A Negotiation Protocol with Conditional Offers for Camera Handoffs

Wiktor Starzyk
University of Ontario Institute of Technology
Oshawa, ON, Canada
wiktor.starzyk@uoit.ca

Faisal Z. Qureshi
University of Ontario Institute of Technology
Oshawa, ON, Canada
faisal.qureshi@uoit.ca

ABSTRACT

This paper explores the idea of conditional offers during camera handoff negotiations. In a departure from contract-net inspired negotiation models that have been proposed for camera handoffs, the current scheme assumes that each camera maintains the state of its neighbouring cameras. To this end, this paper presents a new short-term memory model for maintaining a camera's own state and the state of its neighbouring cameras. The fact that each camera is aware of its surrounding cameras is exploited to generate conditional offers during handoff negotiations. This can result in multiple rounds of negotiations during a single handoff, leading to successful handoffs in situations where one of the cameras that is being asked to take on one more task is unable to take on a new task without relinquishing an existing task. The results demonstrate the advantages of the proposed negotiation model over existing models for camera handoffs.

General Terms

Smart cameras, negotiation models, camera handoff

Keywords

Smart cameras, camera handoff, negotiations models, counter-offers

1. INTRODUCTION

The need for security in public spaces and the plummeting costs associated with camera installations are pushing the growth of video surveillance. Surveilling large environments necessitates the use of multiple cameras, as no single camera is able to observe the entire scene. As the number of cameras grows, it becomes impractical for a human operator to monitor all the video feeds. Consequently, over the last several years, there is much work on camera networks capable of providing video coverage of extended spaces with minimal human intervention. These networks comprise smart cameras: visual sensors with onboard processing and storage and the ability to communicate with other sensor nodes in the vicinity.

Camera control and coordination are important research areas within smart camera networks. Specifically, how best to coordinate the available cameras to carry out the observation tasks? Ideally the control and coordination strategy should be distributed, as centralized schemes do not scale and defeat the *raison d'êter* of smart camera networks. Game-theory [1] based and contract-net [2, 3] inspired auction based schemes have been proposed to address the issue of camera grouping and handoff in smart camera networks. This paper develops a new negotiation protocol for camera handoffs. The proposed model has two novel features. First, it describes a short-term memory model, which enables each camera to maintain the state of its neighbouring cameras. Second, it implements a mechanism for conditional offers, which allows cameras to go through multiple rounds of negotiations during each handoff task. The ability to propose conditional offers leads to successful hand-

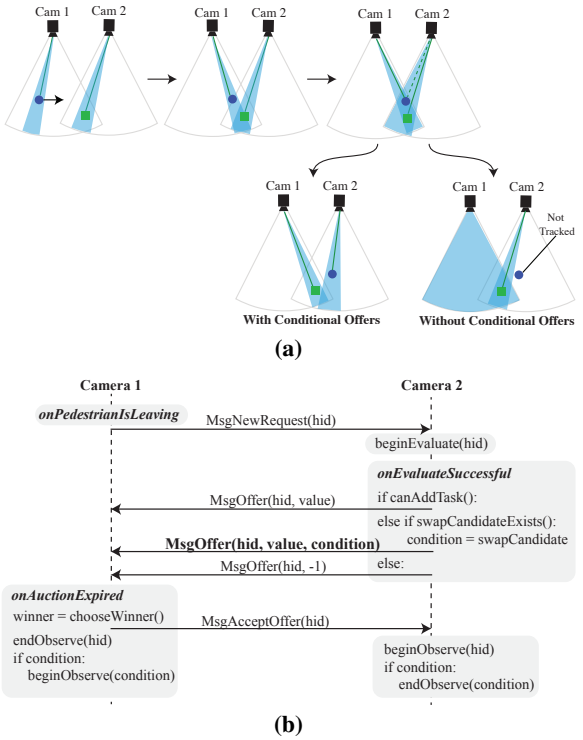


Figure 1: (a) Conditional offers allow handoffs to occur even if a camera is already at is tracking limit. They allow a camera to send an offer with a condition that ensures the auctioning camera takes over a task from the bidding camera. (b) Messages exchanged during the handoff operation. Notice that camera 2 proposes a conditional offer suggesting a task swap.

ICDSC '14 Venice, Italy

offs in situations where a camera that is being asked to take on a new task is unable to do so unless it first terminates an existing task.

Consider the scenario shown in Fig. 1. Camera 1 is observing a pedestrian depicted as the blue circle and camera 2 is tracking an individual depicted as the green square. When camera 1 detects that the blue circle is about to leave its field of view, camera 1 requests camera 2 to take over the tracking (or observation) responsibilities for this pedestrian. Camera 2 is already at its tracking limit, i.e., camera 2 can only observe a single pedestrian at any given time.[1] Consequently, camera 2 is unable to take on the task of observing the blue circle. However, in the scheme proposed here, camera 2 is able to propose the following conditional offer to camera 1: camera 2 agrees to observe the blue circle if camera 1 agrees to observe the green square. Camera 2 comes up with this conditional offer by relying upon its internal model of camera 1. Camera 1 agrees and the two cameras swap their pedestrians, resulting in a successful handoff.

In the above example, we have used tracking limit as the reason why camera 2 is unable to take on a new task, which in turn triggered a conditional offer and the second round of negotiations. Tracking limit is but one of the reasons why a camera might be unsuitable for taking on a new task. It is possible that a camera is unable to carry out a particular combination of tasks simultaneously. For example, a *pan/tilt/zoom* (PTZ) camera might be unable to view two pedestrians at the opposite ends of its field-of-view with the desired resolution. Such situations are also resolved through conditional offers.

The proposed approach is distributed. Camera handoffs are the result of local negotiations between two neighbouring cameras (i.e., cameras that are within the communication range of each other). The lack of a central controller suggests that the proposed technique can be scaled to large camera networks—the scalability properties of the proposed method have yet to be empirically evaluated. We evaluate the proposed negotiation model on a (simulated) network of uncalibrated active *pan/tilt/zoom* (PTZ) cameras. The negotiation model is used to perform handoffs between individual cameras, which are modelled as autonomous behavior-based agents. The handoffs ensure that the camera network is able to track multiple targets as these move in the area under observation, weaving in and out of the field of views of different cameras. We have also compared our technique against the camera handoff scheme that appeared in [3] and the initial results appear promising.

2. LITERATURE REVIEW

The handoff problem refers to the transfer of the tracking responsibilities of a pedestrian from one camera to another. Many different approaches have been proposed, some of which are discussed in [4]. These approaches can typically be placed in one of two groups: 1) handoff function based approaches, and 2) negotiation based approaches.

The handoff function based approaches rely on a learning stage where a handoff function or handoff table can be learned. Javed and Khan propose a way to discover the relationship between camera FOV's by projecting the FOV lines of one camera onto another cameras view to find overlapping regions [5]. Park et al. propose a solution based on constructing a distributed lookup table, which encodes the suitability of a camera to observe a specific location [6]. Jo and Han propose the use of a handoff function defined as the ratio of co-occurrence to occurrence for all pairs of points in two views [7]. Quaritsch et al. propose an approach that relies on a static vision graph that encodes migration regions [8].

Other approaches utilize inter-camera negotiations to complete the handoff. Li and Bhanu develop a game theoretic approach in [9]. When an object is visible in multiple cameras, the best camera is chosen based on its expected utility. They also propose a number of criteria to construct the utility function, such as the number of pixels occupied by the selected target in an image. Their approach eschews spatial and geometric information. Qureshi and Terzopoulos propose an uncalibrated camera network, where cameras form groups and cooperatively carry out the observation tasks [10]. Groups are dynamic arrangements, which are set up through a distributed auction process. Assignment conflicts are modelled as a constraint satisfaction problem whose solution determines appropriate camera assignments. Song et al. present a game-theoretic strategy for cooperative control of a set of decentralized cameras [11]. The cameras work together to track every target in the area at acceptable image resolutions. The camera network can also be tasked to record higher-resolution imagery of a selected target. Qureshi presents a negotiation protocol that allows camera nodes to setup collaborative tasks in a purely distributed manner in [2]. The camera nodes, which are modelled as behaviour based autonomous agents, are able to ask other cameras to take over a task when they are no longer able to meet task requirements.

The work by Esterle et al. on a socio-economic approach to the handoff problem [3] is the closets to the work presented here. They propose a system of self-interested autonomous agents that handoff tasks from one camera to another using a market mechanism. Each camera tries to maximize their own utility by auctioning off tasks it cannot or does not want to complete. This approach is similar to our approach; however, the key difference is our approach does not use self-interested agents that try to maximize their utility. Instead, our agents try to help each-other out when ever possible. This is achieved with the use of a memory module which we introduce.

3. APPROACH

Consider an area under observation by n cameras. The cameras are tasked to observe p pedestrians (targets) throughout their stay in the region given the following constraint: each camera can only track at most o pedestrians. The actual number of tasks a camera can successfully carry out simultaneously at any given instant depends upon multiple factors, such as the type of the camera, computational load on the camera, the desired resolution of the captured imagery, etc. Whether or not the cameras have overlapping field-of-views (FOVs), camera handoffs are necessary to successfully carry out the task outlined above. We now describe our negotiation model for camera handoff. Implementation details are provided as supplementary material.

3.1 Smart Camera Nodes

This work is focused on the negotiation model with conditional offers for camera handoff, and it makes the following assumptions about the camera nodes:

- a camera is able to track pedestrians (targets) assigned to it;
- a camera is able to use appearance based signatures for acquiring a new pedestrian (target) for tracking;
- a camera is able to detect when it loses track of a pedestrian; and
- PTZ cameras are able to select appropriate values for pan, tilt and zoom settings to observe the assigned pedestrians.

[1]The tracking limit of 1 is only used for illustrative purposes in this example. The proposed scheme does not assume that each camera can only track a single individual.

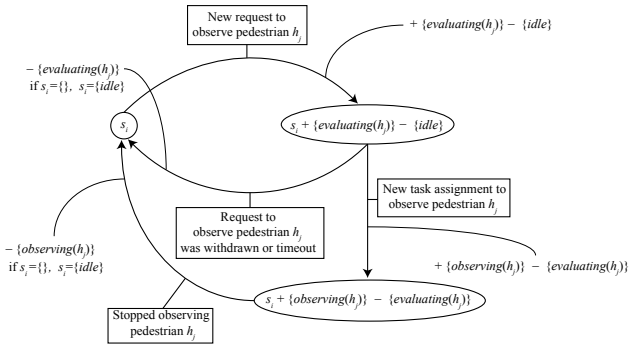


Figure 2: The state machine used by our Camera Controller. States are shown in ovals and transitions are shown in boxes. Additions and subtractions are shown next to each transition arc. *Courtesy of [2].*

These assumptions are motivated (and supported) by recent advances in pedestrian detection, recognition and tracking [12] and PTZ camera tracking [13]. Similar assumptions have been made by others [2, 3].

We treat each camera as a highly capable behavior-based agent. Each camera node is aware of the pedestrians (or targets) present in its FOV. This is easily accomplished by using a pedestrian detection routine. The following algorithm is used to maintain the set of pedestrians $\mathcal{H}_i$ seen by a camera c_i between times $t - t^{\text{forget}}$ and t, where t represent the current time. $ts(\cdot)$ is an operator that is used to set and retrieve the time-stamps for elements of $\mathcal{H}_i$; it operates on sets. To make things concrete $ts(\mathcal{S}) = t$ sets the time-stamps of every element of set $\mathcal{S}$ equal to t; where as, $ts(\mathcal{S})$ assumes that $\|\mathcal{S}\| \leq 1$ and returns the time-stamp of the element or the current time if the set is empty. $ts(\mathcal{S})$ for $\|\mathcal{S}\| > 1$ is undefined.

Require: $\mathcal{H}_i$ {The set of pedestrians seen by camera i between times $t - t^{\text{forget}}$ and now (t).}
Ensure: Updated $\mathcal{H}_i$
1: Capture a frame at time t
2: Use pedestrian detection (recognition) routines to construct a possibly empty set $\mathcal{H}_i^t$ of pedestrians found in frame I.
3: $\mathcal{H}_i^{\text{existing}} = \mathcal{H}_i^t \bigcap \mathcal{H}_i$ {The next four lines update the time-stamps of existing pedestrians in $\mathcal{H}_i$}
4: $\mathcal{H}_i = \mathcal{H}_i \setminus \mathcal{H}_i^{\text{existing}}$
5: $ts(\mathcal{H}_i^{\text{existing}}) = t$
6: $\mathcal{H}_i = \mathcal{H}_i \bigcup \mathcal{H}_i^{\text{existing}}$
7: $\mathcal{H}_i^{\text{new}} = \mathcal{H}_i^t \setminus \mathcal{H}_i^{\text{existing}}$
8: $ts(\mathcal{H}_i^{\text{new}}) = t$
9: $\mathcal{H}_i = \mathcal{H}_i \bigcup \mathcal{H}_i^{\text{new}}$ {Add the previously unseen pedestrians into $\mathcal{H}_i$}
10: **for all** $h \in \mathcal{H}_i$ **do**
11: **if** $t - ts(\{h\}) > t^{\text{forget}}$ **then**
12: $\mathcal{H}_i = \mathcal{H}_i \setminus \{h\}$ {Prunning stale entries from $\mathcal{H}_i$}
13: **end if**
14: **end for**

At any given time, each camera may be engaged in several activities. A camera might be tracking multiple individuals, it might be evaluating its suitability for tracking a new target, or in case of PTZ cameras, a camera might be performing a visual search in the pan/tilt/zoom space to fixate and zoom in on a pedestrian. Ignoring the innards of a camera node, it is possible to keep track of these activities using a finite state machine proposed in [2] (see Fig. 2). The state of a camera node represents the activities it is currently executing. Given this list of activities, we maintain the activity set $\mathcal{A}_i$ for a camera c_i as follows:

Require: The current activity set $\mathcal{A}_i$.
Ensure: The updated activity set $\mathcal{A}_i$.
1: **if** camera c_i is not engaged in activities at the moment **then**
2: **return** $\mathcal{A}_i = \Phi$
3: **end if**
4: **for all** $h \in \mathcal{H}_i$ **do**
5: **if** Camera is tracking h **then**
6: $\mathcal{A}_i = \mathcal{A}_i \bigcup \{Observing(h)\}$
7: **end if**
8: **if** Camera is evaluating its suitability for tracking h **then**
9: $\mathcal{A}_i = \mathcal{A}_i \bigcup \{Evaluating(h)\}$
10: **end if**
11: **end for**

Whether or not a camera is tracking an individual or evaluating its suitability for tracking a new individual is dictated by the tasks assigned to the camera. Other caveats are: 1) a camera cannot be both $Observing$ and $Evaluating$ the same individual and 2) an idle camera cannot be engaged in $Observing$ or $Evaluating$ any individual. Typically these tasks are assigned automatically through camera handoff negotiations described in the next section. These tasks can also be assigned by an operator manually or through heuristics of the form: track every individual that enters a specific region.

3.1.1 Memory Model

In contrast to existing models of smart camera nodes, our scheme explicitly models short-term memory of a camera node. Each camera uses the short-term memory to store its own state (the list of pedestrians in the near past plus the list of currently active tasks, i.e., $\mathcal{H}$ and $\mathcal{A}$) and the state of its neighbouring camera nodes. Each camera manages its memory to keep it up-to-date. Stale items present in the memory are automatically removed. Presently, we assume that each camera has unbounded memory. Furthermore, we employ time-stamps to decide when to remove the an item from the memory.

Below we discuss how a camera manages the states of its neighbouring cameras. Say $\mathcal{C}$ represents the set of cameras and $\mathcal{C}_i$ represents the neighbours of c_i then c_i will store the current state s_j for each camera $c_j \in \mathcal{C}_i$. Additionally, c_i will also keep track of previous states of camera c_j. Remembered states are time-stamped and older states are automatically forgotten after some time.

Lets consider the following example to illustrate the memory model. Consider a camera c_2 that is currently tracking pedestrians h_2 and h_3. c_2 is also evaluating its suitability for tracking pedestrian h_4. Previously, c_2 was tracking h_1; however, h_1 is no longer tracked by c_2. In this scenario a neighbouring camera c_1 will store the following information about c_2 in its memory:

- $Observing(c_2, h_2, t)$;
- $Observing(c_2, h_3, t)$;
- $Evaluating(c_2, h_4, t)$; and
- $Know(c_2, h_1, t^-)$.

Notice that each memory item is time-stamped. $Know(c_2, h_1, t^-)$ indicates that camera c_1 believes that c_2 has seen the pedestrian h_1 at some previous time t^-.

Cameras share states voluntarily during negotiation messages or via a periodic state update message. In the above scenario, camera c_2 will periodically send $\mathcal{H}_2$ and $\mathcal{A}_2$ to the neighbouring camera c_1. c_1 breaks down the received state into memory items and compares

these with the current c_2 items stored in its memory. One of the following four things can happen:

- if an item is already found in the memory, its time-stamp is updated;
- if an Observing item is received that matches with a stored Evaluating item, the Evaluating item is replaced with the Observing item;
- if no item matches with an already stored Observing or Evaluating item, the Observing (or Evaluating) item is replaced by a Know item; or
- if a received item does not match with a stored item (using the criteria implicit in the last three cases), the received item is time-stamped and inserted into the memory.

The following example clarifies these rules. Say c_1 receives the following information from c_2 at time $t^+ > t$:

$$\mathcal{A}_2 = \{Observing(h_2), Observing(h_4)\}$$

and

$$\mathcal{H}_2 = \{h_1, h_2, h_3, h_4\},$$

where $ts(h_1) = t^-$, $ts(h_2) = t^+$, $ts(h_3) = t$ and $ts(h_4) = t^+$. c_1 will update its information about c_2 as follows:

- $Observing(c_2, h_2, t^+)$;
- $Know(c_2, h_3, t)$;
- $Observing(c_2, h_4, t^+)$; and
- $Know(c_2, h_1, t^-)$.

If $t^+ - t^- > t^{\text{forget}}$ then item $Know(c_2, h_1, t^-)$ will be removed from the memory of c_1. There are many schemes for selecting t^{forget}; however, for the results presented in this paper, we use a single value of t^{forget} for all items.

In conclusion, the items stored in the short-term memory of a camera take one of the following forms:

- $Observing(c_i, h_j, t)$;
- $Evaluating(c_i, h_j, t)$;
- $Know(c_i, h_j, t)$.

Here $c_i \in \mathcal{C}$, $h_j \in \mathcal{H}$ and t refers to the time-stamp of a particular item. Notice that this form allows a camera to store both its own state and the states of its neighbouring cameras. Secondly, it is relatively easy to write queries involving cameras and pedestrians on the data stored in the short-term memory of a camera, such as (1) pick a neighbouring camera that is currently observing pedestrian h_j, (2) how many neighbouring cameras are observing a particular pedestrian, (3) is their some neighbouring camera that knows about a particular pedestrians, etc. When considering the memory model discussed here, it is worthwhile to remember two things: (1) each camera only maintains the memory state of its first-hop neighbours, i.e., cameras that share an edge in the communication graph, and (2) we do not impose any guarantees that the state of the neighbouring camera is perfectly synchronized. The second item is of particular importance. The communication overhead to guarantee that each camera maintains a "perfectly synchronized copy of the state of its neighbours" is explosive and therefore unattainable in a real scenario.

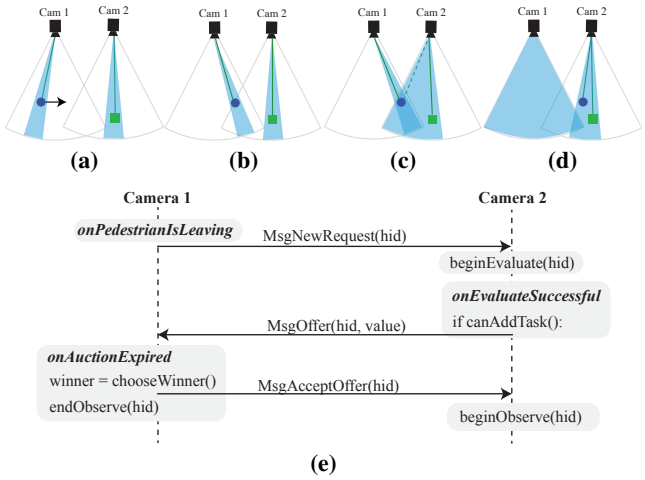


Figure 3: A negotiation is initiated by a camera when it notices that it will not be able to complete a task. (a) Cam 1 is observing a pedestrian that looks like a blue circle as it moves across a room. Cam 2 is observing the pedestrian that looks like a green square. (b) Cam 1 notices that the pedestrian that looks like a blue circle will soon leave its viewing region, so it asks Cam 2 for help. (c) After receiving the request for help, Cam 2 evaluates its suitability for the task. (d) Cam 2 decides to accept the task and is now observing both pedestrians. (e) Messages exchanged during the handoff operation.

Others, including [3], have developed techniques for generating vision and communication graphics within the context of camera networks. A side-effect of explicitly modelling the state of neighbouring cameras is that both communication and vision graphs are automatically learnt. The goal is not to learn the overlaying communication graph or vision graph for the whole network. Rather, our goal is to learn a local snapshot at each node. For stationary nodes, we can assume the snapshot to be static; whereas, for non-stationary nodes, it is best to rely on the forgetting mechanism built into the memory model to maintain a time-varying snapshot.

3.2 Negotiations

Inter-camera negotiations allow observation tasks to be handed off from one camera to another as a person moves around an observation area. Generally speaking, however, a camera can initiate a negotiation whenever it wants another camera to take over an observation task:

- Typically a camera initiates a negotiation when it detects that the pedestrian in question is about to leave its field of view; or
- A camera can also initiate a negotiation when it wants to free up resources for a task requested by some other camera.

Existing negotiation schemes for camera handoff do not start negotiations when a camera wants to release resources that are currently being used by an observation task.

Figure 3 illustrates the basic mechanics of negotiations for camera handoff [3, 2]. Here, camera 1 is viewing a pedestrian that looks like a blue circle as it walks across a room. Camera 2 is viewing a pedestrian that looks like a green square. After some time, camera 1 notices that the pedestrian that looks like a blue circle is approaching the edge of the cameras viewing region. If it goes any

further, the camera will not be able to track it. As a result, camera 1 initiates negotiations with camera 2 to take over the task of observing the blue circle. In response camera 2 computes its suitability for taking on this task and returns this information to camera 1.[2] The negotiations are successful and camera 2 starts observing the pedestrian represented by a blue circle. Often camera 1 will initiate negotiations with more than one neighbouring cameras.

As stated earlier, a key difference between the current work and existing negotiation schemes for camera handoff is that here each camera maintains the state of neighbouring cameras. This allows a camera to be selective when considering neighbouring cameras for handoff. Consider the following scenario: camera 1 has the following information in its memory:

- $Observing(c_2, h_2, t^+)$;
- $Know(c_2, h_3, t)$;
- $Observing(c_2, h_4, t^+)$;
- $Know(c_2, h_1, t^-)$; and
- $Observing(c_3, h_6, t^+)$.

Since camera c_3 is only observing a single pedestrian h_6, camera c_1 first requests a handoff with c_3. If that fails, it can request c_2 to take over the task. Along similar lines, in a calibrated camera network, where each camera not only knows the state of neighbouring cameras, but also their observation constraints, cameras may be able to predict the outcome of negotiations without actually exchanging any messages.

3.3 Conditional Offers

Camera nodes are not modelled as *self-interested* agents and are always ready to takeover an observation task from a neighbouring camera. Still there are situations where a camera cannot accept a task, even if it can meet all task requirements. A camera node has a limited set of resources and is able to track at most o pedestrians at any given time. If a camera node is at this tracking limit, it cannot accept any new tasks. One potential solution to this problem is conditional offers. Conditional offers give camera nodes the ability to swap tasks or to terminate existing tasks in order to take on new, more pressing tasks.

4. RESULTS

We have developed a PTZ camera network simulator for evaluating the proposed camera handoff technique (see Fig. 4). The simulator, which is implemented in Python, is able to simulate the movement of pedestrians, camera logic, camera motors (for PTZ cameras), inter-camera communication and camera sensing. Pedestrians motions are scripted via a sequence of *move* and *stand* commands. Each *move* command identifies a location plus a radius. This allows us to re-run a scenario with randomly generated paths. For the tests presented here, the radius is set to 200 units (or 2 meters in simulation space).

To simplify the testing process, we have made a few simplifying assumptions. We assume perfect tracking, which means that if a pedestrian is inside a camera's viewing region, it can be tracked perfectly. We also ignore any occlusions between pedestrians. The visibility of a pedestrian is calculated by projecting the pedestrians 3D bounding box onto a camera's image plane and computing the

[2]Many schemes exist in the literature for computing the suitability of a camera to an observation task [2, 3, 14]. Here we do not concern ourselves with how this suitability is computed.

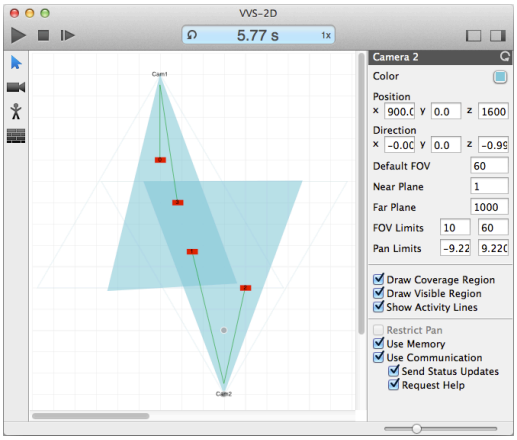


Figure 4: A screenshot of our 2D camera network simulator.

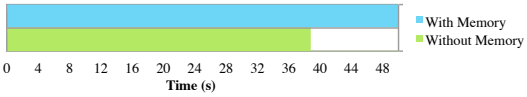


Figure 6: Scenario 1: The tracking history of pedestrian 1 with and without the use of a cameras' memory for storing the internal state of the neighbouring cameras. When using memory, a tasks swap can occur and the pedestrian 1 is seamlessly tracked through out his stay in the region.

percentage of the image taken up by the person. Recent advances in pedestrian tracking suggest that these assumptions are valid for camera networks observing low density (pedestrian) crowds.

4.1 Test Scenarios

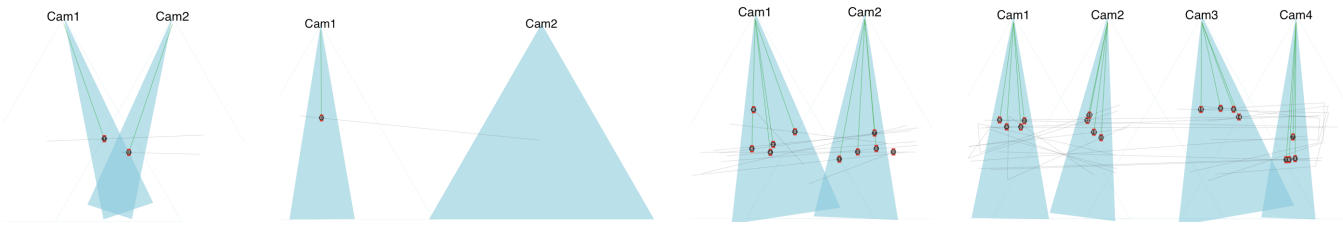
We evaluate our approach on four different scenarios shown in Fig. 5. The first two scenarios are designed to highlight the novel aspects of the proposed approach. Scenarios 3 and 4 are designed to compare our approach with that of Esterle et al. [3]. Lastly we test the proposed approach to track 32 pedestrians as these move along a rectangular path observed by 16 cameras.

4.1.1 Scenario 1

The first scenario highlights the role of memory and conditional offers (Fig. 5(a)). Here each camera is restricted to observe a single individual at any given time. Camera 2 is already observing an individual when camera 1 requests a handoff. In the absence of memory and the ability to propose a conditional offer, handoff will be unsuccessful, since camera 2 is unable to track two pedestrians at the same time. Since camera 2 models the internal state of camera 1, camera 2 proposes to swap the tasks (conditional offer). Camera 1 agrees, which results in a successful handoff. Fig. 6 plots the tracking history of pedestrian 1 with and without memory. Notice that, when using memory, pedestrian 1 is tracked throughout his stay in the region under observation. Without the use of memory, person 1 is lost once he leaves the FOV of camera 1.

4.1.2 Scenario 2

The second scenario also consists of two cameras with non-overlapping FOVs (Fig. 5(b)). It simply demonstrates that the proposed approach can function in situations where cameras' FOVs do not overlap. Fig. 7 plots the tracking history of pedestrian 1 as she moves between the observational ranges of camera 1 and camera 2.



(a) Scenario 1 **(b)** Scenario 2 **(c)** Scenario 3 **(d)** Scenario 4

Figure 5: The first two scenarios highlight the novel aspects of the proposed technique. (a) illustrates the effect of memory on pedestrian tracking. (b) shows that handoffs are possible even when there is no overlap between the cameras' field of views. (c)-(d) The proposed technique is compared with that of Esterle et al. [3] using scenarios 3 and 4.

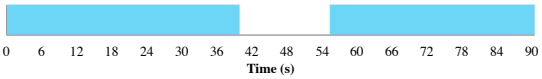


Figure 7: Scenario 2: The tracking history of pedestrian 0. Since the two cameras do not have overlapping viewing regions, the pedestrian does not get tracked for a period of time.

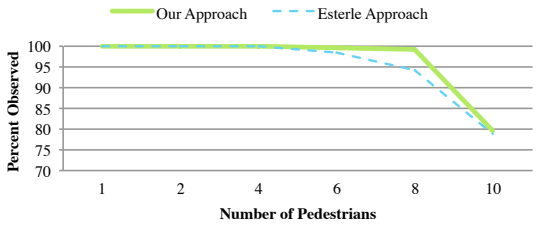


Figure 8: Scenario 3: The percent of pedestrians observed by our approach and that of [3] as the number of pedestrians are increased. The results are averaged over 10 runs.

Notice that camera 2 is able to track the individual as it enters its field of view. Again camera memory is at play here.

4.1.3 Scenario 3

The third scenario is used to compare our approach with that of Esterle et al. [3] (Fig. 5(c)). We run tests with 1, 2, 4, 6, 8, and 10 pedestrians to show how camera network's performance degrades as the number of pedestrians increases. Half of the pedestrians begin in camera 1's viewing region and walk towards camera 2's viewing region. The other half start in camera 2's viewing region and walk in the opposite direction (Fig. 5c). We ran the simulation 10 times for each approach and averaged the results. Fig. 8 shows that our approach performed slightly better than the Esterle et al. approach once there are around 6 people in the scene. Fig. 9 shows how smooth our approach is compared to the Esterle approach. Our approach is able to complete seamless handoffs, even when the cameras are tracking the max number of people. This is due to the memory module allowing cameras to swap tasks when cameras are at their tracking limits. The Esterle et al.'s approach has to wait until a pedestrian leaves a cameras viewing region before being able to accept a new task. This results in a short period of time where a pedestrian will not be tracked.

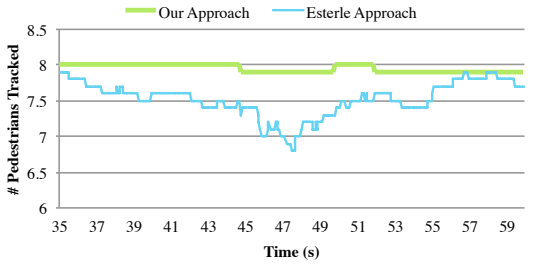


Figure 9: Scenario 3: Average number of pedestrians tracked over time in Scenario 3 with 8 people.

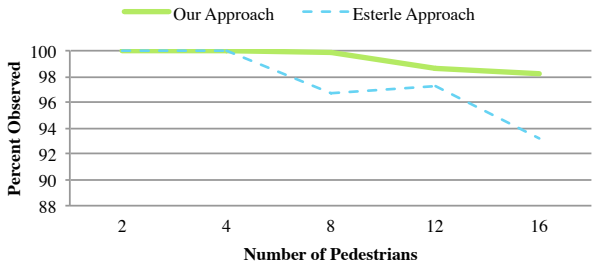


Figure 10: Scenario 4: The percent of pedestrians observed as the number of pedestrians increased in Scenario 4. The results are averaged over 10 runs.

4.1.4 Scenario 4

In our final scenario, we have four cameras positioned in a row with pedestrians walking in both directions. We ran tests with 2, 4, 8, 12, and 16 pedestrians to again show how the cameras performance degrades with more pedestrians. Again we ran each test 10 times for both our approach and that of Esterle et al. s. A comparison of the percent of pedestrians observed as the number of pedestrians is increased is shown in Fig. 10. As in scenario 3, our approach outperforms Esterle et al.'s approach as the number of pedestrians increases.

4.2 Scenario 5

This scenario consists of 32 pedestrians walking along a rectangular path. 16 cameras are tasked with observing these individuals (see Fig. 11). The pedestrians are tracked for 3000 simulation steps. Using this scenario, we have evaluated the proposed model under 4 settings: 1) each camera can track no more than 4 pedestrians, 2) each camera can track no more than 4 pedestrians and the cam-

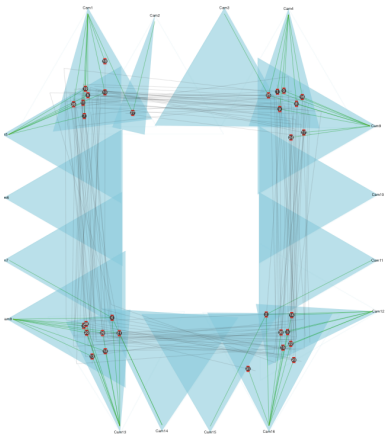


Figure 11: Scenario 5: 16 cameras are tasked to observe 32 pedestrians as these move along a rectangular path.

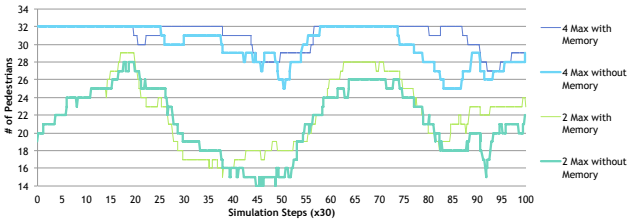


Figure 12: Scenario 5: The numbers of pedestrians tracked over time.

eras do not maintain the state of their neighbours, 3) each camera can track no more than 2 pedestrians and 4) each camera can track no more than 2 pedestrians and the cameras do not maintain the state of their neighbours. Fig. 12 plots the number of pedestrians tracked for each setting over time. On average 30.987 pedestrians are tracked when cameras can track up to 4 pedestrians. This value is reduced to 29.993 when cameras are not allowed to propose conditional offers (no memory). Similarly, on average the cameras track 22.193 pedestrians when each camera is allowed to observe only 2 pedestrians at any given time. This value is reduced to 21.316 when cameras are not allowed to propose conditional offers. While these results suggest that benefits of conditional offers are marginal, it is worth keeping in mind that conditional offers only effect situations where tasks swaps are possible. Perhaps, this scenario does not contain many such situations. In any case, scenario 1 clearly makes the case of conditional offers.

5. CONCLUSIONS

In this paper we presented a new short-term memory model for maintaining a camera's own state and the states of neighbouring cameras. We use this memory model to introduce the concept of conditional offers during handoff negotiations. Conditional offers allow successful handoff in situations where existing approaches would fail. One such example is when a camera is at its tracking limit and cannot track any more pedestrians. With the use of the memory model, cameras are able to negotiate a swap of tasks if the pedestrians being tracked are visible in both cameras.

We have evaluated our approach on scenarios with 2, 4 and 16 cameras and up to 32 pedestrians. We have compared our approach to the Esterle et al. scheme that appeared in [3] and the results seem to suggest that our approach outperforms the handoff approach developed in [3]. Here we focus on the idea of using the memory to support handoff negotiations, there are other uses of memory which we hope to explore in the future, such as distributed dynamic load balancing in camera networks.

6. REFERENCES

[1] Y. Li and B. Bhanu, "A Comparison of Techniques for Camera Selection and Hand-Off in a Video Network," *Distributed Video Sensor Networks*, 2009.

[2] F. Z. Qureshi, "Collaborative sensing via local negotiations in ad hoc networks of smart cameras," *Proceedings of the Fourth ACM/IEEE International Conference on Distributed Smart Cameras - ICDSC '10*, p. 190, 2010.

[3] L. Esterle and P. R. Lewis, "A socio-economic approach to online vision graph generation and handover in distributed smart camera networks," *ACM Transactions on Sensor Networks*, vol. 0, no. 0, pp. 1–24, 2011.

[4] Y. Li and B. Bhanu, "A Comparison of Techniques for Camera Selection and Hand-Off in a Video Network," *Distributed Video Sensor Networks*, 2009.

[5] O. Javed and S. Khan, "Camera handoff: tracking in multiple uncalibrated stationary cameras," *IEEE Computer Society Workshop on Human Motion*, 2000.

[6] J. Park, P. Bhat, and A. Kak, "A look-up table based approach for solving the camera selection problem in large camera networks," *Proceedings of the International Workshop on Distributed Smart Cameras*, 2006.

[7] Y. Jo and J. Han, "A new approach to camera hand-off without camera calibration for the general scene with non-planar ground," *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks - VSSN '06*, p. 195, 2006.

[8] M. Quaritsch, M. Kreuzthaler, B. Rinner, H. Bischof, and B. Strobl, "Autonomous Multicamera Tracking on Embedded Smart Cameras," *EURASIP Journal on Embedded Systems*, vol. 2007, pp. 1–10, 2007.

[9] Y. Li and B. Bhanu, "Utility-based dynamic camera assignment and hand-off in a video network," *Second ACM/IEEE International Conference on Distributed Smart Cameras*, pp. 1–9, 2008.

[10] F. Z. Qureshi and D. Terzopoulos, "Multi-camera Control through Constraint Satisfaction for Persistent Surveillance," *2008 IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance*, pp. 211–218, Sep. 2008.

[11] B. Song, C. Soto, A. K. Roy-Chowdhury, and J. A. Farrell, "Decentralized camera network control using game theory," in *Second ACM/IEEE International Conference on Distributed Smart Cameras*, 2008, pp. 1–8.

[12] M. Enzweiler and D. M. Gavrila, "Monocular pedestrian detection: Survey and experiments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 12, pp. 2179–2195, Dec 2009.

[13] C. Micheloni, B. Rinner, and G. Foresti, "Video analysis in pan-tilt-zoom camera networks," *IEEE Signal Processing Magazine*, vol. 27, no. 5, pp. 78–90, Sep 2010.

[14] Y. Li and B. Bhanu, "Utility-based camera assignment in a video network: A game theoretic framework," *Sensors Journal, IEEE*, vol. 11, no. 3, pp. 676–687, 2011.