Energy Efficient Intrusion Detection in Camera Sensor Networks

Primoz Skraba[1] and Leonidas Guibas[2]

[1] Department of Electrical Engineering
Stanford University, Stanford, CA 94305
primoz@stanford.edu
[2] Department of Computer Science
Stanford University, Stanford, CA 94305
guibas@stanford.edu

Abstract. The problem we address in this paper is how to detect an intruder moving through a polygonal space that is equipped with a camera sensor network. We propose a probabilistic sensor tasking algorithm in which cameras sense the environment independently of one another, thus reducing the communication overhead. Since constant monitoring is prohibitively expensive with complex sensors such as cameras, the amount of sensing done is also minimized. To be effective, a minimum detection probability must be guaranteed by the system over *all* possible paths through the space. The straightforward approach of enumerating all such paths is intractable, since there is generally an infinite number of potential paths. Using a geometric decomposition of the space, we lower-bound the detection probability over all paths using a small number of linear constraints. The camera tasking is computed for set of example layouts and shows large performance gains with our probabilistic scheme over both constant monitoring as well as over a deterministic heuristic.

1 Introduction

Until recently, research in wireless sensor networks (WSN) has focused primarily on low cost, low bandwidth sensors. With dropping costs and advances in imaging technology, there is now increasing interest in camera sensor networks. Several platforms have already been developed for image/video acquisition in a sensor network setting [17, 13]. Cameras provide a higher level of sensor information, but also use more of the limited resources available to a wireless sensor node and so present a new set of challenges. If used continuously, cameras consume too much energy to operate on battery power. While applications such as target tracking usually require constant monitoring, it is unlikely that all areas being monitored will see continuous activity over time. We can achieve significant energy savings if we reduce the amount of sensing a camera node must do, with the goal of first detecting the target(s). Detection can provide a wake-up mechanism for more expensive higher level services such as tracking [16, 2], identity management [24], and occupancy reasoning [27].

J. Aspnes et al. (Eds.): DCOSS 2007, LNCS 4549, pp. 309–323, 2007.

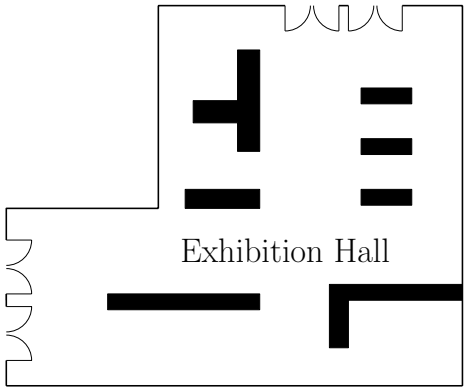


Fig. 1. An exhibition hall with two entrances/exits, allowing for several possible paths through it

In this paper we propose an energy efficient approach to detection in camera sensor networks. Our approach provides two benefits. First, after the initial setup is complete, *no* communication between nodes is required. The amount of sensing done is also minimized, while still providing guarantees on detection quality. Minimizing the use of the cameras is important since they consume as much energy as communication [18]. The algorithm itself is simple: at each time step, a camera independently decides to sense a frame with an assigned probability. These are set so that probabilistic guarantees on detection can be made.

Consider the scenario of tracking people with a camera sensor network in an exhibition hall with the floor plan in Figure 1. In the evening, the building is empty and the network should no longer monitor continuously because there is no one to track. When a person is detected, the network can wake up and begin tracking once again. One possible solution is to have the cameras continuously monitor only the entrances. This either assumes that the cameras have been specifically deployed for the purpose of detection or the cameras which face the entrances will be overused and may soon run out of energy. In this paper, we assume that the cameras are spread out to cover most of the space (as they would be deployed for monitoring or tracking). Surveillance and tracking are canonical applications of camera sensor networks, often aimed at detecting an adverserial intruder. The objective of the intruder is to move between sets of points called sources and destinations. Sources can be thought of as entrances into a space and destinations as secure areas (e.g. a bank vault). Although the intruder model is natural in security applications, the idea of focusing on paths between certain points in space is more general. For example in building monitoring, a destination may be a particular office or all points of a certain distance from the entrance, limiting how far an intruder may travel before being detected.

First we present related work and introduce the models used for the cameras and the space. After introducing the necessary geometric preprocessing of the camera layout, the problem of tasking the cameras subject to a global detection

probability constraint is posed in the framework of convex optimization. A deterministic algorithm with a detection probability of 1 is presented for comparison. Finally, analysis of the performance and validation of the algorithms are presented.

2 Related Work

In WSNs, work in tasking sensors has primarily addressed maintaining a minimum level of coverage over the entire space of the network. This is usually referred to as the k-coverage problem. Several centralized and distributed algorithms have been proposed for uniform [1, 15, 12] or differentiated coverage [26]. These all, however, assume a local sensor coverage model[1]. For camera sensor networks, a deployment algorithm which meets coverage constraints over a space was recently proposed in [10]. These types of coverage problems are all closely related the classical Art Gallery Problem and its many variants. For further details, the reader is directed to [21].

In [20, 25, 19], algorithms for finding maximal exposure and maximal breach paths through a sensor network with local sensors are described. The maximal exposure path is a path which the intruder is exposed to the most sensors. More relevant is the maximal breach path, which is the path of minimal exposure through a sensor network. This work again assumes constant sensing and finds the path through the network which maximally distances itself from all the sensors. The idea of barrier coverage was proposed recently in [14]. Barrier coverage refers to when active sensors form a barrier so that an intruder cannot pass through the network undetected. Results on the probability of random sampling needed to achieve a barrier in the network were given. However, the analysis was done for dense random networks equipped with local sensors and the barriers are static. Energy conservation through limiting the sensing to a small part of the network was also considered in [11, 23]. In [11], the activation pattern follows a user-defined path through the sensor network as a sentry. In [23], the activation pattern is a sweep across the network. Both schemes assume a simple topology and do not handle sensing holes.

Our work incorporates many similar ideas to the ones mentioned above. However, to the best of our knowledge, there is no prior work on providing energy efficient detection in a non-local camera sensor network.

3 Model

The detection area is modeled as a two dimensional polygonal space. It need not be convex or simple, as seen in the Exhibition Hall example in Figure 1. The space may have occlusions as well as regions not covered by any cameras. Sources and destinations are modeled as points in the space. These may be entrances or

[1] A local sensors refers to a sensor that can only detect things close to itself. The circular or Gaussian sensor model fall into this category.

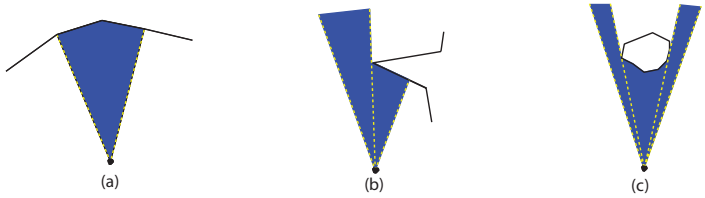


Fig. 2. Examples of visibility regions for cameras: (a) shows the simplest case where two constraint edges suffice, (b) shows an occlusion from one side, where an additional constraint edge is needed and (c) shows an occlusion in the middle of the camera field of view, where two additional constraint edges are needed

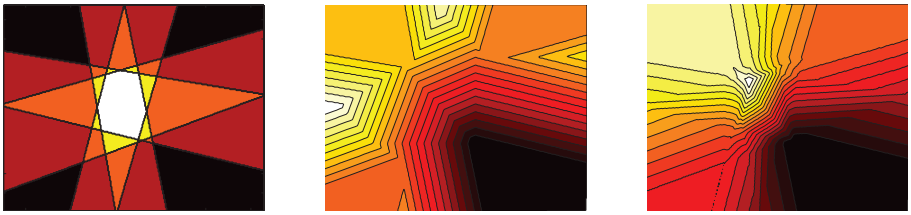
"areas of interest." A point may be a source in one setting and a destination in another. The camera's views are modeled by simple two dimensional cones. It is assumed that detection is uniform with probability 1 inside the cone and 0 outside. More complicated models of camera coverage such as a limited depth of view and varying probability of detection can be incorporated with minimal changes to the framework. Time is divided into discrete steps called *frames*. For energy consumption, we assume that taking a shot within a frame by a camera has a fixed cost. This implies that minimizing the energy used by the node to sense is equivalent to minimizing the probability that it senses during a frame. A shortcoming of this model is that it ignores the energy lost when the node is powering the camera up and powering it down. In future work, these costs will be accounted for.

We assume that the cameras know their locations, orientations, and have been properly calibrated. This could be done using structure from motion and automatic registration. The specific problem of camera calibration has been addressed in [8, 9, 22]. Furthermore, we assume that the cameras know the layout of the space in which they are deployed. This could be done by manually uploading a floorplan or learning it automatically with a scanning device [3]. There are no constraints on placement of the cameras or their viewing angles. The locations of sources and destinations along with their corresponding detection constraints are user-defined parameters and assumed to be known.

4 Geometric Preprocessing

The first step is to understand how the cameras cover the space. Beginning with the empty polygonal space, the two edges which make up a camera cone are placed into the space as constraint edges. In the presence of occlusions additional constraint edges are needed to define the boundary of the visibility cone of the camera. Some examples of where additional constraints are needed are shown in Figure 2. To find all the constraint edges, we implement the rotational sweep algorithm described in [10].

The constraint edges of all the camera cones decompose the space into polygonal faces. Each face is characterized by its neighboring faces and the combination



(a) (b) (c)

Fig. 3. (a) The coverage of 4 cameras facing each other in a rectangular space. The brighter the color the higher the coverage. (b) The distance cost function over the space from a point in the lower righthand corner, where the cost is 1 if the space is covered by a camera and 0 otherwise. Note the distortion due to the uncovered areas in the corners. (c) The cost function where the cost of traveling through the face is proportional to the distance times the number of cameras which see the face.

of cameras which see it. The computation of the arrangement of the visibility cones from the polygonal space and camera constraint edges is a well-studied problem in computational geometry with known efficient solutions [7]. To simplify the implementation, we compute the polygonal faces using the constrained Delaunay triangulation of the space and the camera constraint edges. The faces are reconstructed by joining adjacent triangles which are seen by the same combination of cameras.

From this decomposition, we extract the *spatial adjacency graph* (SAG). The SAG is an undirected graph $G(v, e)$ where each vertex represents a face in the space and two vertices are adjacent if the corresponding faces are adjacent. Two faces are adjacent if an intruder can move between the faces without entering any other face or equivalently, that the faces share a common vertex.

5 Tasking the Cameras

The algorithm tasks the cameras so that whichever path from the source to the destination the intruder chooses, there is high probability that he will be detected. First a simplified intruder motion model is considered. The results are then extended to a more general and realistic motion model.

5.1 Assigning Probabilities in a Graph

Consider any path the intruder can traverse through the space. The continuous path can be mapped to the SAG by marking the corresponding nodes for each face the path crosses. We first assume that at each time step, the intruder may only move one hop in the SAG. Under this simple motion model, each time slot can be considered an independent trial.

Given a path $\mathcal{P} = \{v_1, v_2, \ldots, v_n\}$ the probability of evasion[2] is

$$P_E = \prod_{v_i \in \mathcal{P}} p(v_i)\,, \tag{1}$$

where $p(v_i)$ is the probability that node v_i is not covered. Taking the logarithm of Equation 1, we get

$$\log(P_E) = \sum_{v_i \in \mathcal{P}} \log(p(v_i))\,. \tag{2}$$

Setting the cost of a node v_i to $\log(p(v_i))$, the cost of path $\mathcal{P}$ is defined as sum of the node costs along the path.

The probability that a node is covered depends on which cameras see the corresponding face and the probability with which they are on. If the set of cameras which cover node v_i is denoted by $\mathcal{C}(v_i)$, node v_i is uncovered if and only if the entire set of cameras $\mathcal{C}(v_i)$ are off. The cost of a node is then given by

$$\log(p(v_i)) = \sum_{c_j \in \mathcal{C}(v_i)} \log(1 - p(c_j))\,, \tag{3}$$

where $p(c_j)$ is the probability that camera c_j is on. Note that if a node is not covered by any cameras, its cost is 0 and negative otherwise.

Making the change of variables

$$\begin{aligned} x(c_j) &= -\log(1 - p(c_j))\,, \\ \epsilon &= -\log(P_E) \end{aligned}$$

and substituting into Equation 2, ϵ becomes a linear function of camera weights $x(c_i)$ along a given path. To ensure that P_E is suitably small, we need to ensure that no path of cost less than ϵ exists.

This is a difficult problem because the total cost of the path depends on the individual weights assigned to the cameras. One way to ensure that there is no path of cost less than ϵ is to enumerated all paths between the source and destination. With all the paths as constraints, we can optimize the weight vector $x(c_i)$ over any convex cost function using standard tools from convex optimization [4]. For example, to minimize the maximum amount of time any camera is on, we minimize $||x||_\infty$. Although this is an LP, the number of paths grows exponentially with the number of nodes in G. To reduce the number of constraints, we apply the following lemma:

Lemma 1. *If the cost of a node is set to $\frac{1}{d}$, where d is the shortest distance from the source to the destination through the node, then the total cost of the path cannot be less than 1.*

Proof. Since the length of the shortest path is d and the cost of each along the path step is at least $\frac{1}{d}$, it follows that the cost of any path is at least 1.

[2] That is, the probability that the intruder will not be detected.

Using this lower bound, we can provide a constraint for each node individually rather than along paths. If we set the cost of the node v_i to $\frac{\epsilon}{d(v_i)}$ where $d(v_i)$ is path length from v_i to the source plus the path length from v_i to the destination, then by Lemma 1, no path from the source to the destination will have a cost less than ϵ. The constraint for each node becomes

$$\sum_{c_j \in C(v_i)} \log(1 - p(c_j)) \geq \frac{\epsilon}{d(v_i)}\,, \quad \forall i\,. \tag{4}$$

The number of constraints is reduced from the number of paths to the number of nodes.

5.2 Assigning Probabilities in Continuous Space

Only assuming an upper bound on the speed of the intruder, we extend the results to the continuous domain. Given the framerates of the cameras, we use the maximum speed of the intruder to convert distance from standard units to frames. For example, if the frame rate of the camera is 15 frames/sec and the maximum speed of the intruder is 3 m/s then the conversion factor is 5 frames/m. The cost of $\mathcal{P}$ is the probability of evasion travelling along that path at the maximum speed. If $\mathcal{P}$ is within the visual field of a camera for a length of n frames and the probability of a camera taking a frame is p, the probability of evasion along $\mathcal{P}$ is $(1-p)^n$.

To determine the probability of each camera taking a frame, we need to be able to find the distance between two points in a polygonal space. An efficient algorithm is known from computational geometry/robot motion planning. We briefly outline the algorithm, but refer the reader to Chapter 15 of [7] for further details. The algorithm first constructs the *visibility graph* for a space S. The visibility graph consists of nodes representing the vertices of polygonal obstacles in S. An edge connects two vertices if the vertices are visible to one another (i.e. a straight line between the vertices does not intersect any obstacle). The weight of the edge is set as its Euclidean length. To find the shortest distance between two points, we place the points into the visibility graph and add the appropriate edges in the same way (to all visible vertices). Dijkstra's algorithm is used to compute the shortest path through the graph — exactly the shortest path through the space.

There may be regions of space not covered by any cameras which should not contribute to the computed distance as they have a cost of 0. These "holes" act as shortcuts through the space. To account for the holes, the visibility graph must be augmented. In addition to the vertices of the obstacles, there must be a vertex in the visibility graph for each uncovered face. Edges are added if the uncovered face is visible from a vertex. The weight of the edge is the shortest distance from the vertex to the polygon representing the uncovered face. In this augmented visibility graph, the shortest distance returned by Dijkstra will be correct. By comparing Figure 3(a) and Figure 3(b), we see that the distortion introduced by holes can be significant.

The camera probabilities must be set such that no path from the source to the destination has a smaller cost than ϵ. Unlike in the graph case where paths could be enumerated, in the continuous space there are an infinite number of paths. By extending Lemma 1 to the continuous case, we show that one constraint per face is sufficient to guarantee the minimum cost over all paths. The key idea is to find a shortest path *passing through* each face. A path is considered to pass through a face if it intersects or touches any part of the boundary of that face (i.e. it does not have to enter the face). Now we can state the following theorem.

Theorem 1. *For a path through the space from a source to a destination, its length is defined as the distance the path traverses through covered faces. The length does not increase when the path travels through uncovered faces or equivalently each uncovered face is mapped to a point. If the cost of each covered face is set to $\frac{\epsilon}{d}$, where d is the length (as defined above) of the shortest path from the source to the destination passing through the face, then all paths through the space will have a cost of at least ϵ.*

Proof. First note that by definition, any uncovered face has a fixed cost of 0. Therefore, if an uncovered path exists from the source to the destination then the minimum cost from source to destination is 0 and an intruder can traverse the path without being detected with probability 1. Consider a face and the point in the face lying on the shortest path from the source to the destination through the face. In general, neither the point nor the path are unique, however since the value is unique, since it is defined as the minimum. If the shortest total path length is d then we set the cost of the face to $\frac{\epsilon}{d}$. By Lemma 1, all other faces that the path goes through must have equal or greater cost. Therefore, the total cost of the path will be at least ϵ. By similar argument, any other path through the face will have an equal or higher cost, because all sections of the path will have an equal or higher cost.

Since the cost in each face is split between several cameras, all the faces must be sampled. Each face represents a combination of cameras and we do not know a priori which combination of constraints will be active. However, by finding the minimum length path through every face, all possible combinations of cameras that occur in the space are enumerated. If a path of cost lower than ϵ and larger than 0 existed, it would imply that the path crosses a face where the above constraint is not met. Since the constraint for each face is enumerated, this is a contradiction.

5.3 Algorithm

Given a space and camera positions, we find the decomposition of space into faces. Each face is marked with the combination of cameras which see it. Here we only consider the case of one source and one destination but the algorithm extends to multiple sources and destinations. For each face i, we sample the boundary and use the visibility graph algorithm to find the smallest distance from the source to the destination through the face, d_i, by computing the shortest

distance to the source and the destination from each point along the boundary of the face. Defining the vector a_i, where $a_{ij} = 1$ if camera j sees face i and 0 otherwise. The optimization problem can be written as

$$\begin{aligned} \text{minimize} \quad & f(x)\,, \\ \text{subject to} \quad & a_i^T x > \frac{\epsilon}{d_i} \quad \forall i\,, \text{and} \\ & x > 0 \end{aligned}$$

where $f(x)$ can be any convex function. Then we change $x(c_i)$ back to $p(c_i)$ by

$$p(c_i) = 1 - e^{-x(c_i)} \tag{5}$$

to obtain the vector of camera probabilities.

5.4 Special Case

The special case where we assign one probability to all the cameras can be solved exactly rather than using the constraint at each face. Minimizing this probability is equivalent to minimizing the ∞-norm of the probability vector. The optimal probability can be found if the minimum exposure path with all cameras on is known. The cost of a path through a face is the length of the path through the face multiplied by the number of cameras which see the face, which is the total number of potential frames which the intruder will be exposed to while inside the face. Since each face can have a different cost, the visibility graph cannot be used for computing the minimum exposure path. To find the minimum exposure path, we discretize the space into a grid with 8 neighbors per node. The incoming edge into each node is assigned a cost of the number of cameras which see the node. Then Dijkstra's algorithm is applied to the grid graph to find the minimum cost path from the source to the destination. This gives a good approximation of the true minimum cost. It is simple to show that the discretization error is less than 8% if the source and destination are reasonably far apart compared to the distance between grid points. The distance function with all the cameras on for a simple case is shown in Figure 3(c) where the minimum cost path will clearly need not be the Euclidean shortest path.

The minimum cost is equivalent to the total number of potential frames which will be taken along the path. The required probability is set so that the minimum exposure cost has at least a cost of ϵ. All the camera probabilities are then set to this value. This method only works because the relative weighting of the faces do not change. Changing the relative costs of the faces results in a more difficult version of setting the node weights of the graph in section 5.1, since there is an uncountable number of paths.

6 Deterministic Algorithm

We present a simple heuristic for deterministically detecting intruders as a benchmark to compare the performance of our randomized strategy with. The idea

is similar to the algorithm presented in [23]. We try to sweep the space using the cameras while maintaining a barrier between the uncleared or contaminated areas and the destination. By maintaining a barrier, we prevent the intruder from getting closer to the destination and the cleared area from becoming re-contaminated. If we can grow this area, we will eventually clear the entire space.

The deterministic approach has some inherent drawbacks. First, it requires synchronization between the nodes, resulting in communication overhead. Secondly, regions which are not covered by any cameras can never be cleared. This implies that there must always be a separating sweep from the destination to the first hole in coverage, since a potential intruder may hide in it. For the purposes of this comparison, we ignore both issues and compare the deterministic heuristic with the random scheme. The algorithm consists of two parts. First, a schedule of faces which must be turned on at a given time is computed. Then a set of cameras are found which cover the required faces.

To find a set of faces which maintain a barrier and increase the cleared region, we return to the SAG described in Section 4. The sweep begins at the destination, so we mark the appropriate face in the SAG as the choice at time 0. At time 1, all of its adjacent faces are marked to be on. This fulfills both criteria of maintaining a barrier with uncleared regions and expanding the cleared area. In general, if the chosen set of faces at time t is $\mathcal{S}(t)$, then $\mathcal{S}(t+1)$ will be all the adjacent faces of the faces in $\mathcal{S}(t)$ which have not yet been cleared. This is repeated until the source is reached. For the purposes of comparison, we sweep the entire region before repeating the sweep beginning at the initial face.

The second part of the algorithm requires us to choose a set of cameras at each time instance. For each time instance, we must solve the *minimal hitting set problem*. Each face has a set of cameras which see it. The set of faces to be covered at time t form a collection of sets of cameras. We must choose a set of cameras which cover all the sets in the collection. Since, we are solving this problem many times, cameras are chosen using the following method. At $t = 0$, all cameras are assigned a weight of 0. First, all the cameras in the collection are sorted in increasing order according to their weight. From the minimum weight, we find the camera which appears in the most sets (i.e. sees the most faces). This camera is chosen and its weight is increased by 1. It is removed from the list and the procedure is repeated until all the sets are removed. The re-weighting gives preference to cameras which have been chosen fewer times.

7 Simulation Experiments

In this section, we investigate the performance of the algorithm through simulations. The geometric preprocessing was implemented in CGAL [5] and the optimization was done using the CVX [6] package in MATLAB. The four layouts shown in Figure 4(a)-(d) were tested. Layout 1 and 2 are rectangular rooms with differing camera layouts. Layout 1 corresponds to a real camera deployment of 16 webcams. Layout 2 has 14 cameras placed more evenly around the room. Layouts 3 and 4 are larger with multiple occlusions. Although there are

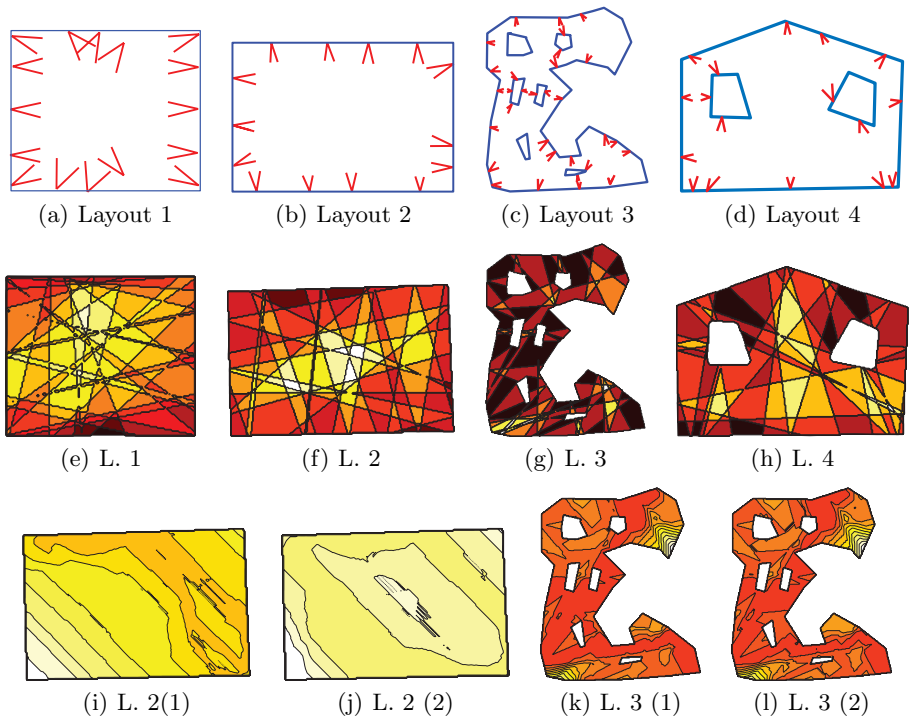
more cameras in the latter layouts (31 and 17 respectively), the coverage is not as dense as in the first two cases. The extent of the coverage is shown in Figure 4(e)-(h) where more densely covered areas are lighter in color. For each layout, we compare the results in terms of load balancing between cameras and the total amount of energy spent.

The results summarized in Table 1 were obtained by setting the probability of evasion to 0.01. We compare optimizing the 2-norm and the ∞-norm by bounding the detection probability on individual faces ($||p||_2(1)$; $||p||_\infty||(1)$), the special case of solving for one parameter ($||p||_\infty(2)$) and the deterministic algorithm. The first three columns show the the 2-norm, the 2-norm normalized by the number of cameras, and the ∞-norm of the probability vector respectively. With the probability vector set to the solution, the maximum probability of evasion along the minimum exposure path was computed and is shown in the fourth column of the table. The exception is the deterministic scheme which has a probability of evasion of 0. In all 4 layouts, the deterministic scheme used the cameras much more than any of the probabilistic schemes, illustrating the price of setting P_E to 0.

In Layouts 1 and 2, locally enforcing the minimum detection probability at each face results in overcoverage. The resulting maximum P_E was at least an order of magnitude smaller than the desired 0.01. At the same time the total energy used ($||p||_2$) is at most twice the amount used in the optimal single parameter case. In Layout 3, the performance of the three probabilistic schemes is comparable. The overcoverage is minimal because most of the area is sparsely covered and most of the paths are of similar length. Layout 4 is lies somewhere between. The maximum P_E is still an order of magnitude smaller than the single parameter case, but all other values are comparable.

These results show that the optimization technique provides very good performance giving quite low probabilities of evasion with each camera taking only a small number of frames. Unfortunately, the approximation in the local enforcement of constraints using Theorem 1, makes it difficult to set the true probability of evasion to a desired value. This means that some experimentation is necessary if we do not want overcoverage in the network. The effect of the optimization on coverage be seen in Figure 4(i)-(l). This shows the cost of paths from the source to the destination over the space when the probabilities are set to the 2-norm solution for Layouts 2 and 3. For Layout 2, the coverage becomes much more uniform than in the case of one parameter, while Layout 3 exhibits almost no change.

The notion of load balancing is important because it prevents certain cameras from being overused. A good metric of load balancing is obtained by comparing the average value of the probability vector to its ∞-norm. The average value is given by $||p||_2/N$ in Table 1. The single parameter case obviously has a ratio of 1 in all cases. The other algorithms did nearly as well. For method (1), the maximum was always less than twice the average. Given a particular layout it may not be desirable to achieve perfect load balancing as there may be some cameras which inherently need to do more sensing than others. The deterministic



(a) Layout 1 (b) Layout 2 (c) Layout 3 (d) Layout 4

(e) L. 1 (f) L. 2 (g) L. 3 (h) L. 4

(i) L. 2(1) (j) L. 2 (2) (k) L. 3 (1) (l) L. 3 (2)

Fig. 4. (a)-(d) 4 example layouts. (e)-(h) Coverage maps for the 4 layouts. The lighter the color of a point, the more cameras cover that point. (i) The minimum total cost from the source to the destination for layout 2 with all the cameras set to one probability. The darkest region is the minimum cost path. (j) The minimum total cost path with the prob. from the approx. optimizaion (k) and (l) The same for layout 3.

algorithm also performed well. However, when there were disjoint faces covered by one camera, it was impossible prevent turning one camera on multiple times.

8 Conclusions and Further Work

We have presented an algorithm for detection where cameras take frames with an assigned probability rather than continuously monitor the space. The camera probabilities can be optimized over a desired convex cost function with relatively few constraints. Once the probabilities are set, no further coordination is required between the nodes. Experimentation shows that although each camera samples infrequently and independently, global guarantees are made on the detection probability.

From the experimentation, we see that locally enforcing the probability of evasion at each face results in overcoverage. Whether it is feasible to efficiently encode the minimum cost over all paths when cameras have variable probabilities is an open question. Furthermore, we assumed that the main energy cost in

Table 1. Results for the 4 layouts. (1) refers to the approximate technique; (2) refers to the optimal ∞-norm value; Det.is the deterministic algorithm

Layout	Method	$\|\|p\|\|_2$	$\|\|p\|\|_2/N$	$\|\|p\|\|_\infty$	Max P_E
1	$\|\|p\|\|_2(1)$	0.0608	0.0152	0.0262	0.0005
	$\|\|p\|\|_\infty(1)$	0.0785	0.0196	0.0262	0.00002
	$\|\|p\|\|_\infty(2)$	0.0320	0.0080	0.008	0.01
	Det.	0.0675	0.2166	0.4545	-
2	$\|\|p\|\|_2(1)$	0.0512	0.0137	0.0193	0.001
	$\|\|p\|\|_\infty(1)$	0.0646	0.0173	0.0193	0.00012
	$\|\|p\|\|_\infty(2)$	0.0344	0.0092	0.0092	0.01
	Det.	1.387	0.3707	0.3722	-
3	$\|\|p\|\|_2(1)$	0.1921	0.0345	0.0357	0.0094
	$\|\|p\|\|_\infty(1)$	0.1947	0.0353	0.0353	0.0093
	$\|\|p\|\|_\infty(2)$	0.1954	0.0351	0.0351	0.01
	Det.	1.400	0.3501	0.3510	-
4	$\|\|p\|\|_2(1)$	0.0817	0.0198	0.0228	0.0009
	$\|\|p\|\|_\infty(1)$	0.0871	0.0211	0.0228	0.0008
	$\|\|p\|\|_\infty(2)$	0.0614	0.0149	0.0149	0.01
	Det.	0.7717	0.1038	0.5729	-

acquiring frames was in actually sampling the frame. Taking the cost of powering on and off into account introduces correlation between the detection probabilities over time complicating the problem significantly.

Acknowledgement. The authors wish to acknowledge support by NSF grants CNS-0435111, CNS-0626151, ARO grant W911NF-06-1-0275, and the DoD Multidisciplinary University Research Initiative (MURI) program administered by ONR under Grant N00014-00-1-0637.

References

[1] Abrams, Z., Goel, A., Plotkin, S.: Set k-cover algorithms for energy efficient monitoring in wireless sensor networks. In: IPSN '04: Proceedings of the third international symposium on Information processing in sensor networks, pp. 424–432. ACM Press, New York (2004)

[2] Arora, A., Dutta, P., Bapat, S., Kulathumani, V., Zhang, H., Naik, V., Mittal, V., Cao, H., Demirbas, M., Gouda, M., Choi, Y., Herman, T., Kulkarni, S., Arumugam, U., Nesterenko, M., Vora, A., Miyashita, M.: A line in the sand: a wireless sensor network for target detection, classification, and tracking. Comput. Networks 46(5), 605–634 (2004)

[3] Biber, P., Fleck, S., Wand, M., Staneke, D., Strasser, W.: First experiences with a mobile platform for flexible 3d model aquisition in indoor and outdoor environments - the waglele. In: 3D-ARCH'2005: 3D Virtual Reconstruction and Visualization of Complex Architectures (2005)

[4] Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)

[5] Cgal: Computational geometry algorithms library. http://www.cgal.org
[6] Cvx: Matlab software for disciplined convex programming. http://http://www.stanford.edu/~boyd/cvx/
[7] de Berg, M., van Kreveld, M., Overmars, M., Schwartzkopf, O.: Computational Geometry - Algorithms and Applications. Springer, Heidelberg (2000)
[8] Devarajan, D., Radke, R.J., Chung, H.: Distributed metric calibration of ad hoc camera networks. ACM Trans. Sen. Netw. 2(3), 380–403 (2006)
[9] Devarajan, D., Radke, R.J., Chung, H.: Distributed metric calibration of ad hoc camera networks. ACM Trans. Sen. Netw. 2(3), 380–403 (2006)
[10] Erdem, U.M., Sclaroff, S.: Automated camera layout to satisfy task-specific and floor plan-specific coverage requirements. Comput. Vis. Image Underst. 103(3), 156–169 (2006)
[11] Gui, C., Mohapatra, P.: Virtual patrol: a new power conservation design for surveillance using sensor networks. In: IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks, Piscataway, NJ, USA, p. 33. IEEE Press, New York (2005)
[12] Huang, C.-F., Tseng, Y.-C.: The coverage problem in a wireless sensor network. In: WSNA '03: Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications, pp. 115–121. ACM Press, New York (2003)
[13] Kulkarni, P., Ganesan, D., Shenoy, P., Lu, Q.: Senseye: a multi-tier camera sensor network. In: MULTIMEDIA '05: Proceedings of the 13th annual ACM international conference on Multimedia, pp. 229–238. ACM Press, New York (2005)
[14] Kumar, S., Lai, T.H., Arora, A.: Barrier coverage with wireless sensors. In: Santosh Kumar, T.H. (ed.) MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking, pp. 284–298. ACM Press, New York (2005)
[15] Kumar, S., Lai, T.H., Balogh, J.: On k-coverage in a mostly sleeping sensor network. In: Santosh Kumar, T.H. (ed.) MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking, pp. 144–158. ACM Press, New York (2004)
[16] Li, D., Wong, K., Hu, Y., Sayeed, A.: Detection, classification, and tracking of targets. IEEE Signal Processing Magazine 19(2), 17–30 (2002)
[17] Margi, C.B., Lu, X., Zhang, G., Stanek, G., Manduchi, R., Obraczka, K.: Meerkats: A power-aware, self-managing wireless camera network for wide area monitoring. In: Workshop on Distributed Smart Cameras (DSC-06) (October 2006)
[18] Margi, C.B., Petkov, V., Obraczka, K., Manduchi, R.: Characterizing energy consumption in a visual sensor network testbed. In: 2nd International IEEE/Create-Net Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (March 2006)
[19] Megerian, S., Koushanfar, F., Potkonjak, M., Srivastava, M.B.: Worst and best-case coverage in sensor networks. IEEE Transactions on Mobile Computing 4(1), 84–92 (2005)
[20] Megerian, S., Koushanfar, F., Qu, G., Veltri, G., Potkonjak, M.: Exposure in wireless sensor networks: theory and practical solutions. Wirel. Netw. 8(5), 443–454 (2002)
[21] O'Rourke, J.: Art gallery theorems and algorithms. Oxford University Press, Oxford (1987)
[22] Rekletis, I.M., Dudek, G.: Automated calibration of a camera sensor network. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 401–406, Edmonton Alberta, Canada (August 2-6, 2005)

[23] Ren, S., Li, Q., Wang, H., Zhang, X.: Design and analysis of wave sensing scheduling protocols for object-tracking applications. In: Prasanna, V.K., Iyengar, S., Spirakis, P.G., Welsh, M. (eds.) DCOSS 2005. LNCS, vol. 3560, Springer, Heidelberg (2005)
[24] Shin, J., Guibas, L., Zhao, F.: A distributed algorithm for managing multi-target identities in wireless ad-hoc sensor networks. In: Proc. 2nd International Workshop on Information Processing in Sensor Networks (IPSN03), Palo Alto, CA, pp. 223–238. Springer, Heidelberg (2003)
[25] Veltri, G., Huang, Q., Qu, G., Potkonjak, M.: Minimal and maximal exposure path algorithms for wireless embedded sensor networks. In: SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems, pp. 40–50. ACM Press, New York (2003)
[26] Yan, T., He, T., Stankovic, J.A.: Differentiated surveillance for sensor networks. In: SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems, pp. 51–62. ACM Press, New York (2003)
[27] Yang, D., Gonzalez-Banos, H., Guibas, L.: Counting people in crowds with a real-time network of image sensors. In: Proc. IEEE ICCV (2003)