# Surveillance Video Querying With A Human-in-the-Loop

Michael Stonebraker[1], Bharat Bhargava[2], Michael Cafarella[3], Zachary Collins[1], Jenna McClellan[1], Aaron Sipser[1], Tao Sun[1], Alina Nesen[2], KMA Solaiman[2], Ganapathy Mani[2], Kevin Kochpatcharin[2], Pelin Angin[2], James MacDonald[4]
stonebraker@csail.mit.edu,bbshail@purdue.edu,michjc@umich.edu,{zcollins,jenmc,asipser,taosun}@mit.edu
{anesen,ksolaima,manig,kkochpat,pangin}@purdue.edu,Jim.MacDonald@ngc.com
[1]MIT, [2]Purdue University, [3]University of Michigan, [4] Northrop Grumman Corporation

## ABSTRACT

SurvQ is a video monitoring system appropriate for surveillance applications such as those found in security and law enforcement. It performs real time object property identification and stores all data in a scalable DBMS. Standing queries implemented as database triggers are supported. SurvQ contains novel adaptive machine learning and algorithmic property classification. The application of SurvQ to assist the West Lafayette (IN) police department at identifying suspects in video is described. This paper also describes the basics of the SurvQ architecture and its human-in-the-loop interface designed to accelerate everyday police investigations.

**ACM Reference Format:**
Michael Stonebraker[1], Bharat Bhargava[2], Michael Cafarella[3], Zachary Collins[1], Jenna McClellan[1], Aaron Sipser[1], Tao Sun[1], Alina Nesen[2], KMA Solaiman[2], Ganapathy Mani[2], Kevin Kochpatcharin[2], Pelin Angin[2], James MacDonald[4]. 2020. Surveillance Video Querying With A Human-in-the-Loop. In *Workshop on Human-In-the-Loop Data Analytics (HILDA'20), June 14–19, 2020, Portland, OR, USA*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3398730.3399192

## 1 INTRODUCTION

Urban areas have many video cameras continuously recording some field of view. There are fixed cameras on light poles, cameras on city vehicles, and cameras on police personnel. In addition, there are surveillance cameras on private property, NEST doorbells, interior spaces, and many private vehicles. The resulting video streams are crucial for police officers when solving a range of everyday crimes, but often entail hours of tedious manual examination, thereby wasting scarce police resources that could be put to better use.

MIT and Purdue researchers have developed a Surveillance Video Querying system, SurvQ, for surveillance video information. The Chesterfield, NH police department is assisting in providing mission requirements. The West Lafayette,IN police department has provided video data from cameras in the downtown area plus redacted police dispatch reports. The SurvQ prototype must operate at sufficient scale to handle the West Lafayette use case (about 200 fixed and mobile cameras), and with an analysis loop that gives rapid answers to busy law enforcement officers. SurvQ has been designed so it is readily adaptable to larger deployments and other surveillance applications, such as those found in securing military bases and disaster recovery.

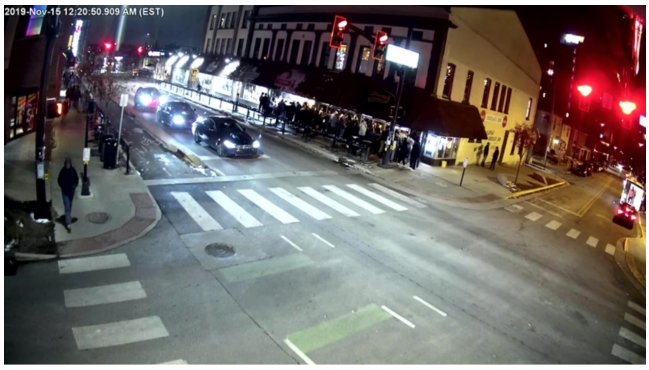



**Figure 1: An example of surveillance footage from West Lafayette, IN.**

**The Detective In the Loop** – The primary focus of SurvQ is to assist humans performing analytical tasks: that is, police detectives solving crimes and tracking individual suspects. A typical scenario is an incident report of the form "assault reported at time *XXX* in location *YYY*. Suspect is of medium height, wearing jeans and a baseball cap". In this particular use case from West Lafayette, a suspect matching the description was spotted on a public bus camera, and that led to his arrest. At the present time, examining video footage is a manual process for police, and takes hours and hours of time. The request from both police departments is simple: "please help us be more efficient at searching video to track suspects". The general use case is to find video frames matching a given description in a given geographic area and time range. Detectives want both an off-line system to search historical data and a real-time (standing query) facility with short response time (under 60 seconds). An example video frame from a West Lafayette street is shown in Figure 1. In addition to accepting video queries via direct human input, one way to make detectives more effective would be to draw query specifications from multiple fused sources, such as tweets or hand-written police reports.

**Surveillance Video and Detective Use Cases** – Surveillance video has a collection of notable characteristics. First, it is often fairly low resolution and therefore difficult to process with sophisticated techniques. Second, the lighting is usually poor, because


*HILDA'20, June 14–19, 2020, Portland, OR, USA*

ACM ISBN 978-1-4503-8022-5/20/06…$15.00
https://doi.org/10.1145/3398730.3399192

| White | Black | Hispanic | Asian |
|---|---|---|---|
| Male | Female | Tattoos | Beard |
| Bald | Hair color | Sandals | Shoes |
| Boots | Jeans | Pants | Shorts |
| T-shirt | Baseball hat | Jacket | Tall |
| Shorts | Walking | Running | Motorcycle |
| Bicycle | Truck | Passenger car | Skateboard |
| Smoking | Backpack | Headphones | |

**Table 1: Video properties of interest to law enforcement.**

of glare, night conditions, fog, rain, or snow. Third, interesting objects are often in the background, often facing away from the camera, and are usually closer to thumbnails than images. Fourth, interesting objects often have properties that are rarely observed, so obtaining training data may be a challenge for building machine learning (ML) systems.

The West Lafayette Police department shared the 31 object properties they are most interested in, and these are shown in Table 1. Several things should be noted about this property set.

Some properties are relatively rare. For example, smoking is rare on college campuses and sandals are very rare in December in Indiana. Trying to find rare events by using traditional machine learning techniques that depend on obtaining large amounts of training data is not likely to succeed.

Some properties are visually very small. Tattoos in surveillance video are a few pixels. Of course, this makes recognition difficult.

The set of interesting properties for the police application is surprisingly modest. To meet their needs, it is easy for computer scientists to hypothesize all the possible data values that might be extracted from an image. Also, police queries — as we saw in interviews with police officers and by examining police reports — are driven by a relatively small number of query types. We believe this fact can help both in designing an efficient system and in addressing citizens' privacy concerns.

**Design Considerations** – Training data is challenging. In an early experiment we demonstrated that building a classifier to find people wearing jeans by training on high quality web images failed to recognize jeans in surveillance video. Hence, transfer learning may not work well. In addition, many of the properties in Table 1 are subjective, and humans may differ on whether they are present. In other words, the input is very noisy.

We face problems of scale. West Lafayette has more than 100 cameras. The administration wishes to retain all video for months. This quickly becomes a terascale to petascale problem.

Traditional deep learning is expensive at scale. We anticipate there will be additional properties of interest to the police beyond those in Table 1. As we have demonstrated, our transfer learning attempt failed to find jeans. Building a training set for the properties of Table 1 is a tedious manual process. In addition, model runtime at scale is costly. It is not clear that a deep learning solution is affordable by the City of West Lafayette.

**Overall Approach** – SurvQ applies property recognizers to the video streams and loads both video and the properties into a Postgres database [3]. To achieve scalability, it can be optionally be loaded into Citus[1], a parallel multi-node terascale extension of Postgres. The detective spends his or her time querying and navigating this database. In addition, Purdue researchers are working on parsing text from tweets and police reports. Data from Bureau of Motor Vehicles records and exchange of messages among police officers could be included in the future.

**Nontechnical Deployment Concerns** – This paper is focused on technical questions, but there are substantial nontechnical issues around deploying such a system. Some countries have deployed digital surveillance systems that are totalitarian. Some communities face overpolicing. Although there are some technical approaches that could possibly limit abuse, such as recent work in fairness in machine learning models, the challenges are quite broad and unlikely to be solved solely by technical measures. It is not possible to address them adequately in a short paper, nor solely from a computing perspective. Obtaining the efficiency benefits of systems like this one, while limiting the potential for abuse, is a broad challenge for both the field and society overall.

**Organization of this paper** – We cover related work in Section 2, then discuss basic SurvQ architecture in Section 3. We describe the user's workflow in Section 4 and provide some initial experimental results in Section 5.

## 2 RELATED WORK

Querying over video is a substantial research problem that draws on work in several areas of computer science.

Although queryable video monitoring systems have existed for some time, the neural network revolution in image processing has changed many of the system opportunities as well as research challenges. Earlier systems were limited to recognizing simpler objects, such as license plates or faces [5, 14]. One line of work assumes that video frames will be processed by a convolutional neural network (CNN) and is primarily concerned with optimizing their execution. NoScope [8] offers several optimizations, including training of inexpensive proxy models and selective frame differencing. The Tahoma [4] system creates multiple physical representations of videos, combined with creation of proxy models, in order to choose the most efficient one at query time. BlazeIt [7] offers optimizations for aggregation and limit queries that again rely on proxy model training. Focus [6] achieves runtime gains with a combination of inexpensive but low-quality CNNs to build an approximate query index, plus expensive high-quality CNNs after using the prebuilt index.

Many of these systems assume the existence of CNN-training capacity that would not be reasonable in our police use case. Their applications also focus heavily on traffic video use cases (whether fixed-camera or car-mounted) where they must detect (1) many examples of (2) common and (3) visually clear phenomena for downstream use by (4) analytical pipelines. In contrast, we are concerned with a human who often needs (1) a single example of (2) rare and (3) visually obscure phenomena for downstream use by (4) a human-intensive investigation. As a result, SurvQ does not have to generate huge quantities of results; it can exploit the natural duplication of imagery common in video in order to find a small number of difficult but high-value query results.

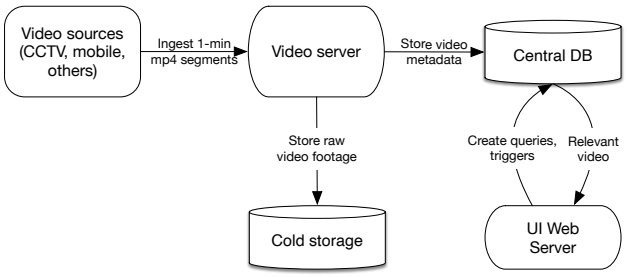



**Figure 2: SurvQ video ingestion and retrieval architecture**

Other video query systems focus on traditional systems-centric optimization methods. SVQ (Streaming Video Queries) [15] is a system for running declarative SQL-style video stream queries, with a focus on counts and spatial constraints on objects in a frame. It optimizes query execution by applying a set of inexpensive filters (such as object counts) on video frames before running expensive object detection algorithms, thereby eliminating frames that have low potential of being a match for the query. Optasia [12] is a large-scale relational video query processing system that focuses on surveillance camera data. Its optimizations mainly focus on deduplicating work and choosing chunks for parallelization. Some of its approaches might be useful for SurvQ. VideoStorm [16] is an analytics system that uses a compute cluster to process thousands of concurrent analytical queries. It is primarily concerned with scheduling and resource allocation questions around those queries.

Some researchers have focused on detecting a larger set of items in video imagery. The Panorama system [17] represents the problem of unknown objects as an unbounded vocabulary problem. This system uses an ML-heavy approach that asks users to manually label unknown objects before automatically retraining novel image classifiers. This is an interesting human annotation problem, but is both computationally heavyweight and likely unnecessary in a concrete application with a relatively fixed set of objects. Techniques such as zero-shot [10] and one-shot [9] learning have been proposed for supporting new object properties, but they require significant manual intervention for model retraining as well as the provision of metadata and/or more labels, *i.e.*, identifying more object properties. As video surveillance applications usually have time-starved and non-technical users, these techniques are difficult to implement.

All of the above systems focus on the data system, without extensive attention paid to the human in the visual analytical loop.

## 3 ARCHITECTURE

The architecture of SurvQ is divided into *ingestion* and *retrieval* systems. These two systems may be operating in parallel.

### 3.1 Data Ingestion

Figure 2 shows the initial data ingress step. SurvQ consumes video feeds in real time or retrospectively. When video data arrives at the Video Server, SurvQ archives it in storage and then applies a a pipeline of processing steps:

(1) Video is converted to MP-4 (if it is not already captured in this format) and down-sampled to one frame per second (there is no sense running property identification more often than this, and we may be able to run less frequently).
(2) YOLO [13] is used to identify *people* objects in each frame. YOLO was chosen because it is very efficient at run time and has the best chance of "keeping up" with the large number of video streams. YOLO also has built-in detection for some of the properties from Table 1, for example bicycles.
(3) Each YOLO-detected object is then further examined to discern its *object properties*, initially the features in Table 1.

Next, property identification is performed using three different approaches:

**Color analysis.** The combination of YOLO class and some color analysis can yield a simple but effective property detector. We have segmented all YOLO-detected person objects into "sections" (e.g. lower half, upper 10% etc.). If the dominant color of the lower half is "blue" then chances are the person is wearing jeans. Between shape analysis, color analysis and common sense reasoning, we can detect about half of the objects in Table 1.

**Traditional deep learning.** Purdue is applying traditional deep learning to the object property identification problem. This requires tagging imagery from West Lafayette with labels to construct training data. We have found that a suitable detector requires O(1000) images to be successful. Student labor is being used for this substantial task.

**Transfer learning.** Purdue trained a CNN-based attribute detection classifier on the PA-100K dataset [11]. PA-100K dataset contains 100,000 pedestrian images from real outdoor surveillance cameras annotated with 26 attributes. We use YOLO as our backend before the frames are passed to the pretrained classifier. We created a mapping between 26 features from PA100K dataset and 31 features of interest in Table 1 based on whether they are visually synonymous: for example, *short sleeves* in the PA-100K dataset is mapped to *T-shirt* in our taxonomy. All of the features from PA100K are similar to some properties of interest to us, and we could find an exact mapping for more than one-third of the properties.

Currently, all three classifiers are being actively improved. However, a few results are already apparent. First, color/shape analysis works well and has the great advantage that training data is not required. Second, transfer learning is proving difficult, because of the variance in quality of frames between training data set and surveillance video. This same issue was a problem in transfer learning to identify jeans. Our experience is that data derived from dissimilar video is not useful in helping to solve our problem. We are hopeful, however, that our classifiers trained on West Lafayette video will work successfully on New Hampshire video. Lastly, training a novel deep learning system is a challenge because of the amount of training data required. Without a ready data set of tagged images, generating training data is a very expensive proposition. Without student labor, the cost could well be prohibitive.

**Active Learning and Color Management** – Active learning is well understood in a deep learning context. However, of particular interest to us is feedback to our color algorithms for automatic improvement. Colors perceived by the camera can be influenced

by the time of day, weather, and other physical properties in the scene so that simply detecting "blue" on a person figure requires some adjustment. To perform color analysis in a segmented area, the RGB value at each pixel is retrieved. The set of RGB values for the standard colors (red, green, yellow, *etc.*) is defined in a reference color map. To infer the color of a pixel, we calculate the color distance to each standard color and choose the closest one [2].

The color for any region of interest was then found by majority vote. To incorporate active learning, we plan to test moving the reference colors in color space based on user feedback.

Finally, the output of the classifiers is stored in Postgres along with suitable metadata, and pointers to the archived raw video.

## 3.2 Data Retrieval

The right-hand side of Figure 2 shows the simple architecture we use for querying video. At run time, our system expects a user query, most likely derived from information in a police incident report, such as the West Lafayette document seen in Figure 3. (We also have code that parses the actual incident reports to extract the description of the incident.) The user's query is converted to SQL and defined as a trigger to the Postgres DBMS. In this way, a historical query is run to find the data of interest in the past, and a Postgres trigger will find data of interest as it is loaded into the DBMS in the future. Section 4 describes in detail the query and interaction cycle from the user's perspective.

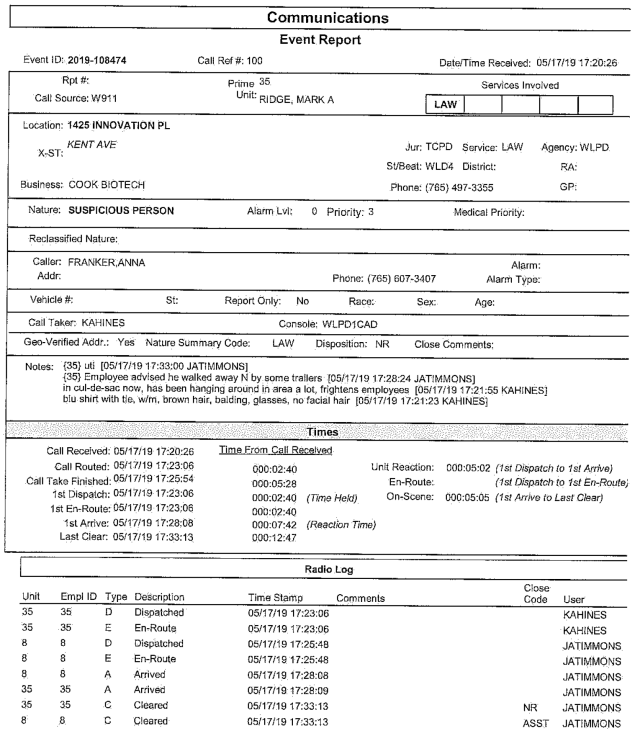
We currently ingest parsed tweets into the database. In the future, we expect to search both video and tweets for the properties of interest in Table 1. Note that the scope of a trigger system can be multiple tables, so joining multiple data sources is straightforward.

# 4 USER WORKFLOW: THE POLICE DETECTIVE IN THE LOOP

Police detectives receive information about events they need to investigate in the form of an incident report, like that of Figure 3. These reports contain information about the event that occurred, often including details of any suspects involved. We have two interfaces to obtain data about an incident. The first is a form-based UI shown in Figure 4. With it, detectives can input the incident details easily. The form has fields to filter on time, location and suspect characteristics. The results are translated into SQL queries and triggers, allowing non-programmers to easily interact with the system. The second system automatically parses West Lafayette incident reports to obtain required information.

An example of the analytical interaction steps a detective may take with SurvQ is shown in Figure 5. In (1) the user will visit the creation page for the incident and enter the appropriate details. Upon submission, the user is redirected to an investigation page that can be revisited at any time. The investigation page contains all the details relevant for the incident. This includes event information, processing progression and matching video clips. In (2) there are three possible *viewpoints* the user can choose:

> **List:** Displays all returned results compactly so the user can quickly view all matches.
> **Map:** Aggregates video that occurred in close proximity and displays the resulting clusters on a map



**Communications**

**Event Report**

Event ID: **2019-108474** Call Ref #: 100 Date/Time Received: 05/17/19 17:20:26

Rpt #: Prime 35 Services Involved
Call Source: W911 Unit: RIDGE, MARK A LAW

Location: **1425 INNOVATION PL**
X-ST: *KENT AVE* Jur: TCPD Service: LAW Agency: WLPD
St/Beat: WLD4 District: RA:
Business: COOK BIOTECH Phone: (765) 497-3355 GP:

Nature: **SUSPICIOUS PERSON** Alarm Lvl: 0 Priority: 3 Medical Priority:

Reclassified Nature:

Caller: FRANKER,ANNA Alarm:
Addr: Phone: (765) 607-3407 Alarm Type:

Vehicle #: St: Report Only: No Race: Sex: Age:

Call Taker: KAHINES Console: WLPD1CAD

Geo-Verified Addr.: Yes Nature Summary Code: LAW Disposition: NR Close Comments:

Notes: {35} ati [05/17/19 17:33:00 JATIMMONS]
{35} Employee advised he walked away N by some trailers [05/17/19 17:28:24 JATIMMONS]
in cul-de-sac now, has been hanging around in area a lot, frightens employees [05/17/19 17:21:55 KAHINES]
blu shirt with tie, w/m, brown hair, balding, glasses, no facial hair [05/17/19 17:21:23 KAHINES]

**Times**

| | | | |
|---|---|---|---|
| Call Received: 05/17/19 17:20:26 | Time From Call Received | | |
| Call Routed: 05/17/19 17:23:06 | 000:02:40 | Unit Reaction: | 000:05:02 *(1st Dispatch to 1st Arrive)* |
| Call Take Finished: 05/17/19 17:25:54 | 000:05:28 | En-Route: | *(1st Dispatch to 1st En-Route)* |
| 1st Dispatch: 05/17/19 17:23:06 | 000:02:40 *(Time Held)* | On-Scene: | 000:05:05 *(1st Arrive to Last Clear)* |
| 1st En-Route: 05/17/19 17:23:06 | 000:02:40 | | |
| 1st Arrive: 05/17/19 17:28:08 | 000:07:42 *(Reaction Time)* | | |
| Last Clear: 05/17/19 17:33:13 | 000:12:47 | | |

**Radio Log**

| Unit | Empl ID | Type | Description | Time Stamp | Comments | Close Code | User |
|---|---|---|---|---|---|---|---|
| 35 | 35 | D | Dispatched | 05/17/19 17:23:06 | | | KAHINES |
| 35 | 35 | E | En-Route | 05/17/19 17:23:06 | | | KAHINES |
| 8 | 8 | D | Dispatched | 05/17/19 17:25:48 | | | JATIMMONS |
| 8 | 8 | E | En-Route | 05/17/19 17:25:48 | | | JATIMMONS |
| 8 | 8 | A | Arrived | 05/17/19 17:28:08 | | | JATIMMONS |
| 35 | 35 | A | Arrived | 05/17/19 17:28:09 | | | JATIMMONS |
| 35 | 35 | C | Cleared | 05/17/19 17:33:13 | | NR | JATIMMONS |
| 8 | 8 | C | Cleared | 05/17/19 17:33:13 | | ASST | JATIMMONS |

**Figure 3: The populated incident report**

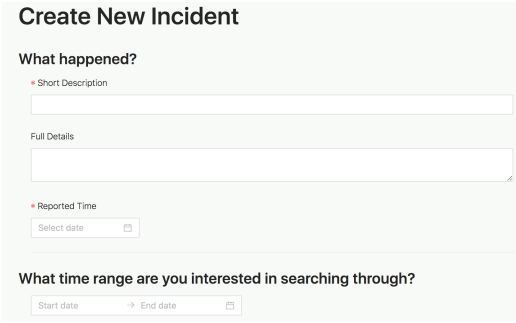



**Figure 4: The incident creation form.**

> **Timeline:** Aggregates video that occurred close in time and displays the bucketed groups in order

These three different viewpoints are shown in Figure 6. In step (3), the investigator will search through the returned video. When an investigator finds a video useful, they mark it as important. Each viewpoint can apply a filter to display only marked video. In step (4), after they've gone through the video, they can select a different viewpoint to make additional passes over the video data. This search-and-mark process comprises much of the detective's analytical work. In the future, we believe that moving cameras (mounted, say, on a police car) may be a potential source of novel video data, and will pose new interaction challenges for investigators attempting to find suspects in the video database.

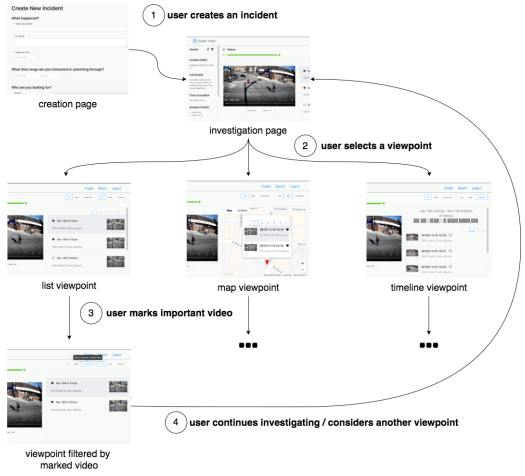



**Figure 5: Example actions a user might take with SurvQ.**

Additional video may enter the system after initial creation and investigation. Investigators following an incident in real time can use SurvQ to subscribe to long-running queries and thereby receive alerts about new data. Postgres triggers are made on incident creation to check for property matches. Users can see new notifications in the investigation view and home page for easy viewing and access.

## 5 RUNTIME PERFORMANCE: PRELIMINARY RESULTS

We have run our ingest system and our query system in parallel on real and simulated queries and data. To assess performance, we consider SurvQ as three components: video ingest, YOLO processing, and database activity. Our goal was to make SurvQ performant enough to handle an urban camera deployment, including the West Lafayette use case.

We maintain a collection of web servers to handle video upload. Upsteam processing must generate video in 1-minute MP-4 files. Future work would be required to handle other video formats. A low cost web server can support 4-5 concurrent video feeds.

YOLO processing runs well on a GPU-equipped server. A single server can perform object recognition and color analysis in approximately 15 seconds per 1 minute of video. Thus, each YOLO processing server can handle around 4 incoming feeds.

Our major performance concern was how our use of trigger functions in Postgres would scale. In SurvQ, a trigger function for each incident is run every time YOLO results are inserted. We run YOLO on video at a rate of 1 frame per second, or 60 frames per minute. West Lafayette surveillance data averages 5 persons per frame. Since the West Lafayette use case has at most 200 video sources, we would expect our system to receive around 60 * 5 * 200 = 60000 inserts per minute. We need to be able to handle these insertions in under 60 seconds worth of time. Figure 7 shows that trigger invocation can easily keep up.

As such, the dominant cost for West Lafayette is the number of YOLO servers to process the incoming video load. Given their computing budget, it is not cost effective to perform ingest-time property identification on all video. Instead we have implemented

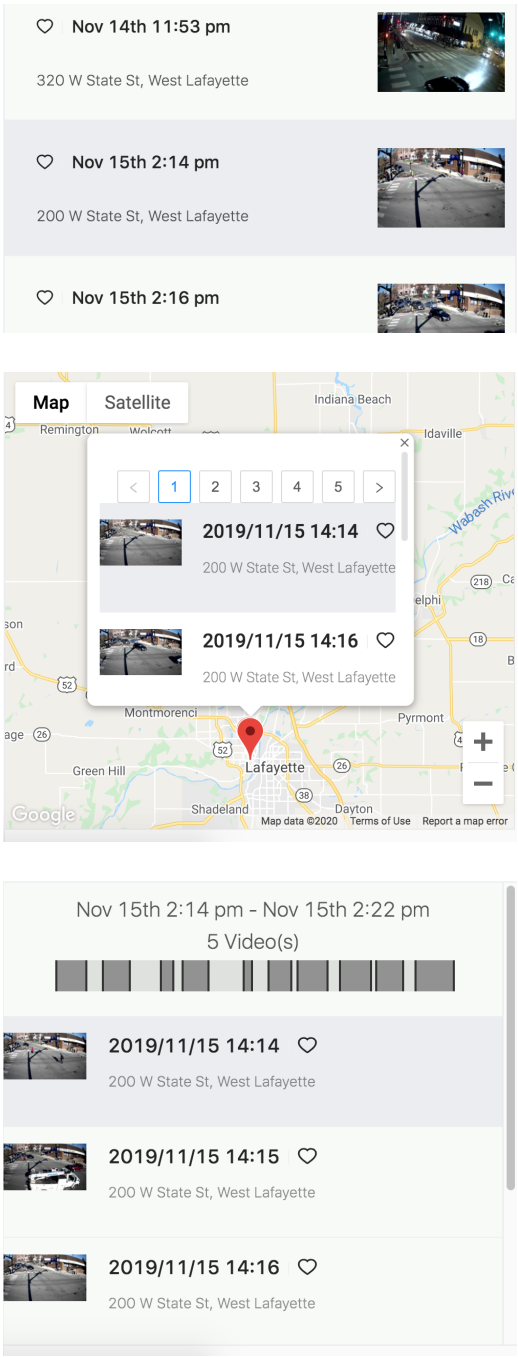



**Figure 6: The *list*, *map*, and *timeline* viewpoints.**

resource-available classification via a priority system. In this way, highest priority feeds are classified at ingest time, and deferred processing is performed when necessary. Of course, when processing is deferred, it is performed at query time. Hence, in the worst case, only video relevant (in space and time) to an incident is classified. Our current system assigns a priority to a video feed equal to the number of incidents it matches in space and time. We expect to investigate more complex schemes in the future.