Activity Topology Estimation for Large Networks of Cameras

Anton van den Hengel, Anthony Dick, Rhys Hill
School of Computer Science
University of Adelaide
Adelaide, 5005, Australia
{anton,ard,rhys}@cs.adelaide.edu.au

Abstract

Estimating the paths that moving objects can take through the fields of view of possibly non-overlapping cameras, also known as their activity topology, is an important step in the effective interpretation of surveillance video. Existing approaches to this problem involve tracking moving objects within cameras, and then attempting to link tracks across views. In contrast we propose an approach which begins by assuming all camera views are potentially linked, and successively eliminates camera topologies that are contradicted by observed motion. Over time, the true patterns of motion emerge as those which are not contradicted by the evidence. These patterns may then be used to initialise a finer level search using other approaches if required. This method thus represents an efficient and effective way to learn activity topology for a large network of cameras, particularly with a limited amount of data.

1. Introduction

The proliferation of surveillance cameras throughout public places has far outpaced the development of software to monitor the video they generate. This has meant that although networks of cameras have been installed to monitor large facilities, their effectiveness is limited by a lack of coordination.

A key step towards automating the monitoring of video from many cameras is to generate an understanding of the paths which targets may take between their fields of view. This activity topology information is the foundation for many fundamental tasks in networked surveillance, such as tracking an object across the network. Although it could be derived manually for small sets of cameras, this approach does not scale to large networks, where cameras may frequently be added, malfunction or be moved.

The aim here is not to compute an exact geographical map of surveillance camera locations. Instead, we wish to determine an approximate distance between pairs of cameras. This distance is measured in terms of time taken to transit between a pair of cameras. Cameras whose fields of view overlap have zero transit time, whereas those that are at opposite ends of a corridor may have a transit time of tens of seconds. Note that this distance may not always reflect the physical separation of the cameras.

In line with this goal, we use a novel representation for activity topology estimation that is not based on tracking objects within each camera. Instead it relies on information that is easier to derive—the presence or absence of objects within each field of view. This information is adequate for topology determination, and is fast to work with, enabling the method to scale to large camera networks.

Previously, activity topology estimation has been approached as a problem of learning the probability of transitions between fields of view (FOVs) from corresponding tracks [5]. However, the correspondence between tracks in different images must be supplied a priori as training data, which limits the scale of the network for which it is useful. Dick et al. [2] suggest an alternate approach whereby learn activity topology is represented by a Markov model. This does not require correspondences, but does need a training phase and does not scale well. Ellis et al. [3] do not require correspondences or a training phase, instead observing motion over a long period of time and accumulating appearance and disappearance information in a histogram. This approach has been extended by Stauffer [8] and Tieu et al. [10] to include a more rigorous definition of a transition based on statistical significance, and by Gilbert et al. [4] to incorporate a coarse to fine topology estimation. These methods show promise for larger scale applications, but have only been demonstrated on networks of less than 10 cameras.

Rahimi et al. [7, 6] perform experiments on configurations of several cameras involving non-overlapping FOVs. One experiment [6] involves calculating the 2D position and 1D orientation of a set of overhead cameras viewing a common ground plane by manually recording the paths followed by people in each camera's FOV. It is shown that a simple smoothness prior is enough to locate the cameras and reconstruct a path where it was not visible to any camera. In another experiment [7], pre-calibrated cameras are mounted on the walls of a room so that they face horizontally. In this case, the 2D trajectory of a person is recovered as they move around in the room, even when the person is not visible to any camera.

On a larger scale, Brand et al. [1] consider the problem of locating hundreds of cameras distributed about an urban landscape. Their method relies on having internal camera calibration and accurate orientation information for each camera, and enough cameras viewing common features to constrain the solution. Given this information the method can localise the cameras accurately due to the constraints imposed by viewing common scene points from different viewpoints.

These methods all rely on observing and analysing large amounts of video in order to determine topology. They also require comparisons to be made between every pair of cameras in a network. The number of pairs of cameras grows exponentially with the number of cameras in the network, rendering exhaustive pairwise comparisons of large volumes of data infeasible.

This paper describes a novel method for determining which pairs of cameras may be profitably interrogated in order to estimate their relative topology, and those which may be ignored. The method is computationally fast, and does not rely on accurate tracking of objects within each camera view. In contrast to most existing methods for activity topology determination, it does not attempt to build up evidence for camera proximity over time. Instead, it uses observed activity to rule out topologies over time. By doing this it can rapidly home in on likely camera layouts by eliminating the least likely first. This is an easier decision to make, especially when a limited amount of data is available. It is also based on the observation that it is impossible to prove a positive connection between cameras—any correlation of events could be coincidence—whereas it is possible to prove a negative connection by observing an object in one camera while not observing it at all in another.

2. Problem formulation

Consider a set of cameras whose images are partitioned into a grid of windows. The activity topology of the network can be represented as the connections between these windows. We assume that we can detect people entering the scene reliably enough to determine their lowest visible extent. This may be their point of contact with the ground, or the point at which the lower portion of their body becomes occluded. As people move around the environment monitored by the cameras, their lowest visible extent moves from one window to another. Occupation is thus determined by whether a person's lowest visible extent falls within the boundaries of a particular window.

By observing the occupancy of each window over time, we can create a table of the form shown in Table 1, where 1 implies that a window is occupied and 0 implies that it is vacant. If two windows are images of exactly the same region in the world, we would expect their corresponding columns in this table to match exactly. There are situations whereby this may not be the case in practice, which we consider later. However the observation that underlies the work presented here is that a window which is occupied at a particular time instant cannot correspond to any other that is simultaneously unoccupied. Given that windows tend to be unoccupied far more often than occupied, this is a critical observation. Determining the topological connections between windows by looking for correlation in their occupancy statistics is necessarily a slow process because simultaneous occupancy may occur coincidentally. Simultaneous occupancy and vacancy of the same space, however, is physically impossible. Rather than attempt the slow process of gathering positive information as to the connections between windows, we seek negative information allowing the instant elimination of impossible connections. We refer to such connections as having been *excluded*.

		Window			
		w_1	w_2	w_3	w_4
Time	t_1	1	0	1	0
	t_2	0	1	0	0
	t_3	0	0	1	0
	t_4	1	0	0	1
	t_5	0	0	1	0
		$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 1. An example occupancy table

3. Determining window overlap

Let the set of windows over all cameras be $\mathcal{W} = \{w_1 \dots w_n\}$. Corresponding to each window w_i is an occupancy vector $\mathbf{o}_i = (o_{i1}, \dots, o_{iT})'$ with o_{it} set to 1 if window w_i is occupied at time t, and 0 if not. The *exclusive-or* $a \oplus b$ of two binary numbers a and b is 1 only if either a or b is 1 and the other is 0. If we define the same operator to apply to two vectors $\mathbf{a} = (a_1, \dots, a_k)'$ and $\mathbf{b} = (b_1, \dots, b_k)'$ as

$$\mathbf{a} \oplus \mathbf{b} = \max_{i=1}^{k} a_i \oplus b_i$$

then the exclusive-or of the two vectors $\mathbf{a}$ and $\mathbf{b}$ is 1 if a single pairwise comparison $a_i \oplus b_i$ is 1.

It can be inferred that two windows w_i and w_j do not overlap if the exclusive-or $\mathbf{o}_i \oplus \mathbf{o}_j$ of the corresponding occupancy vectors $\mathbf{o}_i$ and $\mathbf{o}_j$ is 1. This comparison requires very little time to calculate, even for long vectors. The occupancy vectors on which it is based do not need to be collected over long periods, but rather only as long as is necessary to eliminate obviously non-overlapping windows. This is thus an extremely efficient means of eliminating windows which do not overlap.

3.1. Improving robustness

Two windows w_i and w_j are unlikely to cover exactly the same area of the scene. It is thus possible that two overlapping windows might be simultaneously occupied and vacant and therefore that the exclusive-or of the corresponding occupancy vectors might incorrectly indicate that they do not overlap. An example of such a situation is illustrated in Figure 1. This problem can be rectified by including the

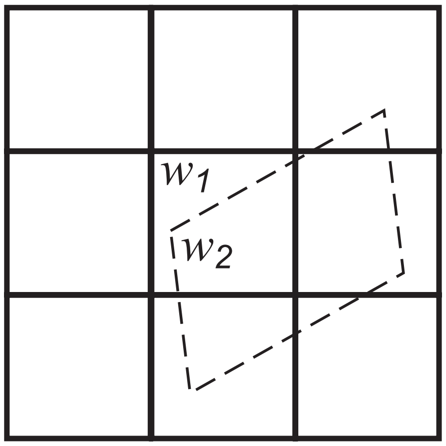

Figure 1. Overlapping windows

neighbours of a particular window when registering its occupancy. We thus introduce a padded occupancy vector $\mathbf{p}_i$ which has element p_{it} set to 1 when window w_i or any of its neighbours is occupied at time t. Calculating the exclusive-or of two padded occupancy vectors $\mathbf{p}_i$ and $\mathbf{p}_j$ does not determine the overlap of the corresponding windows. In fact it is even more likely that $\mathbf{p}_i \oplus \mathbf{p}_j$ will deliver a false positive than the original $\mathbf{o}_i \oplus \mathbf{o}_j$ given that returning a 0 result requires that all the neighbours of the windows w_i and w_j also overlap.

We now introduce a uni-directional version of the exclusive-or defined such that $a \ominus b$ is 1 only if a is 1 and b is 0. If we define the corresponding vector operator in the manner used above for two vectors $\mathbf{a}$ and $\mathbf{b}$ then

$$\mathbf{a} \ominus \mathbf{b} = \max_{i=1}^{k} a_i \ominus b_i. \tag{1}$$

A robust mechanism for determining whether two windows w_i and w_j overlap is thus to calculate $\mathbf{o}_i \ominus \mathbf{p}_j$ on the basis of the occupancy vector $\mathbf{o}_i$ and the padded occupancy vector $\mathbf{p}_j$. The padding described so far accommodates for occupancy misalignments in the spatial domain. The same process may be carried out over the temporal domain to allow accommodate errors in the synchronisation between cameras.

The measure $\mathbf{o}_i \ominus \mathbf{p}_j$ is not symmetric, so it is possible that $\mathbf{o}_i \ominus \mathbf{p}_j \neq \mathbf{o}_j \ominus \mathbf{p}_i$. This may seem counter intuitive, but in fact reflects the conservative nature of the padding process. A response to the asymmetry of the measure might be to require that both $\mathbf{o}_i \ominus \mathbf{p}_j$ and $\mathbf{o}_j \ominus \mathbf{p}_i$ identify the windows as excluded before a conclusion is drawn. This approach is, however, only suitable for situations in which it is expected that every window over the entire camera network will exhibit the occupancy necessary to calculate exclusion. In most practical networks, it is likely that some windows will be in a position whereby they will never be occupied. If we accept that a window pair do not overlap if either $\mathbf{o}_i \ominus \mathbf{p}_j$ or $\mathbf{o}_j \ominus \mathbf{p}_i$ identify an exclusion then every window in the network may be processed. It is still not possible to process every possible pair of windows, but the overlap of every window which may be occupied with every other window in the network may be calculated.

3.2. Accumulating evidence

Rather than making a hard decision about window overlap based on a single contradiction, it is possible to accrue evidence for overlap probabilistically. To do this, we measure support for the hypothesis of window overlap by using a hypothesis test based on the likelihood ratio. First, we define events, at some time t, as follows:

- A: $p_{jt} = 0$
- B: $o_{it} = 1$
- V: windows w_i and w_j overlap
- $\overline{V}$: windows w_i and w_j do not overlap

Events A and B together define a contradiction, as stated in the previous section. We wish to compute the likelihood of a single contradiction, given the binary hypotheses V and $\overline{V}$, in order to compute the likelihood ratio

$$\frac{\Pr(AB|V)}{\Pr\left(AB|\overline{V}\right)}.$$

We first compute the likelihood of a contradiction occurring if the windows overlap, written as:

$$\Pr(AB|V) = \Pr(A|BV)\Pr(B|V).$$

Assuming that both camera fields of view are about the same scale, the window w_i is completely covered by the neighbourhood of w_j if windows w_i and w_j overlap. The probability $\Pr(A|BV)$ is therefore primarily governed by an error rate (the rate of missed detections of occupancy) which we call C. When an occupancy event is missed by the detection process it is still possible that $p_{jt} = 1$ because other detections may have filled the gap. We compensate for this eventuality by multiplying C by an estimate of the probability that p_{jt} would be 0 if the detection failed. Therefore the probability is given by

$$\Pr(A|BV) = C \times \frac{count(\mathbf{p}_j = 0)}{T}$$

where T is the total number of observations (which is the length of $\mathbf{p}_j$). The other term in the likelihood can be computed as

$$\Pr(B|V) = \frac{count(\mathbf{o}_i = 1)}{T}$$

To compute $\Pr\left(AB|\overline{V}\right) = \Pr\left(A|B\overline{V}\right) \Pr\left(B|\overline{V}\right)$ we first note that if the windows do not overlap, A and B are independent. Therefore

$$\Pr\left(A|B\overline{V}\right) = \Pr(A) = \frac{count(\mathbf{p}_j = 0)}{T}$$

and, as $\Pr(B|V) = \Pr\left(B|\overline{V}\right)$, the likelihood ratio is given by

$$\frac{\Pr(AB|V)}{\Pr\left(AB|\overline{V}\right)} = \frac{\Pr(A|BV)}{\Pr\left(A|B\overline{V}\right)} = C$$

In other words, the plausibility of the hypothesis that the windows overlaps is multiplied by C, the tracking error rate (which we expect to be very low), for each contradiction that occurs. This means that the overall probability of a pair of windows overlapping is given by C^K, where K is the number of contradictory observations. This can be calculated by defining an operator $\oslash$ such that, for two vectors $\mathbf{a}$ and $\mathbf{b}$ of length k, returns K:

$$\mathbf{a} \oslash \mathbf{b} = \sum_{i=1}^{k} a_i \ominus b_i. \qquad (2)$$

3.3. Zones

It is often the case that cameras in a surveillance network form natural clusters. For example, a network in a building may contain one or more cameras in each office, cameras monitoring corridors and public spaces, and cameras monitoring the building exterior. This suggests a decomposition of the topology acquisition and tracking problem into surveillance zones. Each room may be a zone with a number of overlapping cameras, as may a corridor, the set of cameras monitoring a lobby, and so on. The defining feature of a zone is that cameras within a zone have high transition frequency, and transitions can take place in many ways, whereas transitions between zones are more tightly constrained. For example, the only transition between an outdoor zone and a lobby zone may be through the front door of the building. If these zones can be detected, they greatly simplify subsequent network tracking problem.

The form of the occupancy vectors allows them to be to be merged using a logical *or* operator. The merged occupancy vector $\mathbf{m}$ may thus be constructed on the basis of a set of vectors $\{\mathbf{a}_1 \dots \mathbf{a}_n\}$ as

$$\mathbf{m}(\{\mathbf{a}_1 \dots \mathbf{a}_n\}) = \left[\bigcup_{i=1}^{n} a_{it}\right]_{t=1\dots T}. \qquad (3)$$

Merging a set of occupancy vectors in this manner results in a vector representing the total occupancy of all the corresponding windows. Such a merged occupancy vector cannot be used for the same purposes as an occupancy vector representing a single window, however. This is because a 1 in the merged occupancy vector does not imply occupancy of the entirety of the corresponding combined window area. As an example of this we would expect that the result of $\mathbf{m}(\{\mathbf{a}_1 \dots \mathbf{a}_n\}) \ominus \mathbf{a}_1$ would be 1. Note that the result of $\mathbf{a}_1 \ominus \mathbf{m}(\{\mathbf{a}_1 \dots \mathbf{a}_n\})$, however, will always be 0.

Merged occupancy vectors may thus be used as the second operand to the $\ominus$ operator. Given the robustness issues outlined in Section 3.1, and the fact that padded occupancy vectors may be used as the second operand to the $\ominus$ operator, merging is performed only on padded occupancy vectors. The overlap of a window w_i with the set of windows $\{w_1 \dots w_n\}$ may thus be determined by calculating $\mathbf{o}_i \ominus \mathbf{m}(\{\mathbf{p}_1 \dots \mathbf{p}_n\})$.

Merging padded occupancy vectors within zones allows a hierarchical approach to adding new cameras to the activity topology. Each new camera is compared to smaller and smaller groups (zones) of cameras until a lack of overlap can be shown or the correct location identified.

4. Determining activity topology

We now describe how this formulation is used to cluster cameras into zones and derive an approximate topology linking zones. Our methodology lies between that of online updating, where the topology estimate is updated every frame, and offline learning, where the topology is learnt in a separate learning period, before the system is run.

Initially it is assumed that each window within a camera view is related to every other window in every other camera. Our goal is to discount those relations which our observations contradict.

4.1. Background subtraction

We detect moving objects within a camera view using the Stauffer and Grimson background subtraction method [9]. To derive a single position from a foreground blob, we use connected components and take the midpoint of the low edge of the bounding box of each blob. This corresponds

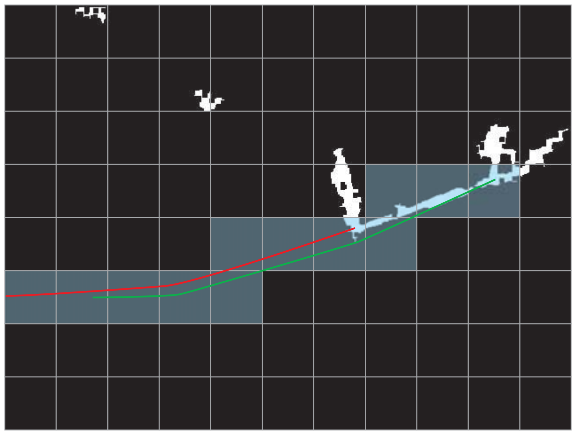

Figure 2. Using background subtraction to determine occupancy

approximately to the lowest visible extent of the person in the image. No tracking is necessary given that no identity information is used by the method. Figure 2 illustrates this process.

4.2. Updating the topology estimate

The background subtraction process produces occupancy information which may be used to update the topology estimate at every time step using the procedure outlined in Section 3.1. It is also possible to accrue this information in order to apply it in a batch using the process described in Section 3.2. New occupancy information is generated with every new frame retrieved from the camera, which may occur as often as 30 times a second. Significant batches of occupancy information can thus be acquired in very short periods of time. Thus, rather than requiring the processing of large amounts of video in order to determine topology the method is capable of generating relevant information as soon as occupancy may be measured. This means not only that topology information is accessible quickly, but also that the method may be applied in situations where only short segments of video are available. This may be the case, for instance, when a camera is moving from one location to another (such as in the case of a pan-tilt-zoom camera).

5. Results

5.1. Synthetic Testing

Synthetic data was generated so as to simulate a network of 50 cameras. The fields of view of the randomly generated cameras are shown in Figure 3. Figure 3 also shows the paths of the pedestrians (generated by an autoregression process) from which occupancy has been calculated. The important measure of the pedestrian activity in the network is the number of frames exhibiting occupancy. In the testing that was carried out $1,000$ frames were generated for each camera with $1,255$ window occupancies identified across the network over this period. The field of view of each camera has been divided into 100 windows, giving a total of $5,000$ windows for the synthetic camera network.

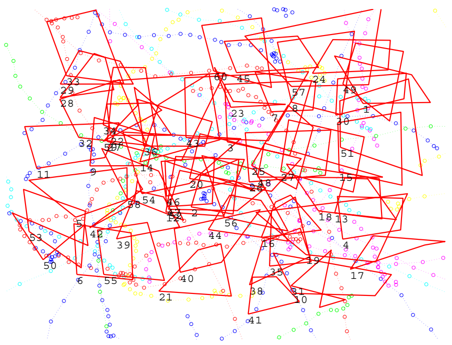

Figure 3. Synthetic camera network fields of view and tracks

The testing carried out consisted of calculating the value of $\mathbf{o}_i \ominus \mathbf{p}_j$ for every pair of windows w_i and w_j using equation (1) from Section 3.1. These calculations were carried out progressively, every 100 frames, which corresponds to less than 4 seconds of video at 30 frames per second, or 20 seconds of video at 5 frames per second. Over each interval the number of occupied windows was calculated, along with the number of exclusions. Figure 4 shows these numbers plotted against each other, for calculations made every 100 frames. The number of exclusions identified should be compared against the total number of exclusions for the network which is approximately 2.5×10^7. It should be noted also that for the measured exclusions to reach this total would require an exhaustive set of occupancies to be observed. Figure 4 shows that information about the topology of the network is gained very quickly, but that the rate of information gain slows over time. In order to determine the veracity of the results an algebraic analysis of overlap was carried out. Each of the measured exclusions was com-

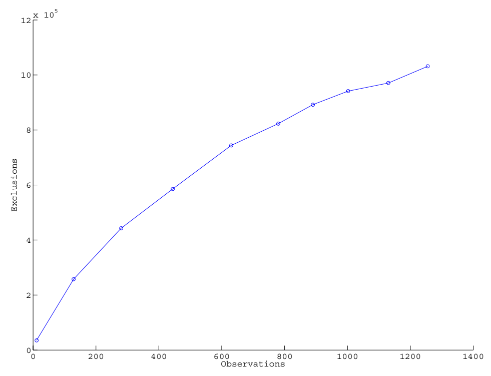

Figure 4. Exclusions vs. observed occupancies for synthetic testing

pared against its analytic equivalent and none were found to be in error.

5.2. Real image testing

In order to test the method on real imagery, 84 seconds of video was recorded from a set of 4 partially overlapping cameras. Example frames from these video streams are shown in Figure 5. Each frame was divided into 225 windows, and $1,471$ window occupancies measured. The total number of windows for the network is $1,125$, but only 156 of these were ever occupied over the course of testing. As in the synthetic case the testing consisted of calculating the value of $\mathbf{o}_i \ominus \mathbf{p}_j$ for every pair of windows w_i and w_j using equation (1) from Section 3.1. A total of $252,105$ exclusions were calculated by the method. The total number of pairs of windows is $1,265,625$, but given that the majority of the windows do not see traffic it would be impossible for the method to approach this number. Given that the true overlap of the cameras is unknown, it was not possible to test the veracity of every exclusion. However, manual inspection of all excluded areas corresponding to a randomly selected set of 20 windows did not uncover any errors.

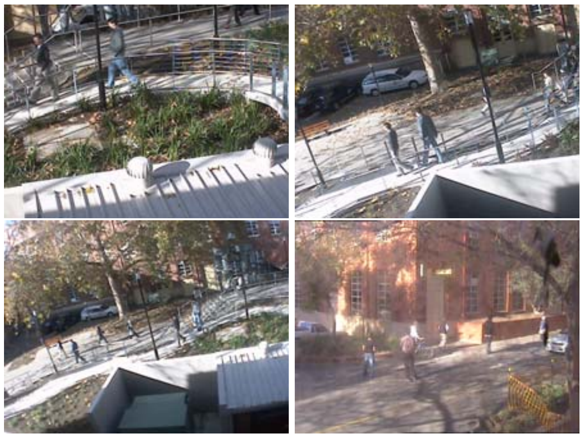

Figure 5. Frames used in real image testing

6. Conclusion

This paper has described a method for automatically determining activity topology in large camera networks. The method is based on the process of eliminating impossible connections rather than the slower process of building up positive evidence of activity. The method succeeds in both synthetic and real-image testing, and generates useful information after only small amounts of video have been processed.

References

[1] M. Brand, M. Antone, and S. Teller. Spectral solution of large-scale extrinsic camera calibration as a graph embedding problem. In *Proc. 8th European Conference on Computer Vision*, pages 262–273, 2004.

[2] A. Dick and M. J. Brooks. A stochastic approach to tracking objects across multiple cameras. In *Proc. Australian Joint Conference on Artificial Intelligence (AI'04)*, pages 160–170, 2004.

[3] T. J. Ellis, D. Makris, and J. Black. Learning a multi-camera topology. In *Joint IEEE Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS)*, pages 165–171, 2003.

[4] A. Gilbert and R. Bowden. Tracking objects across cameras by incrementally learning inter-camera colour calibraion and patterns of activity. In *European Conference on Computer Vision*, volume 2, pages 125–136, 2006.

[5] O. Javed, Z. Rasheed, K. Shafique, and M. Shah. Tracking across multiple cameras with disjoint views. In *Proc. IEEE International Conference on Computer Vision*, pages 952–957, 2003.

[6] A. Rahimi, B. Dunagan, and T. Darrell. Simultaneous calibration and tracking with a network of non-overlapping sensors. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 187–194, 2004.

[7] A. Rahimi, B. Dunagan, and T. Darrell. Tracking people with a sparse network of bearing sensors. In *European Conference on Computer Vision*, pages 507–518, 2004.

[8] C. Stauffer. Learning to track objects through unobserved regions. In *IEEE Computer Society Workshop on Motion and Video Computing*, pages II: 96–102, 2005.

[9] C. Stauffer and W. E. L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747–757, 2000.

[10] K. Tieu, G. Dalley, and W. Grimson. Inference of non-overlapping camera network topology by measuring statistical dependence. In *Proc. IEEE International Conference on Computer Vision*, pages II: 1842–1849, 2005.