Approximate Initialization of Camera Sensor Networks*

Purushottam Kulkarni, Prashant Shenoy, and Deepak Ganesan

Department of Computer Science
University of Massachusetts, Amherst, MA 01003
{purukulk, shenoy, dganesan}@cs.umass.edu

Abstract. Camera sensor networks—wireless networks of low-power imaging sensors—have become popular recently for monitoring applications. In this paper, we argue that traditional vision-based techniques for calibrating cameras are not directly suitable for low-power sensors deployed in remote locations. We propose approximate techniques to determine the relative locations and orientations of camera sensors without any use of landmarks or positioning technologies. Our techniques determine the degree and range of overlap for each camera and show this information can be exploited for duty cycling and triggered wakeups. We implement our techniques on a Mote testbed and conduct a detailed experimental evaluation. Our results show that our approximate techniques can estimate the degree and region of overlaps to within 10% of their actual values and this error is tolerable at the application-level for effective duty-cycling and wakeups.

1 Introduction

1.1 Motivation

Wireless sensor networks have received considerable research attention over the past decade, and rapid advances in technology have led to a spectrum of choices of image sensors, embedded platforms, and communication capabilities. Consequently, camera sensor networks— networks consisting of low-power imaging sensors [18, 19]—have become popular for applications such as environmental monitoring and surveillance.
Regardless of the end-application, camera sensor networks perform several common tasks such as object detection, recognition, and tracking. While object detection involves determining when a new object appears in range of the camera sensors, recognition involves determining the type of the object, and tracking involves using multiple camera sensors to continuously monitor the object as it moves through the environment. To effectively perform these tasks, the camera sensor network needs to be *calibrated* at setup time. Calibration involves determining the location and orientation of each camera sensor. The location of a camera is its position (3D coordinates) in a reference coordinate system, while orientation is the direction in which the camera points. By determining these parameters for all sensors, it is possible to determine the viewable range of each camera and what portion of the environment is covered by one or more cameras. The relationship with other nearby cameras, in particular, the overlap in the

* This work was supported in part by National Science Foundation grants EEC-0313747, CNS-0219520, CNS-052072 and EIA-0098060.

viewable ranges of neighboring cameras can be determined. This information can be used by applications to determine which camera should be used to sense an object at a certain location, to triangulate the position of an object using overlapping cameras, and to handoff tracking responsibilities from one camera to another as the object moves.
Calibration of camera sensors is well-studied in the computer vision community and a number of techniques to accurately estimate the location and orientation of cameras have been proposed [8, 22, 24]. These techniques assume coordinates of few landmarks are known a priori and use the projection of these landmarks on the camera's image plane, in conjunction with principles of optics, to determine a camera's coordinates and orientation.[1] In certain cases locations of landmarks are themselves determined using range estimates from known locations; for instance, a positioning technology such as Cricket can be used to determine the coordinates of landmarks from known beacon locations. However, these techniques are not feasible for deployments of ad-hoc low power camera sensors for the following reasons: (i) **Resource constraints:** Vision-based techniques for accurate calibration of cameras are compute intensive. Low-power cameras do not have the computation capabilities to execute these complex mathematical tasks. Further, images of low-power cameras are often of low fidelity and not well suited for high precision calibration, (ii) **Availability of landmarks:** In many scenarios, ad-hoc camera sensor networks are deployed in remote locations for monitoring mountainous and forest habitats or for monitoring natural disasters such as floods or forest fires. No landmarks may be available in remote inhabited locations, and infrastructure support such as positioning technologies may be unavailable or destroyed, making it difficult to define new landmarks.
One solution that eliminates the need to use landmarks is it to equip each camera sensor with a positioning device such as GPS [4] and a directional digital compass [6], which enable direct determination of the node location and orientation. However, today's GPS technology has far too much error to be practical for calibration purposes (GPS can localize an object to within 5-15m of its actual position). Ultrasound-based positioning and ranging technology [16] is an alternative which provides greater accuracy. But use of additional hardware with low-power cameras both consumes more energy and in some cases, can be prohibitive due to its cost. As a result, accurate calibration is not always feasible for initialization of resource-constrained camera sensor networks with limited or no infrastructure support.
Due to these constraints, in this paper we ask a fundamental question: *is it possible to initialize camera sensors without the use of known landmarks or without using any positioning technology?* In scenarios where accurate camera calibration may not always be feasible, determining relative relationships between nearby sensor nodes may be the only available option. This raises the following questions:

– How can we determine relative locations and orientations of camera sensors without use of known landmarks or positioning infrastructure?
– What kind of accuracy can these approximate initialization techniques provide?
– What is the performance of applications based on approximate initialization?

[1] Vision-based calibration techniques can also determine a camera's internal parameters such as the camera focal length and lens distortion, in addition to external parameters such as location and orientation.

1.2 Research Contributions

To address the above challenges, in this paper, we propose novel *approximate initialization* techniques for camera sensors. Our techniques rely only on the inherent picture-taking ability of cameras and judicious use of on-board computational resources to initialize each camera relative to other cameras in the system. No infrastructure support for beaconing, range estimation or triangulation is assumed. Our initialization techniques are computationally lightweight and easily instantiable in environments with little or no infrastructure support and are well suited for resource-constrained camera sensors.
Our techniques rely on two key parameters—the *degree of overlap* of a camera with other cameras, and the *region of overlap* for each camera. We present approximate techniques to estimate these parameters by taking pictures of a randomly placed reference object. To quantify the accuracy of our methods, we implement two techniques—duty-cycling and triggered wakeup—that exploit this initialization information.
We have implemented our initialization techniques on a testbed of Cyclops [18] cameras and Intel Crossbow Motes [14] and have conducted a detailed evaluation using the testbed and simulations. Our experiments yield the following results:

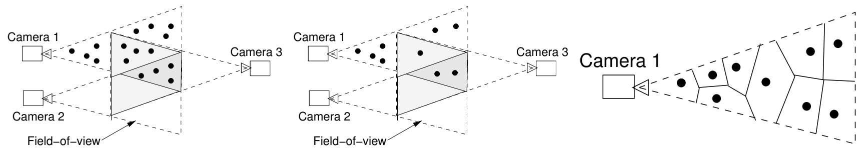
– Our approximate initialization techniques can estimate both k-overlap and region of overlap to within 10% of the actual values.
– The approximation techniques can handle and correct for skews in the distribution of reference point locations.
– The application-level accuracy using our techniques is 95-100% for determining the duty-cycle parameter and 80% for a triggered wakeup application.

2 Problem Formulation

We consider a wireless network of camera sensors deployed in an ad-hoc fashion with no a priori planning. Each sensor node is assumed to consist of a low-power imaging sensor such as the Cyclops [18] or the CMUCam [19] connected to an embedded sensor platform such as the Crossbow Mote [14] or the Telos [15]. No positioning hardware is assumed to be present on the nodes or in the environment. Given such an ad-hoc camera sensor network, our goal is to determine the following parameters for each sensor node:

– *Degree of overlap*, which is the fraction of the viewable range that overlaps with other nearby cameras; specifically we are interested in the *k-overlap*, which is the fraction of the viewable region that overlaps with exactly k other cameras.
– *Region of overlap*, which is the spatial volume within the viewable region that overlaps with another camera. While the degree of overlap indicates the extent of the viewable region that overlaps with another camera, it does not indicate *which* portion of the viewable range is covered by another camera. The region of overlap captures this spatial overlap and is defined as the 3D intersection of the viewable regions of any pair of cameras.

Our goal is to estimate these parameters using the inherent picture-taking capability of cameras. We assume the presence of a reference object that can be manually placed at different locations in the environment; while the coordinates of the reference object



(a) Uniform distribution (b) Skewed Distribution (c) Weighted Approximation

Fig. 2. k-overlap estimation with distribution of reference points.

are *unknown*, the sensors can take pictures to determine if the object can be viewed simultaneously by two or more cameras from a particular location. Our goal is to design techniques that use this information to determine the degree and region of overlap for the various nodes. The physical dimensions of the reference object as well as the focal length f of each camera is assumed to be known a priori.

3 Approximate Initialization

In this section, we describe approximate techniques to determine the *degree of overlap* and *region of overlap* for camera sensors.

3.1 Determining the Degree of Overlap

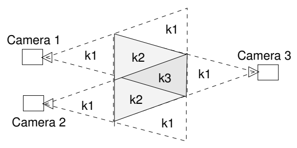


Fig. 1. Different degrees of overlap (k-overlap) for a camera.

As indicated earlier, degree of overlap is defined by the k-***overlap***, which is the fraction of the viewing area simultaneously covered by *exactly* k cameras. Thus, 1-overlap is the fraction of a camera's viewable region that does not overlap with any other sensor; 2-overlap is the fraction of region viewable to itself and one other camera, and so on. This is illustrated in Figure 1 where k_1 denotes the region covered by a single camera, k_2 and k_3 denote the regions covered by two and three cameras, respectively. It follows that the union of the k-overlap regions of a camera is exactly the total viewable range of that camera (i.e., the sum of the k-overlap fractions is 1). Our goal is to determine the k-overlap for each camera, $k = 1 \dots n$, where n is the total number of sensors in the system.

Estimating k-overlap Our approximate technique employs random sampling of the three-dimensional space to determine the k-overlap for each camera sensor. This is done by placing an easily identifiable reference object at randomly chosen locations and by having the camera sensors take pictures of the object. Let each object location be denoted as a reference point (with unknown coordinates). Each camera then processes its pictures to determine which reference points are visible to it. By determining the subset of the reference points that are visible to multiple cameras, we can estimate the k-overlap fractions for various sensors. Suppose that r_i reference points from the total set are visible to camera i. From these r_i reference points, let r_i^k denote the reference points that are simultaneously visible to exactly k cameras. Assuming an uniform distribution of reference points in the environments, the k-overlap for camera i is given by

$$O_i^k = \frac{r_i^k}{r_i} \tag{1}$$

Depending on the density of reference points, error in the estimate of O_i^K can be controlled. The procedure is illustrated in Figure 2(a), where there are 16 reference points visible to camera 1, of which 8 are visible only to itself, 4 are visible to cameras 1 and 3 and another 4 to cameras 1, 2, and 3. This yields a 1-overlap of 0.5, 2-overlap and 3-overlap of 0.25 for camera 1. k-overlaps for other cameras can be similarly determined.

Handling skewed reference point distributions The *k-overlap* estimation technique presented above assumes uniform distribution of reference points in the environment. In reality, due to the ad-hoc nature of the deployment and the need to calibrate the system online in the field, the placement of reference objects at randomly chosen locations will not be uniform. The resulting error due to a non-uniform distribution is illustrated in Figure 2(b), where our technique estimates the 1-, 2- and 3-overlap for camera 1 as $\frac{2}{3}, \frac{2}{9}, \frac{1}{9}$ as opposed to the true values of $\frac{1}{2}, \frac{1}{4} and \frac{1}{4}$ respectively. Thus, we need to enhance our technique to correct for skews in the reference point distribution.
The basic idea behind our enhancement is to assign a weight to each reference point, where the weight denotes the volume that it represents. Specifically, points in dense populated region are given smaller weights and those in sparely populated regions are given higher weights. Since *a higher weight can compensate for the scarcity of reference points in sparely populated region*, we can correct for skewed distributions of reference points. Our enhancement is based on the computational geometry technique called ***Voronoi tessellation*** [5]. In two dimensions, a Voronoi tessellation of a set of points is the partitioning of the plane into convex polygons such that all polygons contain a single generating point and all points within a polygon are closest to the corresponding generating point. Figure 2(c) shows a skewed distribution of reference points in the 2D viewing area of a camera and the corresponding Voronoi tessellation. Each reference point in the camera is contained within a cell, with all points in a cell closest to the corresponding reference point. Given a skewed distribution of reference points, it follows that densely situated points will be contained within smaller polygons, and sparsely situated points in larger polygons. Since the size of each polygon is related to the density of the points in the neighborhood, it can be used as an approximation of the area represented by each point. Voronoi tessellations can be extended to points in three dimensions, with each point contained with a 3D cell instead of a polygon.
Using Voronoi tessellation, each reference point is assigned a weight that is approximately equal to volume of the cell that it lies in. The k-overlap is then computed as

$$O_i^k = \frac{w_i^k}{w_i} \tag{2}$$

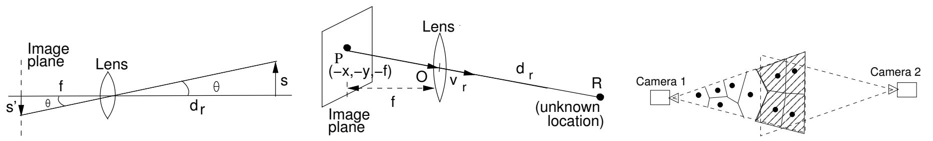
where w_i^k is the cumulative weight of all reference points that are simultaneously visible to exactly k cameras and w_i is the total weight of all the cells in the viewable region of camera i. Observe that when the reference points are uniformly distributed, each point gets an equal weight, and the above equation reduces to Equation 1. Note that Voronoi tessellation requires the coordinates of reference points in order to partition the viewable region into cells or polygons. Section 3.2 describes how to approximately estimate this in .

Approximate Tessellation Since tessellation is a compute-intensive procedure that might overwhelm the limited computational resources on a sensor node, we have developed an approximation. Instead of tessellating the 3D viewing region of a camera into polyhedrons, a computationally expensive task, the viewing region is discretized into smaller cubes. For each cube, the closest viewable reference point from the center of the cube is calculated. The volume of the cube is added to the weight of that reference point. When all cubes are associated and their volumes added to the respective reference points, the weight of each reference points is in proportion to the density of points in the vicinity—points in less dense regions will have higher weights than points in less dense regions, thereby yielding an approximation of the tessellation process.

3.2 Determining the Region of Overlap

Since k-overlap only indicates the extent of overlap but does not specify *where* the overlap exists, our techniques also determine *region of overlap* for each camera. Like before, we assume a reference object placed at randomly chosen locations. Using these points, first a Voronoi tessellation of the viewing area is obtained for each camera. The *region of overlap* for any two cameras C_i and C_j is simply the the union of cells containing all reference points simultaneously visible to the two cameras. Figure 3(c) shows the Voronoi tessellation of the 2D viewing region of camera 1, the reference points viewable by cameras 1 and 2, and the approximate *region of overlap* (shaded region) for (C_1, C_2). Thus, our approximate tessellation (described in Section 3.1) can be used to determine the region of overlap for all pairs of cameras in the system.

Estimating reference point locations As indicated before, the tessellation process requires the locations of reference points. Since no infrastructure is available, we present a technique to estimate these locations using principles of optics. A key insight is that *if each camera can determine the coordinates of visible reference points relative to itself, then tessellation is feasible—absolute coordinates are not required.* Assuming the origin lies at the center of the lens, the relative coordinates of a point are defined as (d_r, v_r), where d_r is its distance from the origin, and $\boldsymbol{v_r}$ is a vector from the origin in the direction of the reference point that defines its orientation in 3D space.
We illustrate how to determine the distance d_r from the camera in 2-dimensions. We have assumed that the size of the reference object is known a prior, say s. The focal length f is also known. Then the camera first estimates the size of the image projected by the object—this is done by computing the bounding box around the image, determining the size in pixels and using the size of the CMOS sensor to determine the size of those many pixels. If s' denotes the size of the image projected by the reference object



(a) Object and image relation in 2D (b) Estimation of reference point location (c) Region of overlap using Voronoi tessellation

Fig. 3. Region of overlap estimation using reference points and Voronoi tessellation.

on the camera, then from Figure 3(a) , the following condition holds

$$tan\theta = \frac{s}{d_r} = \frac{s'}{f} \tag{3}$$

Since s, s' and f are known, d_r can be computed. A similar idea holds in 3D space where instead of size, area of the object has to be considered.
Next, to determine the orientation of the reference point relative to the camera, assume that the reference object projects an image at pixel coordinates (x, y) on the image plane of the camera. Then the vector $\boldsymbol{v_r}$ has the same orientation as the vector that joins the centroid of the image to center of the lens (i.e., the origin). As shown in Figure 3(b), the vector $\boldsymbol{PO} = (x, y, f)$ has the same orientation as $\boldsymbol{v_r}$, where O is the origin and P is the centroid of the image with coordinates $(-x, -y, -f)$. Since (x, y) can be determined by processing the image and f is known, the relative orientation of the reference point can be determined.

4 Applications

In this section, we describe how camera that are initialized approximately can satisfy application requirements.

4.1 Duty-Cycling

Duty-cycling is a technique to operate sensors in cycles of ON and OFF durations to increase lifetime while providing the desired event-detection reliability and also to bound the maximum time to detect an event. The duty-cycling parameter d is commonly defined as the fraction of time a sensor is ON. An important criteria in deciding the duty-cycle parameter is the degree of overlap. Sensors with high coverage redundancy can be operated at low duty cycles to provide desired event detection probability, whereas those with lower redundancy will require higher duty cycles. One of the several techniques to estimate the duty-cycle parameter based on degree of overlap is as follows,

$$d_i = \sum_{k=1}^{n} O_i^k \times \frac{1}{k} \tag{4}$$

where, d_i is the duty-cycle parameter of camera i, O_i^k the fraction of k-overlap with the neighboring cameras and n the total number of cameras. The intuition is to duty-cycle each camera in proportion to its degree of overlap with neighboring cameras.

4.2 Triggered Wakeup

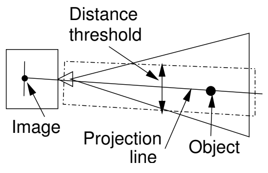
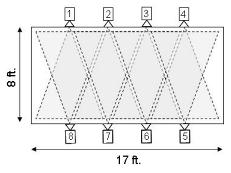


Fig. 4. Region of overlap for triggered wakeup.

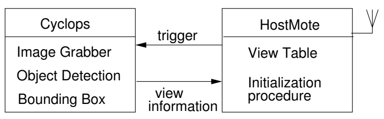
Object tracking involves continuous monitoring of an object—as the object moves from the range of one camera to another, tracking responsibilities are transferred via a handoff. Since cameras may be duty-cycled, such a handoff involves a triggered wakeup to ensure that the destination camera is awake. A naive solution is to send triggered wakeups to all overlapping cameras and have one of them take over the tracking. While doing so ensures seamless handoffs, it is extremely wasteful in terms of energy by triggering unnecessary wakeups. A more intelligent technique is to determine the trajectory of the object and using the region of overlap determine which camera is best positioned to take over tracking duties and only wake it up. However, since the object's location cannot be calculated without knowledge of accurate camera parameters, its trajectory can not be accurately determined. The only known information about the object is its image on the camera's image plane—the object is known to lie along a line that connects the image to the center of the lens. As shown in Figure 4, we refer to this line as the *projection line*, the line on which the object must lie. We can exploit this information to design an intelligent triggered wakeup technique. Any camera whose region of overlap intersects with the projection line can potentially view the object and is a candidate for a handoff. To determine all such cameras, we first determine the set of reference points within a specific *distance threshold* of the line (see Figure 4). To determine these reference points, equidistant points along the length of the projection line are chosen and reference points within the distance threshold are identified. Next, the set of neighboring cameras that can view these reference points is determined (using information gathered during our initialization process). One or more of these camera can then be woken up. Depending on the extent of overlap with the projection line, candidate cameras are prioritized and woken up in priority order—the camera with highest overlap has the highest probability of detecting the object on wakeup and is woken up first. Two important parameters of the scheme are the *distance threshold* and the *maximum number of cameras* to be woken up. A large distance threshold will capture many reference points and yield many candidates for wakeup, while a small threshold will ignore overlapping cameras. The maximum number of cameras to be woken up bounds the redundancy in viewing the same object by multiple cameras—a small limit may miss the object whereas a large limit may result in wasteful wakeups. We discuss the effect of these parameters as part of the experimental evaluation.

5 Prototype Implementation

System Design The approximate initialization procedure involves taking pictures of reference points (or objects). Reference points are objects like a ball with a unique color or a light source, that can be easily identified by processing images at each camera. Each camera after taking a picture, processes the image to determine if it can view a reference point. If a reference point is visible, it calculates the location of the reference point on its



(a) Network Setup



(b) Software architecture

Fig. 5. Setup and software architecture of prototype implementation.

image plane and if possible estimates the location of the reference point. The location can be estimated using an approximation of the distance of the reference point from the camera. The distance can be determined if dimensions of the reference object are known a priori along with the size of it's image on the camera's image plane. The image location and distance of object information is exchanged with all other cameras in the network. The data recorded at each camera can be stored as table of tuples,

$$< R_k : C_i, u_i, v_i, d_i, C_j, u_j, v_j, d_j ... >$$

where, R_k is the k^{th} reference point visible to camera i, (u_i, v_i) is the projection location of the reference point in the image plane and d_i is the distance of the reference point from the camera. The tuple also stores information from each camera that can view the reference point simultaneously. Multiple reference points are generated by placing the reference object at several locations.
The network setup for our prototype implementation is shown in Figure 5(a). The network consists of 8 cameras covering a region of $8ft \times 6ft \times 17ft$. The camera are equidistantly placed on the longest side, each at a height of $3ft$ facing each other and viewing inside the cubical volume. The depth-of-view for each camera is $8ft$ and the horizontal and vertical viewing regions are $7ft$ and $6ft$ respectively. The setup is used to estimate and compare k-overlap and region of overlap for each camera.

Hardware Components We used the Cyclops [18] camera sensor in our prototype implementation to evaluate the approximate initialization techniques. The Cyclops camera sensor consists of a ADCM 1700 CMOS camera module, and supports image resolutions of 32x32, 64x64 and 128x128. Image resolution of 128x128 is used in the experimental evaluation. The Cyclops node also has an on-board ATMEL ATmega128L micro-controller, 512 KB external SRAM and 512 KB Flash memory. The on-board processing capabilities of the Cyclops are used for object detection and to detect the size of object's image. Each Cyclops sensor is connected to a Crossbow Mote (referred to as the HostMote) and they communicate via the I2C interface. The HostMote is also used to receive and send wireless messages and store initialization information on behalf of the Cyclops. A mote is also used as a remote control to send synchronized sampling triggers to detect reference points during the initialization process. A glowing ball (a translucent ball fitted on a light bulb) is used as a reference object and is manually placed at several locations to generate reference points for initialization.

Software Components: Both the Cyclops sensors and the Motes run TinyOS [21]. Each Cyclops communicates with it's attached mote using the I2C interface and the motes communicate with each other via their wireless interface (see Figure 5(b)).
Cyclops Onboard Tasks: Each Cyclops is responsible for taking images and processing them locally to detect the reference objects. On receiving a trigger from the HostMote each Cyclops takes a picture and processes it to detect and recognize reference objects. The results are communicated back to the HostMote.
HostMote Tasks: The HostMote drives each Cyclops to detect reference objects and stores all the initialization information for each camera. Once an reference object is detected, the HostMote estimates the distance of the object from the camera and transmits a broadcast message indicating visibility of the reference object, coordinates of the object on it's image plane and distance of object from the camera. Further, the HostMote receives similar broadcasts from other nodes and maintains the *ViewTable*, a table of tuples representing viewability information of each reference point.
Trigger Mote Tasks: The trigger mote is used as a remote control for synchronized detection of the reference object. Once a reference object is placed in a location, the trigger mote sends a wireless broadcast trigger to all HostMotes, which in turn trigger the attached Cyclops sensors.

6 Experimental Evaluation

In this section we present a detailed experimental evaluation of the approximate initialization techniques using both simulation and implementation based experiments. Specifically, we evaluate the accuracy of the approximate initialization procedure in estimating the *degree of overlap* and *region of overlap* of camera sensors. In addition, we evaluate the effect of skew in location of reference points on the accuracy of estimation. Further, we also evaluate the performance of an triggered wakeup application which demonstrates effective use of the region of overlap information.

6.1 Simulation Setup

The simulation setup used for evaluation consisted of a cubical region with dimensions 150x150x150. Two cases, one with 4 cameras and the other with 12 cameras are used. In the first case, 4 cameras are placed at locations (75,0,75), (75,150,75), (0,75,75), (150,75,75), oriented perpendicular to the side plane looking inwards. The k-overlap at each camera is as follows: 1-overlap: 0.54, 2-overlap: 0.23, 3-overlap: 0.07 and 4-overlap: 0.16. In the second case, additional 8 cameras are placed at the 8 corners of the cube and each of them is oriented inwards with the central axis pointing towards the center of the cube.
An uniform distribution of reference points was simulated by uniformly distributing points in the cubical viewing region. To simulate a skewed distribution, a fraction of reference points were distributed in a smaller region at the center of the viewing region and the rest were distributed in the entire viewing area. For example, a region of size 25x25x25 at the center of the viewing region, in different cases, had atleast 25%, 33%, 50%, 66% and 75% of total points within its boundary. We also used restricted regions of sizes 50x50x50 and 75x75x75 with varying fractions of skew in our evaluation.

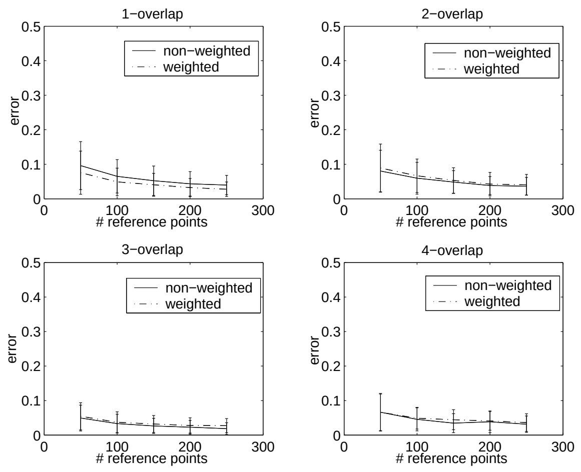


Fig. 6. Evaluation of k-overlap estimation scheme with uniform distribution of reference points.

6.2 Degree of overlap estimation

In this section we present evaluation of the techniques used to estimate *k-overlap*, the degree of overlap metric, and its use to estimate the duty-cycling parameter.

Initialization with uniform distribution of reference points Figure 6 plots the error in k-overlap estimation using the four camera setup with uniform distribution of reference points. The absolute difference in the approximate estimation and the exact k-overlap fraction averaged over the 4 cameras is reported as *error*. The error in k-overlap estimation using both the non-weighted and weighted techniques is similar. Figure 6 also plots the effect of number of viewable reference points— reference points viewable by atleast a single camera— on k-overlap estimation. The error in k-overlap estimation decreases with increase in number of reference points for both the non-weighted and weighted schemes. Error in 1-overlap estimation with the weighted scheme decreases from 0.075 to 0.04 with 50 and 150 reference points respectively.

Initialization with skewed distribution of reference points Figure 7 plots the k-overlap estimates with non-uniform distribution of reference points. The results are averaged for the different fractions of skew within a restricted region of 25x25x25. As seen from the figure, the weighted scheme accounts for skew better than the non-weighted scheme—with most benefits for 1-overlap and 4-overlap estimation. The non-weighted scheme performs poorly as it only counts the number of simultaneously viewable points, while the weighted scheme accounts for the spatial distribution of the points. Further, with increase in number of reference points, the error with the weighted scheme decrease, whereas that with the non-weighted scheme remains the same. Figure 8(a) plots the k-overlap with 150 reference points, and it shows that the weighted

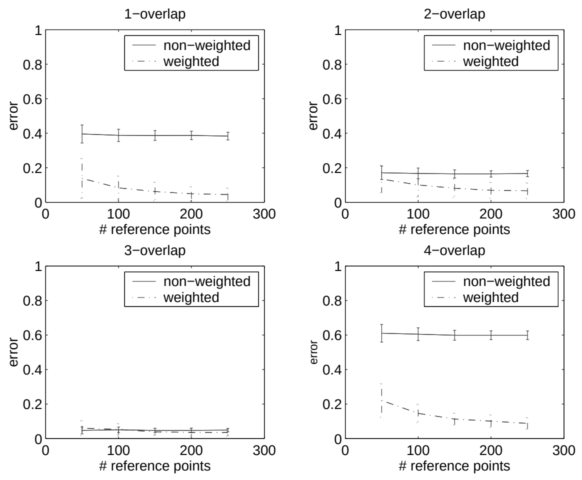
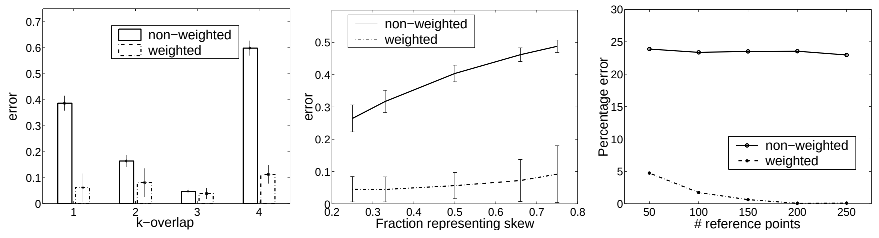


Fig. 7. Evaluation of weighted k-overlap estimation with skewed distribution of reference points.



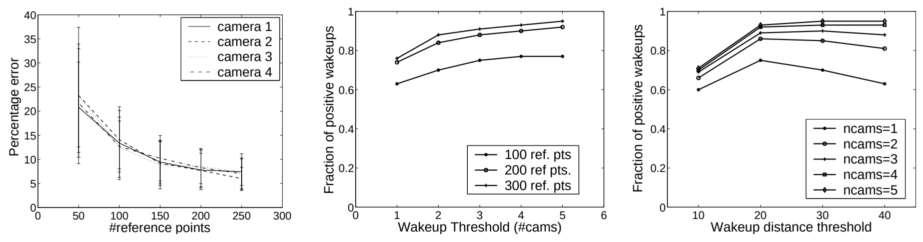
(a) k-overlap (b) Effect of skew (c) Duty-cycle parameter

Fig. 8. Evaluation of the weighted k-overlap estimation scheme.

scheme performs better than the non-weighted scheme. The error with the non-weighted scheme for 1 and 4 overlap is worse by a factor of 4 and 6 respectively.

Figure 8(b) plots error in estimation of 1-overlap with 150 reference points and varying skew. As skew increases, so does the error in both non-weighted and weighted schemes—error with the weighted scheme being smaller than the non-weighted scheme. The increase in error is also more gradual with the weighted scheme as compared to the non-weighted scheme. The error with the non-weighted scheme increases from 0.26 to 0.49 with increase in skew fraction from 25% to 75% and the corresponding values for the weighted scheme are 0.045 and 0.09 respectively.

Duty-Cycling The percentage error in duty-cycle parameter estimation (see Section 4.1) using the k-overlap estimates is shown in Figure 8(c). As seen from the figure, error using the non-weighted scheme is close to 24% and remains unchanged with increase in



(a) Effect of number of reference points (b) Effect of number of cameras (c) Effect of distance threshold

Fig. 9. Region of overlap estimation and wakeup heuristic performance.

reference points. Whereas, error with the weighted scheme is 5% even with only 50 points and decreases very close to zero with more than 150 points.
From the results presented above, we conclude that the weighted k-overlap estimation scheme is well suited to estimate degree of overlap of cameras. The scheme performs identical to the non-weighted scheme with uniform distribution of reference points and significantly better with non-uniform distributions. The application-level error in determining the duty-cycle parameter using the weighted scheme is close to zero.

6.3 Region of overlap estimation

In this section we present evaluation of *region of overlap* estimation and the triggered wakeup heuristic that uses this estimate. Figure 9(a) plots results evaluating the effect of number of reference points on region of overlap estimation. The percentage error reported is the absolute error in estimated volume corresponding to a region of overlap and the exact volume. As seen in Figure 9(a), with uniform distribution of reference points, the percentage error of all four cameras follows a similar trend. With 50 reference points the percentage error for the four cameras is between 21-23% and with 100 reference points is 12-14%. With higher number of reference points the error decreases and so does the standard deviation. With 200 reference points the error is 7-8% and with 250 points is 6-7%. *The above results show that region of overlap between pair of cameras can be estimated with low error—6-7% with uniform distribution in our setup.*

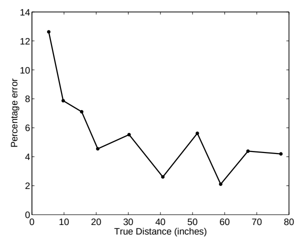
Wakeup Heuristic Next, we evaluate effectiveness of the wakeup heuristic based on the region of overlap estimates with the 12-camera setup. Figure 9(b) plots the effect of maximum number of cameras triggered on the fraction of positive wakeups, i.e., fraction of cases when atleast one of the triggered cameras could view the object. As seen from the figure, with increase in maximum number of cameras triggered per wakeup, the fraction of positive wakeups increases. Further, the fraction also increases with increase in total reference points in the environment. The fraction of positive wakeups with a maximum of 2 cameras to be triggered is 0.7 and 0.88 for 100 and 300 reference points respectively with a distance threshold (see Section 4.2) of 20 inches. With a maximum of 5 cameras to be triggered the corresponding fractions are 0.77 and 0.93

Camera	Error
1	1.5%
2	7.1%
3	4.9%
4	5.8%
5	8.7%
6	3.1%
7	7.9%
8	6.7%

(a) k-overlap error

Camera	Error
1	2.4%
2	2%
3	6.4%
4	10.8%
5	3%
6	4.7%
7	4.3%
8	0.65%

(b) Region-of-overlap error



(c) Distance estimation error

Fig. 10. Initialization using prototype implementation.

respectively. The fraction of positive wakeups is over 0.8 with a maximum of 2 wakeups per trigger. *The result shows that the wakeup heuristic based on region of overlap estimate can achieve high fraction of positive wakeups—close to 80% accuracy with 2 cameras woken up per trigger.*
Another parameter that influences the performance of the heuristic is the *distance threshold*—the distance along the projection of the object's image used to approximate overlapping cameras. As shown in Figure 9(c), with increase in distance threshold from 10 to 20 with 200 reference points, the fraction of positive wakeups increases and remains relatively constant for a maximum 2, 3, 4 and 5 triggered cameras. With just one camera to be woken up for each trigger, the fraction of positive wakeups decreases with further increase (beyond 20) in distance threshold. This indicates that the distance threshold is an important factor affecting the performance of the heuristic and for our setup a threshold of 20 yields best performance.

6.4 Implementation Results

In this section, we evaluate the estimation of k-overlap and region of overlap using our prototype implementation. As described in Section 5, we used 8 cameras in our setup and a glowing ball(1.5 inches in diameter) as a reference object. The object was manually placed at several locations to approximate an uniform distribution of reference points. Table 10(a) shows the average k-overlap percentage error at each camera. The percentage error in k-overlap estimation over all cameras is 2-9%.
We also evaluate the accuracy of region of overlap estimate between pairs of cameras in the 8-camera setup. Figure 10(b) tabulates the average percentage error estimating the region of overlap between pairs of cameras. The average error in estimating the region of overlap between pairs of cameras varies form 1-11% for our setup. An important factor that affects the region of overlap estimate is the distance estimate of the object from the camera. Figure 10(c) plots the percentage error in estimating the distance of the object from the camera based on its image size. As can been from the figure, the error is varies from 2-12%. For our setup, the region of overlap estimates show that the error is below 11% inspite of the error in distance estimation of the object.
Our results show that the approximate initialization techniques are feasible in real-world deployments and for our setup had errors close to 10%.

7 Related Work

Several techniques have been developed by the vision community for accurate camera calibration that use a set of reference points with known locations [22, 24]. In sensor networks, techniques like [9, 10, 23] are specialized to estimate only the extrinsic parameters of cameras in an exact manner, typically using additional infrastructure or hardware. Further, distributed techniques proposed in [20, 13] are suited to calibrate networked cameras. In [20], cameras collaborate to track an object and reason about consistent camera location and orientations for observed images. The technique simultaneously solves both the object tracking and camera calibration problem. Examples of systems that use accurately calibrated cameras for video surveillance and tracking are [12, 17]. All the above techniques estimate exact parameters of camera, whereas our work focuses on approximate initialization of camera networks with no or limited infrastructure support and camera nodes with limited resources.
Positioning and locationing techniques for sensor nodes other than cameras have also been well studied. These techniques depend on a beaconing infrastructure and use several modalities—Active Badge [1] uses IR signals, Active Bat [2] and Cricket [16] use ultrasound signals and RADAR [3] uses RF signals. Further, GPS [4] is an example of an outdoor localization system, which can localize object to within 5–15 meters of their actual location. While these methods can be used to localize cameras and in some cases to estimate their orientation, they either have high error or are not suitable for low-power resource constrained camera sensor networks.
There exist several types of camera sensor nodes, each with different resources and capabilities. The Cyclops [18] and CMUCam [19] are examples of low-power nodes capturing low-resolution images with limited computation capabilities. XYZ [11] is a power-aware sensor platform which can be equipped with image sensors. Panoptes [7] is a camera sensor node comprising of a webcam capturing high-resolution images and a Intel StrongARM PDA processor for reasonably high computation resources. Even more sophisticated camera nodes are those with pan-tilt-zoom capabilities connected to PDA or laptop-class devices. In this work, we are interested in developing techniques for low-power resource constrained camera nodes, and our solutions can be applied to more powerful nodes as well.

8 Conclusions

In this paper, we argued that traditional vision-based techniques for accurately calibrating cameras are not directly suitable for ad-hoc deployments of sensors networks in remote locations. We proposed approximate techniques to determine the relative locations and orientations of camera sensors without any use of landmarks or positioning technology. By randomly sampling the environment with a reference object, we showed how to determine the degree and range of overlap for each camera and how this information can be exploited for duty cycling and triggered wakeups. We implemented our techniques on a Mote testbed. Our experimental results showed that our approximate techniques can estimate the degree and region of overlaps to within 10% of their actual values and this error is tolerable at the application-level for effective duty-cycling and wakeups.

References

1. Andy Harter and Andy Hopper. A Distributed Location System for the Active Office. *IEEE Network*, 8(1), January 1994.
2. Andy Ward and Alan Jones and Andy Hopper. A New Location Technique for the Active Office. *IEEE Personal Communications*, 4(5):42–47, October 1997.
3. P. Bahl and V. N. Padmanabhan. RADAR: An In-building RF-based User Location and Tracking System. In *IEEE INFOCOM* , Volume 2, pages 775–784, March 2000.
4. R. Bajaj, S. L. Ranaweera, and D. P. Agrawal. GPS: Location-tracking Technology . *Computer*, 35(4):92–94, March 2002.
5. M. Berg, M. Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry*. Springer, Second edition, 2000.
6. Sparton SP3003D Digital Compass. http://www.sparton.com/.
7. W. Feng, B. Code, E. Kaiser, M. Shea, W. Feng, and L. Bavoil. Panoptes: A Scalable Architecture for Video Sensor Networking Applications. In *ACM Multimedia*, 2003.
8. B. K. P. Horn. *Robot Vision* . The MIT Press , First edition, 1986.
9. X. Liu, P. Kulkarni, P. Shenoy, and D. Ganesan. Snapshot: A Self-Calibration Protocol for Camera Sensor Networks. In *IEEE/CreateNet BASENETS*, October 2006.
10. D. Lymberopoulos, A. Barton-Sweeny, and A. Savvides. Sensor Localization and Camera Calibration using Low Power Cameras. Technical Report, Yale University, 2005.
11. D. Lymberopoulos and A. Savvides. XYZ: A Motion-Enabled, Power Aware Sensor Node Platform for Distributed Sensor Network Applications. In *IPSN*, April 2005.
12. M. Chu and J. E. Reich and F. Zhao. Distributed Attention for Large Video Sensor Networks. In *Intelligent Distributed Surveillance Systems*, 2004.
13. W. Mantzel, H. Choi, and R. Baraniuk. Distributed Camera Network Localization. In *Asilomar Conference on Signals, Systems, and Computers*, volume 2, November 2004.
14. Crossbow Wireless Sensor Platform. http://www.xbow.com/
15. J. Polastre, R. Szewczyk, and D. Culler. Telos: Enabling Ultra-Low Power Wireless Research. In *IPSN/SPOTS*, April 2005.
16. N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket Location-Support System. In *ACM MOBICOM*, pages 32–43, August 2000.
17. R. Collins and A Lipton and T. Kanade. A System for Video Surveillance and Monitoring. In *American Nuclear Society (ANS) Eighth International Topical Meeting on Robotics and Remote Systems*, 1999.
18. M. Rahimi, R. Baer, O. I. Iroezi, J. C. Garcia, J. Warrior, D. Estrin, and M. Srivastava. Cyclops: In Situ Image Sensing and Interpretation in Wireless Sensor Networks. In *ACM SENSYS*, pages 192–204, November 2005.
19. A. Rowe, C. Rosenberg, and I. Nourbakhsh. A Low Cost Embedded Color Vision System. In *International Conference on Intelligent Robots and Systems*, 2002.
20. S. Funiak and C. Guestrin and M. Paskin and R. Suthankar. Distributed Localization of Networked Cameras. In *IPSN*, April 2006.
21. TinyOS Website. http://www.tinyos.net/.
22. R. Y. Tsai. A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses. *IEEE Journal of Robotics and Automation*, RA-3(4):323–344, August 1987.
23. F. Y. Wang. A Simple and Analytical Procedure for Calibrating Extrinsic Camera Par ameters. *IEEE Transactions on Robotics and Automation*, 20(1):121–124, February 2004.
24. Z. Y. Zhang. A Flexible New Technique for Camera Calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, November 2000.