have prior information that a VO does not, such as knowledge that a particular individual carries a specific bag or rides a unique bike (e.g., Fig. 1-B). Further, even with a semantic definition, detecting private information is difficult. State-of-the-art computer vision algorithms commonly miss objects or produce erroneous classification labels in favorable video conditions [74]; performance steeply degrades in more challenging conditions such as poor lighting, distant objects, and low resolution, all of which are common in public video. Taken together, the problem is that denaturing systems cannot guarantee whether or not a private object was left in the video, and thus fail to provide a formal notion of privacy (violating Goal 1).

Denaturing also falls short from the analyst’s perspective. First, it inherently precludes (safe) queries that aggregate over private information (violating Goal 2). For example, an urban planner may wish to count the number of people that walk in front of camera A and then camera B. Doing so requires identifying and cross-referencing individuals between the cameras (which is not possible if they have been denatured), but the ag-

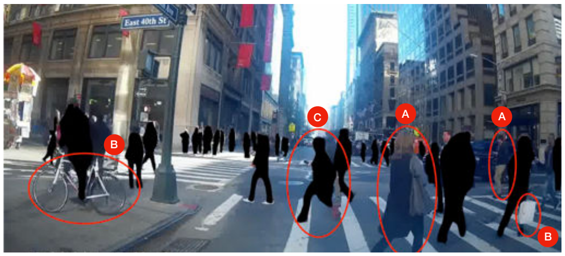


Figure 1: A video clip after (silhouette) denaturing exemplifying some of its shortcomings: (A) entirely missed detections, (B) potentially-identifying objects not incorporated in privacy definition, (C) silhouette may reveal gait.

gregate count may be large and safe to release.[2] Second, obfuscated objects are not naturally occurring and thus video processing pipelines are not designed to handle them. If the analyst's processing code and models have not been trained explicitly on the type of obfuscation the VO is employing, it may behave in unpredictable and unbounded ways (violating Goal 2).

3.2 Differential Privacy

Differential Privacy (DP) is a strong formal definition of privacy for traditional databases [37]. It enables analysts to compute aggregate statistics over a database, while protecting the presence of any individual entry in the database. DP is not a privacy-preserving mechanism itself, but rather a goal that an algorithm can aim to satisfy. Informally speaking, an algorithm satisfies DP if adding or removing an individual from the input database does not noticeably change the output of computation, almost as if any given individual were not present in the first place. More precisely,

Definition 3.1. Two databases D and D' are *neighboring* if they differ in the data of only a single user (typically, a single row in a table).

Definition 3.2. A randomized algorithm $\mathcal{A}$ is ϵ-*differentially private* if, for all pairs of neighboring databases (D,D') and all $S \subseteq Range(\mathcal{A})$:

$$\Pr[\mathcal{A}(D) \in S] \leq e^{\epsilon} \Pr[\mathcal{A}(D') \in S] \tag{3.1}$$

A non-private computation (e.g., computing a sum of bank balances) is typically made differentially private by adding random noise sampled from a Laplace distribution to the final result of the computation [37]. The scale of noise is set proportional to the *sensitivity* (Δ) of the computation, or the maximum amount by which the computation's output could change due to the presence/absence of any one individual. For instance, suppose a database contains a value $v_i \in V$ for each user i, where $l \leq v_i \leq u$. If a query seeks to sum all values in V, any one individual's v_i can influence that sum by at most $\Delta = u-l$, and thus adding noise with scale $u-l$ would satisfy DP.

Challenges. Determining the sensitivity of a computation is the key ingredient of satisfying DP. It requires understanding (a) how individuals are delineated in the data, and (b) how the aggregation incorporates information about each individual. In the tabular data structures that DP was designed for, these are straightforward. Each row (or a set of rows sharing a unique key) typically represents one individual, and queries are expressed in relational algebra, which describes exactly how it aggregates over these rows. However, these answers do not translate to video data; we next discuss the challenges in the context of several applications of DP to video analytics.

Regarding *requirement (a)*, as described in §3.1, it is difficult and error-prone to determine the full set of pixels in a video that correspond to each user (including all potentially identifying objects). Accordingly, prior attempts of applying DP concepts to video analytics [66, 67] that rely on perfectly defined and detected private information (via CV) fall short in the same way as denaturing approaches (violating Goal 1).

Regarding *requirement (b)*, typical video processing algorithms (e.g., ML-based CV models) are not transparent about how they incorporate private objects into their results. Thus, without a specific query interface, the "tightest" possible bound on the sensitivity of an arbitrary computation over a video is simply the entire range of the output space. In this case, satisfying DP would add noise greater than or equal to any possible output, precluding any utility (violating Goal 2).

Given that DP is well understood for tables, a natural idea would be for the VO to use their own (trusted) model to first convert the video into a table (e.g., of objects in the video), then provide a DP interface over *that table*[3] (instead of directly over the video itself). However, in order to provide a guarantee of privacy, the VO would need to completely trust the model that creates the table. This entirely precludes using a model created by the *untrusted* analyst (violating Goal 3).

4 Event Duration Privacy

We will first formalize (ρ, K, ϵ)-privacy, then provide the intuition for what it protects and clarify its limitations.

4.1 Definition

We consider a video V to be an arbitrarily long sequence of frames, sampled at f frames per second, recorded directly from a camera (i.e., unedited). A "segment" $v \subset V$ of video is a contiguous subsequence of those frames. The "duration" of a segment $d(v)$ is measured in real time (seconds), as opposed to frames. An "event" e is abstractly *anything* that is visible within the camera's field of view.

As a running example, consider a video segment v in which individual x is visible for 30 seconds before they enter a building, and then another 10 seconds when they leave some time later. The "event" of x's visit is comprised of one 30-second segment, and another 10-second segment.

[2] As a workaround, the VO could annotate denatured objects with query-specific information, but this would conflict with Goal 3.

[3] This would be equivalent to adding DP to an existing video analytics interface, such as [30, 47], which treat the video as a table of objects.

Definition 4.1 ((ρ, K)-bounded events)**.** An event e is (ρ,K)-bounded if there exists a set of $\leq K$ video segments that completely contain[4] the event, and each of these segments individually have duration $\leq \rho$.

(Ex). The tightest bound on x's visit is $(\rho=30s, K=2)$. To be explicit, x's visit is also (ρ,K)-bounded for any $\rho \geq 30s$ and $K \geq 2$.

Definition 4.2 ((ρ, K)-neighboring videos)**.** Two video segments v, v' are (ρ,K)-neighboring if the set of frames in which they differ is (ρ,K)-bounded.

(Ex). One potential v' is a hypothetical video in which x was never present (but everything else observed in v remained the same). Note this is purely to denote the strength of the guarantee in the following definition, the VO does not actually construct such a v'.

Definition 4.3 ((ρ, K, ϵ)-event-duration privacy)**.** A randomized mechanism $\mathcal{M}$ satisfies (ρ,K,ϵ)-event-duration privacy [5] iff for all possible pairs of (ρ, K)-neighboring videos v, v', any finite set of queries $Q = \{q_1, q_2, ...\}$ and all $S_q \subseteq Range(\mathcal{M}(\cdot, q))$:

$$Pr[(\mathcal{M}(v,q_1),...,\mathcal{M}(v,q_n)) \in S_{q_1} \times \cdots \times S_{q_n}] \leq$$
$$e^{\epsilon} Pr[(\mathcal{M}(v',q_1),...,\mathcal{M}(v',q_n))) \in S_{q_1} \times \cdots \times S_{q_n}]$$

Guarantee. (ρ, K, ϵ)-privacy protects all (ρ, K)-bounded events (such as x's visit to the building) with ϵ-DP: informally, if an event is (ρ,K)-bounded, an adversary cannot increase their knowledge of whether or not the event happened by observing a query result from $\mathcal{M}$. To be clear, (ρ, K, ϵ)-privacy is *not* a departure from DP, but rather an extension to explicitly specify what to protect in the context of video.

4.2 Choosing a Privacy Policy

The VO is responsible for choosing the parameter values (ρ, K) ("policy") that bound the class of events they wish to protect. They may use domain knowledge, employ CV algorithms to analyze durations in past video from the camera, or a mix of both. Regardless, they express their goal to Privid solely through their choice of (ρ,K).

Automatic setting of (ρ,K). The primary reason (ρ,K,ϵ)-privacy is *practical* is that, despite their imperfections, today's CV algorithms are capable of producing good estimates of the maximum duration any individuals are visible in a scene. We provide some evidence of this intuition over three representative videos from our evaluation. For each video, we chose a 10-minute segment and manually annotate the duration of each individual (`person` or `vehicle`), i.e., "Ground Truth", then use state-of-the-art object detection and tracking to estimate the durations and report the maximum ("CV"). Our results, summarized in Table 1, show that, while object detection misses a non-trivial fraction of bounding boxes, the tracking algorithm is able to fill in the gaps for enough trajectories to capture a conservative estimate of the maximum duration. In other words, for our three videos, using these algorithms to parameterize a (ρ,K,ϵ)-private system would successfully capture the duration of, and thus protect the privacy of, *all* individuals, while using them to implement any prior approach would not.

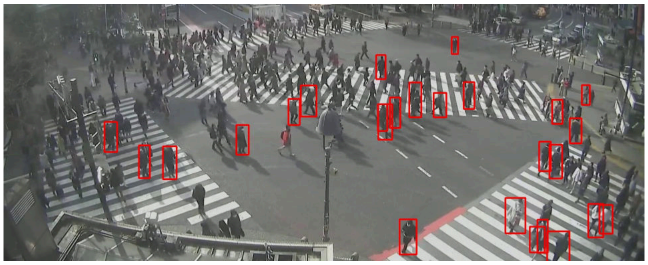


Figure 2: The results of a state-of-the-art object detection algorithm (filtered to "person" class) on one frame of `urban`. The algorithm misses 76% of individuals in the frame, but is *still* able to produce a conservative bound on the maximum duration of all individuals (Table 1).

Video	**Maximum Duration**		**% Objects**
	Ground Truth	CV Estimate	**CV Missed**
`campus`	81 sec	83 sec	29%
`highway`*	316 sec	439 sec	5%
`urban`	270 sec	354 sec	76%

Table 1: Despite the imperfection of current CV algorithms (exemplified by % objects they failed to detect), they still produce a conservative estimate on the duration of any individual's presence. *For the purposes of this experiment, we ignored cars that were parked for the entire duration of the segment.

Relaxing the set of private individuals. Sometimes protecting *all* individuals is unnecessary. Consider a camera in a store; employees will appear significantly longer and more frequently than customers (e.g., 8 hours every day vs. 30 minutes once a week), but if the fact that the employees work there is public knowledge, the VO can pick a policy (with smaller ρ and K) that only bounds the appearance of customers.

Generic policies. Alternatively, the VO can choose a policy to place a generic limit on the (temporal) granularity of queries. Consider a policy (ρ = 5min, K = 1). Suppose individual x stops and talks to a few people on their way to work each morning, but each conversation lasts less than 5 minutes. Although the policy does not protect x's presence or even the fact that they often stop to chat on their way to work, it *does* protect the timing and contents of each conversation.

4.3 Privacy Guarantees in Practice

In Privid's implementation of (ρ,K,ϵ)-privacy (described in the following section), the policy provides a relative reference point: events that exactly match the policy (i.e., made up of *exactly* K segments each of duration ρ) are protected

[4] A set of segments is said to completely contain an event if the event is not visible in any frames outside of those segments.

[5] We chose to use ϵ-DP rather than the more general (ϵ, δ)-DP for simplicity, since the difference is not significant to our definition. Our definition could be extended to (ϵ,δ)-DP without additional insights.

with ϵ-DP, while events that are visible for shorter or longer durations are protected with a proportionally (w.r.t. the duration) stronger or weaker guarantee, respectively.

Theorem 4.1. Consider a camera with a fixed policy (ρ,K,ϵ). If an individual x's appearance in front of the camera is bound by some $(\hat{\rho},\hat{K})$, then PRIVID effectively protects x with $\hat{\epsilon}$-DP, where $\hat{\epsilon}$ is $O(\frac{\hat{\rho}\hat{K}}{\rho K})\epsilon$, which grows (degrades) as $(\hat{\rho},\hat{K})$ increase while (ρ,K,ϵ) are fixed, and the constants do not depend on the query. We provide a formal proof in §A.1.

For example, given $(\rho = 1hr, K = 1)$, PRIVID would protect an a single 2-hour appearance with $\sim 2\epsilon$-DP (weaker) or a single half-hour appearance with $\sim \frac{1}{2}\epsilon$-DP (stronger).

Graceful degradation. An important corollary of this theorem is that privacy degrades "gracefully". As an event's $\hat{\rho}$ increases further from ρ (or $\hat{K}$ from K), its effective $\hat{\epsilon}$ increases linearly, yielding a progressively weaker guarantee. (The reverse is true, as $\hat{\rho}$ and $\hat{K}$ decrease, it yields a stronger guarantee). Thus, if $\hat{\rho}$ (or $\hat{K}$) is only *marginally* greater than ρ (or K), then the event is not immediately revealed in the clear, but rather is protected with $\hat{\epsilon}$-DP, which is still a DP guarantee, only marginally weaker: a malicious analyst has only a marginally higher probability of detecting x in the worst case. This in effect *relaxes* the requirement that (ρ,K) be set strictly to the maximum duration an individual could appear in the video to achieve useful levels of privacy. We generalize and provide a visualization of this degradation in §A.2.

Repeated appearances. The larger the time window of video a query analyzes, the more instances an individual may appear within the window, even if each appearance is itself bounded by ρ. Consider our example individual x and policy $(\rho = 30s, K = 2)$ from §4.1. In the query window of a single day d, x appears twice; they are properly (ρ,K)-bounded and thus the event "x appeared on day d" is protected with ϵ-DP. Now, consider a query window of one week; x appears 14 times (2 times per day), so the event "x appeared sometime this week" is $(\rho,7K)$-bounded and thus protected with (weaker) 7ϵ-DP. However, the more specific event "x appeared on day d" (for any d in the week) is *still* (ρ,K)-bounded, and thus still protected with the same ϵ-DP. In other words, while an analyst may learn that an individual appeared *sometime* in a given week, they cannot learn on which day they appeared. Thus, in order to get greater certainty, the analyst must give up temporal granularity.

Multiple cameras. When an individual appears in front of multiple cameras, their privacy guarantees are analogous to the previous case of repeated appearances in a single camera. If they appear in front of N different cameras, where the event of their appearance in camera i is protected with $\hat{\epsilon}_i$-DP, then the event of their appearance across all the cameras is protected with $\sum_i \hat{\epsilon}_i$-DP. Suppose for 10 cameras, $\sum_{i=1}^{N} \hat{\epsilon}_i$ is large enough for the adversary to detect their appearance with high confidence. Then while the adversary can infer that a person appeared *somewhere* across the 10 cameras, the adversary cannot learn *which* cameras they appeared in or when; appearances within individual cameras are still protected by ϵ-DP.

5 PRIVID

In this section, we present PRIVID, a privacy-preserving video analytics system that satisfies (ρ,K,ϵ)-privacy (§2.2 Goals 1 and 2) and provides an expressive query interface which allows analysts to supply their own (untrusted by PRIVID) video-processing code (Goal 3).

5.1 Overview

PRIVID supports *aggregation* queries, which process a "large" amount of video data (e.g., several hours/days of video) and produce a "small" number of bits of output (e.g., a few 32-bit integers). Examples of such tasks include counting the total number of individuals that passed by a camera in one day, or computing the average speed of cars observed. In contrast, PRIVID does not support a query such as reporting the location (e.g., bounding box) of an individual or car within the video frame. PRIVID can be used for one-off ad-hoc queries or standing queries running over a long period, e.g., the total number of cars per day, each day over a year.

The VO decides the level of privacy provided by PRIVID. The VO chooses a privacy policy (ρ,K) and privacy budget (ϵ) for each camera they manage. Given these parameters, PRIVID provides a guarantee of (ρ,K,ϵ)-privacy (Theorem 5.2) for all queries over all cameras it manages.

To satisfy the privacy guarantee, PRIVID utilizes the standard Laplace mechanism from DP [37] to add random noise to the aggregate query result before returning the result to the analyst. The key technical pieces of PRIVID are: (i) providing analysts the ability to specify queries using arbitrary untrusted code (§5.2), (ii) adding noise to results to guarantee (ρ,K,ϵ)-privacy for a single query (§5.5), and (iii) extending the guarantee to handle multiple queries over the same cameras (§5.6).

5.2 PRIVID Query Interface

Execution model. PRIVID requires queries to be expressed using a *split-process-aggregate* model in order to tie the duration of an event to the amount it can impact the query output. The target video is split temporally into chunks, then each chunk is fed to a separate instance of the analyst's processing code, which outputs a set of rows. Together, these rows form a traditional tabular database (untrusted by PRIVID since it is generated by the analyst). The aggregation stage runs a SQL query over this table to produce a raw result. Finally, PRIVID adds noise (§5.5) and returns *only* the noisy result to the analyst, not the raw result or the intermediate table.

Query contents. A PRIVID query must contain (1) a block of statements in a SQL-like language, which we introduce below and call PRIVIDQL, and (2) video processing executables.

(1) PrividQL statements. A valid query contains one or more of *each* of the 3 following statements. We provide an example in §5.7.1 and the full grammar in §E of [33].

- `SPLIT` statements choose a segment of video (camera, start and end datetime) as input, and produce a set of video chunks as output. They specify how the segment should be split into chunks, i.e., the chunk duration and stride between chunks.
- `PROCESS` statements take a set of `SPLIT` chunks as input, and produce a traditional ("intermediate") table. They specify the executable that should process the chunks, the schema of the resulting table, and the maximum number of rows a chunk can output (max_rows, necessary to bound the sensitivity, §5.5). Any rows output beyond the max are dropped.
- `SELECT` statements resemble typical SQL `SELECT` statements that operate over the tables resulting from `PROCESS` statements and output a (ρ,K,ϵ)-private result. They must have an aggregation as the final operation. Privid supports the standard aggregation functions (e.g., `COUNT`, `SUM`, `AVG`) and the core set of typical operators as internal relations. An aggregation must specify the range of each column it aggregates (just as in related work on DP for SQL [50]). Each `SELECT` constitutes at least one data release: one for a single aggregation or multiple for a `GROUPBY` (one for each key). Each data release receives its own sample of noise and consumes additional privacy budget (§5.6). In order to aggregate across multiple video sources (separate time windows and/or multiple cameras), the query can use a `SPLIT` and `PROCESS` for each video source, and then aggregate using a `JOIN` and `GROUPBY` in the `SELECT`.

(2) `PROCESS` executables. Executables take one chunk as input, and produce a set of rows (e.g., one per object) as output.

5.3 Providing Privacy Despite Blackbox Executables

When running a Privid query, an analyst can observe only two pieces of information: (1) the query result, and (2) the time it takes to receive the result.

Query result. In order to link an event's duration to its impact on the output, Privid ensures that the output of processing a chunk i can *only* be influenced by what is visible in chunk i (not any other chunk j). Then, an individual can *only* impact the outputs of chunks in which they appear, and the duration of their appearance is directly proportional to their contribution to the output table.

To achieve this, Privid processes each chunk using a separate instance of the analyst's executable, each running in its own isolated environment. This environment enforces that the executable can read *only* the video chunk, camera metadata, and a random number generator, and can output *only* values formatted according to the `PROCESS` schema. However, the executable may use arbitrary operations (e.g., custom ML models for CV tasks).

Execution time. To prevent the execution time from leaking any information, we must add two additional constraints. First, each chunk must complete and return a value within a pre-determined time limit T, otherwise a default value is returned for that chunk (both T and the default value are provided by the analyst at query time).[6] Second, Privid only returns the final aggregated query result after $|chunks| \cdot T$. By enforcing these constraints, the observed return time is only a property of the query itself, not the data.

Implementation. Our prototype implementation (described in §D of [33]) satisfies these requirements using standard Linux tools. Alternatively, a deployment of Privid could use related work [8, 24, 35] on strong isolation with low overhead.

5.4 Interface Limitations

The main limitation of Privid's query interface is the inability to write queries that maintain state across separate chunks. However, in most cases this does not preclude queries, it simply requires them to be expressed in a particular way. One broad class of such queries are those that operate over *unique* objects. Consider a query that counts cars. A straightforward implementation might detect `car` objects, output one row for each object, and count the number of rows. However, if a car enters the camera view in chunk i and is last visible in chunk $i+n$, the `PROCESS` table will include n rows for the same car instead of the expected 1. To minimize overcounting, the executable can incorporate a license plate reader, output a `plate` attribute for each car, and then `count(DISTINCT plate)` in the `SELECT` (as in §5.7.1).

Suppose instead the query were counting people, who do not have globally unique identifiers. To minimize overcounting, the `PROCESS` executable could choose to output a row only for people that *enter* the scene *during that chunk* (and ignore any people that are already visible at the start of a chunk).

Privid's aggregation interface imposes some limitations beyond traditional SQL (detailed in §E of [33], e.g., the `SELECT` must specify the range of each column), but these are equivalent to the limitations of DP SQL interfaces in prior work.

5.5 Query Sensitivity

The sensitivity of a Privid query is the maximum amount the final query output could differ given the presence or absence of any (ρ,K)-bounded event in the video. This can be broken down into two questions: (1) what is the maximum number of rows a (ρ, K)-bounded event could impact in the analyst-generated intermediate table, and (2) how much could each of these rows contribute to the aggregate output. We discuss each in turn.

Contribution of a (ρ,K) event to the table. An event that is visible in even a single frame of a chunk can impact the output of that chunk arbitrarily, but due to Privid's isolated execution environment, it can *only* impact the output of that chunk, not any others. Thus, the number of rows a (ρ, K)-bounded event could impact is dependent on the number of chunks it spans (an event spans a set of chunks if it is visible in at least one frame of each).

[6] Timeouts can impact query accuracy, hence analysts should first profile their code to select a conservative limit T.

In the worst case, an event spans the most contiguous chunks when it is first visible in the last frame of a chunk. Given a chunk duration c (same units as ρ) a single event segment of duration ρ can span at most $\mathsf{max\_chunks}(\rho)$ chunks:

$$\mathsf{max\_chunks}(\rho) = 1 + \lceil \frac{\rho}{c} \rceil \tag{5.1}$$

Definition 5.1 (Intermediate Table Sensitivity). Consider a privacy policy (ρ, K), and an intermediate table t (created with a chunk size of c_t and maximum per-chunk rows $\mathsf{max\_rows}_t$). The *sensitivity* of t w.r.t (ρ, K), denoted $\Delta_{(\rho,K)}$, is the maximum number of rows that could differ given the presence or absence of any (ρ, K)-bounded event:

$$\Delta_{(\rho,K)}(t) \leq \mathsf{max\_rows}_t \cdot K \cdot \mathsf{max\_chunks}(\rho) \tag{5.2}$$

Proof. In the worst case, none of the K segments overlap, and each starts at the last frame of a chunk. Thus, each spans a separate $\mathsf{max\_chunks}(\rho)$ chunks (Eq. 5.1). For each of these chunks, all of the $\mathsf{max\_rows}$ output rows could be impacted. □

Sensitivity propagation for (ρ, K)-bounded events. Prior work [45, 50, 57] has shown how to compute the sensitivity of a SQL query over *traditional* tables. Assuming that queries are expressed in relational algebra, they define the sensitivity recursively on the abstract syntax tree. Beginning with the maximum number of rows an individual could influence in the input table, they provide rules for how the influence of an individual propagates through each relational operator and ultimately impacts the aggregation function.

Unlike prior work on propagating sensitivity recursively, the intermediate tables in Privid are untrusted, and thus require careful consideration to ensure the privacy definition is rigorously guaranteed. In this work, we determined the set of operations that can be enabled over Privid's intermediate tables, derived the sensitivity for each, and proved their correctness. Many rules end up being analogous or similar to those in prior work, but `JOIN`s are different. We provide a brief intuition for these differences below. Fig. 9 in §B contains the complete definition for sensitivity of a Privid query.

Privacy semantics of untrusted tables. As an example, consider a query that computes the size of the intersection between two cameras, `PROCESS`'d into intermediate tables t_1 and t_2 respectively. If $\Delta(t_1) = x$ and $\Delta(t_2) = y$, it is tempting to assume $\Delta(t_1 \cap t_2) = \min(x, y)$, because a value needs to appear in both t_1 and t_2 to appear in the intersection. However, because the analyst's executable can populate the table arbitrarily, they can "prime" t_1 with values that would only appear in t_2, and vice versa. As a result, a value need only appear in either t_1 or t_2 to show up in the intersection, and thus $\Delta(t_1 \cap t_2) = x + y$.

Theorem 5.1. Privid's sensitivity definition (Fig. 9, §B) provides (ρ, K, ϵ)-privacy for a query Q over V.

We provide the formal proof in §B.

5.6 Handling Multiple Queries

In traditional DP, the parameter ϵ is viewed as a "privacy budget". Informally, ϵ defines the total amount of information that may be released about a database, and each query consumes a portion of this budget. Once the budget is depleted, no further queries can be answered.

Rather than assigning a single global budget to an entire video, Privid allocates a separate budget of ϵ to each frame of a video. When Privid receives a query Q over frames $[a, b]$ requesting budget ϵ_Q, it only accepts the query if *all* frames in the interval $[a - \rho, b + \rho]$ have sufficient budget $\geq \epsilon_Q$, otherwise the query is denied (Alg. 1 Lines 1-3). If the query is accepted, Privid then subtracts ϵ_Q from each frame in $[a, b]$, but *not* the ρ margin (Alg. 1 Lines 4-5). We require sufficient budget at the ρ margin to ensure that any single segment of an event (which has duration at most ρ) cannot span two temporally disjoint queries (§B).

Note that since each `SELECT` in a query represents a separate data release, the total budget ϵ_Q used by a query is the sum of the ϵ_i used by each of the i `SELECT`s. The analyst can specify the amount of budget they would like to use for each release (via a `CONSUMING` clause, defined in §E of [33], see example in §5.7.1).

Putting it all together. Algorithm 1 presents a simplified (single video) version of the Privid query execution process. We provide the full algorithm in §G of [33].

Algorithm 1: Privid Query Execution (simplified)

```
   Input  : Query $Q$, video $V$, interval $[a,b]$, policy $(\rho, K, \epsilon)$
   Output : Query answer $A$
1  foreach frame $f \in V[a-\rho : b+\rho]$ do
2      if $f.budget < \epsilon_Q$ then
3          return DENY
4  foreach frame $f \in V[a:b]$ do
5      $f.budget$ -= $Q.budget$
6  chunks ← Split $V[a:b]$ into chunks of duration $c$
7  $T$ ← Table(schema)
8  foreach $chunk \in chunks$ do
9      rows ← $F(chunk)$ // in isolated environment
10     $T$.append(rows)
11 $r$ ← execute PrividQL query $S$ over table $T$
12 $\Delta_{(\rho,K)}$ ← compute recursively over the structure of $S$ (§5.5)
13 $\eta$ ← $Laplace(\mu = 0, b = \frac{\Delta}{\epsilon_Q})$
14 $A$ ← $r + \eta$
```

Theorem 5.2. Consider an adaptive sequence (§2.3) of n queries $Q_1, \ldots, Q_n$, each over the same camera C, a privacy policy (ρ_C, K_C), and global budget ϵ_C. Privid (Algorithm 1) provides $(\rho_C, K_C, \epsilon_C)$-privacy for all $Q_1, \ldots, Q_n$.

We provide the formal proof in §B.

5.7 Example Queries

5.7.1 Benevolent Query

Suppose a VO provides access to `camA` via Privid, with a policy $(\rho=60s, K=2)$. The city transportation department wishes to collect statistics about vehicles passing `camA` during October 2021. We formulate two questions as a Privid query:

```
-- Select 1 month time window from camera, split into chunks
SPLIT camA
   BEGIN 10-01-2021/12:00am END 11-01-2021/12:00am
   BY TIME 10sec STRIDE 0sec
   INTO chunksA;
-- Process chunks using analyst's code, store outputs in tableA
PROCESS chunksA USING traffic_flow.py TIMEOUT 1sec
   PRODUCING 20 ROWS
   WITH SCHEMA (plate:STRING="", type:STRING="", speed:NUMBER=0)
   INTO vehiclesA;
-- S1: Number of unique cars per day
SELECT day,COUNT(DISTINCT plate) FROM vehiclesA WHERE type=="car"
      GROUP BY day CONSUMING eps=0.5;
-- S2: Average speed of trucks
SELECT AVG(range(speed, 30, 60)) FROM vehiclesA WHERE type=="truck"
      CONSUMING eps=0.5;
```

The `SPLIT` selects 1 month of video from `camA`, then divides the frames into a list of 10-second-long chunks (267k chunks total). The `PROCESS` first creates an empty table based on the `SCHEMA` (3 columns). Then, for each chunk, it starts a fresh instance of `traffic_flow.py` inside a restricted container, provides the chunk as input, and appends the output as rows to `vehiclesA`. The executable `traffic_flow.py` contains off-the-shelf object detection and tracking models, a license plate reader, and a speed estimation algorithm (source in §F of [33]).

The first `SELECT` filters all cars, then counts the "distinct" license plates to estimate the number of *unique* cars per day. Each day is a separate data release with an independent sample of noise. The second `SELECT` filters all trucks, then computes the average speed across the entire month of footage. It uses the same input video as the first select, and thus draws from the same budget, so in aggregate the two `SELECT`s consume $\epsilon=1.0$ budget from all frames in October 2021.

5.7.2 Malicious Query Attempt

Now consider a malicious analyst Mal who wishes to determine if individual x appeared in front of `camA` each day. Assume x's appearance is bound by the VO's (ρ,K) policy.

To hide their intent, Mal disguises their query as a traffic counter, mimicking S_1 from the previous example. They write identical query statements, but their "`traffic_flow.py`" instead includes specialized models to detect x. If x appears, it outputs 20 rows (the maximum) with random values for each of the columns, otherwise it outputs 0 rows. This adds 20 rows to the corresponding daily count for each chunk x appears.

Amplification attempt. Due to the isolated environment (§5.3), the `PROCESS` executable can only output rows for a chunk if x *truly appears.* It has no way of saving state or communicating between executions in order to artificially output rows for a chunk in which x does not appear. It could output more than 20 rows for a single chunk, but Privid ignores any rows beyond the `PROCESS`'s explicit max (20), so this would not increase the count. Increasing the rows per chunk parameter would also be pointless: Privid would compute a proportionally higher sensitivity and add proportionally higher noise.

Side channel attempt. The executable could try to encode the entire contents of a frame in a row of the table, either by encoding it as a string, or a very large number of individual integer columns. But in either case, the analyst cannot view the table directly or even a single row directly, it can only compute noisy aggregations over entire columns.

Summary. Privid would compute the sensitivity of S_1 (identical in both the benevolent and malicious cases) as $\Delta_{(60,2)}(Q) \leq 20\cdot 2\cdot(1+\lceil\frac{60}{10}\rceil)=280$ rows, meaning it would add noise with scale 280 to each daily count. Regardless of how Mal changes her executable, it cannot output more than 280 rows based on x's presence. Thus, even if she observed a non-zero value ~ 280, she could not distinguish whether it is a result of the noise or x's appearance.

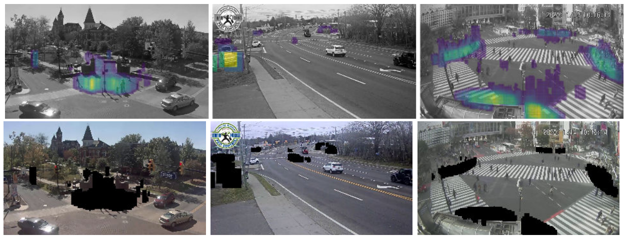
Mal's query gets a useless result, because her target (x's appearance) was close in duration to the policy. In contrast, the benevolent query can get a useful result because the duration of its target (the set of *all cars'* appearances) far exceeds the policy. Privid's noise will translate to $\mathcal{L}^-1(p=0.99, u=0, b=\frac{\Delta}{\epsilon}=\frac{280}{0.5}) \leq 2200$ cars with 99% confidence. If, for example, there are an average of 10 cars in each chunk (and thus 86000 in one day), 2200 represents an error of $\pm 2.5\%$.

6 Query Utility Optimization

The noise that Privid adds to a query result is proportional to both the privacy policy (ρ,K) and the range of the aggregated values (the larger the range, the more noise Privid must add to compensate for it). In this section we introduce two optional optimizations that Privid offers analysts to improve query accuracy while maintaining an equivalent level of privacy: one reduces the ρ needed to preserve privacy (§6.1), while the other reduces the range for aggregation (§6.2).

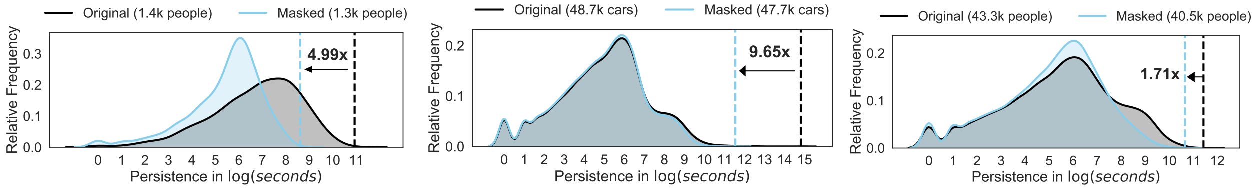
6.1 Spatial Masking

Observation. In certain settings, a few individuals may be visible to a camera for far longer than others (e.g., those sitting on a bench or in a car), creating a heavy-tailed distribution of presence durations. Fig. 3 (top row) provides some representative examples. Setting $(\rho,\, K)$ to the maximum



(a) `campus` (b) `highway` (c) `urban`

Figure 3: (Top) Heatmap measuring the maximum time any object spent in each pixel, noramlized to the max (yellow) per video. (Bottom) The resulting masks used for our evaluation, chosen from the list of masks automatically generated using the algorithm in §I of [33].



(a) campus **(b) highway** **(c) urban**

Figure 4: The distribution of private objects' durations (persistence) is heavy tailed. Applying the mask from Fig. 3 significantly lowers the maximum duration, while still allowing most private objects to be detected. The key denotes the total number of private objects detectable before and after applying the mask. The dotted lines highlight the maximum persistence, and the arrow text denotes the relative reduction.

duration in such distributions would result in a large amount of noise needed to protect just those few individuals; all others could have been protected with a far lower amount of noise. We observe that, in many cases, lingering individuals tend to spend the majority of their time in one of a few fixed regions in the scene, but a relatively short time in the rest of the scene. For example, a car may be parked in a spot for hours, but only visible for 1 minute while entering/leaving the spot.

Opportunity. Masking fixed regions (i.e., removing those pixels from all frames prior to running the analyst's video processing) in the scene that contain lingering individuals would drastically reduce the *observable* maximum duration of individuals' presence, e.g., the parked car from above would be observable for 1 minute rather than hours. This, in turn, would permit a policy with a smaller ρ, but an equivalent level of privacy–all appearances would still be bound by the policy. Of course, this technique is only useful to an analyst when the remaining (unmasked) part of the scene includes all the information needed for the query at hand, e.g., if counting cars, masking sidewalks would be reasonable but masking roads would not.

Optimization. At camera-registration time, instead of providing a single (ρ,K) policy per camera, the VO can provide a (fixed) list of a few frame masks and, for each, a corresponding (ρ,K) policy that would provide equivalent privacy when that mask is applied. At query time, the analyst can (optionally) choose a mask from the list that would minimally impact their query goal while maximizing the level of noise reduction (via the tighter (ρ,K) bound). If a mask is chosen, PRIVID applies it to all video frames *before* passing it to the analyst's PROCESS executable (the analyst only "sees" the masked video), and uses the corresponding (ρ,K) in the sensitivity calculation (§5.5).

To aid the analyst in discovering a useful set of masks (i.e., those that reduce (ρ,K) as much as possible using the fewest pixels), we provide an algorithm in §I.2 of [33]. Regardless of how they are chosen, the masks themselves are static (i.e., the same pixels are masked in every frame regardless of its contents), and the set of available masks is fixed. Neither depend on the query or the target video. Further, the mask itself does not reveal how the analyst generated it or which specific objects contributed to it, it only tells the analyst that some objects appear for a long duration in the masked region.

Noise reduction. We demonstrate the potential benefit of masking on three queries (Q1-Q3) from our evaluation (Table 3). Given the query tasks (counting unique people and cars), we chose masks that would maximally reduce ρ without impacting the object counts; the bottom row of Fig. 3 visualizes our masks. Fig. 4 shows that these masks reduce maximum durations by 1.71-9.65×. In §I.1 of [33] we show that masking provides similar benefits for 7 additional videos evaluated by BlazeIt [47] and MIRIS [30].

Video	Max(frame)	Max(region)	Reduction
campus	6	3	2.00×
highway	40	23	1.74×
urban	37	16	2.25×

Table 2: Reduction in max output range from splitting each video into distinct regions. Reduction shows the factor by which the noise could be reduced. 2× cuts the necessary privacy level in half.

Masking vs. denaturing. Although masking is a form of denaturing, PRIVID uses it differently than the prior approaches in §3.1, in order to sidestep their issues. Rather than attempting to dynamically hide individuals as they move through the scene, PRIVID's masks cover a *fixed* location in the scene and are publicly available so analysts can account for them in their query implementation. Also, masks are used as an optional modification to the input video; the rest of the PRIVID pipeline, and thus its formal privacy guarantees, remain the same.

6.2 Spatial Splitting

Observation. (1) At any point in time, each object typically occupies a relatively small area of a video frame. (2) Many common queries (e.g., object detections) do not need to examine the entire contents of a frame at once, i.e., if the video is split spatially into regions, they can compute the same total result by processing each of the regions separately.

Opportunity. PRIVID already splits videos temporally into chunks. If each chunk is further divided into spatial regions and an individual can only appear in one of these chunks at a time, then their presence occupies a relatively smaller portion of the intermediate table (and thus requires less noise to protect). Additionally, the maximum duration of each individual region may be smaller than the frame as a whole.

Optimization. At camera-registration time, PRIVID allows VOs to manually specify boundaries for dividing the scene into regions. They must also specify whether the boundaries are soft (individuals may cross them over time, e.g., between two crosswalks) or hard (individuals will never cross them, e.g., between opposite directions on a highway). At query time, analysts can optionally choose to spatially split the video using these boundaries. Note that this is in addition to, rather than in replacement of, the temporal splitting. If the boundaries are soft, tables created using that split must use a chunk size of 1 frame to ensure that an individual can always be in at most 1 chunk. If the boundaries are hard, there are no restrictions on chunk size since the VO has stated the constraint will always be true.

Noise reduction. We demonstrate the potential benefit of spatial splitting on three videos from our evaluation (Q1-Q3). For each video, we manually chose intuitive regions: a separate region for each crosswalk in `campus` and `urban` (2 and 4, respectively), and a separate region for each direction of the road in `highway`. Table 2 compares the range necessary to capture all objects that appear within one chunk in the entire frame compared to the individual regions. The difference (1.74-2.25×) represents the potential noise reductions from splitting: noise is proportional to $\max(\text{frame})$ or $\max(\text{region})$ when splitting is disabled or enabled, respectively.

Grid split. To increase the applicability of spatial splitting, Privid could allow analysts to divide each frame into a grid and remove the restrictions on soft boundaries to allow any chunk size. This would require additional estimates about the max size of any private object (dictating the max number of regions they could occupy at any time), and the maximum speed of any object across the frame (dictating the max number of regions they could move between). We leave this to future work.

7 Evaluation

The evaluation highlights of Privid are as follows:

1. Privid supports a diverse range of video analytics queries, including object counting, duration queries, and composite queries; for each, Privid increases error by 1-21% relative to a non-private system, while protecting all individuals with (ρ, K, ϵ)-privacy (§7.2).
2. Privid enables VOs and analysts to flexibly and formally trade utility loss and query granularity while preserving the same privacy guarantee (§7.3).

7.1 Evaluation Setup

Datasets. We evaluated Privid primarily using three representative video streams (`campus`, `highway` and `urban`, screenshots in Fig. 3) that we collected from YouTube spanning 12 hours each (6am-6pm). For one case study (multi-camera), we use the Porto Taxi dataset [58] containing 1.7mil trajectories of all 442 taxis running in the city of Porto, Portugal from Jan. 2013 to July 2014. We apply the same processing as [42] to emulate a city-wide camera dataset; the result is the set of timestamps each taxi would have been visible to each of 105 cameras over the 1.5 year period.

Implementation. We implemented Privid in 4k lines of Python. We used the Faster-RCNN [63] model in Detectron-v2 [71] for object detection, and DeepSORT [69] for object tracking. For these models to work reasonably given the diverse content of the videos, we chose hyperparameters for detection and tracking on a per-video basis (details in §H of [33]).

Privacy policies. We assume the VO's underlying privacy goal is to "protect the appearance of all individuals". For each camera, we use the strategy in §6.1, to create a map between masks and (ρ, K) policies that achieve this goal.

Query parameters. For each query, we first chose a mask that covered as much area as possible (to get the minimal ρ) without disrupting the query. The resulting ρ values are in Table 3. We use a budget of $\epsilon = 1$ for each query. We chose query windows sizes (W), chunk durations (c), and column ranges to best approximate the analyst's expectations for each query (as opposed to picking optimal values based on a parameter sweep, which the analyst is unable to do).

Baselines. For each query, we compute `error` by comparing the output of Privid to running the same exact query implementation without Privid. We execute each query 1000 times, and report the mean accuracy value ± 1 standard deviation.

7.2 Query Case Studies

We formulate five types of queries to span a variety of axes (target object class, number of cameras, aggregation type, query duration, standing vs. one-off query). Fig. 5 displays results for Q1-Q3. Table 3 summarizes the remaining queries (Q4-Q13).

Case 1: Q1-Q3 (Counting private objects over time). To demonstrate Privid's support for standing queries and short (1 hour) aggregation durations, we `SUM` the number of *unique* objects observed *each hour* over the 12 hours.

Case 2: Q4-Q6 (Aggregating over multiple cameras with complex operators). We utilize `UNION`, `JOIN`, and `ARGMAX` to aggregate over cameras in the `Porto Taxi Dataset`. Due to the large aggregation window (1 year), Privid's noise addition is small (relative to the other queries using a window on the order of hours) and accuracy is high.

Case 3: Q7-Q9 (Counting non-private objects, large window). We measure the fraction of trees (non-private objects) that have bloomed in each video. Executed over an entire network of cameras, such a query could be used to identify the regions with the best foliage in Spring. Relative to Case 1, we achieve high accuracy by using a longer query window of 12 hours (the status of a tree does not change on that time scale), and minimal chunk size (1 frame, no temporal context needed).

Case 4: Q10-Q12 (Fine-grained results using aggressive masking). We measure the average amount of time a traffic signal stays red. Since this only requires observing the light itself, we can mask *everything else*, resulting in a ρ bound of 0 (no private objects overlap these pixels), enabling high accuracy and fine temporal granularity.

Case 5: Q13 (Stateful query). We count only the individuals that enter from the south and exit at the north. It requires a larger chunk size (relative to Q1-Q3) to maintain enough state within a single chunk to understand trajectory.

Case #	Q#	Query Description	Query Parameters	Video	ρ	Query Output	Error
Case 2	Q4	Average Taxi Driver Working Hours (`union` across 2 cameras)	$\lvert W\rvert = 365$ days, $c = 15$ sec, Agg = `avg`, range $= (0,16)$	`porto10`, `porto27`	[45, 195] sec	5.87 hrs	$5.86\% \pm 0.18\%$
Case 2	Q5	Average # Taxis Traversing 2 Locations on Same Day (`intersection` across 2 cameras)	$\lvert W\rvert = 365$ days, $c = 15$ sec, Agg = `avg`, range $= (0,300)$	`porto10`, `porto27`	[45, 195] sec	131 taxis	$0.20\% \pm 0.13\%$
Case 2	Q6	Identifying Camera with Highest Daily Traffic (`argmax` across all 105 cameras)	$\lvert W\rvert = 365$ days, $c = 15$ sec, Agg = `argmax`	`porto0`, ..., `porto104`	[15, 525] sec	`porto20`	0%
Case 3	Q7	Fraction of trees with leaves (%)	$\lvert W\rvert = 12$ hrs, $c = 1$ frame, Agg = `avg`, range $= (0,100)$	`campus`	49 sec	15/15 = 1.00	$0.10\% \pm 0.11\%$
	Q8			`highway`	6.21 min	3/7 = 0.43	$1.76\% \pm 1.90\%$
	Q9			`urban`	3.34 min	4/6 = 0.67	$0.61\% \pm 0.66\%$
Case 4	Q10	Duration of Red Light (seconds)	$\lvert W\rvert = 12$ hrs, $c = 30$ min, Agg = `avg`, range $= (0,300)$	`campus`	1 frame	75 sec	$0\% \pm 1.4 \times 10^{-4}\%$
	Q11			`highway`	1 frame	50 sec	$0\% \pm 2.1 \times 10^{-4}\%$
	Q12			`urban`	1 frame	100 sec	$0\% \pm 1.0 \times 10^{-4}\%$
Case 5	Q13	# Unique People (Filter: trajectory moving towards campus)	$\lvert W\rvert = 12$ hrs, $c = 10$ sec, Agg = `sum`, range $= (0,5)$	`campus`	49 sec	576 people	$20.31\% \pm 2.60\%$

Table 3: Summary of query results for Q4-Q13. For Case 3 and 5, we use the same masks (and thus ρ) from Fig. 3. For Case 4, we mask all pixels except the traffic light to attain $\rho = 0$. For Case 2 we do not use any masks.

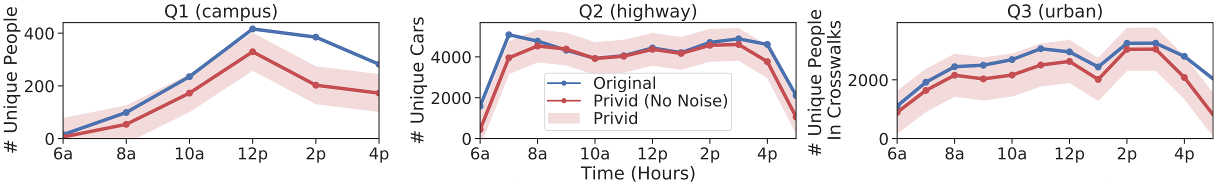


Figure 5: Time series of Privid's output for Case 1 queries. "Original" is the baseline query output without using Privid. "Privid (No Noise)" shows the raw output of Privid before noise is added. The final noisy output will fall within the range of the red ribbon 99% of the time.

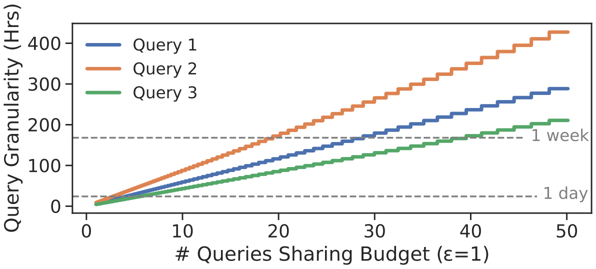


Figure 6: Given a fixed query and accuracy target, decreasing the amount of budget used by each query allows more queries to be executed over the same video segment, but requires a proportionally coarser granularity. The x-axis plots the number of queries evenly sharing a budget of $\epsilon = 1$, thus $x = 10$ means 10 instances of the same exact query over the same video segment, each using a budget of $\frac{1}{10}$. We fix the accuracy target to be 99% of values having error $\leq 5\%$.

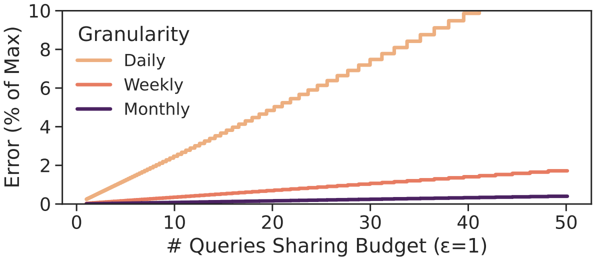


Figure 7: Given a fixed query and granularity, decreasing the amount of budget used by each query allows more queries to be executed over the same video segment, but results in proportionally higher error. The x-axis is the same as Fig. 6. Each line corresponds to Q1 using a different granularity. The y-axis plots the error for 99% of values. Error is the amount of noise added relative to the maximum query output. For example, in Q4, the final output is the average number of working hours in the range [0,16]. Thus an error of 1% would mean the noisy result is within 0.16 hours of the true result.

7.3 Budget-Granularity Tradeoff

Analysts have two main knobs for each query Q to navigate the utility space: (1) the fraction ϵ_Q of the total budget ϵ used by that query, and (2) the duration (granularity) of each aggregation (i.e., "one value per day for a month" has a granularity of one day). The query budget is inversely proportional to both the query granularity and error (the expected value of noise Privid adds relative to the output range). Thus, to decrease the amount of budget per query (or equivalently, increase the number of queries sharing the budget), an analyst must choose a (temporally) coarser result, a larger expected error bound, or both. Fig. 6 shows that, for example, 5 instances of Query 3 could release results daily or 40 instances of Query 3 could release results weekly, while achieving the same expected accuracy. Fig. 7 shows that, for example, 20 separate instances of Query 1 ($x = 20$) executed over the same target video could each expect 4.8% error if they release one result daily, 0.7% error if they release one weekly, or 0.16% error if they release one monthly. Importantly, this tradeoff is transparent to analysts: Figs. 6 and 7 rely only on information that is publicly available to analysts and did not require executing any queries.

7.4 Analyzing Sources of Inaccuracy

Privid introduces two sources of inaccuracy to a query result: (1) intentional noise to satisfy (ρ,K,ϵ)-privacy, and (2) (unintentional) inaccuracies caused by the impact of splitting and masking videos before executing the video processing. Fig. 5 shows these two sources separately for queries Q1-Q3 (Case 1): the discrepancy between the two curves demonstrates the impact of (2), while the shaded belt shows the relative scale of noise added (1). In summary, the scale of noise added by Privid allows the final result to preserve the trend of the original.

8 Using Privid

In this section, we summarize the set of decisions that both the VO and analyst need to make when interacting with Privid.

8.1 Video Owner

First, the VO must register a set of cameras with Privid. For *each* camera, they must supply: (1) a (ρ,K) bound (or more generally a map of masks to bounds), (2) a privacy budget ϵ, and (3) some metadata describing the scene to analysts (e.g., a short video clip, since they cannot view the camera feed directly). All of this is public to analysts. Below we provide general suggestions for the VO, but ultimately they are responsible for choosing these values. Privid only handles enforcing a given policy.

(1) (ρ, K) bounds. In most cases, we expect the VO will record a sample of video, measure durations of objects of interest using off-the-shelf tracking algorithms, and then set the bound to the longest duration.

To provide better utility for analysts, the VO can offer a menu of static masks that remove some of the scene in exchange for tighter noise bounds than the original policy (which is itself mapped to the empty mask). Note that the VO must explicitly choose a (ρ, K) policy for each mask. A mask is only useful if it reduces the amount of time the longest objects are visible, which enables a tighter bound while protecting the same set of individuals.

The VO may draw masks manually or generate them automatically, e.g., by analyzing past trends from the camera. In general, we expect masks to be static properties of each scene, dependent only on dynamics of the scene type, rather than behaviors of any individuals. However, it is ultimately the VO's responsibility to ensure any masks it provides do not reveal anything private, such as a person's silhouette. Privid focuses on preventing the leakage of privacy when answering queries. It does not make any guarantees about the mask itself.

(2) Budget ϵ. As in any deployment of DP, the choice of ϵ is subjective. Academic papers commonly use $\epsilon \approx 1$ [52] while recent industry deployments have used $1 < \epsilon < 10$ [27, 36, 56]. Note that in Privid, this budget is *per-frame* (§5.6); two queries aggregating over disjoint time ranges of the same video draw from separate budgets. The only Privid-specific consideration for choosing ϵ is that cameras with overlapping fields of view should share the same budget.

(3) Metadata. The VO should release a sample video clip[7] representative of the scene so that analysts can calibrate their executable[8] and query[9] accordingly. Any privacy loss resulting from the one-time release of this single clip is limited, and can be manually vetted by the VO. Optionally, the VO can release additional information to aid analysts, such as the camera's GPS coordinates, make, or focal length settings.

8.2 Analyst

In order to formulate a Privid query the analyst must make the following decisions. For each decision, we provide an example for the query in §5.7.1 (counting cars crossing a virtual line on a highway).

Choose a mask (from the list provided by the VO) based on the query goal. For example, they should select a mask that covers as much of the scene as possible without covering the area near the virtual line. This would significantly reduce the bound by removing parking spots and intersections where objects linger.

Choose a chunk size based on the amount of context needed. A larger chunk size permits more context for each execution of the `PROCESS`, but results in more noise (§5.5). Thus, the analyst should choose the smallest chunk size that captures their events of interest. For example, 1 second is likely sufficient to capture cars driving past a line. If they instead wanted to calculate car speed, they would need a larger chunk size (e.g., 10 seconds) and less restrictive mask to capture more of the car's trajectory.

Choose upper bound on number of output rows per chunk based on the expected (via the video sample) level of activity in each chunk. For counting cars over a short chunk, especially in less busy scenes, each chunk may see 1-2 cars and thus need 1-2 rows. For calculating speed over a larger chunk, especially in more busy scenes, each chunk will see more cars and may need 10 or 100 rows.

Create a `PROCESS` executable. This involves tuning their CV models based on the scene (via the sample video), and combining all tasks into a single executable. For example, their executable may include an object detector to find cars, an object tracker to link them to trajectories, a license plate reader to link cars across cameras or prevent double counting, and an algorithm to compute speed or determine car model.

Choose query granularity and budget. The query granularity and budget are directly proportional to accuracy. Given a fixed value for each, improving one requires worsening another proportionally. We elaborate upon this tradeoff in §7.3.

9 Ethics

In building Privid, we *do not* advocate for the increase of public video surveillance and analysis. Instead, we observe that it is already prevalent and seek to improve the privacy landscape. Privid's accuracy and expressiveness makes it palatable to add formal privacy to existing analytics, and lowers the barrier to deployment. If privacy legislation is introduced, Privid could be one way to ensure compliance.

Acknowledgements. We thank Hari Balakrishnan, Matt Lentz, Dave Levin, Amy Ousterhout, Jennifer Rexford, Eugene Wu, the NSDI reviewers, and our shepherd, Jonathan Mace, for their helpful feedback and suggestions. This work was partially supported by a Sloan fellowship and NSF grants CNS-2153449, CNS-2152313, CNS-2140552, and CNS-2151630.

[7] While a clip is not needed in principle, without it, the analyst "runs blind" and will not have confidence in the correctness of their results.

[8] ML models may perform better when retrained on a particular scene.

[9] For example, queries must specify bounds on the amount of output per chunk, which depend on the amount of activity in the scene.

References

[1] Absolutely everywhere in beijing is now covered by police video surveillance. https://qz.com/518874/.

[2] Are we ready for ai-powered security cameras? https://thenewstack.io/are-we-ready-for-ai-powered-security-cameras/.

[3] British transport police: Cctv. http://www.btp.police.uk/advice_and_information/safety_on_and_near_the_railway/cctv.aspx.

[4] Can 30,000 cameras help solve chicago's crime problem? https://www.nytimes.com/2018/05/26/us/chicago-police-surveillance.html.

[5] Data generated by new surveillance cameras to increase exponentially in the coming years. http://www.securityinfowatch.com/news/12160483/.

[6] Detection leaderboard. https://cocodataset.org/#detection-leaderboard.

[7] Epic domestic surveillance project. https://epic.org/privacy/surveillance/.

[8] nsjail. https://github.com/google/nsjail.

[9] Oakland bans use of facial recognition. https://www.sfchronicle.com/bayarea/article/Oakland-bans-use-of-facial-recognition-14101253.php.

[10] Paris hospitals to get 1,500 cctv cameras to combat violence against staff. https://bit.ly/2OYiBz2.

[11] Powering the edge with ai in an iot world. https://www.forbes.com/sites/forbestechcouncil/2020/04/06/powering-the-edge-with-ai-in-an-iot-world/.

[12] San francisco is first us city to ban facial recognition. https://www.bbc.com/news/technology-48276660.

[13] Video analytics applications in retail - beyond security. https://www.securityinformed.com/insights/co-2603-ga-co-2214-ga-co-1880-ga.16620.html/.

[14] The vision zero initiative. http://www.visionzeroinitiative.com/.

[15] What's wrong with public video surveillance? https://www.aclu.org/other/whats-wrong-public-video-surveillance, 2002.

[16] Abuses of surveillance cameras. http://www.notbored.org/camera-abuses.html, 2010.

[17] Mission creep-y: Google is quietly becoming one of the nation's most powerful political forces while expanding its information-collection empire. https://www.citizen.org/wp-content/uploads/google-political-spending-mission-creepy.pdf, 2014.

[18] Mission creep. https://www.aclu.org/other/whats-wrong-public-video-surveillance, 2017.

[19] How retail stores can streamline operations with video content analytics. https://www.briefcam.com/resources/blog/how-retail-stores-can-streamline-operations-with-video-content-analytics/, 2020.

[20] The mission creep of smart streetlights. https://www.voiceofsandiego.org/topics/public-safety/the-mission-creep-of-smart-streetlights/, 2020.

[21] Video analytics traffic study creates baseline for change. https://www.govtech.com/analytics/Video-Analytics-Traffic-Study-Creates-Baseline-for-Change.html, 2020.

[22] What is computer vision? ai for images and video. https://www.infoworld.com/article/3572553/what-is-computer-vision-ai-for-images-and-video.html, 2020.

[23] P. Aditya, R. Sen, P. Druschel, S. Joon Oh, R. Benenson, M. Fritz, B. Schiele, B. Bhattacharjee, and T. T. Wu. I-pic: A platform for privacy-compliant image capture. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '16, page 235–248, New York, NY, USA, 2016. Association for Computing Machinery.

[24] A. Agache, M. Brooker, A. Iordache, A. Liguori, R. Neugebauer, P. Piwonka, and D.-M. Popa. Firecracker: Lightweight virtualization for serverless applications. In *17th {usenix} symposium on networked systems design and implementation ({nsdi} 20)*, pages 419–434, 2020.

[25] Amazon. Rekognition. https://aws.amazon.com/rekognition/.

[26] G. Ananthanarayanan, Y. Shu, M. Kasap, A. Kewalramani, M. Gada, and V. Bahl. Live video analytics with microsoft rocket for reducing edge compute costs, July 2020.

[27] Apple Differential Privacy Team. Learning with privacy at scale. *Apple Machine Learning Journal*, 1(8), 2017.

[28] M. Azure. Computer vision api. https://azure.microsoft.com/en-us/services/cognitive-services/computer-vision/, 2021.

[29] M. Azure. Face api. https://azure.microsoft.com/en-us/services/cognitive-services/face/, 2021.

[30] F. Bastani, S. He, A. Balasingam, K. Gopalakrishnan, M. Alizadeh, H. Balakrishnan, M. Cafarella, T. Kraska, and S. Madden. Miris: Fast object track queries in video. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, SIGMOD '20, page 1907–1921, New York, NY, USA, 2020. Association for Computing Machinery.

[31] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468, 2016.

[32] Z. Cai, M. Saberian, and N. Vasconcelos. Learning complexity-aware cascades for deep pedestrian detection. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 3361–3369, Washington, DC, USA, 2015. IEEE Computer Society.

[33] F. Cangialosi, N. Agarwal, V. Arun, J. Jiang, S. Narayana, A. Saarwate, and R. Netravali. Privid: Practical, privacy-preserving video analytics queries (extended version). https://arxiv.org/abs/2106.12083.

[34] A. Chattopadhyay and T. E. Boult. Privacycam: a privacy preserving camera using uclinux on the blackfin dsp. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[35] G. Chrome. minijail0. https://google.github.io/minijail/.

[36] B. Ding, J. Kulkarni, and S. Yekhanin. Collecting telemetry data privately. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3571–3580. Curran Associates, Inc., 2017.

[37] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In S. Halevi and T. Rabin, editors, *Theory of Cryptography*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284, Berlin, Heidelberg, Mar. 2006. Springer.

[38] I. Ghodgaonkar, S. Chakraborty, V. Banna, S. Allcroft, M. Metwaly, F. Bordwell, K. Kimura, X. Zhao, A. Goel, C. Tung, et al. Analyzing worldwide social distancing through large-scale computer vision. *arXiv preprint arXiv:2008.12363*, 2020.

[39] Google. Cloud vision api. https://cloud.google.com/vision, 2021.

[40] K. Hsieh, G. Ananthanarayanan, P. Bodik, S. Venkataraman, P. Bahl, M. Philipose, P. B. Gibbons, and O. Mutlu. Focus: Querying large video datasets with low latency and low cost. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 269–286, 2018.

[41] IBM. Maximo remote monitoring. https://www.ibm.com/products/maximo/remote-monitoring, 2021.

[42] S. Jain, G. Ananthanarayanan, J. Jiang, Y. Shu, and J. E. Gonzalez. Scaling Video Analytics Systems to Large Camera Deployments. In *ACM HotMobile*, 2019.

[43] S. Jain, X. Zhang, Y. Zhou, G. Ananthanarayanan, J. Jiang, Y. Shu, V. Bahl, and J. Gonzalez. Spatula: Efficient cross-camera video analytics on large camera networks. In *ACM/IEEE Symposium on Edge Computing (SEC 2020)*, November 2020.

[44] J. Jiang, G. Ananthanarayanan, P. Bodik, S. Sen, and I. Stoica. Chameleon: scalable adaptation of video analytics. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, pages 253–266. ACM, 2018.

[45] N. Johnson, J. P. Near, and D. Song. Towards practical differential privacy for sql queries. *Proceedings of the VLDB Endowment*, 11(5):526–539, 2018.

[46] P. Kairouz, S. Oh, and P. Viswanath. The composition theorem for differential privacy. *IEEE Transactions on Information Theory*, 63(6):4037–4049, 2017.

[47] D. Kang, P. Bailis, and M. Zaharia. Blazeit: optimizing declarative aggregation and limit queries for neural network-based video analytics. *Proceedings of the VLDB Endowment*, 13(4):533–546, 2019.

[48] D. Kang, J. Emmons, F. Abuzaid, P. Bailis, and M. Zaharia. Noscope: optimizing neural network queries over video at scale. *Proceedings of the VLDB Endowment*, 10(11):1586–1597, 2017.

[49] D. Kang, J. Guibas, P. Bailis, T. Hashimoto, and M. Zaharia. Task-agnostic indexes for deep learning-based queries over unstructured data. *arXiv preprint arXiv:2009.04540*, 2020.

[50] I. Kotsogiannis, Y. Tao, X. He, M. Fanaeepour, A. Machanavajjhala, M. Hay, and G. Miklau. Privatesql: A differentially private sql query engine. *Proc. VLDB Endow.*, 12(11):1371–1384, July 2019.

[51] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017.

[52] Y.-H. Kuo, C.-C. Chiu, D. Kifer, M. Hay, and A. Machanavajjhala. Differentially private hierarchical count-of-counts histograms. *Proceedings of the VLDB Endowment*, 11.11:1509—1521, 2018.

[53] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua. A convolutional neural network cascade for face detection. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5325–5334, June 2015.

[54] Y. Li, A. Padmanabhan, P. Zhao, Y. Wang, G. H. Xu, and R. Netravali. Reducto: On-Camera Filtering for Resource-Efficient Real-Time Video Analytics. SIGCOMM '20, page 359–376, New York, NY, USA, 2020. Association for Computing Machinery.

[55] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 936–944, July 2017.

[56] A. Machanavajjhala, D. Kifer, J. M. Abowd, J. Gehrke, and L. Vilhuber. Privacy: Theory meets practice on the map. In *ICDE*, 2008.

[57] 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*, SIGMOD '09, page 19–30, New York, NY, USA, 2009. Association for Computing Machinery.

[58] 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*, 14(3):1393–1402, 2013.

[59] D. A. Osvik, A. Shamir, and E. Tromer. Cache attacks and countermeasures: the case of aes. In *Cryptographers' track at the RSA conference*, pages 1–20. Springer, 2006.

[60] J. R. Padilla-López, A. A. Chaaraoui, and F. Flórez-Revuelta. Visual privacy protection methods: A survey. *Expert Systems with Applications*, 42(9):4177–4195, 2015.

[61] C. Percival. Cache missing for fun and profit, 2005.

[62] R. Poddar, G. Ananthanarayanan, S. Setty, S. Volos, and R. A. Popa. Visor: Privacy-preserving video analytics as a cloud service. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 1039–1056, 2020.

[63] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[64] J. Stanley and A. C. L. Union. *The Dawn of Robot Surveillance: AI, Video Analytics, and Privacy.* American Civil Liberties Union, 2019.

[65] Y. Sun, X. Wang, and X. Tang. Deep convolutional network cascade for facial point detection. In *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '13, pages 3476–3483, Washington, DC, USA, 2013. IEEE Computer Society.

[66] H. Wang, Y. Hong, Y. Kong, and J. Vaidya. Publishing video data with indistinguishable objects. *Advances in database technology : proceedings. International Conference on Extending Database Technology*, 2020:323 – 334, 2020.

[67] H. Wang, S. Xie, and Y. Hong. Videodp: A universal platform for video analytics with differential privacy. *arXiv preprint arXiv:1909.08729*, 2019.

[68] J. Wang, B. Amos, A. Das, P. Pillai, N. Sadeh, and M. Satyanarayanan. A scalable and privacy-aware iot service for live video analytics. In *Proceedings of the 8th ACM on Multimedia Systems Conference*, pages 38–49. ACM, 2017.

[69] N. Wojke, A. Bewley, and D. Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3645–3649. IEEE, 2017.

[70] H. Wu, X. Tian, M. Li, Y. Liu, G. Ananthanarayanan, F. Xu, and S. Zhong. Pecam: Privacy-enhanced video streaming and analytics via securely-reversible transformation. In *ACM MobiCom*, October 2021.

[71] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[72] X. Yu, K. Chinomi, T. Koshimizu, N. Nitta, Y. Ito, and N. Babaguchi. Privacy protecting visual processing for secure video surveillance. In *2008 15th IEEE International Conference on Image Processing*, pages 1672–1675. IEEE, 2008.

[73] H. Zhang, G. Ananthanarayanan, P. Bodik, M. Philipose, P. Bahl, and M. J. Freedman. Live video analytics at scale with approximation and delay-tolerance. In *NSDI*, volume 9, page 1, 2017.

[74] X. Zhu, Y. Wang, J. Dai, L. Yuan, and Y. Wei. Flow-guided feature aggregation for video object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 408–417, 2017.

A Relative Privacy Guarantees

A.1 Proof

In this section, we provide a proof for Theorem 4.1. We begin with a lemma that will be helpful for the proof:

Lemma A.1. Consider an individual x whose appearance is bound by $(\hat{\rho}, \hat{K})$ in front of a camera whose policy is (ρ, K, ϵ). For every query Q there exists $\alpha, \beta \in \mathbb{R}$ such that $\alpha K(1+\beta\rho) \leq \Delta_{(\rho,K)}(Q) \leq \alpha K(2+\beta\rho)$.

Proof. Any Privid query must contain some aggregation agg as the outer-most relation, and thus we can write $Q := \Pi_{agg}(R)$. $\Delta_{(\rho,K)}(Q)$ is defined in Figure 9 for five possible aggregation operators, which are each a function of $\Delta_{(\rho,K)}(R)$ (the sensitivity of their inner relation R).

First, we will prove these bounds are true for the inner relation $\Delta_{(\rho,K)}(R)$ by induction on R (all rules for $\Delta_{(\rho,K)}(R)$ given by Figure 9):
Case (Base): $R := t$ When R is an intermediate Privid table t, its sensitivity is given directly by Equation 5.2, where $\alpha = \mathsf{max\_rows}_t$ and $\beta = 1/c$. Note, the $(1+\cdots)$ and $(2+\cdots)$ in the lemma inequalities bound $\lceil \frac{\rho}{c} \rceil$.
Case (Selection): $R := \sigma(R')$. When R is a selection from R', $\Delta_{(\rho,K)}(R) = \Delta_{(\rho,K)}(R')$. If $\Delta_{(\rho,K)}(R')$ is bound by the inequalities in the lemma statement, then $\Delta_{(\rho,K)}(R)$ is too.
Case (Projection): $R := \Pi(R')$. Same as selection.
Case (GroupBy and Join): $R := \gamma(R_1 \bowtie \ldots \bowtie R_i)$ When R is a `join` of relations R_i proceeded by a `GroupBy`, $\Delta_{(\rho,K)}(R) = \sum_{i=1}^{N} \Delta_{(\rho,K)}(R_i)$. Let $\Delta_{(\rho,K)} R_i$ be parameterized by α_i and β_i. If each of $\Delta_{(\rho,K)}(R_i)$ are bound by the inequalities in the lemma, then $\sum_i \Delta_{(\rho,K)}(R_i)$ is as well, but with $\alpha = \sum_{i=1}^{N} \alpha_i$ and $\beta = \sum_{i=1}^{N} \beta_i$.

Finally, each of the supported aggregation operators only involve multiplying $\Delta_{(\rho,K)}(R)$ by constants (with respect to ρ and K), and thus these constants can be subsumed into α. □

We now restate Theorem 4.1 for the reader's convenience:

Theorem A.2. Consider a camera with a fixed policy (ρ, K, ϵ). If an individual x's appearance in front of the camera is bound by some $(\hat{\rho}, \hat{K})$, then Privid effectively protects x with $\hat{\epsilon}$-DP, where $\hat{\epsilon}$ is $O(\frac{\hat{\rho}\hat{K}}{\rho K})\epsilon$, which grows (degrades) as $(\hat{\rho}, \hat{K})$ increase while (ρ, K, ϵ) are fixed, and the constants do not depend on the query.

Proof. Recall from §5 that Privid uses the Laplace mechanism: it returns $Q(V) + \eta$ to the analyst, where $Q(V)$ is the raw query result, and $\eta \sim \text{Laplace}(0, b)$, $b = \frac{\Delta_{(\rho,K)}(Q)}{\epsilon}$ and $\Delta_{(\rho,K)}(Q)$ is the global sensitivity of the query over any (ρ, K)-neighboring videos. Note that the sensitivity is purely a function of the query, and thus Privid samples noise using the same scale b regardless of how long any individual is actually visible in the video.

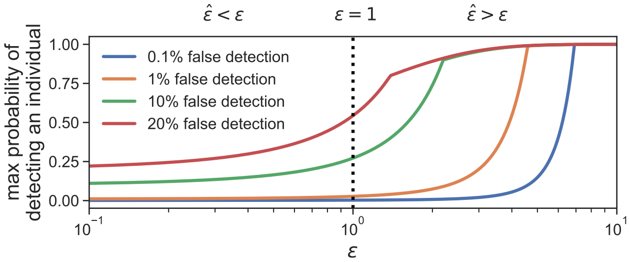


Figure 8: Plot of Equation A.4 for a few different levels of α. Note that the x-axis is plotted for absolute values of ϵ and is using a log scale. The y-axis is the maximum probability that an adversary with a given confidence level could detect whether or not x was present. If one draws a vertical line at the value of ϵ being enforced (e.g., we mark $\epsilon = 1$ here), the trend to the left shows how privacy is improved for individuals who are visible for less time, and the right shows how it degrades for those who are visible for more.

By Theorem B.2, this mechanism provides ϵ-DP for all (ρ, K)-bounded events. If we rearrange the equation for b so that $\epsilon = \frac{\Delta_{(\rho,K)}(Q)}{b}$, we can equivalently say that Privid guarantees $\frac{\Delta_{(\rho,K)}(Q)}{b}$-DP for all (ρ, K)-bounded events. Or, more generally, that a particular instantiation of Privid with policy $p = (\rho, K, \epsilon)$ guarantees $\hat{\epsilon}$-DP for all $(\hat{\rho}, \hat{K})$-bounded events in query Q, where [10]

$$\hat{\epsilon}_p(\hat{\rho}, \hat{K}, Q) = \frac{\Delta_{(\hat{\rho},\hat{K})}(Q)}{b} = \frac{\Delta_{(\hat{\rho},\hat{K})}(Q)}{\Delta_{(\rho,K)}(Q)/\epsilon} = \frac{\Delta_{(\hat{\rho},\hat{K})}(Q)}{\Delta_{(\rho,K)}(Q)}\epsilon$$

In other words, for a fixed policy, $\hat{\epsilon}$ defines the effective level of protection provided to an event as a function of the *event's* (not policy's) $(\hat{\rho}, \hat{K})$ bound.

From Lemma A.1, we can bound $\hat{\epsilon}$ as $\frac{\alpha\hat{K}(1+\beta\hat{\rho})}{\alpha K(2+\beta\rho)}\epsilon \leq \hat{\epsilon} \leq \frac{\alpha\hat{K}(2+\beta\hat{\rho})}{\alpha K(1+\beta\rho)}\epsilon$. To see where this comes from, note that $\hat{\epsilon}$ is minimized when the numerator is minimized (the lower bound from Lemma A.1) and the denominator is maximized (the upper bound from Lemma A.1). The same logic applies to the upper bound on $\hat{\epsilon}$.

We can simplify both bounds by first canceling α and then picking units of time such that $\beta = 1$ (β has dimensions of chunks per unit time). Thus,

$$\hat{\epsilon} \approx \frac{\hat{\rho}\hat{K}}{\rho K}\epsilon \tag{A.1}$$

□

A.2 Degradation of Privacy

Although $\hat{\epsilon}$ provides a way to quantify the level of privacy provided to each individual, it can be difficult to reason about relative values of ϵ and what they ultimately mean for privacy in practice. We can use the framework of binary hypothesis testing to develop a more intuitive understanding and ultimately visualize the degradation of privacy as a function of $\hat{\epsilon}$ relative to ϵ.

[10] Note the difference in subscript in the numerator and denominator.

Consider an adversary who wishes to determine whether or not some individual x appeared in a given video V. They submit a query Q to the system, and observe only the final result, A, which Privid computed as $A = Q(V) + \eta$, where η is a sample of Laplace noise as defined in the previous section. Based on this value, the adversary must distinguish between one of two hypotheses:

$$\mathcal{H}_0 : x \text{ does not appear in } V$$
$$\mathcal{H}_1 : x \text{ appears in } V$$

We write the false positive P_{FP} and false negative P_{FN} probabilities as:

$$P_{FP} = \mathbb{P}(x \in V | \mathcal{H}_0)$$
$$P_{FN} = \mathbb{P}(x \notin V | \mathcal{H}_1)$$

From Kairouz [46, Theorem 2.1], if an algorithm guarantees ϵ-differential privacy ($\delta = 0$), then these probabilities are related as follows:

$$P_{FP} + e^{\epsilon} P_{FN} \geq 1 \tag{A.2}$$
$$P_{FN} + e^{\epsilon} P_{FP} \geq 1 \tag{A.3}$$

Suppose the adversary is willing to accept a false positive threshold of $P_{FP} \leq \alpha$. In ther words, they will only accept $\mathcal{H}_1$ (x is present) if there is less than α probability that x is not actually present.

Rearranging equations A.2 and A.3 in terms of the probability of correctly detetecting x is present $(1 - P_{FN})$, we have:

$$1 - P_{FN} \leq \quad e^{\epsilon} P_{FP} \leq \quad e^{\epsilon} \alpha$$
$$1 - P_{FN} \leq \quad e^{-\epsilon}(P_{FP} - (1 - e^{\epsilon})) \leq \quad e^{-\epsilon}(\alpha - (1 - e^{\epsilon}))$$

Then, for a given threshold α, the probability that the adversary *correctly* decides x is present is *at most* the minimum of these:

$$\mathbb{P}(x \in V | \mathcal{H}_1) \leq \min\{e^{\epsilon}\alpha, e^{-\epsilon}(\alpha - (1 - e^{\epsilon}))\} \tag{A.4}$$

In Fig. 8, we visualize A.4 as a function of ϵ for 4 different adversarial confidence levels (α=0.1%,1%,10%,20%). As an example of how to read this graph, suppose Privid uses a $(\rho = 60s, K = 1, \epsilon = 1)$ policy ($\epsilon = 1$ marked with the dotted line). An individual who appears 3 times for $< 60s$ each is $(\rho = 60s, K = 3)$-bounded, and thus has an effective $\hat{\epsilon} = 3$ relative to the actual policy for most queries (Eq. A.1). If an adversary has a $\alpha = 1\%$ confidence level, then they would have at most a $\sim$ 20% chance of correctly detecting the individual appeared, even though they appeared for far more than the policy allowed. We can also see that, for sufficiently small values of ϵ (e.g., $\epsilon < 1$), even if the adversary has a very liberal confidence level (say, 20%), a marginal increase in $\hat{\epsilon}$ relative to ϵ only gives the adversary a marginally larger probability of detection than they would have had otherwise.

An important takeaway is that, when an individual exceeds the (ρ, K) bound protected by Privid, their presence is not immediately revealed. Rather, as it exceeds the bound further, $\hat{\epsilon}$ increases, and it becomes more likely an adversary could detect the event.

B Privid Sensitivity Definition

Figure 9 provides the complete definition of sensitivity for a Privid query.

Lemma B.1. Given a relation R, the rules in Figure 9 are an upper bound on the global sensitivity of a (ρ, K)-bounded event in an intermediate table t.

Proof. Proof by induction on the structure of the query.
Case: t. $\Delta_{\mathcal{P}}(t)$ is given directly by Equation 5.2.
Case: $R' := \sigma_{\theta}(R)$. A selection may remove some rows from R, but it does not add any, or modify any existing ones, so in the worst case an individual can be in just as many rows in R' as in R and thus $\Delta_{\mathcal{P}}(R') \leq \Delta_{\mathcal{P}}(R)$ and the constraints remain the same. If θ includes a LIMIT $= x$ condition, then R' will contain at most x rows, regardless of the number of rows in R.
Case: $R' := \Pi_{a,\ldots}(R)$. A projection never changes the number of rows, nor does it allow the data in one row to influence another row, so in the worst case an individual can be in at most the same number of rows in R' as in R ($\Delta_{\mathcal{P}}(R') \leq \Delta_{\mathcal{P}}(R)$) and the size constraint $\tilde{C}_s(R)$ remains the same. If the projection transforms an attribute by applying a stateless function f to it, then we can no longer many assumptions about the range of values in a ($\tilde{C}_r(R', a) = \varnothing$), but nothing else changes because the stateless nature of the function ensures that data in row cannot influence any others.
Case: GroupBy. A `GROUP BY` over a fixed set of a n keys is equivalent to n separate queries that use the same aggregation function over a $\sigma_{\text{WHERE}col=key}(R)$. If the column being grouped is a user-defined column, Privid requires that the analyst provide the keys directly. If the column being grouped is one of the two implicit columns (chunk or region), then the set of keys is not dependent on the contents of the data (only its length) and thus are fixed regardless.
Case: Join. Consider a query that computes the size of the intersection between two cameras, `PROCESS`'d into intermediate tables t_1 and t_2 respectively. If $\Delta(t_1) = x$ and $\Delta(t_2) = y$, it is tempting to assume $\Delta(t_1 \cap t_2) = \min(x, y)$, because a value needs to appear in both t_1 and t_2 to appear in the intersection. However, because the analyst's executable can populate the table arbitrarily, they can "prime" t_1 with values that would only appear in t_2, and vice versa. As a result, a value need only appear in either t_1 or t_2 to show up in the intersection, and thus $\Delta(t_1 \cap t_2) = x + y$ (the sum of the sensitivities of the tables). □

Theorem B.2. Consider an adaptive sequence (§2.3) of n queries $Q_1, \ldots, Q_n$, each over the same camera C, a privacy policy (ρ_C, K_C), and global budget ϵ_C. Privid (Algorithm 1) provides $(\rho_C, K_C, \epsilon_C)$-privacy for all $Q_1, \ldots, Q_n$.

Proof. Consider two queries Q_1 (over time interval I_1, using chunk size c_1 and budget ϵ_1) and Q_2 (over I_2, using c_2 and ϵ_2). Let $v_1 = V[I_1]$ be the segment of video Q_1 analyzes and $v_2 = V[I_2]$ for Q_2. Let E be a (ρ, K)-bounded event.

Notation

$\mathcal{P}$	Privacy policy for each camera: $\{(\rho,K)_c \ \forall \ c \ \in$ cameras$\}$
$\Delta_{\mathcal{P}}(R)$	Maximum number of rows in relation R that could differ by the addition or removal of any (ρ,K)-bounded event.
$\tilde{C_r}(R,a)$	Range constraint: range of attribute a in R
$\tilde{C_s}(R)$	Size constraint: upper bound on total number of rows in R
$\varnothing$	Indicates that a relational operator leaves a constraint unbound. If this constraint is required for the aggregation, it must be bound by a predecessor. If it is not required, it can be left unbound.

Aggregation Functions

Function	**Definition**	**Constraints**	**Sensitivity ($\mathbf{\Delta(Q)}$)**
Count	$Q := \Pi_{\text{count}(*)}(R)$	Δ	$1 \cdot \Delta(R)$
Sum	$Q := \Pi_{\text{sum}(a)}(R)$	$\Delta, \tilde{C_r}$	$\Delta(R) \cdot \tilde{C_r}(R,a)$
Average	$Q := \Pi_{\text{avg}(a)}(R)$	$\Delta, \tilde{C_r}, \tilde{C_s}$	$\frac{\Delta(R) \cdot \tilde{C_r}(R,a)}{\tilde{C_s}(R)}$
Std. Dev	$Q := \Pi_{\text{stddev}(a)}(R)$	$\Delta, \tilde{C_r}, \tilde{C_s}$	$\Delta(R) \cdot \tilde{C_r}(R,a) / \sqrt{\tilde{C_s}(R)}$
Argmax	$Q := \Pi_{\text{argmax}(a)}(R)$	$\Delta, a \in K$	$\max_{k \in K} \Delta(\sigma_{a=k}(R))$

Relational Operators

Operator	**Type**	**Definition**	$\mathbf{\Delta_{\mathcal{P}}(R')}$	$\mathbf{\tilde{C_r}(R',a_i)}$	$\mathbf{\tilde{C_s}(R')}$
Base Case	Base Table	R	$mr \cdot K \cdot (1 + \lceil \frac{\rho}{c} \rceil)$	$\varnothing$	$\varnothing$
Selection (σ)	Standard selection: rows from R that match WHERE condition	$R' := \sigma_{\text{WHERE}(\dots)}(R)$	$\Delta_{\mathcal{P}}(R)$	$\tilde{C_r}(R,a_i)$	$\tilde{C_s}(R)$
	Limit: first x rows from R	$R' := \sigma_{\text{LIMIT}=x}(R)$	$\Delta_{\mathcal{P}}(R)$	$\tilde{C_r}(R,a_i)$	$\min(x, \tilde{C_s}(R))$
Projection (Π)	Standard projection: select attributes $a_i, \dots$ from R	$R' := \Pi_{a_i,\dots}$	$\Delta_{\mathcal{P}}(R)$	$\tilde{C_r}(R,a_i)$	$\tilde{C_s}(R)$
	Apply (user-provided, but stateless) f to column a_i	$R' := \Pi_{f(a_i),\dots}$	$\Delta_{\mathcal{P}}(R)$	$\varnothing$	$\tilde{C_s}(R)$
	Add range constraint to column a_i	$R' := \Pi_{a_i \in [l_i,u_i],\dots}$	$\Delta_{\mathcal{P}}(R)$	$[l_i,u_i]$ if $a_i \neq \varnothing$ $\tilde{C_r}(R,a_i)$ otherwise	$\tilde{C_s}(R)$
GroupBy (γ)	Group attribute(s) (g_i) are `chunk` (or binned `chunk`) or `region`	$R' := {}_{g_j,\dots}\gamma_{\text{agg}(a_i),\dots}$ $g_j :=$ chunk$\mid$bin(chunk)	Equation 5.2	$\Delta(\text{agg}(a_i))$	$\frac{\tilde{C_s}(R)}{\text{(bin size)}}$
	Group attribute(s) (g_j) are *not* `chunk` or `region`	$R' := {}_{g_j,\dots}\gamma_{\text{agg}(a_i),\dots}$	$\Delta_{\mathcal{P}}(R)$	$\varnothing$	$\varnothing$
	... discrete set of keys provided for each group (constrains size)	$R' := {}_{g_j \in K_j,\dots}\gamma_{\text{agg}(a_i),\dots}$	...	...	$\Pi_j \lvert K_j \rvert$
	... aggregation constrains range: $agg(a_i) \in [l_i,u_i]$	$R' := {}_{g_j,\dots}\gamma_{\text{agg}(a_i) \in [l_i,u_i],\dots}$	...	$[l_i,u_i]$ if $a_i \neq \varnothing$ $\tilde{C_r}(R,a_i)$ otherwise	...
Joins* ($\bowtie$)	*When *immediately* preceeded by GroupBy *over the same key(s)* ... equijoin on g_j (intersection on g_j) ... outer join on g_j (union on g_j)	$R' := {}_g\gamma_{\text{agg}(a)}(R_1 \bowtie_g \dots \bowtie_g R_n)$ $R' := {}_g\gamma_{\text{agg}(a)}(R_1 ⟗_g \dots ⟗_g R_n)$	$\sum_{i=1}^{n} \Delta_{\mathcal{P}}(R_i)$	(GroupBy rules)	(GroupBy rules)

Figure 9: Full set of rules for Privid's sensitivity calculation.

Case 1: I_1 and I_2 are not ρ-disjoint The budget check (lines 1-3 in Algorithm 1) ensures that these two queries must draw from the same privacy budget, because their effective ranges overlap by at least one frame (but may overlap up to all frames). By Theorem 5.1, Privid is (ρ,K,ϵ_1)-private for Q_1 and (ρ,K,ϵ_2)-private for Q_2. By Dwork [37, Theorem 3.14], the combination of Q_1 and Q_2 is $(\rho,K,\epsilon_1+\epsilon_2)$-private.

Case 2: I_1 and I_2 are ρ-disjoint In other words, $I_1+\rho < I_2-\rho$, thus the budget check (lines 1-3) allows these two queries to draw from entirely separate privacy budgets. Since the intervals are ρ-disjoint, and all segments in E must have duration $\leq \rho$, it is not possible for the same segment to appear in even a single frame of *both* intervals.

Let K_1 be the number of segments contained in I_1, each of duration $\leq \rho$, and K_2 be the remaining segments contained in I_2, each of duration $\leq \rho$. In other words, E is (ρ,K_1)-bounded in v_1 and (ρ,K_2)-bounded in v_2. Since E has at most K segments, $K_1+K_2 \leq K$. We need to show that the probability of observing both A_1 and A_2 if the inputs are the actual segments v_1 and v_2 is close (e^ϵ) to the probability of observing those values if the inputs are the neighboring segments v_1' and v_2':

$$\frac{\Pr[A_1 = Q_1(v_1), A_2 = Q_2(v_2)]}{\Pr[A_1 = Q_1(v_1'), A_2 = Q_2(v_2')]} \leq \exp(e)$$

Since the probability of observing A_1 is independent of observing A_2 (randomness is purely over the noise added by Privid):

$$\frac{\Pr[A_1 = Q_1(v_1), A_2 = Q_2(v_2)]}{\Pr[A_1 = Q_1(v_1'), A_2 = Q_2(v_2')]}$$

$$\leq \frac{\Pr[A_1 = Q_1(v_1)]\Pr[A_2 = Q_2(v_2)]}{\Pr[A_1 = Q_1(v_1')]\Pr[A_2 = Q_2(v_2')]}$$

$$\leq \frac{\frac{1}{2b_1}\exp(-\frac{|A_1 - Q_1(v_1)|}{b_1})\frac{1}{2b_2}\exp(-\frac{|A_2 - Q_2(v_2)|}{b_2})}{\frac{1}{2b_1}\exp(-\frac{|A_1 - Q_1(v_1')|}{b_1})\frac{1}{2b_2}\exp(-\frac{|A_2 - Q_2(v_2')|}{b_2})}$$

(By Algorithm 1, Line 13)

$$= \exp(\frac{|A_1 - Q_1(v_1')| - |A_1 - Q_1(v_1)|}{b_1} + \frac{|A_2 - Q_2(v_2')| - |A_2 - Q_2(v_2)|}{b_2})$$

If K_1 segments are in v_1 and K_2 segments are in v_2, the numerator of each fraction above is the sensitivity of a (ρ, K_1)-bounded event and a (ρ, K_2)-bounded event, respectively. b_1 and b_2 are the amount of noise actually added to the query, which are both based on K:

$$\leq \exp(\frac{\Delta_{(\rho,K_1)}(Q_1)}{\Delta_{(\rho,K)}(Q_1)/\epsilon} + \frac{\Delta_{(\rho,K_2)}(Q_2)}{\Delta_{(\rho,K)}(Q_2)/\epsilon})$$

$$= \exp(\epsilon \cdot (\frac{K_1(\lceil \frac{\rho}{c_1} \rceil + 1)}{K(\lceil \frac{\rho}{c_1} \rceil + 1)} + \frac{K_2(\lceil \frac{\rho}{c_2} \rceil + 1)}{K(\lceil \frac{\rho}{c_2} \rceil + 1)}))$$

(by Equation 5.2)

$$= \exp(\epsilon \cdot (\frac{K_1}{K} + \frac{K_2}{K})) \quad \text{(recall } K \geq K_1 + K_2\text{)}$$

$$\leq \exp(\epsilon)$$

□

C Query Details

C.1 Case 1 Query Statements

Case 1: Query 1

```
SPLIT campus
   BEGIN 06-01-2019/06:00am END 06-01-2019/06:00pm
   BY TIME 30sec STRIDE 0sec
   BY REGION
   WITH MASK C1
   INTO campusChunks;
PROCESS campusChunks USING count_ppl_campus.py TIMEOUT 1sec
   PRODUCING 1 ROWS
   WITH SCHEMA (ppl:NUMBER=0)
   INTO campusTable;
SELECT hour,sum(RANGE(ppl,0,6)) from campusTable
   GROUP BY hour
   CONSUMING eps=1.0;
```

Case 1: Query 2

```
SPLIT highway
   BEGIN 06-01-2019/06:00am END 06-01-2019/06:00pm
   BY TIME 30sec STRIDE 0sec
   BY REGION
   WITH MASK H2
   INTO highwayChunks;
PROCESS highwayChunks USING count_cars.py TIMEOUT 1sec
   PRODUCING 1 ROWS
   WITH SCHEMA (cars:NUMBER=0)
   INTO highwayTable;
SELECT hour, sum(RANGE(cars,0,100)) from highwayTable
   GROUP BY hour
   CONSUMING eps=1.0;
```

Case 1: Query 3

```
SPLIT urban
   BEGIN 06-01-2019/06:00am END 06-01-2019/06:00pm
   BY TIME 30sec STRIDE 0sec
   BY REGION
   WITH MASK U2
   INTO urbanChunks;
PROCESS campusChunks USING count_ppl_urban.py TIMEOUT 1sec
   PRODUCING 1 ROWS
   WITH SCHEMA (ppl:NUMBER=0)
   INTO campusTable;
SELECT hour, sum(RANGE(ppl,0,23)) from campusTable
   GROUP BY hour
   CONSUMING eps=1.0;
```

C.2 Case 2: Complex Sensitivity Example

The code block for Case 2 contains Queries 4-6, which are computed over the same set of intermediate tables.

To demonstrate the sensitivity computation for a complex Privid query, we focus on Query 4. This query aims to estimate the typical working hours of taxis in the city of Porto, Portugal; it first computes the difference between the first and last time each taxi (identified by plate) was seen (by either camera 10 or 27) on a given day, then averages across all taxis and days (over a year).

In order to ensure all variables needed for the aggregation are properly constrained, we make two assumptions: most taxis will not work more than 16 hours in a day, and there are roughly 300 public taxis in Porto (based on public information). We can express this query in relational algebra as follows:

$$\Pi_{\text{Avg(hrs)}}(\sigma_{\text{limit(plates)=300}}({}_{\text{plate,day}}\gamma_{\text{range(chunks)}\in[0,16]}(t_1 \cup t_2)))$$

Case 2: Queries 4-6

```
-- Repeat for portoCam1...portoCam127:
SPLIT portoCam1
   BEGIN 07-01-2013/12:00am END 07-01-2014/12:00am
   BY TIME 15sec STRIDE 0sec
   INTO chunks1;
-- Repeat for chunks1...chunks127:
PROCESS chunks1 USING porto.py TIMEOUT 1sec
   PRODUCING 3 ROWS
   WITH SCHEMA (plate:STRING="")
   INTO table1;

-- Query 4: Average Taxi Working Hours
SELECT avg(avg_shift) FROM
   (SELECT plate,avg(RANGE(shift, [0,16])) FROM
      (SELECT plate,day,(max(chunk)-min(chunk) as shift) FROM
         table10 UNION table27 GROUP BY plate,day(chunk))
   GROUP BY plate LIMIT 300)
CONSUMING eps=0.33;
-- Query 5: # Taxis Traversing Both Locations On Same Day
SELECT day,count(DISTINCT plate) FROM
   (SELECT day,plate FROM
      table10 INNER JOIN table27 ON
      (table10.plate=table27.plate AND table10.day=table27.day)
   )
   GROUP BY day
CONSUMING eps=0.33;
-- Query 6: Camera with highest daily traffic
SELECT argmax(arg=cam, target=avg_daily) FROM
  (SELECT "cam1" as cam, avg(daily) as avg_daily FROM
   (SELECT day,count(DISTINCT plate) as daily FROM
    table1 GROUP BY day))
  UNION
  // ...
  UNION
  (SELECT "cam127" as cam, avg(daily) as avg_daily FROM ...)
CONSUMING eps=0.33;
```

We use the policy $\mathcal{P} = \{(\rho = 45s, K = 1)_{c_1}, (195s, 1)_{c_2}\}$ (the max observed persistence over historical data for each camera) and an ϵ of 1.

First, we compute the base sensitivity of each table. The `SPLIT` statement specifies the video will be split into 15 second chunks with 0 stride, and that each chunk will produce a maximum of 3 rows. With this we can compute: $\Delta_{\mathcal{P}}(t_1) = \lceil\frac{(45*\texttt{fps}-1}{15*\texttt{fps}}\rceil + 1 = 4 \cdot 3 = 12$ and $\Delta_{\mathcal{P}}(t_2) = \lceil\frac{195*\texttt{fps}-1}{15*\texttt{fps}}\rceil + 1 = 14 \cdot 3 = 42$. When we combine them with a union, their sensitivities add: $\Delta_{\mathcal{P}}(t_1 \cup t_2) = 12 + 42 = 54$. The `GROUP BY` creates a new table with a row per plate per day, and constrains the range of the aggregate value `shift` to $[0,16]$ (`range`(a,b) returns $|b-a|$, i.e., the time between the first and last appearance of a taxi on a given day), but we don't know how many unique plates there might be, so the size $\tilde{C}_s(\gamma(...))$ is unconstrained. We add σ_{limit} to manually enforce a maximum of 300 plates per day, which gives us a constraint $\tilde{C}_s(\sigma(...)) = 300\text{plates} * 365\text{days} = 109,500$. We now have all the constraints necessary to compute the sensitivity of the average aggregation: $\Delta_{\mathcal{P}}^{\text{AVG}}(R) = \frac{\Delta_{\mathcal{P}}(R)\tilde{C}_r(R,\text{shift},)}{\tilde{C}_s(R)} = \frac{54 \cdot 16}{109{,}500} = 0.0079$. Since Privid uses the Laplace mechanism to add noise, we can use the inverse CDF of the Laplace distribution to bound the expected error based on Δ with a given confidence level. For example, $\mathcal{L}^{-}1(p = 0.999, u = 0, b = \frac{\Delta}{\epsilon} = \frac{0.0079}{0.33}) \leq 0.15$ hours with 99.9% confidence.