CrowdAR: Augmenting Live Video with a Real-Time Crowd

Elliot Salisbury, Sebastian Stein and Sarvapali D. Ramchurn
Agents, Interaction and Complexity Research Group
Department of Electronics and Computer Science
University of Southampton, UK
{e.salisbury,ss2,sdr}@ecs.soton.ac.uk

Abstract

Finding and tracking targets and events in a live video feed is important for many commercial applications, from CCTV surveillance used by police and security firms, to the rapid mapping of events from aerial imagery. However, descriptions of targets are typically provided in natural language by the end users, and interpreting these in the context of a live video stream is a complex task. Due to current limitations in artificial intelligence, especially vision, this task cannot be automated and instead requires human supervision. Hence, in this paper, we consider the use of real-time crowdsourcing to identify and track targets given by a natural language description. In particular we present a novel method for augmenting live video with a real-time crowd.

Introduction

Surveillance video feeds are already prevalent in today's society, used by security firms, private owners, and search and rescue workers. For example, security firms use CCTV footage to detect illegal activity and track suspects, while search and rescue workers use UAVs equipped with cameras to assess a situation and find areas of interest. However, these feeds are typically reviewed retrospectively (e.g., after a crime takes place or after the UAV flight), and while potentially helpful information can be extracted in this way, it does not enable useful actions to be taken live (e.g., catching the criminal in the act, or flying the UAV to get a better vantage point).

In order to enable more timely response, these applications would currently require constant human observation to extract the information from the live feeds. For example, security firms need to detect suspicious activity or find and track suspects from a given natural language description (e.g., a witness report describing the person and the clothes they are wearing). Likewise, search and rescue teams use UAVs to search for areas of interest that may differ from one situation to another (e.g., finding missing persons, locating damaged buildings, or identifying floating plane debris), and frequently change appearance (e.g., due to lighting conditions, cloud cover, or different architecture).

In these examples, using an expert to view these surveillance feeds is not only expensive, but the observer will eventually succumb to fatigue, as these tasks are typically tedious (CCTV) or visually taxing (tagging damaged buildings), and thus crucial information may be missed. Unless such jobs can be automated, significantly scaling up a surveillance system will not be possible.

Automating these tasks would require artificial intelligence that is capable of understanding natural language. Given that the search tasks are highly variable and the targets are different each time with few or no prior examples, natural language is the easiest way for end users to convey what they are looking for [Reid et al., 2013]. Furthermore, automation would require computer vision algorithms that can detect suspicious activity [Little et al., 2013] and reliably detect and track any number of targets in the video feed, despite ever changing conditions (e.g., lighting conditions, orientation, partial occlusion). A lot of the work in this area is still in its infancy, and automation is still a long way off [Law and von Ahn, 2011].

Real-time crowdsourcing can bridge this gap. Real-time crowdsourcing [Bernstein et al., 2011] is the process of outsourcing work to multiple, simultaneously connected online workers. In so doing, the task appears automated but retains the human intelligence required to understand context and natural language. In our case, we can use a real-time crowd to find and tag search targets in a live video feed. These annotations can then be aggregated and used to augment the live feed with the crowdsourced data, thereby reducing the cognitive load required by the observer. For example, this could be used to inform a security guard where a suspect matching a witness report is located, aiding in their immediate apprehension, or these annotations could be used to inform a UAV pilot of damage, enabling the pilot to make more effective in-flight decisions, such as surveying the areas of most damage. Furthermore, real-time crowdsourcing enables many more people to observe the live camera feed, and in so doing, they are more likely to observe and accurately identify important details (i.e., finding suspects, search targets) than a single expert could. Additionally, while a single observer may become fatigued, a real-time crowdsourced team is a tireless workforce. More specifically, individuals of the crowd that tire and stop contributing to the task, can be replaced by new crowd members. As

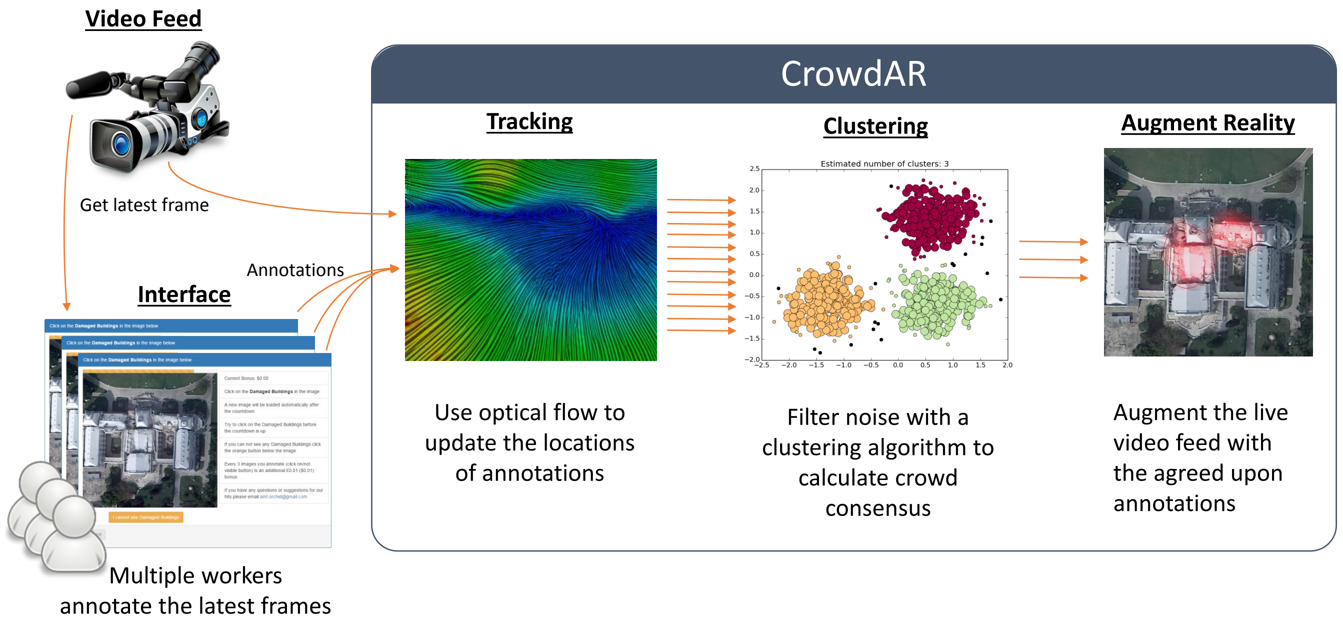

Video Feed
Get latest frame
Annotations
Interface
Multiple workers annotate the latest frames
CrowdAR
Tracking
Use optical flow to update the locations of annotations
Clustering
Estimated number of clusters: 3
Filter noise with a clustering algorithm to calculate crowd consensus
Augment Reality
Augment the live video feed with the agreed upon annotations

Figure 1: CrowdAR process diagram

such, the pool of workers is always being replenished with fresh individuals.

In order to enable this augmentation, a number of important research challenges have to be addressed. First, an open system of crowd workers provides no guarantee on worker reliability, so we need multiple workers to provide simultaneous input such that their tags can be aggregated and filtered in real-time. Therefore, aggregating this input in real-time requires a process that has a low computational cost. Second, as there is an inevitable latency between crowd worker input and the live video feed, mechanisms must be developed to reduce the visible effects of this lag. Finally, the aggregated annotations can then be used to augment the video feed in real-time. Existing work on real-time crowdsourcing has streamed live CCTV to the crowd, asking them to alert an expert when suspicious activity was detected [Tewksbury, 2012; Trottier, 2014]. Other work has explored systems that stream live video from phones, surveillance cameras, and robots, and encourage active interaction with members of the crowd [Lasecki et al., 2013b; 2011; 2013a; Laput et al., 2015; Salisbury, Stein, and Ramchurn, 2015]. However, these systems cannot track targets, given by a natural language description, in a live video feed.

To address these challenges we introduce a novel framework for augmenting live video using a real-time crowd. Specifically, we advance the state of the art in the following ways:

1. We present Crowd Augmented Reality, or CrowdAR, an application that augments a live video feed with tags based on a natural language description using a real-time crowd.

2. We describe a novel method for calculating consensus in real-time from tags of the same search target given at spatially and temporally different locations in a video feed.

3. We test this system using a real crowd hired from Amazon Mechanical Turk, on real surveillance footage and simulated UAV footage. In total, 349 crowd members took part, with up to 28 simultaneous users. We show that CrowdAR is more accurate at reporting the locations of search targets, and identifies up to twice as many search targets when compared to a single observer.

The rest of this paper is structured as follows. Next we discuss the related work. Then we describe the live video tagging system. Finally we evaluate the system in two different scenarios and present the results.

Related Work

Since the rise of crowdsourcing, the research community has put considerable effort into developing tools for annotating large datasets of images. For example, LabelMe [Russell et al., 2008] is an online tool to annotate images for use in computer vision research, and the ESP Game [von Ahn and Dabbish, 2004] is a similar tool that used gamification to engage crowd workers to label images. Later this work was expanded to locate those labels in the image [von Ahn, Liu, and Blum, 2006]. These online platforms have a number of applications, from labelling computer vision datasets [Deng et al., 2009], to digital humanitarian aid [Lin et al., 2013; Meier, 2014].

Subsequently, extending these platforms to the problem of video annotation requires similar techniques to that of still images, but performing the same procedure per frame can be time-consuming and wasteful. Videos have some consistency between frames and this can be exploited to intelligently interpolate annotations between frames [Vondrick, Patterson, and Ramanan, 2013].

Using the crowd to annotate video is not a new concept. LabelMe extended their platform to handle precise video annotation [Yuen et al., 2009], and likewise, work has been done to use the crowd to annotate regions of interest in video playback [Carlier et al., 2010].

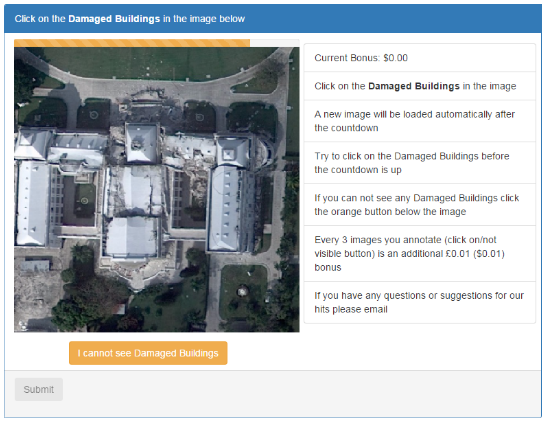
However, these past approaches are applied to offline video, and are not suitable for our application, as we require annotations to be done live, while the video feed is playing. Furthermore, for our applications we typically do not need the exact shape of the object, so providing a high fidelity annotation (i.e., a precise outline) is unnecessary, especially since simply requesting the location of the target in the video from many crowd workers is enough to calculate a rough outline.

Previous work on real-time crowdsourcing has broadcast live CCTV footage online to detect illegal immigration across the Mexico-US border [Tewksbury, 2012], or to detect shoplifting in convenience stores [Trottier, 2014]. These approaches used an alert model, which asks the crowd to continuously watch the feed and earn money when they press a button to alert an expert that something illegal is currently happening. This encouraged a high number of false positives, and a large number of workers to quickly abandon the task [Dunphy et al., 2015].

Instead, a more engaging approach would be to encourage active participation of the workers. There are a number of existing applications that do this. These applications use active participation of real-time crowds observing video streams to perform a variety of similar tasks. For example, Chorus [Lasecki et al., 2013b] uses workers to observe live video captured from a cellphone and converse with the person behind it. Legion [Lasecki et al., 2011] and CrowdDrone [Salisbury, Stein, and Ramchurn, 2015] observe a live feed from a robot and frequently provide input to a shared control interface.

More relatedly, a number of applications have used real-time crowds as a human-driven sensor, providing a stream of data about a video feed. LegionAR [Lasecki et al., 2013a] uses the crowd to provide activity labels for actors in a live video stream, whereas Glance [Lasecki et al., 2014] codes the timings of behavioural cues. However, these approaches only work on known or detectable actors, and do not provide the location of the targets in the scene. Likewise, Zensors [Laput et al., 2015] uses a real-time crowd as a human sensor. Zensors also uses an automated machine learning approach, to detect when a video stream has changed significantly and needs new crowd judgement. Furthermore they attempt to learn features in the image that correlate with the crowd judgements to eventually remove the need for hiring a crowd. However, Zensors only provides a number of datatypes (e.g., YesNo, Number, Scale) and aggregates crowd opinion using a voting methodology. Thus, their approach works on simple discrete datatypes, but given that the location of a search target is not a single discrete position, but rather a continuous area, changing in size, shape and location over time, their approach does not fit our needs.

In the following section, we discuss a system that engages simultaneously connected workers in annotating frames of a live video feed. Thus, enabling us to augment the video feed shown to an expert.


Click on the **Damaged Buildings** in the image below

Current Bonus: $0.00

Click on the **Damaged Buildings** in the image

A new image will be loaded automatically after the countdown

Try to click on the Damaged Buildings before the countdown is up

If you can not see any Damaged Buildings click the orange button below the image

Every 3 images you annotate (click on/not visible button) is an additional £0.01 ($0.01) bonus

If you have any questions or suggestions for our hits please email

Figure 2: The interface presented to the real-time crowd workers.

CrowdAR

In this section we outline the architecture of CrowdAR, a system for augmenting live video with the annotations of a real-time crowd, and discuss the trade-offs we considered during its design. CrowdAR consists of four key components (see Figure 1). The first of these is a web interface that a real-time crowd interacts with, hired from any crowdsourcing platform (e.g., Amazon Mechanical Turk[1]). The second is a tracking system that continously moves a worker's previously provided tags to their most likely location in the current frame of the live feed. The third, a clustering algorithm, aggregates workers' input to reduce noise. Finally, the fourth component augments the live video with the clusters. These components will be described in more detail through the remainder of this section. Thus, CrowdAR addresses three main challenges. The first challenge is to create an **interface** suitable for eliciting rapid input from a real-time crowd. The second is in **reducing perceived latency** of the annotations used to augment the video, and the third challenge is to filter the noise of unreliable crowd workers by **calculating consensus**. We will cover these in turn now.

Interface

Given that CrowdAR augments live video, we require workers' annotations to be received, and a consensus to be reached, in real-time. Thus we require multiple workers viewing the same video feed simultaneously. Therefore, to ensure that we have a simultaneous crowd from the start of the video feed, we use the retainer model [Bernstein et al., 2011]. The retainer model is a method of pre-hiring a crowd and alerting them to all participate simultaneously when the task is ready. Once a crowd has been pre-hired, the video stream is started. The retained workers are alerted that the task is ready, and they are transferred to a web interface,

[1]`https://www.mturk.com`

where they can annotate the video feed, tagging search targets from a given description (e.g., guy in striped sweater, damaged buildings).

A naive implementation of the web interface (used by previous examples [Tewksbury, 2012; Trottier, 2014]) is to simply stream the live video to the workers. Eliciting real-time input would require workers to click and drag their mouse over the search target, continuously streaming their annotations back to CrowdAR. The benefit of this approach is that worker annotations would have the least latency, and are thus more relevant to the live frame. However, workers can only track a single target, and so this approach cannot address the need to track multiple targets.

Instead we developed an interface that shows a worker the most recent frame from the video stream (see Figure 2), the worker is given a short period of time to view the frame and identify if and where the search target is present in the frame. After the time is up, a new frame is loaded. This requires less bandwidth, allows workers to annotate multiple targets per frame, and because the targets are not moving it is easier for workers to accurately move their mouse and click where desired, and thus the input is likely more accurate. Furthermore, we add a button to the interface to indicate that the worker cannot find the target in the frame. This enables us to identify whether a lack of annotation was due to this, or due to worker inactivity.

This still frame interface works as follows[2]. The crowd worker, w, is presented with a still frame, f_t, which is the most recent frame of the video at time t. Above the frame, the worker observes a bar counting down d seconds until the next frame is presented. The worker may then click on the frame any number of times, indicating where they believe the search target to be. These annotations are denoted by $\alpha_{w,t}$.

$$\alpha_{w,t} = \{a_1, a_2, \ldots, a_n\} \tag{1}$$

Where a_i is an x, y coordinate in the frame. After the countdown, all of the worker's annotations, $\alpha_{w,t}$, are transmitted to CrowdAR for further processing. If the worker does not tag anything in this frame then $\alpha_{w,t} = \emptyset$.

Unfortunately, this interface now introduces significant delay between the current frame of the video and the received tags. In what follows, we describe methods to help rectify this, but if the value of d is too long, or the search targets move too fast, then the targets will be out of frame of the video before we even receive the worker's tags.

Reducing Perceived Latency

There is always some unavoidable delay between the received annotations, $\alpha_{w,t}$, for frame f_t, and the current frame of the live video feed, $f_{t'}$.

$$t' - t = d + l \tag{2}$$

Where d is the duration of the still frame task, and l is the delay caused by network latency. This means that the annotations used to augment the video may no longer be relevant or in the correct location. We require a mechanism to reduce this latency such that the augmentations remain relevant to the live feed. Thus, CrowdAR uses two techniques, preloading and tracking, to help rectify this inherent lag.

Preloading Just in time preloading is used so that the worker does not need to wait l seconds for the next frame to load. We measure the network latency, l, between the CrowdAR system and each worker. Thus, the worker's interface should start preloading the next frame $d - l$ seconds into the countdown. This ensures that the worker is always viewing the most recent frame, without needing to wait for it to download, thus increasing their input rate.

Tracking Computer vision techniques are used to further reduce the effects of network, l, and interface latency, d. Specifically, CrowdAR uses dense optical flow [Farnebäck, 2003] to track workers' annotations through the video feed, ensuring that they are tracking the movement of the search target. Optical flow is a technique to determine the movement between two frames at the pixel level. Calculating the movement of pixels between every frame allows us to iterate through from the frame that was annotated to the most current live frame, updating the tag's location. Optical flow performs best on feature rich video feeds as in our example scenarios, but may fail to track if the target becomes occluded. This is rarely an issue in the case of aerial footage, but CCTV footage may be affected depending on the placement of the camera. To rectify this, and to reduce the effects of noise introduced by tracking, we timeout tracked annotations after 10 seconds. Workers frequently click on the same target, ensuring that not all annotations timeout.

Let O be a function that transforms an annotation, a_i, from its location in frame f_t to its tracked location in the current frame, $f_{t'}$.

$$O(a_i, f_t, f_{t'}) \rightarrow a'_i \tag{3}$$

Then we propagate all of the workers' annotations for frame f_t to the live frame, $f_{t'}$, giving the following.

$$\alpha'_{w,t} = \{O(a_i, f_t, f_{t'}) | a_i \in \alpha_{w,t}\} \tag{4}$$

This method is then applied to all of the worker's annotations.

$$A'_w = \bigcup_{t'-10<t<t'} \alpha'_{w,t} \tag{5}$$

Calculating Consensus

Because CrowdAR is an open platform, workers of varying skill may participate. Thus it is possible that some workers will prove unreliable and provide incorrect input, either accidentally or maliciously. In order to filter this noise, we require multiple workers' input with which we can calculate a consensus.

To that end, once the annotations are tracked to the most current frame, we need to identify if there is indeed something relevant worth augmenting the video with, or if the annotation can be considered just noise. An annotation is considered relevant if multiple crowd workers have tagged roughly the same location in the frame. A clustering algorithm is used to calculate the crowd's consensus. A cluster

[2]Watch an example of CrowdAR's interface here: `https://vimeo.com/132924336`

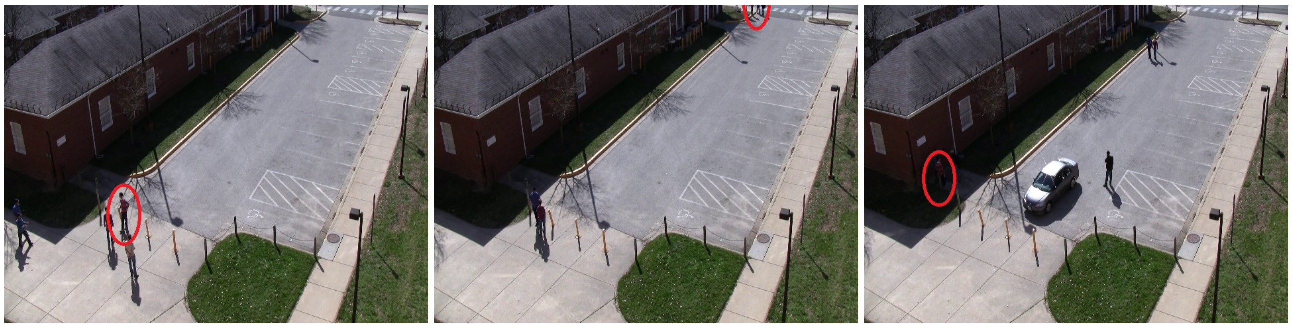

(a) Example Frame (b) Partial Occlusion (c) Poor Illumination

Figure 3: Example frames from the CCTV task, the search target is highlighted with a red circle.

with at least 3 unique workers' annotations would be considered a verified search target. This value can be set by the system designer and will likely depend on the domain and crowd size. More sophisticated techniques for choosing this value have been left for future work.

CrowdAR uses DBSCAN [Ester et al., 1996] as the clustering algorithm because it does not require a predetermined number of clusters (i.e., we don't know how many targets are in a scene), and it can detect clusters of arbitrary shapes (i.e., we don't know the shape of our targets). In particular, clustering allows us to determine a rough size and shape of the target, which, in turn, enables us to better augment the live feed. However, should multiple targets be close to, or even overlap each other, the clustering may consider them as only a single target. In our case of improving surveillance feeds, this is not an issue as an expert is still in the loop.

Let C be a function that calculates the set of clusters from a set of annotations.

$$C(\alpha') \rightarrow \{c_1, c_2, \ldots, c_n\} \tag{6}$$

Where c_i is a set of x, y coordinates in the frame. Thus, $\mathcal{C}$ is the set of clusters from each workers', $w \in W$, complete tracked annotations, A'_w.

$$\mathcal{C} = C(\bigcup_{w \in W} A'_w) \tag{7}$$

These verified annotations can now be used to augment the live video feed. An expert who views the feed could see the cluster overlaid onto the original frame.

$$\Lambda(f_{t'}, \mathcal{C}) \rightarrow \mathbf{f_{t'}} \tag{8}$$

Where Λ is the augmentation function, that renders the clusters, $\mathcal{C}$ onto the current frame, $f_{t'}$. The output, $\mathbf{f_{t'}}$, is then shown to the expert. The implementation of Λ could be done in any number of ways. For example, drawing a bounding box or ellipse around the cluster. The chosen style of augmentation would depend on the end-user. In the case of CrowdAR, we overlay a heatmap generated from each cluster[3,4] (see Figure 7d).

Next we evaluate CrowdAR to understand how such a system, and the crowd workers, perform under varying circumstances.

Empirical Evaluation

Here we evaluate the performance of CrowdAR, described in the previous section. This system works with any video feed, be that live or the playback of pre-recorded footage. The following experiments use pre-recorded video for the purpose of repeatability. CrowdAR was designed to augment surveillance footage, intending to reduce the cognitive load of an observer. With that in mind, we evaluate CrowdAR on two types of surveillance footage, CCTV and aerial surveillance.

Surveillance Footage

We use two different datasets to test the capabilities of this system. First, we use real CCTV footage[3] to investigate CrowdAR's ability to detect a search target given a natural language description (e.g., from a witness report). This is challenging for a single observer as the task is tedious. Second, we use aerial footage[4] to investigate CrowdAR's ability to track multiple search targets (e.g., damaged buildings) in a much more challenging setting. This task is visually taxing and a single worker is unlikely to observe every detail.

CCTV This is a highly tedious task, where, in current practice, experts have to watch multiple monotonous video feeds for prolonged periods of time. CrowdAR's approach to CCTV scales better than current practice. CrowdAR highlights the location of targets in the video feed, thus reducing the need for constant expert observation. Instead, experts could multitask more effectively, only needing to briefly observe the CrowdAR output and note the current location of the tracked suspects.

To be able to measure the accuracy of CrowdAR, in a real-world setting, we require video footage annotated with ground truth to compare the crowd performance against. Hence, we used the VIRAT Video dataset [Oh et al., 2011], which provides labelled surveillance footage of real world scenes.

In this experiment we test the crowd's ability to annotate a video feed given a natural language description. The video

[3]Watch an example of CrowdAR's augmented CCTV footage here: `https://vimeo.com/132313166`

[4]Watch an example of CrowdAR's augmented aerial footage here: `https://vimeo.com/130873053`

contains a number of people waiting in a parking lot, and the crowd workers are asked to identify the search target from a colloquial description, "guy in the striped sweater". This individual enters the video after 10 seconds, and stays visible for the remaining 14 minutes. Watching a fairly uneventful CCTV feed is a tedious task, and a single observer is likely to lose focus. This still remains a challenging problem for computer vision, as the target moves around the scene into areas of shade where the stripes on his top are no longer visible (large changes in illumination), or to partially occlude himself from the shot, such that only his legs are visible (see Figure 3).

Note in practice, we would envisage tedious tasks like this CCTV setting to be working in tandem with an automated process (similar to Zensors [Laput et al., 2015]), in which a computer vision algorithm is used to detect a scene change and request a crowd to check if a target is present, and to then alert an expert.

Aerial Surveillance This is a very visually taxing task, where, in current practice, expert pilots make flight decisions based on what they can observe. However, due to the nature of aerial surveillance, the video feed may move quickly, leaving little time to observe crucial information and make informed decisions. CrowdAR could enable the pilots to make more informed decisions, by highlighting search targets that expert pilots may miss.

Due to the difficulty in obtaining live UAV footage of a disaster scene, we instead generate a realistic video from high resolution aerial imagery. The imagery was sourced from Google Crisis Response[5], and contained imagery gathered of Port Au Prince shortly after the 2010 Haiti earthquake. We choose an arbitrary flight path over the Haitian capital and record a video scanning over the imagery, replicating the movement of an aircraft. The ground truth data is sourced from a UNITAR survey that identified the location and the destruction of every building in Haiti [UNOSAT, 2010]. From this, an outline of all 266 damaged building on the flight path was manually created (see Figure 4).

The crowd was then asked to click on any "damaged building" they see. There are often many damaged buildings in the frame, and a single observer would struggle to identify them all. This is a challenging problem for computer vision to solve, as the damage appears in many forms and cloud cover sometimes obscures the buildings below.

Evaluation Metrics

The overall aim of CrowdAR is to assist an expert in doing their task more effectively. However, measuring expert performance is highly domain-specific and can often be difficult, especially in a repeatable, controlled experiment (e.g., preventing a crime, or achieving good situational awareness in disaster response). Instead, In order to evaluate the performance of CrowdAR, we take a domain-agnostic approach and focus on the quality of the produced annotations, using objective metrics. We compare the aggregated output of CrowdAR to that of a single worker. Both involve

[5]www.google.org/crisisresponse

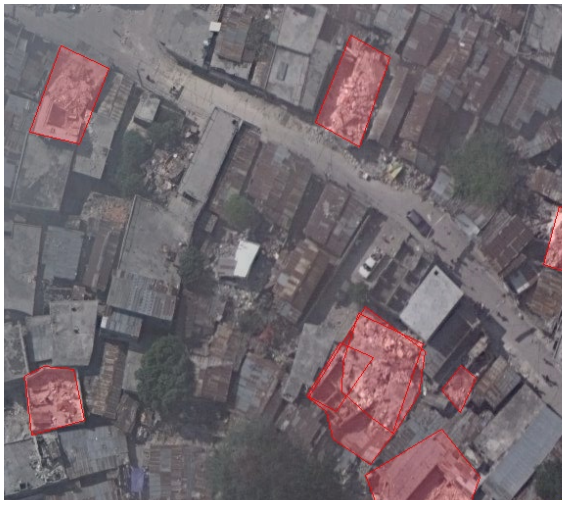


Figure 4: Example frame and ground truth (in red) from the Aerial Surveillance task.

non-experts, but there is no evidence as yet that aggregating many novices' input results in better performance in CrowdAR's real-time setting. Thus, by comparing to a single worker, we show that aggregation is necessary to achieving a high performance. In so doing, we define the following metrics:

- **Precision:** The fraction of true positives, over all classifications, both true and false positives (i.e., $\frac{TP}{TP+FP}$). Put simply, in the case of worker clicks, it is the percentage of their clicks that they correctly identified. For CrowdAR, it is the percentage of clusters that correctly highlight a search target, for every frame.
- **Recall:** The fraction of true positives, over all the possible search targets, both true positives and false negatives (i.e., $\frac{TP}{TP+FN}$). In more detail, in the case of worker clicks, it is the percentage of search targets they correctly identified per frame viewed. For CrowdAR it is the percentage of search targets correctly highlighted by the clusters per frame.

  The actual numerical value of recall is not too important, and relates more to the difficulty of the dataset. Instead we are comparing the difference between individual workers' recall, and CrowdAR's recall.
- **Worker Responses Over Time:** The proportions of different response types given by crowd workers over the duration of the video. We use this to investigate how worker performance reacts to events in a video feed. We measure four types of responses, correct clicks (i.e., true positive), incorrect clicks (i.e., false positive), when a worker indicates they cannot see the target, and when a worker views the frame but does not provide feedback.

Experimental Settings

Throughout the experiments, we set the duration of the still frame interface, d, to 2 seconds. This was chosen, because preliminary experimentation showed this to be a good value to allow for network latency (1.5 seconds on average in our experiments), while balancing the throughput of annotations with their quality. In practice, this duration can be adjusted by the system designer to take into account the complexity of the task, desired quality and network latency.

The total number of active crowd members (those who clicked at least once) varied from 8 to 28 per experiment, there were 349 unique workers in total, and on average, the system received 3 annotations from unique crowd members every second. Workers were paid \$0.01 for every 3 frames they annotated, a milestone bonus of \$0.01 was offered for every 20 frames, encouraging workers to keep going. This works out at a maximum hourly wage of \$6.90 per worker, should the worker annotate every frame they were presented.

In order to determine the effect that crowd size has on precision and recall, we computed these metrics for 100 samples of crowd input for each crowd size (from 3 to 20). Our results found no statistical significance in these metrics, but this was partly due to high levels of variance in the low crowd sizes. This variance decreases as more crowd members are present. In practice, we found that the appropriate crowd size is dependent on the latency of the workers and the video being annotated.

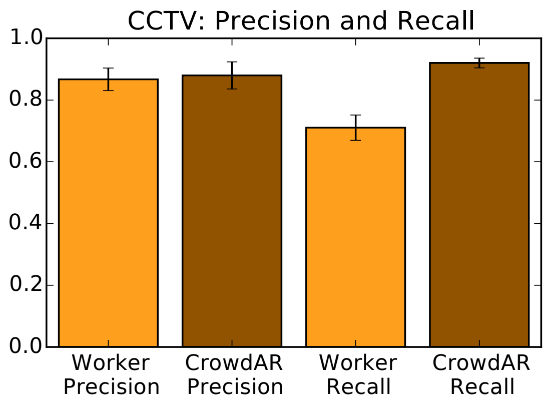
Results

We ran 10 experiments for each video feed. The real-time nature of these experiments is prohibitive to running large numbers of repeats.

Precision and Recall The performance of both the Aerial surveillance and CCTV footage can be seen in Figure 5. For the CCTV footage, a single worker's clicks are on average 87% precise, while CrowdAR's clusters are 88% precise, which is not a significant improvement. However, a single worker on average only recalls (i.e., identifies) the target in 71% of the frames they observe. Whereas CrowdAR, because it has many people observing the same feed, performs significantly[6] better at keeping track, recalling the target in 92% of frames.

CrowdAR's main performance improvements can be seen on more visually challenging video feeds, such as the Aerial footage. In the Aerial footage, we can see that despite the lower precision of individual crowd workers' annotations, 76%, CrowdAR's clustering algorithm is able to filter the noise, and so 91% of the reported damaged buildings are correct, significantly better than any single worker. Furthermore, while a single worker on average only identifies 5% of the damaged buildings observed, CrowdAR is able to correctly identify 11% of the buildings, a significant improvement and more than doubling the performance of a single observer. The recall values for the aerial footage are quite low compared to that of the CCTV footage. This is due to the difficulty of the dataset, while the CCTV footage has only a single target that remains mostly visible for the duration of the video. In contrast, the Aerial dataset has a large

[6]All results reported as significant were confirmed with t-tests at $p < 0.05$ and all figures show 95% confidence intervals.



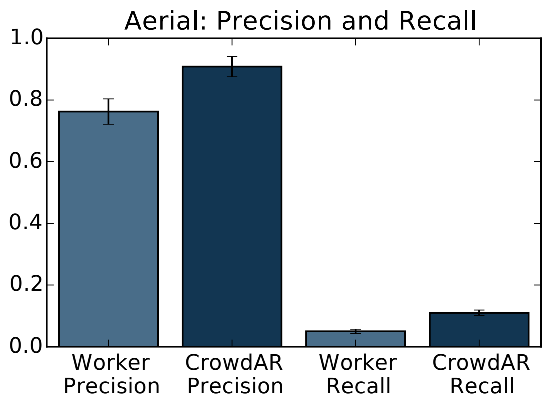


Figure 5: The precision and recall of workers clicks and the tracked clusters of CrowdAR.

number of damaged buildings (i.e., 266), and some show little sign of damage from the air.

Responses Here we measured the responses of workers during a video stream, and their change over time, seen in Figure 6. Notably, these graphs show that the number of workers providing input corresponds to the difficulty of the search task. For example, the CCTV footage has a near constant input rate, whereas the Aerial footage is a visually difficult challenge, and we notice a significant difference in response rate. Workers are busy searching the footage, and it is more likely the countdown will timeout before the worker can provide an input.

Furthermore, for the Aerial footage we observe spikes of correct tags (A1,A3,A4) when large easy to identify buildings enter the frame, and a prolonged period of multiple damaged buildings at A2. Similarly, in the CCTV footage we observe a sudden drop in correct clicks at C1, which corresponds to when the search target is partially occluded (Figure 3b). Notably, the input rate does not change, and instead many workers indicate that they cannot see the target, while others begin tagging the wrong person. As the target slowly walks back into frame, C2, the correct annotations slowly begin to rise as well.

In the aerial footage, and part way through the CCTV

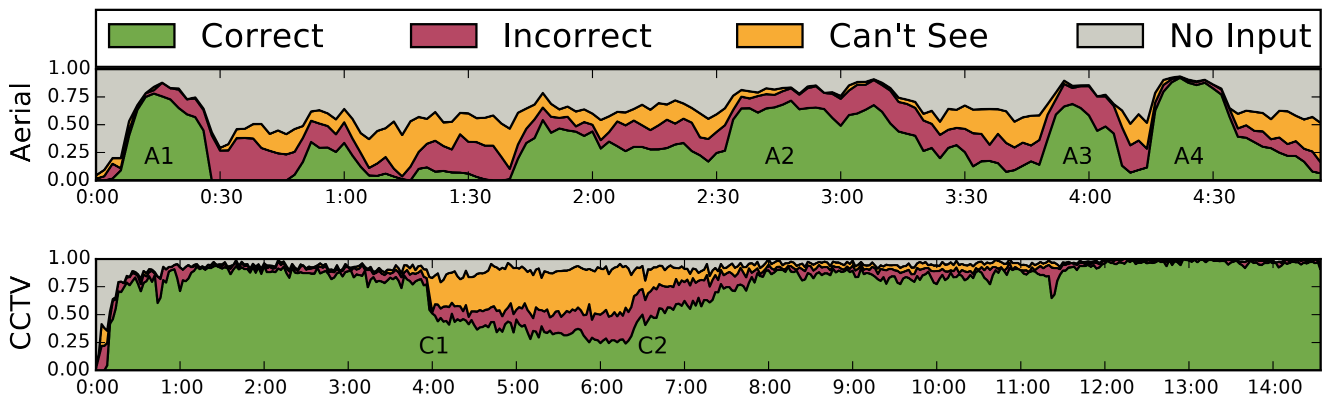


Figure 6: Workers' Inputs over the duration of the video feed.

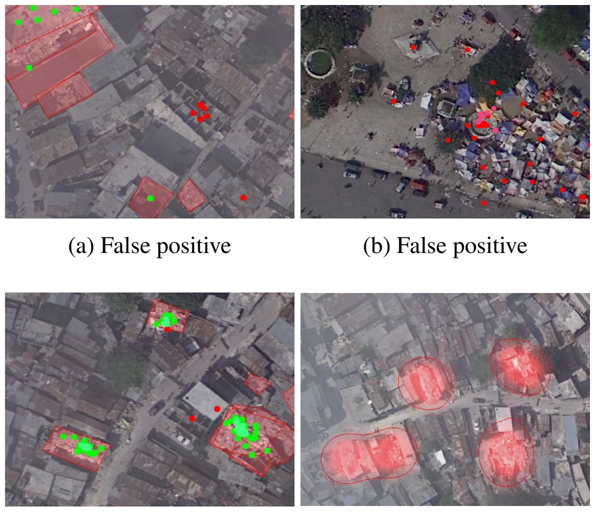


Figure 7: Examples of true and false positives, worker clicks are colored in green (correct) or red (incorrect). In (d) we can see an example of the augmented output, with identified clusters colored in red.

footage, there is a sizable percentage of incorrect clicks. From visual inspection, these tend to arise not because of worker maliciousness, but because of honest mistakes. For example, workers incorrectly identify a different individual also wearing a sweater (seen in lower left of Figure 3b), or tag garden walls that may look like a missing roof, or gravel that may be similar to rubble (see Figure 7a and 7b respectively).

Conclusions and Future Work

CrowdAR annotates more search targets, and with more precision than a single worker could achieve. This performance can help augment live video and reduce the cognitive load of the observer. This could be used to inform an expert where search targets are located, enabling more effective and timely decision making. We show from our results that tracking and clustering real-time input data from the crowd is suitable for augmenting live video. Furthermore, there are a number of future applications for these annotations. For example, providing input to an autonomous path planner onboard a UAV, or in filtering large segments of video with little interest.

CrowdAR could be improved in a number of ways. For example, there are other techniques for tracking workers' tags in a video, such as SIFT, which could be used to detect features on, or around the tag, and look for similar features in the most recent frame [Lowe, 2004; Dalal and Triggs, 2005]. This would improve tracking in scenarios with occlusion, and improve the precision of CrowdAR.

Additionally, the interface speed could be varied per worker, depending on a specific worker's latency and reaction times. This would result in more effective utilisation of workers, providing CrowdAR with data in a more timely manner.

Furthermore, we can see from the difference in worker response rates for easy (CCTV) and difficult (Aerial) footage, that this could be used to represent a measure of confidence from the worker, which could be used to filter noise, further improving precision. Likewise, distributing smaller subsections of the footage amongst the crowd would reduce the visual difficulty of the search task, and thus increase the recall.

Acknowledgments

This research was undertaken as part of the ORCHID project funded by the EPSRC (EP/I011587/1).

References

Bernstein, M. S.; Brandt, J.; Miller, R. C.; and Karger, D. R. 2011. Crowds in Two Seconds: Enabling Realtime Crowd-powered Interfaces. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, 33–42. New York, NY, USA: ACM.

Carlier, A.; Charvillat, V.; Ooi, W. T.; Grigoras, R.; and Morin, G. 2010. Crowdsourced automatic zoom and scroll for video retargeting. In *Proceedings of the International Conference on Multimedia*, MM '10, 201–210. New York, NY, USA: ACM.

Dalal, N., and Triggs, B. 2005. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, 886–893 vol. 1.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 248–255.

Dunphy, P.; Nicholson, J.; Vlachokyriakos, V.; and Briggs, P. 2015. Crowdsourcing and CCTV : the Effect of Interface , Financial Bonus and Video Type.

Ester, M.; Kriegel, H.-P.; Sander, J.; and Xu, X. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, 226–231.

Farnebäck, G. 2003. Two-Frame Motion Estimation Based on Polynomial Expansion. In Bigun, J., and Gustavsson, T., eds., *Image Analysis*, volume 2749 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg. 363–370.

Laput, G.; Lasecki, W. S.; Wiese, J.; Xiao, R.; Bigham, J. P.; and Harrison, C. 2015. Zensors: Adaptive, Rapidly Deployable, Human-Intelligent Sensor Feeds. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, 1935–1944. New York, NY, USA: ACM.

Lasecki, W. S.; Murray, K. I.; White, S.; Miller, R. C.; and Bigham, J. P. 2011. Real-time Crowd Control of Existing Interfaces. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, 23–32. New York, NY, USA: ACM.

Lasecki, W. S.; Song, Y. C.; Kautz, H.; and Bigham, J. P. 2013a. Real-time crowd labeling for deployable activity recognition. *Proceedings of the 2013 conference on Computer supported cooperative work - CSCW '13* 1203.

Lasecki, W. S.; Thiha, P.; Zhong, Y.; Brady, E.; and Bigham, J. P. 2013b. Answering Visual Questions with Conversational Crowd Assistants. In *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '13, 18:1—-18:8. New York, NY, USA: ACM.

Lasecki, W. S.; Gordon, M.; Dow, S. P.; Bigham, J. P.; Koutra, D.; Jung, M. F.; Dow, S. P.; and Bigham, J. P. 2014. Glance: Rapidly Coding Behavioral Video with the Crowd. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, 551–562. New York, NY, USA: ACM.

Law, E., and von Ahn, L. 2011. Human computation. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 5(3):1–121.

Lin, A.-M.; Huynh, A.; Barrington, L.; and Lanckriet, G. 2013. Search and discovery through human computation. In Michelucci, P., ed., *Handbook of Human Computation*. Springer New York. 171–186.

Little, S.; Jargalsaikhan, I.; Clawson, K.; Nieto, M.; Li, H.; Direkoglu, C.; O'Connor, N. E.; Smeaton, A. F.; Scotney, B.; Wang, H.; and Liu, J. 2013. An information retrieval approach to identifying infrequent events in surveillance video. In *Proceedings of the 3rd ACM Conference on International Conference on Multimedia Retrieval*, ICMR '13, 223–230. New York, NY, USA: ACM.

Lowe, D. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 60(2):91–110.

Meier, P. 2014. *Digital Humanitarians: How Big Data Is Changing the Face of Humanitarian Response*. Crc Press. 66–71.

Oh, S.; Hoogs, A.; Perera, A.; Cuntoor, N.; Chen, C. C.; Lee, J. T.; Mukherjee, S.; Aggarwal, J. K.; Lee, H.; Davis, L.; Swears, E.; Wang, X.; Ji, Q.; Reddy, K.; Shah, M.; Vondrick, C.; Pirsiavash, H.; Ramanan, D.; Yuen, J.; Torralba, A.; Song, B.; Fong, A.; Roy-Chowdhury, A.; and Desai, M. 2011. A Large-scale Benchmark Dataset for Event Recognition in Surveillance Video. In *IEEE Computer Vision and Pattern Recognition*, number 2, 527–528.

Reid, D.; Samangooei, S.; Chen, C.; Nixon, M.; and Ross, A. 2013. Soft biometrics for surveillance: an overview. *Machine learning: theory and applications. Elsevier* 327–352.

Russell, B. C.; Torralba, A.; Murphy, K. P.; and Freeman, W. T. 2008. LabelMe: a database and web-based tool for image annotation. *International journal of computer vision* 77(1-3):157–173.

Salisbury, E.; Stein, S.; and Ramchurn, S. 2015. Real-time opinion aggregation methods for crowd robotics. In *Autonomous Agents and Multiagent Systems (AAMAS 2015)*.

Tewksbury, D. 2012. Crowdsourcing homeland security: The Texas virtual borderwatch and participatory citizenship. *Surveillance and Society* 10:249–262.

Trottier, D. 2014. Crowdsourcing CCTV surveillance on the Internet. *Information, Communication & Society* 17(5):609–626.

UNOSAT. 2010. Haiti earthquake 2010: Remote sensing based building damage assessment data.

von Ahn, L., and Dabbish, L. 2004. Labeling Images with a Computer Game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, volume 6 of *CHI '04*, 319–326. New York, NY, USA: ACM.

von Ahn, L.; Liu, R.; and Blum, M. 2006. Peekaboom: a game for locating objects in images. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, 55–64. New York, NY, USA: ACM.

Vondrick, C.; Patterson, D.; and Ramanan, D. 2013. Efficiently Scaling up Crowdsourced Video Annotation. *International Journal of Computer Vision* 101(1):184–204.

Yuen, J.; Russell, B.; Liu, C.; and Torralba, A. 2009. LabelMe video: Building a video database with human annotations. In *Computer Vision, 2009 IEEE 12th International Conference on*, 1451–1458.