Distributed Calibration of Pan-Tilt Camera Network using Multi-Layered Belief Propagation

Ayesha Choudhary[1] Gaurav Sharma[2*] Santanu Chaudhury[1] Subhashis Banerjee[1]
[1] Indian Institute of Technology, Delhi, India. [2] University of Caen, France.
{ayesha, suban}@cse.iitd.ernet.in santanuc@ee.iitd.ernet.in gaurav.sharma@info.unicaen.fr

Abstract

In this paper, we present a technique for distributed self-calibration of pan-tilt camera network using multi-layered belief propagation. Our goal is to obtain globally consistent estimates of the camera parameters for each camera with respect to a global world coordinate system. The network configuration changes with time as the cameras can pan and tilt. We also give a distributed algorithm for automatically finding which cameras have overlapping views at a certain point in time. We argue that using belief propagation it is sufficient to have correspondences between three cameras at a time for calibrating a larger set of (static) cameras with overlapping views. Our method gives an accurate and globally consistent estimate of the camera parameters of each camera in the network.

1. Introduction

In this paper, we present a distributed algorithm for self-calibration of a pan-tilt camera network using multi-layered belief propagation. The goal of our distributed calibration algorithm is to obtain a globally consistent and accurate estimate of each camera's parameters (intrinsic as well as extrinsic) with respect to a global world coordinate system (WCS). As the cameras can pan and tilt, the camera network contains various mutually exclusive sub-networks, where, all cameras in a sub-network view a common region. For distributed calibration, we perform multi-camera self-calibration at each camera in a sub-network and apply belief propagation to obtain consistent camera parameters in each sub-network. We then propagate belief between sub-networks to obtain the globally consistent and accurate estimates of the camera parameters for each camera in the network.

In general, pan-tilt camera networks are well-suited for wide area surveillance. Automated surveillance requires that the camera network be calibrated with respect to a global WCS so that tasks such as 3D-tracking, recognition of objects, activities and events can be effectively performed. Moreover, this also requires that the camera parameters be consistent and accurate with respect to one another, which cannot be achieved by individually calibrating each camera. Self-calibration of a pan-tilt camera network is necessary as it is, in general, difficult and impractical to use an external calibration object.

Distributed calibration is advantageous for pan-tilt camera network, as it is more robust against failures. In case of failure of a camera, the information can be retrieved from its neighbors. Moreover, unlike failure of the central server which may lead to shutting down of the system, failure of a camera does not impact the complete network. Also, in case of distributed calibration, addition of new cameras in the network does not require re-calibration of the complete camera network. Our distributed calibration also leads to making the system scalable, as large camera networks spanning a wide geographical area would contain mutually exclusive sub-networks, thereby, no communication and computation among the cameras of these sub-networks would be necessary for calibration. Therefore, in effect, cameras which do not view a common scene in any of their pan-tilt positions do not affect each other. Therefore, our distributed algorithm calibrates the complete camera network by calibrating smaller sub-networks, making the system scalable.

Distributed calibration of the camera network may lead to inconsistencies in the estimation of the camera parameters since these parameters are computed at each node of the network. We use belief propagation to leverage on the information at each node of the camera network to arrive at a consistent and accurate estimate of the camera parameters of each camera in the network.

The configuration of a pan-tilt camera network is dynamic. The various sub-networks that exist in the system change across time, that is, cameras in different pan-tilt positions become a part of different sub-networks across time. Moreover, within a fixed time interval, a camera can be a part of only one sub-network. We give a technique to au-

*The work was done when Gaurav Sharma was with Multimedia lab, Indian Institute of Technology, Delhi.

tomatically find the sub-networks as well as a method to automatically control the cameras so that they become parts of different sub-networks across time, which is essential for propagating belief across various sub-networks. We discuss the related work in the next section.

2. Related Work

Multi-camera calibration is a well-studied problem in computer vision. Pan-tilt camera network calibration has also become an important area of research. Most of the multi-camera calibration methods are based on centralized processing. As camera networks are becoming larger, distributed algorithms are becoming a necessity. Recently, in [10], an online distributed algorithm has been proposed for cluster based calibration of a large wireless static camera network using features detected on known moving target objects. They assume that the intrinsic parameters are known and that each target object has known multiple distinctive features. In [7], 3D features and geographic hash tables are used while in [5] object motion is used for calibration. Very recently, authors in [4], have proposed a distributed algorithm for calibration of a camera sensor network, where they assume that one of the cameras is calibrated and use epipolar geometry based algorithms at each node to obtain its calibration parameters. They show that a globally consistent solution can be reached in a distributed manner by solving a set of linear equations.

In [1], a method for self-calibration of purely rotating cameras using infinite homography constraint is proposed. Davis et al. [2] present a method for calibrating pan-tilt cameras and introduce a complete model of the pan-tilt rotation occurring around arbitrary axes. Both these methods are for calibrating a single camera and not for calibration of a pan-tilt camera network. Authors in [12], estimate both the internal and external parameters of a pan-tilt camera network without requiring any special calibration object. But, their method is feature based and estimates the camera parameters by using the complete set of images captured at each pan-tilt-zoom configuration of the camera.

Radke et al. [3], give a distributed calibration method for a static camera network using belief propagation. They assume that the cameras form a graph where cameras are the nodes and an edge exists between the nodes if they have overlapping views. In their case, since the cameras are static, the configuration of the network does not change with time and the cameras form one connected graph. We extend this approach for distributed calibration of pan-tilt camera network using multi-layered belief propagation. In our case, many mutually exclusive graphs exist at the same time and the same camera may belong to many different graphs across time. We also address the issues of automatically finding the various graphs in the system. In [3], they assume that the camera network forms a connected graph, whereas we give a method for automatically controlling the cameras to create connected graphs. Also, we propose the use of multi-layered belief propagation, first within a graph for a consistent measure of the camera parameters within the graph, and then between multiple graphs to get a consistent estimate of the camera parameters in the pan-tilt camera network.

The methods in [3, 10, 7, 4] are for distributed calibration of static camera networks while we propose a technique for distributed calibration of pan-tilt camera network. Moreover, unlike [4, 10], we do not require that the internal or external parameters of any camera be known and do not require any external calibration object. Also, unlike [12], our method does not consider every pan-tilt configuration of any camera in the network.

3. Distributed calibration of pan-tilt camera network: an overview

We assume that the camera network has $N \geq 3$ cameras and each camera has a unique number $n \in \{1, 2, \ldots, N\}$ associated with it. We also assume that each camera has a processing unit attached with it and that there exists an underlying communication network such that each camera can communicate with every other camera. A sub-network in a pan-tilt camera network consists of cameras viewing a common area. The cameras which have overlapping views form a complete graph $G = (V, E)$ where, the cameras $C_i \in V$ and edge $e_{ij} \in E$ between cameras C_i and C_j for all cameras in the graph. In a pan-tilt camera network, there may exist many such mutually exclusive graphs at any point in time. Moreover, if a camera pans and/or tilts, then it may cease to remain a part of one graph and become a part of another graph. In Section 5, we give a distributed algorithm for finding these graphs automatically.

We assume that the cameras remain in a certain pan-tilt position for a fixed period of time. During this time interval, the cameras in each graph are considered as static cameras. Corresponding points between the views of the cameras in each graph are found automatically and multi-camera self-calibration is performed at each node of the graph. It is well-known that finding automatic correspondences between multiple views is not an easy problem. We show that by using multi-layered belief propagation it is sufficient to have correspondences between only three cameras at a time for consistent calibration of a larger $N > 3$ static camera network. In Section 6, we give the method to calibrate a large $N > 3$ (static) camera network using multi-layered belief propagation by iteratively calibrating its 3-cliques. We discuss belief propagation and multi-layered belief propagation in Section 7 and discuss how multi-layered belief propagation is applied at each camera in the network. Since the information is combined from

Figure 1. Example of common points found in three images. *Note*: All images are best viewed in color and at a high resolution.

graphs containing the cameras in various pan-tilt configurations, it is unlikely that belief propagation will get stuck in a local minima and hence, globally consistent estimates are achieved.

In Section 8, we give a protocol for automatically controlling the cameras so that they become a part of various sub-networks across time which is necessary for distributed calibration of the pan-tilt camera network. Otherwise, the network will remain divided into mutually exclusive sub-networks and there will be no exchange of information between various pan-tilt views of the same camera across time. To perform multi-layered belief propagation between two graphs containing the same camera in different pan-tilt positions, we need to bring the cameras to their home (*zero* pan and *zero* tilt) position in both the graphs. We show that the camera matrix for the home position of the camera can be computed by automatically finding pairwise correspondences to compute the homography or a sequence of homographies between the camera's pan-tilt view and the home view. We also propose a protocol in Section 9, for aligning all the cameras' home positions to a global WCS, to get a globally consistent estimate of the camera's home position (*zero* pan, *zero* tilt position). In the next section, we give a method for automatically finding correspondences between three images. The same method can be used for finding correspondences automatically between a pair of images.

4. Automatically finding corresponding points between three images

We propose a method for automatically finding corresponding points in three images. It can also be used to find correspondences in a pair of images or more than three images. But, as the number of images increase, the error in correspondences also increase. Let I_1, I_2 and I_3 be three images taken by three different cameras of the same scene. We perform the following steps to automatically find correspondences between the three images. First, compute the SIFT features in all three images and then, compute the SIFT matches between the pairs $I_1 - I_2$, $I_1 - I_3$ and $I_2 - I_3$. Next, find the common SIFT matches between these three pairs, denoted by $X = \{x_1, x_2, x_3\}$ for points in I_1, I_2 and I_3 respectively. Further, refine these points by fitting fundamental matrices between pairs of images and taking points which are common in all the three images. This is done by first fitting fundamental matrix to the pairs $F_{12} = \{x_1, x_2\}$, $F_{13} = \{x_1, x_3\}$ and $F_{23} = \{x_2, x_3\}$ and then, finding the common points between the inliers in F_{12}, F_{13} and F_{23}, say y_1, y_2 and y_3. If the number of points are ≥ 50, then we say that there exists overlap between the three images and y_1, y_2 and y_3 are the correspondences in the three views. Figure 1 shows the common points found between three images taken by three different cameras.

5. Finding the graphs

We develop an algorithm to automatically find the graphs in the network. Starting with the camera with the smallest number that does not belong to any graph currently, say C_i, find the camera with the next smallest number, say C_j, that has an overlap with C_i and which does not belong to any graph. Form a graph $G = (V, E)$ where, $V = \{C_i, C_j\}$ is the set of nodes and $e_{ij} \in E$ is the edge between C_i and C_j. Incrementally, find all those cameras (by automatically finding the corresponding points) which have overlapping views with C_i and C_j and are not a part of any graph currently. Add them as nodes of G and add edges between all the nodes of G. Continue till either there is no camera that does not belong to a graph in the system or no other camera has overlapping views with the nodes in graph G.

Repeat this with all the cameras in the network that are not a part of any graph. In general, there will be more than one graph in the pan-tilt camera network. Moreover, each graph will be a complete graph. *A priori* knowledge of the camera network topology can be used to reduce the amount of communication across cameras as well as the number of computations for SIFT matches. For example, in a wide area pan-tilt camera network it is possible that two sets of cameras are geographically so far apart that there will be no overlapping view between these two sets of cameras. Therefore, no communication or computation needs to be carried out between such mutually exclusive and distant camera sub-sets.

6. Camera calibration within a graph

We assume that the cameras in a graph, say G_k, remain static for a certain time period. Thus, standard multi-camera self-calibration techniques can be used for calibrating the cameras within a graph. In a distributed system, multi-camera calibration is carried out at each node of the graph, G_k. The crucial point here is to automatically find multi-view correspondences at each node. Since this is not an easy task, we show that it is possible to calibrate a graph of size $N > 3$ by calibrating its $3-$cliques and using multi-layered belief propagation to reach a consistent estimate of the camera parameters of all the cameras in the graph.

We consider all possible 3-cliques of the graph G_k. Let

Figure 2. These images are from one pan-tilt camera taken at different pan and tilt positions. To find the homography between (a) and (f), where (f) is the home position, we find a sequence of homographies: between (a) and (b), then (b)=(c) and (d) and then (d) = (e) and (f). The point correspondences for finding the homographies are automatically found as explained in text.

G_k^i be the i^{th} 3-clique of G_k. The corresponding points between the nodes of G_k^i are found automatically as discussed in Section 4. Standard multi-camera self-calibration technique is used at each node of G_k^i to get estimates of camera parameters of each camera in G_k^i. Belief propagation (discussed in Section 7) between the nodes of G_k^i gives a consistent estimate of the camera parameters for each camera in G_k^i. This is done for each of the 3-cliques of G_k, which will not be more than $\binom{n}{3}$ for a graph of size n. Therefore, there will be $\binom{n}{3}$ estimates of each camera after belief propagation is carried out within each 3-clique. Then, multi-layered belief propagation at each node of G_k is carried out between the estimates of the camera parameters of that node in the various (at most $\binom{n}{3}$) 3-cliques. If this procedure is carried out iteratively, then it is not necessary to calibrate all the $\binom{n}{3}$ 3-cliques. It is possible that a consistent estimate of the camera parameters for each camera in G_k can be reached with a lesser number of 3-cliques than $\binom{n}{3}$. Thus, we are able to calibrate the complete graph of $N > 3$ cameras without knowing multi-view correspondences among all the nodes of the graph. Figure 4 shows a result of this technique for calibrating a graph of five cameras by using five 3-cliques of the graph. An important point to be noted here is that the camera matrices have to be aligned to a common WCS for this graph before propagating belief at a node between the subgraphs. The common WCS for this graph can be a predefined WCS or we can take the lowest numbered camera in the graph to be at the origin of the WCS.

7. Belief Propagation within a graph

For distributed calibration of cameras in a graph, say G_k, multi-camera self-calibration is carried out at each node, using the automatically found corresponding points. Therefore, at each node C_i of G_k, we obtain an estimate of the camera parameters P_j^k for all j cameras in G_k. Let y_i be the true camera parameters for the i^{th} camera. Our aim is to find y_i from the estimates of the camera parameters computed at each node of G_k, using belief propagation. The estimates of the camera parameters of all cameras in G_k computed at each node are considered as the beliefs at each node. In general, belief propagation algorithm is used for solving inference problems based on local message passing [11]. Each node updates its beliefs by using the estimates it receives from its neighbors in the form of "messages". These beliefs are iteratively updated until there is no change in the belief at a node. As has been shown in [3], belief propagation can be directly applied on a graph which has cameras viewing a common scene as its nodes. In this case, the update equations are of the form:

$$\tilde{\Sigma}_{i,k} \leftarrow [\Sigma_{i,k}^{-1} + \sum_{j \in N(i,k)} \Sigma_{j,k}^{-1}]^{-1}$$
$$\tilde{\mu}_{i,k} \leftarrow \tilde{\Sigma}_{i,k} * [\Sigma_{i,k}^{-1}\mu_{i,k} + \sum_{j \in N(i,k)} \Sigma_{j,k}^{-1}\mu_{j,k}] \quad (1)$$

Here, $\mu_{i,k}$ and $\Sigma_{i,k}$ are the estimate and covariance of the camera parameters computed at the i^{th} camera C_i in the k^{th} graph, G_k. $N(i,k)$ denotes the set of neighbors of camera C_i in graph G_k. Moreover, the i^{th} node, C_i receives $\mu_{j,k}$ and $\Sigma_{j,k}$ from C_j, its j^{th} neighbor, $j \in N(i,k)$. $\tilde{\mu}_{i,k}$ and $\tilde{\Sigma}_{i,k}$ are the estimates of the camera parameters after belief propagation within graph G_k. The covariance matrix is calculated based on the forward covariance propagation from bundle adjustment. We consider the diagonal terms of the covariance matrix only, resulting in it being a diagonal matrix which is positive definite and invertible. Moreover, we use all the 11 camera parameters [6] as the belief at a node.

7.1. Multi-layered Belief Propagation

Since the graphs are dynamic and the same camera C_i can be a part of two graphs, say G_{k-1} and G_k, in different pan-tilt orientations at different points in time, we perform belief propagation between graphs at each node, C_i, which is common in both G_{k-1} and G_k. Here, the belief at C_i in G_{k-1} is the estimate of the camera matrix of C_i (after belief propagation within G_{k-1}) at its home position, obtained by using the homography between C_i's view in G_{k-1} and the image taken at the home position of C_i. Similarly, the belief at C_i in G_k is the estimate of camera matrix of C_i (after belief propagation within G_k) at home position obtained using homography between the view of C_i in G_k and the home view of C_i.

As is well-known [6], two views of a camera in different pan-tilt positions are related by a 3×3 image to image

homography. Therefore, we automatically compute the homography between the pan/tilt view and the home view of a camera by automatically finding corresponding points between the two images, using SIFT matches further refined by fitting fundamental matrices to the points obtained, as described in Section 4. This homography is then used to get the camera matrix of the home position from the camera matrix of the pan-tilt position. Let $P_{\theta\phi}$ be the camera matrix at pan θ and tilt ϕ position, P_{home} be the camera matrix at the home position, and H be the homography between the home view and the pan-tilt view. Then, if $x = P_{home}X$, $x' = P_{\theta\phi}X$ and $x = Hx'$, $\Rightarrow P_{home} = H * P_{\theta\phi}$. Similarly, we can get to the pan-tilt position as: $P_{\theta\phi} = H^{-1} * P_{home}$. In case, the pan-tilt view of the camera does not have any overlap with the home position's view, a sequence of homographies can be used, again calculated automatically, as shown in Figure 2. Let $\tilde{\mu}_{i,k}$ be the estimate of the camera parameters of C_i after belief propagation within graph G_k, where C_i is in pan θ_k and tilt ϕ_k position. Homography or a sequence of homographies is used to calculate the camera parameters for the home position of C_i, denoted by $P_{i_{home},k}$. These parameters, taken as a vector, are the belief at C_i in G_k denoted by $\mu_{i_{home},k}$. Let $\tilde{\mu}^{k-1}_{i_{home}}$ and $\tilde{\Sigma}^{k-1}_{i_{home}}$ be the estimates of the camera parameters and the covariance matrix after the $(k-1)^{th}$ iteration, at the home position of C_i, of multi-layered belief propagation between $k-1$ graphs containing C_i in different pan-tilt positions. The home position is calculated in each graph using the image-to-image homography before applying the update equations for multi-layered belief propagation. The belief is updated using Equations 2.

$$\tilde{\Sigma}^{k}_{i_{home}} \leftarrow [(\tilde{\Sigma}^{k-1}_{i_{home}})^{-1} + \Sigma^{-1}_{i_{home},k}]^{-1}$$
$$\tilde{\mu}^{k}_{i_{home}} \leftarrow \tilde{\Sigma}^{k}_{i_{home}}[(\tilde{\Sigma}^{k-1}_{i_{home}})^{-1}\tilde{\mu}^{k-1}_{i_{home}} + \Sigma^{-1}_{i_{home},k}\mu_{i_{home},k}] \quad (2)$$

where, $\tilde{\mu}^{k}_{i_{home}}$ denotes the estimate of the camera parameters and $\tilde{\Sigma}^{k}_{i_{home}}$ is the estimate of the covariance matrix of the home position of C_i after the k^{th} iteration.

8. Forming new graphs

The multi-layered belief propagation mechanism can be utilized only if the graphs change across time. We develop a protocol for automatically controlling the pan-tilt of the cameras so that the network configuration changes after a fixed time period. We define a set of landmarks $L = \{L_1, L_2, \ldots, L_m\}$ in the scene with respect to the global WCS. Initially, the graphs are found using the technique discussed in Section 5. Once the estimate of the camera parameters for cameras have been computed in each of these graphs by multi-camera self-calibration and belief propagation within each graph, these cameras are aligned to the global WCS. The camera parameter estimates after alignment are then used for controlling the cameras to form new graphs in the network. The protocol is:

1. For each camera, compute the pan-tilt rotations required to view all the landmarks. (It is possible that a camera may not be able to view all the landmarks, therefore, only those that are visible are considered).
2. For each camera, rotate by the smallest pan-tilt angles such that it views a landmark other than the one it is currently viewing.
3. Send a message to all the other cameras about the new landmark that it is viewing. If it is known *a priori* that two cameras will never have overlapping views, they need not inform each other about the new landmark they are viewing, thereby reducing unnecessary communication.
4. Each camera will have information of all other cameras about the landmark they are viewing. It takes into consideration all the cameras, say set S, that are viewing the same landmark as itself.
5. For each camera, check whether the cameras in its set S form a graph by using the procedure given in Section 5.

This also makes our system scalable as the correspondences have to be calculated among only those cameras which view the same landmark and in step 3, the messages have to be passed only between those cameras which can have overlapping views in some pan-tilt configuration. In general, these will be much smaller in number compared to the size of the camera network. The above algorithm ensures that the graphs in the camera network change over time. This is essential because if the graphs remained static, since they are mutually exclusive no information would be shared between the graphs and it would not be possible to calibrate the complete network. It is possible that there will be cameras which do not have overlapping views with any other camera or graphs that have less than 3 cameras. In the current time period these cameras are not considered for calibration and therefore, remain idle. In the next time period, they shall repeat the above protocol and become part of graphs with ≥ 3 nodes and hence, will be used for calibration and multi-layered belief propagation.

9. Aligning cameras to a global world coordinate system

We want the position and orientation of each camera's home position with respect to a global WCS. Moreover, belief propagation can be carried out only if all the cameras are aligned with respect to a common coordinate system in the world. For the cameras to align themselves to a global

Figure 3. (a) Re-projections after belief propagation within the graph. (b) and (c) Re-projection after randomly choosing camera parameters after multi-camera self-calibration at each node of the graph G_1. The yellow and green '+' denote the reprojections, the red 'o' are the input points.

Table 1. The re-projection statistics for graph, G_1. (Refer Fig 3)

After belief propagation within the graph:			
CamId	1	2	3
mean	0.85	1.32	0.85
std. deviation	0.82	1.36	0.75
Random Set 1:			
CamId	1	2	3
mean	20.00	7.74	21.26
std. deviation	19.37	8.60	20.38
Selected from node	C_1	C_2	C_3
Random Set 2:			
CamId	1	2	3
mean	9.58	20.71	2.12
std. deviation	8.89	33.38	2.76
Selected from node	C_2	C_3	C_1

coordinate system in a distributed manner, we follow the following protocol. Within each graph G_j, the camera with the smallest number is said to be at the origin of a common coordinate system. Its lowest numbered neighbor is said to be on the $x-$axis at a unit distance. These two conditions establish a common coordinate system at the lowest numbered camera, say C_i, in each graph formed in the camera network. All other cameras in G_j are aligned to this common coordinate system. If camera C_i pans/tilts it becomes a part of another graph, say G_k. The two views of camera C_i, in G_j and in G_k are aligned using the pan and tilt rotation matrices between the two pan-tilt positions of the camera. Thus, the cameras in the two graphs are aligned to the common coordinate system at the lowest numbered

Table 2. The re-projection error for the graph with 5 cameras. (Refer Figs. 4, 5, 6)

After multi-layered belief propagation at each node					
CamId	1	2	3	4	5
mean	1.80	2.60	1.79	1.63	2.01
std. deviation	0.77	1.78	0.98	0.90	1.42
Random Set 1:					
CamId	1	2	3	4	5
mean	8.67	14.91	13.34	28.62	10.97
std. deviation	8.35	14.26	9.43	21.21	8.10
Sub-graph	G_1	G_3	G_2	G_5	G_4
Random Set 2:					
CamId	1	2	3	4	5
mean	13.41	8.90	30.08	15.48	6.69
std. deviation	7.25	5.74	16.42	11.31	2.95
Sub-graph	G_3	G_1	G_5	G_4	G_2

camera among the two graphs. This is done for every graph formed. The lowest numbered camera in the network is then aligned to the global WCS in case it is pre-defined. In case the global WCS is not pre-specified, the lowest numbered camera in the network may be assumed to be at the origin of the global WCS.

10. Results and Discussion

We use 6 SONY EVI-D70 PTZ cameras for our experiments. For multi-camera self-calibration, within each graph, we use the code by Svoboda et al. [13] and for detecting and matching SIFT features we use the code by Lowe [9]. Moreover, for fitting the fundamental matrix to the SIFT matches we use Peter Kovesi's code [8]. The objective of our experiments is to show the following: (a) that a pan-tilt camera network can be calibrated in a distributed manner using multi-layered belief propagation; (b) that multi-layered belief propagation leads to accurate and consistent estimates of the camera parameters, both within a graph and across multiple graphs; (c) that it is possible to calibrate a static camera network of size $N > 3$, by knowing correspondences between only 3 views at a time and using multi-layered belief propagation. Figure 3 shows one of the graphs in the network with 3 nodes. The corresponding points among the 3 nodes are found automatically as described in Section 4. Multi-camera self-calibration is carried out at each node of this graph. Therefore, each node C_i computes the camera parameters P_i of all the cameras in the graph. If we randomly select one camera (all its parameters) from each node, for example, P_1 from node C_2, P_2 from C_3 and P_3 from C_1, then as seen in Figure 3(b) and (c) the reprojection error is high and vary based on which

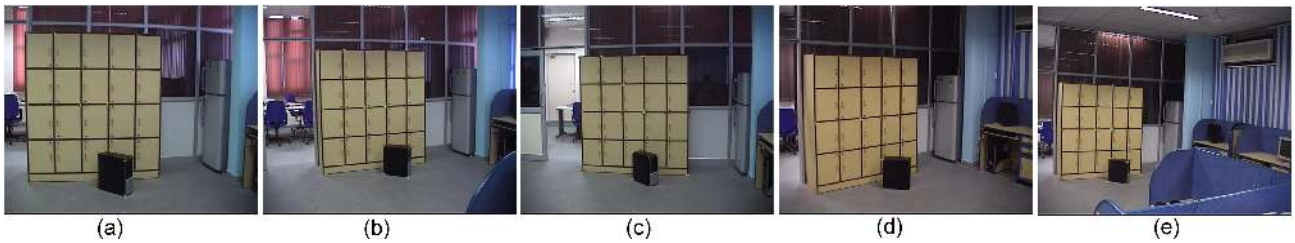

Figure 4. The five cameras are calibrated by first distributed calibration and belief propagation within the 3-cliques and then multi-layered belief propagation across the nodes of the graph. The red circles denote the input points and the green '+' are the reprojections of the $3D$-points found by triangulating the input points using all five cameras.(*Note: same notation for the two images below*)

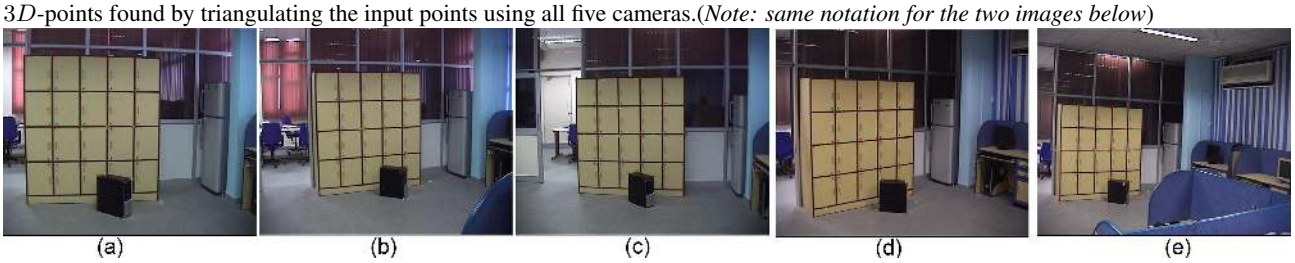

Figure 5. Random set 1: The five cameras are calibrated by first distributed calibration and belief propagation within the 3-cliques and then randomly chosen from the different cliques.

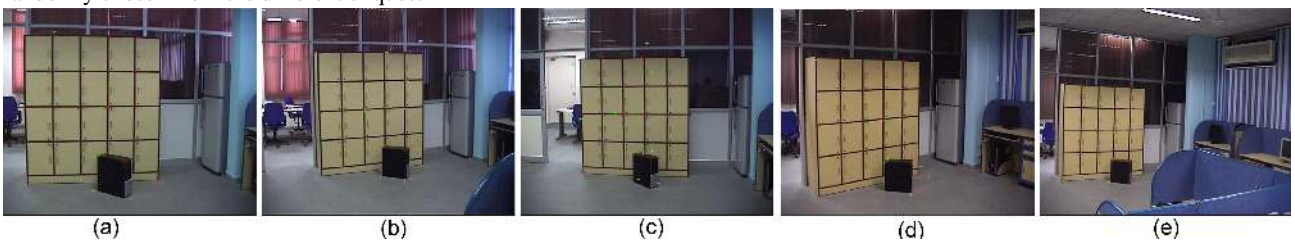

Figure 6. Random set 2: The five cameras are calibrated by first distributed calibration and belief propagation within the 3-cliques and then randomly chosen from the different cliques.

camera is selected from which node. That is, the camera parameters are inconsistent. When belief is propagated within this graph, as seen in Figure 3(a) it leads to a consistent estimate of the camera parameters for each of the three cameras in the graph. The reprojection error statistics are given in Table 1. Figure 4 shows a graph formed by five cameras in various pan-tilt positions. We consider five 3-cliques of the graph for calibrating this graph. First, multi-camera self-calibration and then belief propagation within each of the five 3-cliques is computed. Then, belief is propagated between these 3-cliques at each node of the graph. We consider the camera coordinate system at the first camera to be the global WCS and align each camera with it. We observe that if belief propagation is performed within the 3-cliques and then the cameras are randomly chosen, from the five sub-graphs then the reprojection errors are high and there is an inconsistency in the reprojections, as shown in Figure 5 and Figure 6. Multi-layered belief propagation at the nodes of the graph results in consistent and accurate camera parameters as seen in Figure 4. The re-projection error statistics are given in Table 2. This experiment shows that a static camera network of size $N > 3$ can be calibrated in a distributed manner by knowing correspondences among 3 cameras at a time. Moreover, it is not necessary to consider all possible 3-cliques of the network. Figure 7(a), shows the reprojection on the first camera of the network at its home position. The camera parameters are found by multi-layered belief propagation at C_1 from 7 graphs in the network which contained C_1 in various pan-tilt positions. The multi-layered belief propagation is carried out by finding P_{home} for C_1 in each camera using the automatically computed homography matrices. Figure 7(b) shows the reprojection after belief propagation within the graph containing C_1 at its home position. The reprojection error statistics are given in Table 3. Figure 8 shows that the reprojection of input points in all 6 cameras in their home position is accurate. These reprojections are calculated using the final estimates of the camera parameters for each camera in the pan-tilt camera network. The reprojection error statistics are given in Table 4.

11. Conclusion

We have presented a multi-layered belief propagation based distributed algorithm for self-calibration of a pan-tilt camera network. We have shown that by using multi-

layered belief propagation it is possible to get accurate and globally consistent estimates of the camera parameters for each pan-tilt camera in the network with respect to a global world coordinate system. We have given a method that shows that if multi-layered belief propagation is used, then it is sufficient to know correspondences between three cameras (at a time) for distributed calibration of a large (static) camera network. Our system does not require that all the cameras should have overlapping views at all times. Moreover, we have shown that by propagating beliefs between graphs, it is possible to calibrate cameras in the network even if they do not have any overlap in their views at any point in time. Our method gives an accurate and globally consistent estimate of the camera parameters for the home position of each camera and using the method for automatically finding correspondences in two views, homographies between the home view and any pan/tilt view can be automatically computed. Therefore, it is possible to obtain accurate and globally consistent camera parameters for any pan/tilt position of the pan-tilt cameras in the network with respect to a global world coordinate system.

Table 3. The re-projection statistics for home position of C_1. (Refer Fig 7)

	CamId	mean	std. deviation
Multi-layered BP	1	0.98	0.67
BP within a graph	1	3.13	1.66

Table 4. The re-projection statistics for home position of all cameras after multi-layered belief propagation. (Refer Fig. 8)

CamId	1	2	3	4	5	6
mean	0.98	0.68	0.79	0.81	0.31	0.67
std. deviation	0.67	0.39	0.37	0.47	0.23	0.51

Figure 7. (a) Re-projections after multi-layered belief propagation at the home position of C_1, computed from 7 pan-tilt views of C_1, (b) Reprojection after belief propagation within a graph containing C_1 at its home position. The red 'o' are the input points, the green '+' are the reprojections.

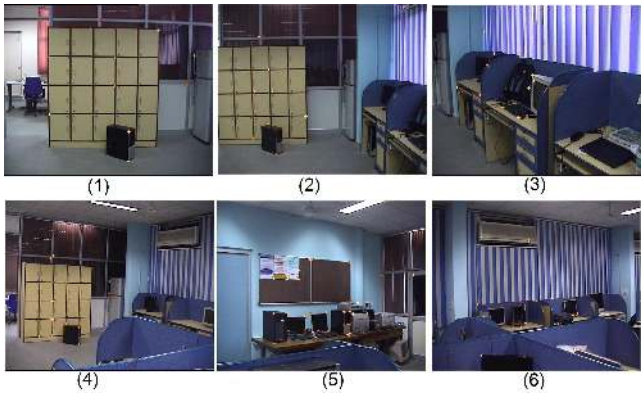

Figure 8. Reprojections of the input points in each of the six cameras in their home positions after the camera network is completely calibrated. The red 'o' are the input points and the yellow '+' are the reprojections.

References

[1] L. D. Agapito, E. Hayman, and I. Reid. Self-calibration of rotating and zooming cameras. *International Journal of Computer Vision*, 45(2):107–127, 2001.

[2] X. Chen, J. Davis, and P. Slusallek. Wide area camera calibration using virtual calibration objects, 2000. In Proc. IEEE Conference in Computer Vision and Pattern Recognition.

[3] D. Devarajan and R. Radke. Calibrating distributed camera networks using belief propagation, 2007. EURASIP JASP Special issue on Visual Sensor Networks.

[4] E. Elhamifar and R. Vidal. Distributed calibration of camera sensor network, 2009. In Proc. International Conference on Distributed Smart Cameras.

[5] S. Funiak, C. Guestrin, M. Paskin, and R. Sukthankar. Distributed localization of networked cameras, 2006. In Proc. Intl. Conf. on Information Processing in Sensor Networks.

[6] R. I. Hartley and A. Zisserman. Multiple view geometry in computer vision, 2004. Cambridge University Press.

[7] J. Jannotti and J. Mao. Distributed calibration of smart cameras, 2006. In Proc. Intl. Workshop on Distributed Smart Cameras.

[8] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing. Available from: <http://www.csse.uwa.edu.au/∼pk/research/matlabfns/>.

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

[10] H. Medeiros, H. Iwaki, and J. Park. Online distributed calibration of a large network of wireless cameras using dynamic clustering, 2008. In Proc. of International Conference on Distributed Smart Cameras.

[11] K. Murphy, Y. Weiss, and M. Jordan. Loopy belief propagation for approximate inference: An empirical study., 1999. UAI.

[12] S. Sinha and M. Pollefeys. Towards calibrating a pan-tilt-zoom camera network, 2004. In Proc. workshop on Omnidirectional Vision and Camera Networks in conjunction with ECCV.

[13] T. Svoboda, D. Martinec, T. Pajdla, J. Bouguet, T. Werner, and O. Chum. Multi-camera self-calibration code, 2005. http://cmp.felk.cvut.cz/švoboda/SelfCal/.