The Design and Implementation of a Wireless Video Surveillance System

Tan Zhang[†], Aakanksha Chowdhery[‡], Paramvir Bahl[‡], Kyle Jamieson[§], Suman Banerjee[†]
[†]University of Wisconsin-Madison, [‡]Microsoft Research Redmond, [§]University College London
{tzhang, suman}@cs.wisc.edu, {ac, bahl}@microsoft.com, k.jamieson@cs.ucl.ac.uk

ABSTRACT

Internet-enabled cameras pervade daily life, generating a huge amount of data, but most of the video they generate is transmitted over wires and analyzed offline with a human in the loop. The ubiquity of cameras limits the amount of video that can be sent to the cloud, especially on wireless networks where capacity is at a premium. In this paper, we present Vigil, a real-time distributed wireless surveillance system that leverages edge computing to support real-time tracking and surveillance in enterprise campuses, retail stores, and across smart cities. Vigil intelligently partitions video processing between edge computing nodes co-located with cameras and the cloud to save wireless capacity, which can then be dedicated to Wi-Fi hotspots, offsetting their cost. Novel video frame prioritization and traffic scheduling algorithms further optimize Vigil's bandwidth utilization. We have deployed Vigil across three sites in both whitespace and Wi-Fi networks. Depending on the level of activity in the scene, experimental results show that Vigil allows a video surveillance system to support a geographical area of coverage between five and 200 times greater than an approach that simply streams video over the wireless network. For a fixed region of coverage and bandwidth, Vigil outperforms the default equal throughput allocation strategy of Wi-Fi by delivering up to 25% more objects relevant to a user's query.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Distributed networks

Keywords

Edge computing; video surveillance; wireless

1. INTRODUCTION

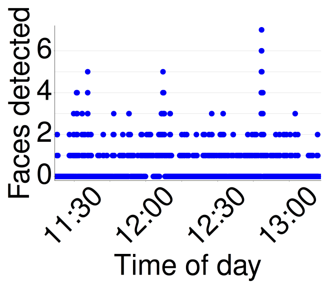
Video surveillance today has become pervasive: it provides inventory control for today's retail stores, security for corporate and educational campuses, and both security and demand monitoring for roadways and rapid transit networks in smart cities. Furthermore, the trend is expected to increase: according to a recent IDC report as much as 5,800 exabytes of video footage will be stored by the year 2020 [15]. But, while cities such as London and Beijing have close to a million cameras deployed today, the vast majority of them are wired into the physical infrastructure, which incurs massive deployment cost and effort. As a result, the coverage these cameras provide is necessarily limited. Smart video cameras currently on the market such as Dropcam [8] use a passive infrared motion detection sensor to transmit video in response to human or animal movement, but miss potentially notable stationary objects or people in a camera's field of view. We therefore see an opportunity for a surveillance system to radically scale by leveraging a user's query to look deeper into the video feed.

This paper examines the possibility of building a wireless video surveillance network over UHF whitespace and unlicensed Wi-Fi spectrum bands for uploading real-time video feeds to the cloud, similar to today's wired video surveillance systems. The important challenge in realizing such a system is the limited capacity of wireless spectrum, far from sufficient to accommodate many simultaneous high-definition video feeds. The bandwidth usage of Dropcam has already been shown to be several order of magnitude higher than peer-to-peer file-sharing applications [3]. Moreover, wireless networks suffer channel losses that need to be overcome by channel coding and retransmission techniques, introducing overhead that further limits the effective capacity of the network. When wireless capacity lags behind the surveillance cameras' demands, queue lengths increase and the system is forced to degrade application performance.

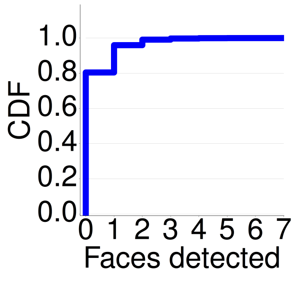
Our approach begins with an analysis of more than 250 hours of real-world video feeds showing that most of the time there is nothing of interest in a video feed. For example, in one of our deployments in a lounge area, we found that over a period of two weeks, 99.8% of the time a state-of-the-art face detection algorithm finds no faces, obviating the need to use wireless capacity to upload video feed to the cloud. Even watching a scene where many people are present such as a busy corridor during a lunchtime period in an office building, state-of-the-art vision algorithms detect very few faces most of the time. Figure 1 shows a time series of faces detected and a CDF of the face count: 80% of the time the face-detection algorithm finds no faces.

We present *Vigil*, a real-time wireless video surveillance system that leverages edge computing [24] to enable wireless video surveillance to scale to many cameras and expand its region of coverage in a crowded wireless spectrum. Vigil's processing flow begins with a user inputting a *query* into the system, such as locating people or objects of interest, counting the number of people passing in an area of interest, or locating people seen nearby a person of interest. Vigil's edge compute nodes (ECNs) then locally process each camera's video feed with lightweight, stateless vision algorithms such as motion and object detection, resulting in a stream of

MobiCom'15, September 7–11, 2015, Paris, France.
© 2015 ACM. ISBN 978-1-4503-3619-2/15/09 ...$15.00.
DOI: http://dx.doi.org/10.1145/2789168.2790123.



(a) Face count time series



(b) Face count CDF

Figure 1: Time series and CDF of a count of the number of faces a state-of-the-art vision algorithm detects during a busy period at an office building.

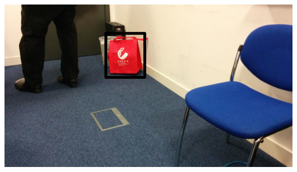
analytic information comprising the indices of significant frames in the raw surveillance video stream. This analytic information triggers the ECNs to upload analytic information, together with significant associated video frames, to the cloud. The ECN also locally stores video footage temporally-close to frames of significance for subsequent retrieval. Later and if needed, Vigil performs a deeper analysis on the video feed (*e.g.* object/face recognition or trajectory synthesis).

Once Vigil has extracted analytic information from cameras' video frames, it has to decide the relative importance of different frames to the query at hand. This allows it to prioritize more important frames over less important frames when wireless capacity is scarce. To quantify frames' relative importance, we propose and evaluate a new metric called *ops* (objects per second) whose objective is to maximize the number of query-specified objects (of interest in the video frame) delivered to the cloud while minimizing the bandwidth required to upload images containing those objects. We also incorporate a priority-based mechanism into Vigil so that queries can incorporate their own logic into the way that Vigil prioritizes traffic, overriding *ops*.

Once Vigil has prioritized frames, a novel traffic scheduling algorithm uploads the frames most relevant to the user's query in reverse-priority order (*i.e.* most-relevant frames first), to ensure efficient utilization of available wireless capacity. This *video content-aware* uploading strategy suppresses a large fraction of unrelated image data, conserving utilization of the scarce wireless medium while simultaneously boosting the responsiveness of the system.

As an additional feature, Vigil scavenges the wireless capacity it conserves to provide public Wi-Fi access. This *hybrid* network design offsets the deployment and maintenance cost of the underlying network infrastructure.

We have deployed Vigil at three sites in two countries under vastly different operational conditions. Two of our deployments are based on TV whitespace networks, and operate in multiple indoor locations and a campus-wide outdoor area. A third deployment is based on a Wi-Fi network and deployed indoors. We compare the performance of Vigil to an "oracle" system running a state-of-the-art vision algorithm on all the frames every camera captures and uploading these frames to the cloud. Depending on the level of activity in a scene, experimental results show that Vigil allows a video surveillance system to support a geographical area of coverage between five and 200 times greater than an approach that simply streams video over the wireless network. For a fixed region of coverage and bandwidth, Vigil outperforms the default equal throughput allocation strategy of Wi-Fi by delivering up to 25% more objects relevant to a user's query.



(a) Object of interest.

(b) Line of people.

Figure 2: The two Vigil use-cases targeted in this paper.

This paper makes the following contributions:

1. **Edge-computing architecture for wireless video surveillance:** We propose and evaluate a novel architecture for wireless video surveillance that leverages edge computing for a highly scalable, real-time wireless video surveillance system. Our proposed architecture partitions its design between computing elements at the edge of the network and in the cloud.
2. **Camera & frame selection to suppress redundancy:** When multiple cameras looking at the same scene capture different views of an object or person of interest to the user's query, Vigil uploads only the frames that best capture the scene. This collaboration *between* cameras distinguishes Vigil from popular solutions on the market such as Dropcam [8].
3. **Content-aware traffic scheduling:** We propose and experimentally evaluate the *ops* metric, showing that it maximizes utility in the sense of objects per second uploaded to the cloud for person-counting applications and applications that recognize objects. We also incorporate a priority-based scheduling mechanism, showing increased utility for security applications.
4. **Hybrid surveillance-access network:** We introduce a hybrid network architecture where ECNs also provide Wi-Fi network access to recoup the cost of deploying and running the surveillance network.

2. MOTIVATION: USE CASES

Security and counter-terrorism. Across our cities, CCTV cameras are installed in underground transport networks, on buses, and in airports and other public areas. We envisage online, real-time processing of the wireless video feed so that law enforcement and counter-terrorism can track public threats in real-time. For example, in the event of multiple co-ordinated attacks on public transport, the video surveillance network can pick out the face of one perpetrator, scan the database of cloud-stored video for other people the perpetrator was spotted with, and then search for those associated persons in real-time as the attack progresses, directing law-enforcement to the locations of the perpetrator's accomplices for intervention.

Locating people or objects of interest. In many situations, people are interested in locating objects or people of interest, such as an "Amber Alert" in the United States, or an unattended bag of a certain color (Figure 2(a)). An airport might choose to continuously run a query on CCTV footage looking for bags that are not held by any person nearby, flagging up unattended baggage to airport authorities. Traffic monitoring systems use vision-based algorithms to detect and count the number of cars on the highways [4, 22].

Customer queue analytics. In places where customers line up for service, such as coffee shops, supermarkets, or amusement parks, management has an interest in knowing numbers of people waiting in line and the dynamics thereof over the course of a day (Figure 2(b)). Cameras are used to track line length, but face or body

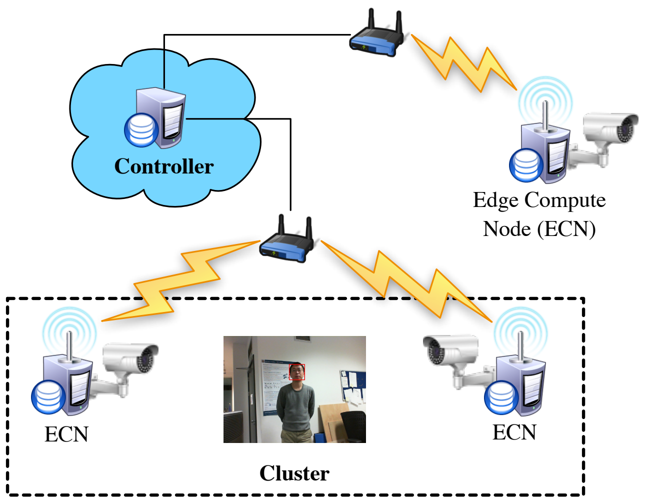


Figure 3: Vigil architecture, in which *Edge computing nodes (ECN)* are connected to camera devices to perform simple vision analytic functions, while uploading a relevant portion of video feed to a *Controller* in the Internet. Cameras monitoring the same areas form a *cluster*.

counting is challenging, as people strike different poses and turn at different angles to the camera. Consequently, a better design is to deploy an array of cameras surrounding the queuing area. The system then fuses their data together to form a more accurate count of the people in line.

3. DESIGN

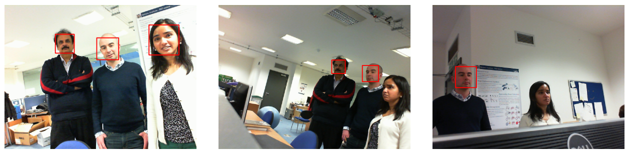
Vigil proposes a novel architecture (Contribution 1 in Section 1) that leverages the computing elements at the edge of the network to minimize bandwidth consumption of a wireless video surveillance network without sacrificing surveillance accuracy. Vigil consists of the two major components shown in Figure 3. The *controller* is located in the cloud and receives users' queries, coordinating all the other parts of the system to answer the query. An *edge compute node* (ECN) is a small computing platform (*e.g.*, a laptop or embedded system) that is attached to a camera to bring cloud resources close to the edge of the network.[1] [2] Each ECN receives the video feed from its connected camera, and executes the first, stateless, stages of computation such as face detection or object recognition. It then periodically uploads analytic results to the controller. The ECNs also perform video compression, indexing and maintaining a short-term store of the video frames they capture. ECNs connect to the controller via wireless links operating over TV whitespace or Wi-Fi bands. The controller runs a frame scheduling algorithm, requesting ECNs to only upload a fraction of relevant video frames to conserve wireless bandwidth.

Vigil further utilizes saved wireless bandwidth to provide public network access, by augmenting each ECN with a Wi-Fi access point. The AP forwards users' traffic over the controller to the Internet, thus providing Internet hotspot functionality for nearby users (Contribution 4 of Section 1).

To improve the accuracy of vision analytic functions, we introduce the notion of a *cluster*: a group of camera nodes monitoring

[1]Note that we use the terms ECN and camera interchangeably in the remainder of the paper.

[2]We discuss the case of connecting multiple cameras to a single ECN in § 7.



(a) Camera 1 **(b)** Camera 2 **(c)** Camera 3

Figure 4: Different views on the same scene can reveal more people at favorable angles to the camera, as captured by Camera 1's view of this scene with three people.

a single geographical area with *substantially overlapping* views, as illustrated in Figure 3. Leveraging a cluster of cameras allows us to capture multiple views of objects from different angles and overcome the limitations of the state-of-the-art visions algorithms (Figure 4). These clusters are constructed during a calibration phase based on the covered area of each camera. Vigil effectively fuses the observations from cameras in a cluster to improve surveillance accuracy without significant wireless bandwidth overhead while existing surveillance systems upload video from each camera to the cloud before executing vision analytic functions.

Design goals and scope. The primary goal of our design is to maximize the number of query-specified objects the system returns while minimizing the bandwidth required to upload the images containing these objects. We also limit the scope of our design, noting the following non-goals:

1. Each vision algorithm has a certain accuracy and degree of confidence in the results it returns. Improving the accuracy of vision algorithms is outside this paper's scope.
2. Enough cameras are present and use a high enough resolution and frame rate so that with high probability, the resulting raw video streams capture objects of interest.
3. In a Vigil deployment, cameras are line-powered, so there are no battery-conservation issues.

The next section describes the ECN in detail, followed by a description of how Vigil's controller prioritizes frames to upload within a cluster of ECNs (§3.2, *Intra-cluster processing*), and arbitrates demand across multiple clusters (§3.3, *Inter-cluster traffic shaping*).

3.1 Edge compute node

We begin by describing the stateless image processing functions performed by each ECN. Each Vigil application implements a callback API `frameUtility`, which returns an integer value evaluating the importance of a video frame to that application. Referring to our two use-cases, queue counting application at a coffee shop defines `frameUtility` to be the number of people visible in the frame. The application that locates people or objects of interest defines `frameUtility` to be one if the object of interest is found in the frame, and zero otherwise. When Vigil receives a new query, the query contains a definition of the `frameUtility` function, which is disseminated to all ECNs. Each ECN calls `frameUtility` on every received frame to generate an array of *analytic data* we denote as `utils`. It then uploads these analytic data to the controller.[3] While we focus on person-counting applications to define `frameUtility` in this paper, Vigil can process any queries that can process a vision analytic function at ECN and output a `frameUtility` (for example, an object or a license plate number is present or not).

[3]As discussed below in Section 3.2.2, an advanced intra-cluster scheduling algorithm also requires ECNs to upload the location of each detected object.

Frame index:	0	1	2	3	4	5	6	7	8	9
`util[1]:`	1	1	3	3	3	3	3	1	1	1
`util[2]:`	1	1	1	1	1	1	1	1	1	1

Figure 5: The intra-cluster frame selection algorithm processes on counts representing the number of objects from each ECN camera in a cluster. This example shows object counts over a certain time epoch from two ECNs.

We note here that camera placement, focus, environmental conditions, motion and many other factors can blur objects and faces of interest to the system. Indeed, based on our experience, vision algorithms such as face detection are more likely to fail on blurred images, and so `frameUtility` implicitly factors the image quality into the number of detected objects.[4] We elaborate further on this design choice in Section 7.

Video storage. Each ECN is equipped with some persistent storage devices to retain all the video frames captured close to the time of detected events. This allows Vigil to support "drill-down" queries, which can be answered by uploading additional video data from ECNs to the cloud for more detailed analysis. Consequently, Vigil provides a time window, such as one or two weeks, within which the system retains important video information, as existing wired surveillance systems do.

3.2 Intra-cluster processing

The Vigil controller runs an intra-cluster algorithm to determine the most valuable frames from cameras within a cluster to upload (Contribution 2 in Section 1). The key challenge is to eliminate redundant observations of multiple cameras within a cluster, capturing the same objects, to minimize communication bandwidth without actually exchanging the redundant frames. This section describes our intra-cluster scheduling algorithm in two iterations: a strawman version described next, and its generalization in Section 3.2.2, which proposes a re-identification approach to check if the same objects are captured by cameras in a cluster.

3.2.1 Basic frame-selection algorithm

The basic algorithm selects frames to upload by examining the frame utility array `utils[c]` that each ECN `c` reports to the controller. We show an example of the controller's view of the frame utility arrays for a cluster containing two ECNs in Figure 5: each element of `util` is an object count captured by a ECN during consecutive time slots. The maximum object count is based on the vision analytic function ECN is running (i.e., practically, the vision analytic algorithm will only detect a limited number of faces in a frame, for example). To reduce protocol overhead, our intra-cluster algorithm operates over a certain number of L_e time slots, referred to as an *epoch* in this section. The controller selects a single ECN in each epoch—the selected ECN then uploads a fraction of its frames to the controller determined by inter-cluster traffic shaping (§3.3 on p. 5). The basic version of our scheduling algorithm proceeds in three steps:

Step 1: The controller sums the L_e object counts from each camera across the epoch, selecting the camera c^* with the highest average counts (most information about the scene) in the epoch. In the running example of Figure 5, the algorithm selects Camera 1 for further processing.

	0	1	2
`sis:`	(2,3)	(7,1)	(0,1)

Figure 6: Step 2 of the Vigil intra-cluster frame selection algorithm aggregates one camera's beginning frame index and utility into a *selected image sequence* (`sis`) array containing a sequence of (frame index, utility) pairs.

Step 2: The second step of the algorithm processes Camera c^*'s counts, finding the frames that begin changes to the scene, and collecting them into a *selected image sequence* array `sis`, as shown in Figure 6. The `sis` array contains pairs of (frame index, utility), sorted by utility, breaking ties in utility by favoring the `sis` element with the longer duration sequence of images.

The frames can only be uploaded at a rate lower than the capacity of the wireless link from the ECN to avoid network congestion and frames from being dropped. To avoid congestion, the controller estimates the link capacity (in bits per second) C available from each ECN by examining the near-term sending rates and the loss rates. The ECN measures a time-averaged packet loss rate L at its wireless link layer, and the physical-layer bit rate R. The ECN then estimates the link capacity as:

$$C = R \cdot (1 - L). \tag{1}$$

Step 3: The final step of the algorithm takes as input the estimated available wireless capacity C, and estimates the number of bits that ECN c^* can upload. We make a simplifying assumption that ECNs within a cluster have similar wireless links to controller because they cover same geographical area of interest. Suppose the size (in bits) of the frames in the `sis` is N_{sis} and the length of the epoch in seconds is T_e. At the end of the epoch, the controller sends a control message to ECN c^* soliciting an upload. If $N_{\text{sis}} \leq C \cdot T_e$, the ECN uploads `sis`, along with all the changing frames as indicated in the `sis` after compression. Otherwise, the ECN uploads a fraction of the compressed images in `sis` in the decreasing order of utility. In the example of Figure 6, `sis[0]` has utility of 3, so frame 2 is uploaded first, followed by frame 7 and 0.

We note that this algorithm is an approximation that will lose information when more than one camera in the cluster sees objects that other cameras in the cluster miss. So, we describe a sophisticated intra-cluster algorithm to select the most valuable frames within a cluster.

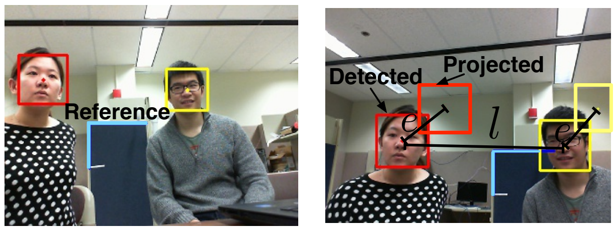
3.2.2 Sophisticated frame-selection algorithm

Vigil's sophisticated intra-cluster frame-selection algorithm specifically targets cases where one camera in a cluster sees objects that other cameras miss to select more than one ECN to upload images during a time epoch.

This algorithm relies on geometry and known locations of the ECNs to detect redundant viewpoints without actually exchanging the redundant frames. The algorithm first identifies duplicates of the same objects from multiple ECNs in overlapping camera views using object re-identification. It then prioritizes video frames from the cluster, factoring the count of "re-identified" objects into the frame utility metric.

Object re-identification. Object re-identification determines redundant objects reported by multiple ECNs in a cluster with overlapping camera views. By selecting the smallest subset of camera

[4] Blurry frames will result in fewer objects being detected, thus the `frameUtility` metric will characterize blurry frames as less useful. Furthermore, the minimum size required in terms of face pixels is encoded in the face or person detector.



(a) Camera 1 **(b)** Camera 2

Figure 7: Two cameras simultaneously capture a scene containing the same two faces. To avoid redundant counting, Vigil projects faces from Camera 1 to Camera 2 based on common reference points denoted by the two blue lines labeled "Reference" in Camera 1's view. If projection error does not exceed the distance between the detected faces, re-identification correctly identifies the two views of each face.

Frame index:	0	1	2	3	4	5
objects[1]:	**a**, **b**, c	**a**, **b**, c	b, **d**	b, d	b, **d**, f, g	b, d, **f**, g
objects[2]:	**a**, **b**	**a**, **b**	**d**, e	e	**d**, h, i	**f**, h, i

Frame index:	0	1	2	3	4	5
util[1]:	3	3	2	2	4	4
util[2]:	0	0	1	1	2	2

Figure 8: The advanced intra-cluster frame selection algorithm operating on unique objects after applying the object re-identification algorithm. Each letter denotes an unique object identified by re-identification. The algorithm chooses ECN 1's camera to be the primary camera view, while debiting the object counts reported by ECN 2 by the number of duplicate objects re-identified (in bold).

views to cover the overlapping views, we identify the unique objects across cameras within a cluster. To achieve this, Vigil uses the following lightweight approach [33] called re-identification.

Figure 7 shows an example where two cameras simultaneously detect the faces of two people in their overlapping views. The re-identification algorithm identifies if the detected face instances belong to the same person or not. It first selects common reference points between the camera views, for e.g., the two blue lines in Figure 7.[5] The reference points are then used to project *any* pixel in Camera 1's view onto the plane of Camera 2's view. The detected face instances are identified as the same person when the distance between projected face and the detected face is below a threshold. For e.g., the error e between the projection and the detected face in Figure 7(b) is much smaller than the distance between the two different faces. We set an empirical value to determine the threshold for this projection error e that accounts for any inaccuracy in marking reference points as benchmarked in Section 5.3. The threshold value is set during a calibration stage and can remain useful for a long period of time for static cameras. This re-identification approach has linear complexity in number of cameras because it projects all of the captured scenes to a common plane to calculate inter-object distance.

To integrate the re-identification technique on top of the basic algorithm Section 3.1, each ECN reports the center coordinates of detected faces and the frame utility in the modified `utils` to the

[5]The reference points are manually determined in a camera calibration phase or extracted using algorithms that extract SIFT image features [17].

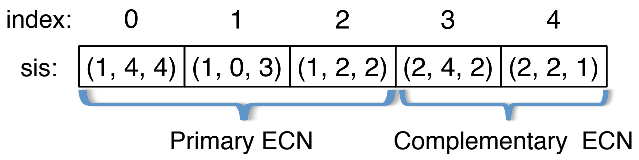


Figure 9: Step 2 of the advanced intra-cluster algorithm generates a selected image sequence (*sis*) array comprising (camera identifier, frame index, utility) tuples.

controller. The controller then performs re-identification using the analytic data in `utils`. Based on the results of object reidentification, The controller then executes the following sophisticated scheduling algorithm:

Step 1: The controller determines an ECN that has maximal average object counts (thus capturing the most information about the scene) to be the *primary* ECN. It then projects the detected objects from other (*complementary*) camera views onto the primary camera view. For each re-identified object, the controller debits the object count of all the complementary ECNs by one. This produces an updated utility array for each ECN. For example, Figure 8 shows unique object counts captured by two ECNs after applying object re-identification, along with their utility arrays. We choose ECN 1 as the primary camera view, while debiting object counts for ECN 2 by the number of duplicate objects re-identified (marked in bold).

Step 2: The controller determines a selected sequence of frames in the *sis* array for each ECN, which comprises tuples of (ECN identifier, frame index, utility). As illustrated in Figure 9, the modified *sis* array includes the frames captured by the primary ECN with changes in its object count (*i.e.*, frames four, zero, two). It also contains frames captured by the complimentary ECNs when a frame captured by the primary ECN fails to cover all the unique objects. For example, frames four and two of the (complementary) ECN 2 are appended to the *sis* array because they include additional unique objects, *i.e.*, objects h and i in frame four and object e in frame two.

Step 3: The controller consults the estimated wireless capacity C to determine whether all the selected frames in *sis* can be uploaded. If not, it prioritizes the selected images from the primary ECN, in decreasing order of their utility value. It then polls the selected images from all the complementary ECNs, in the order of their debited utility value.

Vigil can handle a large number of objects in a single frame as long as the vision analytic function is capable of doing so. The intra-cluster frame selection falls back to uploading each frame when the number of objects per frame are very large for each camera in the scene.

3.3 Inter-cluster traffic shaping

After determining the priority of frames to be uploaded within a cluster, the Vigil controller needs to coordinate upload bandwidth demand across the clusters that are within a wireless contention domain (*i.e.* served by a single access point). To do so, Vigil uses a novel inter-cluster traffic scheduling algorithm that attempts to allocate upload rates to each cluster that maximizing the number of useful objects per second delivered to the application (Contribution 3 in Section 1).

We describe our algorithm in the context of two application scenarios, i.e., counting applications such as customer queue analytics, and security based applications such as tracking a person or finding a suspicious object. These scenarios are monitored by two clusters of ECNs, which can contend with each other for uploading im-

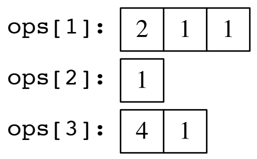


Figure 10: The per-cluster ops queues at the controller measure the utility, in operations per second, of sending the respective image sequences from each cluster.

ages. For the application scenarios related to analyzing a queue of customers or locating an object of interest, Vigil allocates rates proportional to the objects detected per second at each camera cluster. In these scenarios, the application can manually intervene when it detects large number of people or objects. For e.g., in a Starbucks coffee shop, the queues with higher person count could be served faster, and at the airport security checkpoints, the queue with highest person count would use more personnel to manage the queue. Following the notation of Section 3.2, we denote wireless capacity from cluster c as C_c. Since different video frames compress at different ratios, ECNs may also upload unequal frame sizes: we denote the size of the compressed frame in ith index of the selected image sequence from cluster c as L_i^c. Based on the these quantities, we calculate the number of useful objects per second (*ops*) for ith index of the selected image sequence of cluster c:

$$\texttt{ops[c][i]} = \frac{\texttt{sis[c][i].utility}}{L_i^c / C_c}. \tag{2}$$

The numerator of the *ops* metric is a count of objects, while the denominator has units of seconds (bits divided by bits per second). The *ops* thus captures how many useful objects per second the frame at ith index of the selected image sequence from cluster c will deliver if it is scheduled for transmission.

Step 1: Every time epoch, each cluster c uploads its selected image sequence sis, to the controller, which the controller stores in array element sis[c], an array of selected image sequences indexed by cluster number c.

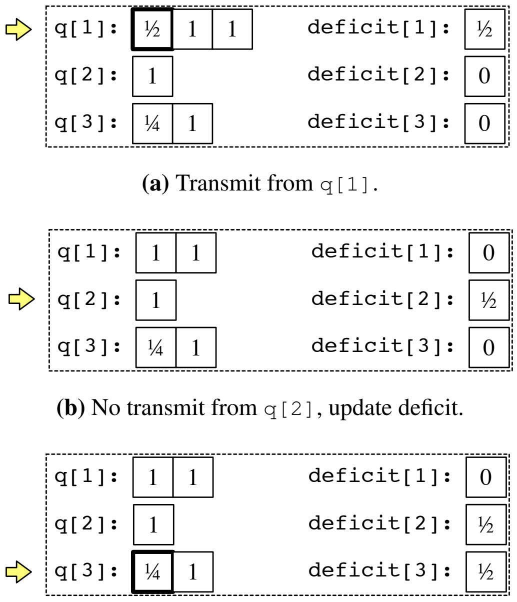
Step 2: For each cluster c and frame group index i, the controller computes ops[c][i] using Equation 2. The controller's state now appears as in Figure 10: a per-cluster queue of image sequences' *ops* values.

Step 3: The controller schedules service to different clusters using a variant of deficit round robin (DRR) scheduling [25] to approximate fair queuing [6]. To define terminology and provide context, we now briefly recall the DRR algorithm, in the context of serving Vigil clusters. Each cluster has a *deficit counter*, which represents the amount of information it is allowed to transmit when it is its turn. Vigil-DRR works by considering clusters in round-robin order. The controller adds a fixed amount of credit, called the *quantum*, to each cluster's deficit counter.[6] If the cluster's deficit counter exceeds the size of the packet, then the cluster transmits the packet and the controller decrements the cluster's deficit counter by the size of the transmitted packet.

Our DRR variant uses the reciprocal of *ops* in place of the packet length, for queue weights

$$\texttt{q[c][i]} = \frac{1}{\texttt{ops[c][i]}}, \tag{3}$$

[6]We describe our setting of quantum in Section 4.



(a) Transmit from q[1].

(b) No transmit from q[2], update deficit.

(c) Transmit from q[3].

Figure 11: The Vigil-DRR algorithm operating over three camera clusters with a quantum of ½.

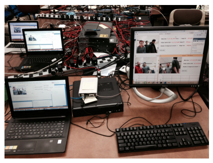
in the case that *ops* is non-zero, and drop the upload in the case that *ops* is zero. Vigil-DRR provides fair air-time to different clusters, each of which may have different wireless throughputs possible to the base station. Vigil-DRR also provides fairness between clusters in terms of the number of seconds per object change (utility). Thus a cluster that can upload a frame containing two object changes compared to a cluster that can upload a frame containing one object change.

Vigil-DRR also maximizes the number of objects per second delivered to the controller. To see this, consider the example of Figure 10 where there are three clusters, with frames at their respective queue heads having *ops* of two, one, and four respectively. If we service the clusters at rates r_1, r_2, and r_3 (bits per second), the number of objects per second uploaded to the controller will be the inner product of the preceding rate vector with the *ops* vector $(2,1,4)$: $\langle (r_1,r_2,r_3),(2,1,4)\rangle = 2\cdot r_1 + 1\cdot r_2 + 4\cdot r_3$. The Cauchy-Schwartz inequality states that we can maximize this inner product (*i.e.*, the number of objects per second uploaded) by choosing (r_1,r_2,r_3) proportional to the *ops* vector. Setting packet length inversely proportional to *ops* in the DRR algorithm accomplishes this, as DRR will schedule packets at rates inversely proportional to the packet length normalized by the wireless throughput.

Figure 11 shows an example of Vigil-DRR in operation over three clusters, with a quantum value of ½. Figure 11(a) shows the initial state of all queues. At the first time-step, the algorithm increments the deficit counter of the first queue by quantum (½ in this example) and then checks the q[] value of the change at the head of the queue. Since it is ½, Vigil-DRR transmits the change and decrements the deficit counter by ½, leaving zero in the deficit counter, as shown in Figure 11(b). At the next time-step, the algorithm increments the deficit counter of the second queue by ½, but the q[] value at the head of the second queue is greater than ½, and no transmission occurs, as shown in Figure 11(c). At the final time-step, the algorithm increments deficit[3] by ½ and transmits, leaving ¼ in the deficit counter. The algorithm then proceeds similarly in a round-robin fashion.

(a) Site #1 lounge.

(b) Site #2 office.


(c) Site #3 lab.

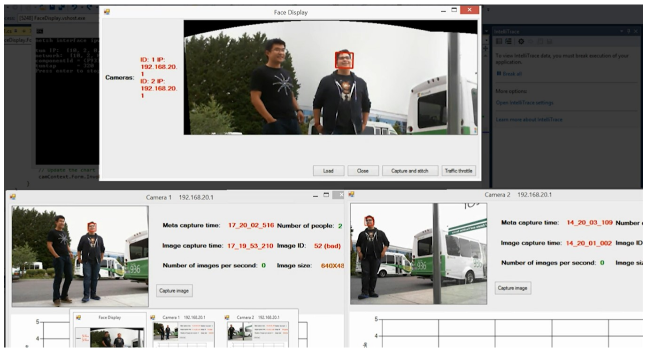

(d) Site #1 outdoor bus stop pilot.

Figure 12: Vigil deployment at three sites.

For certain security-based applications such as intruder detection and tracking a person, the ECN nodes may assign high-priority to the captured frames. We modify the design of Vigil-DRR algorithm similar to MDRR algorithm to allow a high priority queue that allows the frames requesting priority access on the channel to be uploaded immediately overriding the *ops* metric.

4. IMPLEMENTATION

In this section, we describe our implementation of Vigil. With the goal of understanding system performance *in situ*, we have deployed Vigil at three sites. We describe these deployments in the next section, and hardware and software details in Section 4.2 and Section 4.3, respectively.

4.1 Testbed deployments

We deploy a single cluster of Vigil cameras at three sites under vastly different operational conditions.[7] Firstly, we study an outdoor surveillance scenario by deploying a cluster of camera ECN nodes at a shuttle bus stop at Site #1's outdoor campus (Figure 12(d)), where we monitor real-time vision analytic functions such as counting passengers. The ECN nodes in this deployment connect to a controller by long-distance backhaul links over TV whitespaces. Secondly, we study an indoor surveillance scenario in a busy office hallway by deploying a cluster of camera ECN nodes at Site #1, where we monitor the frequency of passers-by. The ECN nodes in this deployment use unlicensed 2.4 GHz Wi-Fi to connect to the controller. Finally, we surveil an open-plan office at Site #2 (Figure 12(b)) and an indoor lab environment at Site #3 (Figure 12(c)), where we monitor working hours and office occupancy. The ECN nodes in these deployments use unlicensed 2.4 GHz Wi-Fi at Site #2 and TV whitespace radios at Site #3 to connect to the controller. The indoor deployments in the three sites have been operational for the last two months, giving us valuable information on traffic patterns.

[7]Privacy of the monitored users is preserved by ensuring that the cameras only capture users who had given prior permission to take part in the study.

4.2 Hardware

This section describes the hardware platform we used to deploy the controller and the ECNs. To implement the ECN nodes, we use laptops running a user-space program to perform image analysis functions on video feeds from the connected cameras. We have used laptops and the Intel Next Unit of Computing (NUC) to prototype our system, but embedded devices (e.g. Gatework routers or NUC) with 500 MHz–1 GHz CPU have enough processing power to run image analysis functions for ECNs. Recent trends in vision are moving toward smart cameras, thereby enabling face recognition, motion detection, and other image analysis tools to be implemented to an increasing extent in hardware [10].

Each laptop is also attached to a Wi-Fi based router to provide public Internet access. Wi-Fi traffic along with vision analytic data are sent over the connected routers to a central controller. Linux-based routers running the OpenWRT operating system upload vision analytic traffic from the ECN nodes to the controller. These router boards control two different types of wireless interface cards for communication in TV whitespaces and the 2.4 GHz ISM band, respectively. In our outdoor deployment, we use TV-band transmitters from Doodle Labs [7] for TV whitespace communications. The radios are configured in a single vacant TV channel at a center frequency of 580 MHz, with a 5 MHz bandwidth.[8] In our indoor deployment, we use off-the-shelf 802.11a/b/g radios operating in the 2.4 GHz band.

We implemented the central controller on a workstation hosting a user-space program that collects ECN traffic for further processing. Both ECN and controller run code implemented in Microsoft Visual Studio on Windows 8.1.

4.3 Software architecture

This section describes the software architecture of the controller and ECNs. The entire software codebase consists of 6,000 lines of C Sharpcode that implements vision analytic algorithms and end-to-end protocols, along with approximately 100 lines of C code for `ath5k` driver modifications.

Virtual networking device: We use the `tun` virtual networking device in OpenVPN [21] software at each camera node. All traffic from Wi-Fi users is directed through this virtual device, which is subsequently captured by our application for traffic shaping, then transmission via the underlying whitespace interface.

MAC layer modifications: We disable the rate adaptation function in the `ath5k` driver, and allow the ECN nodes to control the physical-layer data rate used to send each packet by appending a special bit-rate in the header of each packet to be transmitted. This allows the controller to accurately estimate the wireless capacity C of the link from the ECN node to the controller (equation 1). In our evaluation, we fixed C to isolate the effect of our frame selection algorithms. However, any rate adaptation algorithm can be easily adopted in ECNs to further improve performance.

5. EVALUATION

In this section we evaluate Vigil's performance gains over conventional approaches. Sections 5.1 and 5.2 evaluate the accuracy improvement of Vigil's intra-cluster frame selection (§3.2.1) and inter-cluster traffic shaping (§3.3) components. Section 5.3 presents microbenchmarks that stress-test Vigil's vision and advanced scheduling algorithms (§3.2.2).

[8]Before each experiment, we query a commercial spectrum occupancy database [27] to ensure this channel is vacant.

Frame index:	0	1	2	3	4	5	6	7	8	9
`utility:`	1	1	2	2	2	2	1	1	3	3
Changes:	1		2				1		3	

Figure 13: Frame utilities and frames where utility changes. Uploading all the frames where utility changes gives 100% accuracy relative to a baseline (*i.e.* human validation, or a vision algorithm working on full M-JPEG camera streams).

For object-counting applications, we measure the accuracy of Vigil as the accuracy of object counts relative to baseline data from running vision algorithms on all frames of the raw video streams of all available cameras. If the system uploads all the frames where the object count changed (*i.e.* the zeroth, second, sixth, and eighth frames, as shown in Figure 13) the system will make no errors relative to baseline and has 100% accuracy. Otherwise the accuracy is the percentage of frames that the system uploads where the object count changed.

5.1 Intra-cluster frame selection

In this section, we evaluate the accuracy of Vigil in a cluster of cameras using intra-cluster frame selection (§3.2).

Methodology. In this experiment, we use video traces collected from a cluster of three cameras at Site #3 in a lab environment. The three cameras log video traces synchronously for a duration of 180 seconds, with about 2,000 image frames. The ECN connected to the camera uploads detected face counts and the controller selects which frames are uploaded from ECNs based on the intra-cluster frame selection algorithm.

We choose the ECN's slot time (§3.1, p. 3) to be 100 milliseconds: this strikes a good tradeoff between the detection errors of vision algorithm and responsiveness in detecting people. We configure the epoch time L_e to be five slots: this choice of epoch length reduces the protocol overhead of sending control messages, while enabling use of the best available camera in detecting people. We experimented with other choices of L_e and found end-to-end performance was not sensitive to this parameter.

We compare the accuracy of intra-cluster frame selection in Vigil to two approaches: a *Round-Robin* approach that cycles through all cameras within a cluster in a round-robin manner to upload frames and a *Single-Camera* approach that arbitrarily selects a single camera to upload frames. Note that all approaches only upload the frames where the count of detected faces changes. We constrain the capacity of wireless link from each ECN to the controller and repeat the experiment five times.

Results. Figure 14 shows the performance gains of Vigil as we increase the per-camera available wireless capacity for video traces collected at low, medium and high activity levels. The bandwidth required at each activity level is summarized in table 1. In Figure 14, the bandwidth required at low activity level (at most 16 Kbit/s) is lower than the available per-camera wireless capacity and therefore, both Vigil and *Round-Robin* achieve more than 90% accuracy, while the single camera suffers because of lack of sufficient coverage. Similar results are observed for medium activity level, except Vigil outperforms other approaches when the available per-camera wireless capacity 50 kbps is lower than the bandwidth required for medium activity level (at most 80 kbps). Finally at high activity level, the bandwidth required is much higher than the available per-camera wireless capacity and we observe 23-30% gains for Vigil compared to *Round-Robin* because Vigil prioritizes those frames across cameras that maximize the accuracy.

Level	# people	Change interval(s)	Bandwidth(Kbps)
Low	1	5–15	4–16
Medium	4	1–2	32–80
High	7	0.2–1	100–400

Table 1: Summary of video traces used to benchmark intra-cluster algorithms in terms of the number of participants, the frequency of change in object counts, and required bandwidth to upload the frames where object count changes.

We gather more insight in to why Vigil results in higher accuracy compared to a round-robin or a single camera approach. Figure 15 illustrates that Vigil's accuracy increases with the number of cameras up to the point where no blind spots are left uncovered. In this example, two cameras provide a significant gain over a single camera approach, but subsequently adding more cameras does not improve performance, because Vigil already prioritizes those frames which maximize accuracy.

5.2 Inter-cluster scheduling

In this section, we evaluate the accuracy of Vigil across multiple clusters of cameras by using the inter-cluster traffic scheduling algorithm (§3.3), examining to what extent the system can maintain accuracy as wireless capacity becomes more and more scarce, and more camera clusters contend on the same wireless bandwidth. We compare the accuracy of Vigil's inter-cluster traffic scheduling algorithm to two approaches: an equal throughput allocation which is a throughput-based fairness policy that gives equal throughput to all the camera clusters such as in the case of Wi-Fi and an equal time based allocation which is a time-based fairness policy. Note that all approaches only upload the frames from the selected image sequence `sis` of each ECN.

Methodology. We simulate a network of clusters of cameras that contend over a shared wireless channel. We vary the wireless capacity of this shared channel from 1 Mbps to 20 Mbps to quantify the ability of inter-cluster frame selection to alleviate congestion on the shared wireless medium. We emulate different activity levels by modeling the arrivals at each camera cluster by a Poisson arrival process to emulate the traffic patterns from our real-world deployments, where an increasing rate λ corresponds to higher activity levels. We assume that each arriving person departs after a constant dwell time. A single image in our experiments is 30 Kbytes, which takes approximately 240 ms to upload at one Mbit/s. We choose a Vigil-DRR `quantum` of 100 (seconds/object) so that Vigil-DRR would transmit a three-object frame without cycling round the clusters. We found that Vigil-DRR is not sensitive to our choice of `quantum`. We simulate the system over a time period of approximately one hour.

We evaluate the accuracy of Vigil-DRR algorithm across multiple clusters of cameras. In these experiments, we simulate a network of ten clusters of cameras that contend over a shared wireless bandwidth where each cluster has two cameras. The number of faces detected at each cluster is modeled by a Poisson arrival process, where the rate of the Poisson arrival process λ is set to 2.5 (objects/second) for low activity level, 5 (objects/second) for medium activity level, and 12.5 (objects/second) for high activity level. Note that while Vigil selects the most relevant frames from two cameras in each cluster based on intra-cluster frame selection, but the equal throughput and equal time approach assume one camera per cluster for fair comparison of traffic scheduling.

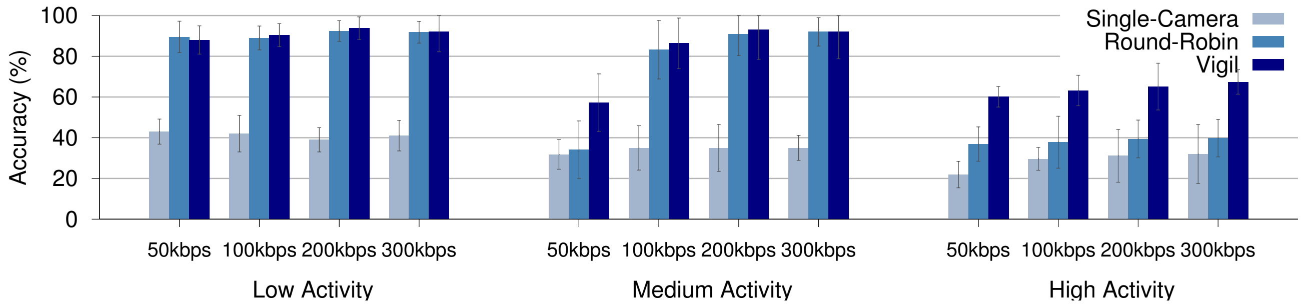


Figure 14: Accuracy of intra-cluster frame-selection in Vigil relative to a single-camera system and a multi-camera system with round-robin scheduling. Error bars show standard deviation of the experiment in varying wireless conditions.

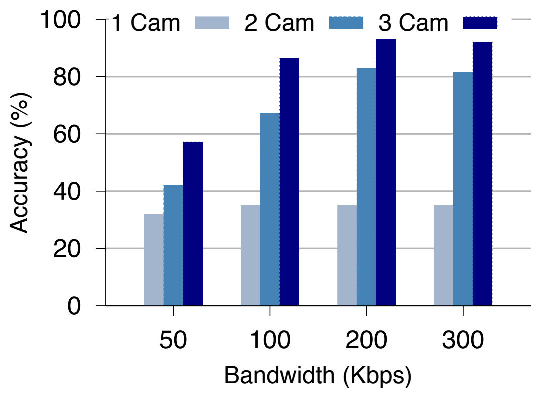


Figure 15: Vigil's accuracy in a single-cluster surveillance network with different number of cameras.

5.2.1 *Equal wireless capacity*

We first consider the scenario where the available wireless capacity is same from each ECN to the controller in ten camera cluster. Figure 16 shows the performance gains of Vigil-DRR as we increase the shared wireless capacity at low, medium and high activity levels. We observe that Vigil-DRR requires more wireless bandwidth to achieve 100% accuracy at higher activity levels. In this example, Vigil-DRR utilizes only 10 Mbit/s at low activity level to achieve 100% accuracy for ten camera clusters, but at high activity level, it achieves only 80% accuracy at bandwidths as high as 20 Mbps. Further, we note that Vigil-DRR significantly outperforms the equal throughput allocation and equal time allocation approach when the shared wireless capacity is not sufficient because it prioritizes the frames with maximum object count, using the *ops* metric. In this example, Vigil-DRR achieves gains of 20-25% over the other two approaches at 5 Mbit/s in low activity level and 10 Mbit/s in medium activity level. Finally, we note that the equal throughput and equal time allocation approaches achieve similar accuracy gains because the available wireless capacity is same from each ECN to the controller.

5.2.2 *Unequal wireless capacity*

Now we consider a scenario where the available wireless capacity from each ECN to the controller varies across ECNs. Figure 17 shows the performance gains of Vigil-DRR when the available wireless capacity from five clusters to ECN is C_1 and from the other five clusters to ECN is C_2 in a network with ten clusters of cameras. We first note that equal throughput allocation approach penalizes the clusters with high wireless capacity C_2 to sacrifice accuracy to ensure all clusters get equal throughputs. On the other hand, equal time allocation approach ensures time-based fairness allowing the clusters with high wireless capacity C_2 to upload more frames than clusters with low capacity C_1. But Vigil-DRR outperforms both these approaches in terms of accuracy for both the clusters with low wireless capacity C_1 and high wireless capacity C_2. Further, the gap in accuracy between the high-capacity and low-capacity clusters is much smaller for Vigil-DRR compared to equal time allocation approach because of maximizing the *ops* metric.

5.3 Vision algorithm microbenchmarks

In this section, we evaluate the two vision algorithms Vigil uses: face detection and a re-identification algorithm to associate faces detected in overlapping camera views.

5.3.1 *Face detection*

This evaluation answers two questions: first, how accurate is the Haar-cascade-classifier-based face detection algorithm used in our system? Second, in terms of that accuracy, what is the impact of various video compression schemes on system bandwidth savings? We compare two state-of-the-art video compression algorithms, M-JPEG and MPEG-4, to determine which fits the design of Vigil best.

Methodology. We use a single camera to record five-minute video traces at two different resolutions. Each trace contains about 7,000 images. Two people arrive and departed randomly in the scene, facing the camera. The distance of the subjects to the camera ranges from two to eight meters. The ground-truth person count is established by visual confirmation. We compress the frames with different state-of-the-art compression algorithms at different levels, and then apply a face detection algorithm on the compressed images to understand the tradeoff of accuracy and bandwidth required.

Results. Figure 18 shows a scatter plot of bandwidth required as the accuracy of the face detection vision algorithm increases. For each video compression algorithm, the accuracy of the face detection increases when it is compressed less because of low information loss. Here Vigil-crop applies M-JPEG compression only on the cropped faces in an image (by replacing the image background with a single RGB color). We observe that Vigil-crop outperforms M-JPEG compression without object cropping by 2–5× in bandwidth savings. It even outperforms the state-of-the-art MPEG-4 algorithm by a factor of two in bandwidth savings for the same accuracy. We therefore choose M-JPEG algorithm with object cropping in Vigil. Further, we note that Vigil allows a wide accuracy and bandwidth tradeoff compared to MPEG-4. This is because MPEG-4 applies delta-based frame compression, which either removes all the details in intermediate frames, or have to keep most of the redundant information.

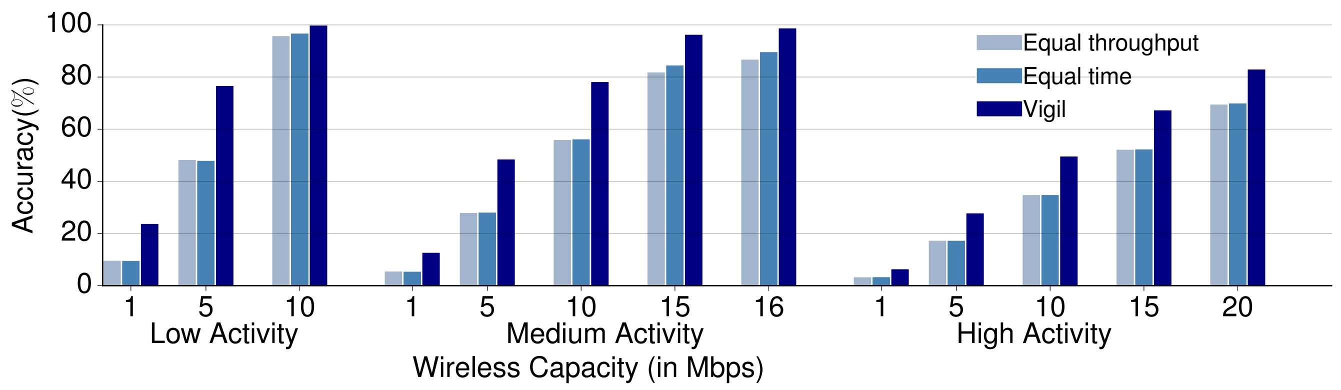


Figure 16: Accuracy of a multi-cluster system as the wireless capacity varies as shown on the x-axis. We compare Vigil with time-based fairness and equal throughput allocation for ten cluster of cameras. Vigil uses two cameras per cluster to select the most relevant frames, but equal throughput and equal time approach assume one camera per cluster for fair comparison.

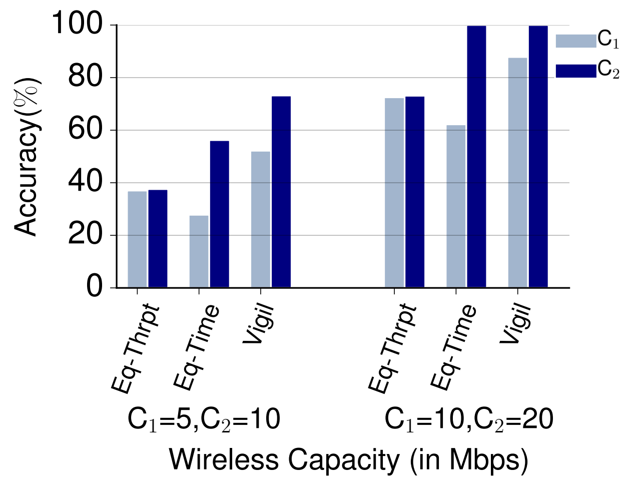


Figure 17: Accuracy of a multi-cluster system with 10 clusters where the wireless capacity of 5 clusters is C_1 and other 5 clusters C_2. Vigil outperforms both equal time-based fairness (Eq-Time) and equal throughput allocation (Eq-Thrpt).

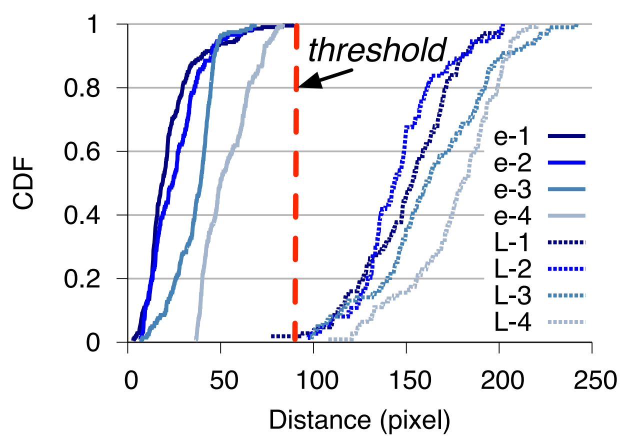


Figure 19: CDF of the distance l between two people's faces and the projection errors e when mapped from one camera view to another (illustrated in Figure 7(b)).

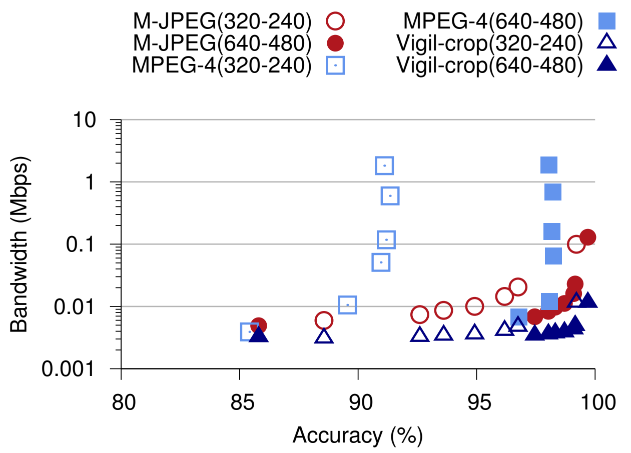


Figure 18: Bandwidth required versus accuracy of the face detection vision algorithm on videos compressed with different video compression algorithms, including our algorithm Vigil-crop that performs M-JPEG on the cropped objects. Only the selected frames in the `sis` array are compressed for this bandwidth calculation.

5.3.2 *Object re-identification*

We evaluate the accuracy of the re-identification algorithm described in Section 3.2.2. This section addresses the question of how well Vigil can tag faces in overlapping camera views as to the same person.

Methodology. In this experiment, two cameras synchronously log video traces at four indoor locations, where each trace consists of 300 images with the faces of same two people detected at different locations. The two cameras have a partially overlapping view as illustrated in Figure 7. The re-identification algorithm projects the faces detected at the first camera onto the corresponding images captured by the second camera. It then calculates the distance between the projection and the faces detected at the second camera, which is the projection error e in Figure 7(b). This projection error is then compared to the distance l between the two people's faces detected in the same camera view.

Results. Figure 19 shows the CDF of the projection error e and the distance l between different faces of all the images at each captured location. We observe that the maximum projection error e is 89 pixels at these locations, and is much lower than l for most of the cases. Thus, we use this maximum e as the threshold of associating projected faces in our implementation. (§ 3.2.2), which

Resolution	Face detection	Re-identification	Compression
320 × 240	32ms	2ms	13ms
640 × 480	80ms	2ms	15ms

Table 2: Average processing delay of different vision analytic functions in Vigil using a laptop with a 2.4GHz dual-core CPU.

can lead to >98.1% re-identification accuracy at all the measured locations. The maximum projection error can be obtained by running this experiment as part of the camera calibration process. The threshold value, set during calibration process, is useful for a long period of time for static cameras. Finally, our current algorithm can fail to distinguish unique objects that are densely located (e.g., within <89 pixel). In such a scenario, we may revert back to the basic intra-cluster scheduling by picking the maximum object counts from all the ECNs. Enabling Vigil to dynamically switch between different scheduling algorithms remains as our future work.

Finally, Table 2 summarizes the processing delay of running different vision analytic functions of Vigil on a standard laptop. The latency is measured based on our video traces at two different resolutions. We observe a low latency of <80ms for all the processing functions, which enables Vigil to promptly capture objects in highly dynamic scenes.

5.4 Area coverage

The following back-of-the-envelope calculation shows that Vigil can achieve significant area coverage gains over systems that stream MPEG-4 video. Assuming that we cluster cameras in groups of four covering a 100 sq. ft. area per cluster, each camera covers an amortized 25 sq. ft. area. Assuming an available capacity of 20 Mbit/s, the status quo approach of deploying a camera stream 1 Mbit/s video (a typical MPEG-4 rate) will support 20 cameras, for a total coverage area of 500 sq. ft. But referring to Table 1, we see that the bandwidth a low activity scene actually requires is only on the order of 10 Kbit/s, while a high activity scene requires approximately 200 Kbit/s. So Vigil can function at data rates ranging from 40–800 Kbit/s per cluster, resulting in 500 clusters for low activity and 25 clusters for high activity. Consequently, Vigil covers between 2,500 sq. ft. and 50,000 sq. ft, resulting in a coverage gain of between 5× and 200× over status quo video streaming.

6. RELATED WORK

We delineate Vigil from prior work in the following three categories, i.e., cloud-based video surveillance systems, vision analytic algorithms, and video compression algorithms.

Cloud-based video surveillance systems. Many of "smart" surveillance systems today such as Dropcam [8]) rely on a wired network to upload camera feed to the cloud for vision analysis. A similar architecture has been explored in early research prototypes like IrisNet [9], Bolt [11], and S3 [29] aiming to coordinate camera sensors at a large scale. While existing systems manage to address various scalability challenges in computation and storage, their wired backhauls lead to high deployment cost and low flexibility in providing pervasive surveillance. To address this limitation, Vigil has explored a wireless video surveillance design, while leveraging specific scheduling techniques combined with edge computing technology to conserve wireless bandwidth.

Another line of work [12, 23]) aims to reduce the latency of uploading data to the cloud, by partitioning computation tasks between mobile sensors and the cloud. Odessa [23] supports interactive perception applications by dynamically offloading parts of computation tasks from mobile devices to the cloud. A recent system Gabriel [12] targets a similar class of augmented reality applications based on a cloudlet architecture, which comprises computation devices located at the edge of network to reduce network latency. In contrast to prior systems, Vigil focuses on an orthogonal problem of wireless bandwidth limitation, and tackles it with various scheduling algorithms among multiple ECNs.

Dao *et al.* [5] present a framework that suppresses uploading of redundant images from smartphones based on feature matching algorithms run on thumbnails of candidate images to be uploaded. Hu *et al.* [14] make a case for offload shaping, using image metrics such as focus, blur detection, and similarity of successive frames. These systems, developed in parallel with Vigil, differ from our work because they make uploading decisions solely on image similarity rather than the presence of objects in a user's query. In addition, Vigil advances the state-of-the-art by using re-identification algorithms running on images from multiple simultaneous cameras and schedules the uploads to match the network conditions as well as the dynamics of the scene.

Vision analytic algorithms. A large body of prior work has leveraged vision algorithms for different applications. For example, Gabriel [12] and Glimpse [13] use a Harr Cascade classifier for face detection. CarSafe [32] and WalkSafe [31] use SIFT based object detection algorithms to detect cars and lanes on the road. Finally, InSight [30] aims to detect clothing patterns by applying Wavelets transformation on the distribution of color pixels. Vigil is different from this prior work, and can give additional gains by leveraging these advanced vision algorithms to make better scheduling decisions.

Video compression algorithms. Motion JPEG (M-JPEG) [1] and MPEG-4 (H.264) [2] are two popular techniques for video compression. We choose M-JPEG in Vigil because it offers a wide range of trace between accuracy and image size (§5.3.1). It also allows the controller to decode each image independently, without waiting for other images in the same group to arrive. This effectively reduces the end-to-end latency of surveillance applications in a lossy wireless network.

Generally, video compression techniques are orthogonal to Vigil as they rely on redundancy between frames to compress the video stream. By sending only changes that maximize objects per second delivered to the cloud, Vigil removes that redundancy, reducing the efficacy of change-based video compression. State-of-the-art video compression standards like MPEG-4 and H.264 use object segmentation to identify moving objects in each frame of a video sequence [26] and cross-frame compression to eliminate the redundant image portions, *e.g.*, background scenery.

The re-identification algorithms we study are part of a family of algorithms that measure image similarity. Other algorithms include perceptual hashing techniques that encode image properties onto strings of bits [16, 19].

7. DISCUSSION

In this section we discuss various the design points of Vigil with the full hindsight of the previous sections.

Required compute resources at edge compute node. Recent trends in vision are moving toward smart cameras, thereby enabling face recognition, motion detection, and other vision tools to be implemented to an increasing extent in hardware [10, 20, 28]. Consequently, simple embedded platforms with 500 MHz–1 GHz CPU are sufficient to implement the vision analytic functions at ECNs.

Amount of ECN-local storage. The local storage at ECN retains video frames captured close in time to significant events in the video stream. MPEG-4 video at sufficient resolution for our application (320×200 or above) consumes a maximum bit-rate of between 384 and 8,000 Kbit/second (172 to 3,600 Mbytes/hour), depending on the encoding rate. The price of hard drive storage has been rapidly falling over the years, reaching USD 0.03/gigabyte in recent months [18], pricing the local storage between ½ and five US cents per hour for MPEG-4 video. Based on these figures, we expect the incremental cost for including storage for most queries requires less than USD 1.00 per ECN, adding a negligible cost to the overall bill.

Cost of distributed processing. Deploying ECNs together with cameras can inevitably increase the infrastructure cost. However, when wireless capacity is limited, the saved bandwidth by Vigil can be used to forward users' traffic, thereby recouping the cost of ECNs. To further reduce the cost, we envision that multiple cameras can connect to a single ECN to upload vision analytic functions of each connected camera to the controller. But this leads higher contention on the wireless medium between camera nodes, and hence, a more complex design of scheduling algorithms that we plan to address in future work.

Hybrid hotspot functionality. The hybrid camera-hotspot functionality can subsidize the cost of a droplet. With the cost of a camera in the range of USD 5–10, ECNs can become ubiquitous.

Choice of an object count-based metric. In Section 3.1 we motivated the use of an object count-based metric (`frameUtility`) for Vigil. We choose an object count-based metric in our design because it is a good first order approximation to frame value in a vast number of surveillance applications such as object identification and tracking. Also, our design is general enough to support any utility function by letting the controller push the utility function to the edge compute node. Therefore, the definition of utility can be modified based on the specific surveillance function performed by the system.

8. CONCLUSION

We began this work with the goal of investigating the feasibility of building a scalable wireless surveillance system, and to that end we built the Vigil system. In the process of building Vigil, we discovered several interesting design points, introducing the *ops* metric to maximize the number of query-specified objects uploaded to the cloud, and using clusters to add redundancy to camera views of a scene. We built and deployed Vigil in three different locations, and with this deployment experience our conclusion is that a system like Vigil is sufficient for most but not all surveillance use cases, but it is especially good for situations where the scene changes are infrequent. Experimental results show that Vigil allows a video surveillance system to support a geographical area of coverage between five and 200 times greater than an approach that simply streams video over the wireless network. For a fixed region of coverage and bandwidth, Vigil outperforms the default equal throughput allocation strategy of Wi-Fi by delivering up to 25% more objects relevant to a user's query. We are currently pursuing future work to even further increase Vigil's utility.

Acknowledgements

We would like to thank Jie Xiong for his help in deployment and valuable suggestions to improve the quality of our work. We are grateful to the anonymous shepherd and reviewers whose comments helped bring the paper to its final form. The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement no. 279976. Tan Zhang and Suman Banerjee are supported in part by the US National Science Foundation through awards CNS-1405667, CNS-1345293 and CNS-1343363.

9. REFERENCES

[1] Motion JPEG Video Codec. http: //www.digitalpreservation.gov/formats/fdd/fdd000063.shtml.

[2] MPEG4/H264 Video Coding Standard. http://mpeg.chiariglione.org/standards/mpeg-4.

[3] S. Biswas, J. Bicket, E. Wong, R. Musaloiu-E, A. Bhartia, and D. Aguayo. Large-scale measurements of wireless network behavior. In *ACM SIGCOMM*, 2015.

[4] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik. A real-time computer vision system for vehicle tracking and traffic surveillance. *Elsevier Transportation Research Part C: Emerging Technologies*, 6(4):271 – 288, 1998.

[5] T. Dao, A. K. Roy-Chowdhury, H. V. Madhyastha, S. V. Krishnamurthy, and T. L. Porta. Managing redundant content in bandwidth constrained wireless networks. In *ACM CoNEXT*, 2014.

[6] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. In *ACM SIGCOMM*, 1989.

[7] Doodle Labs. Broadband TV Whitespace Transceiver. http://www.doodlelabs.com/products/radio-transceivers/sub-ghz-range/470-790-mhz-tvws-100.

[8] Dropcam. Dropcam website. https://www.dropcam.com.

[9] P. B. Gibbons, B. Karp, Y. Ke, S. Nath, and S. Seshan. Irisnet: An architecture for a worldwide sensor web. *Pervasive Computing, IEEE*, 2(4):22–33, 2003.

[10] E. Gudis, L. Pullan, D. Berends, K. Kaighn, G. Van der Wal, G. Buchanan, S. Chai, and M. Piacentino. An embedded vision services framework for heterogeneous accelerators. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, pages 598–603, June 2013.

[11] T. Gupta, R. P. Singh, A. Phanishayee, J. Jung, and R. Mahajan. Bolt: Data management for connected homes. In *NSDI*, 2014.

[12] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan. Towards Wearable Cognitive Assistance. In *ACM MobiSys*, 2014.

[13] S. Han, R. Nandakumar, M. Philipose, A. Krishnamurthy, and D. Wetherall. GlimpseData: Towards continuous vision-based personal analytics. In *ACM MobiSys Workshop on Physical Analytics*, 2014.

[14] W. Hu, B. Amos, Z. Chen, K. Ha, W. Richter, P. Pillai, B. Gilbert, J. Harkes, and M. Satyanarayanan. The case for offload shaping. In *HotMobile*, 2015.

[15] IDC. The Digital Universe in 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East, Dec. 2012. http://www.emc.com/leadership/digital-universe/2012iview/executive-summary-a-universe-of.htm.

[16] S. Kozat, R. Venkatesan, and M. Mihcak. Robust perceptual image hashing via matrix invariants. In *ICIP*, 2004.

[17] D. Lowe. Object recognition from local scale-invariant features, 1999.

[18] mkomo.com. A History of Storage Cost. http://www.mkomo.com/cost-per-gigabyte-update.

[19] V. Monga and B. Evans. Perceptual image hashing via feature points: Performance evaluation and tradeoffs. *IEEE Trans. on Image Processing*, 15(11):3452–3465, 2006.

[20] V. Moshnyaga, K. Hashimoto, and T. Suetsugu. A Hardware Design of Camera-based UserâĂŹs Presence Detector. In *IEEE International conference on Systems, Man and Cybernetics*, 2008.

[21] OpenVPN Technologies. OpenVPN Virtual Networking Device. https://openvpn.net.

[22] B. Płaczek. A real time vehicle detection algorithm for vision-based sensors. In L. Bolc, R. Tadeusiewicz, L. Chmielewski, and K. Wojciechowski, editors, *Computer Vision and Graphics*, volume 6375 of *Lecture Notes in Computer Science*, pages 211–218. Springer Berlin Heidelberg, 2010.

[23] M. Ra, A. Sheth, L. Mummert, P. Pillai, D. Wetherall, and R. Govindan. Odessa: Enabling Interactive Perception Applicaitons on Mobile Devices. In *ACM MobiSys*, 2011.

[24] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies. The Case for VM-Based Cloudlets in Mobile Computing. *IEEE Pervasive Computing*, 8(4):14–23, 2009.

[25] M. Shreedhar and G. Varghese. Efficient fair queuing using deficit round robin. In *ACM SIGCOMM*, 1995.

[26] T. Sikora. The MPEG-4 video standard verification model. *IEEE Trans. on Circuits and Systems for Video Technology*, 7:19–31, 1997.

[27] Spectrum Bridge Inc. Spectrum bridge database. http://www.spectrumbridge.com/Home.aspx.

[28] T. Theocharides, G. Link, N. Vijakrishnan, M. Irwin, and W. Wolf. Embedded hardware face detection. In *IEEE Conf. on VLSI Design*, 2004.

[29] Y.-L. Tian, L. Brown, A. Hampapur, M. Lu, A. Senior, and C. Shu. IBM Smart Surveillance System (S3): Event Based Video Surveillance System with an Open and Extensible Framework. *Mach. Vision Appl.*, 19(5-6):315–327, Sept. 2008.

[30] H. Wang, X. Bao, R. R. Choudhury, and S. Nelakuditi. Insight: Recognizing humans without face recognition. In *ACM HotMobile*, 2013.

[31] T. Wang, G. Cardone, A. Corradi, L. Torresani, and A. T. Campbell. WalkSafe: A Pedestrian Safety App for Mobile Phone Users Who Walk and Talk While Crossing Roads. In *ACM HotMobile*, 2012.

[32] C.-W. You, N. D. Lane, F. Chen, R. Wang, Z. Chen, T. J. Bao, M. Montes-de Oca, Y. Cheng, M. Lin, L. Torresani, and A. T. Campbell. CarSafe App: Alerting Drowsy and Distracted Drivers Using Dual Cameras on Smartphones. In *MobiSys*, 2013.

[33] Z. Zhang, A. Scanlon, W. Yin, L. Yu, and P. L. Venetianer. Video surveillance using a multi-camera tracking and fusion system. In *M2SFA2*, 2012.