Model-Based Pose Estimation by Consensus

Anne Jorstad [1], Philippe Burlina [2], I-Jeng Wang [2], Dennis Lucarelli [2], Daniel DeMenthon [2]
[1] *Applied Mathematics and Scientific Computation, University of Maryland College Park*
[2] *Johns Hopkins Applied Physics Lab*

Abstract

We present a system for determining a consensus estimate of the pose of an object, as seen from multiple cameras in a distributed network. The cameras are pointed towards a 3D object defined by a configuration of points, which are assumed to be visible and detected in all camera images. The cameras are given a model defining the 3D configuration of these object points, but do not know which image point corresponds to which object point. Each camera estimates the pose of the object, then iteratively exchanges information with its neighbors to arrive at a common decision of the pose over the network. We consider eight variations of the consensus algorithm, and find that each converges to a more accurate result than do the individual cameras alone on average. The method exchanging 3D world coordinates penalized to agree with the input model provides the most accurate results. If bandwidth is limited, performing consensus over rotations and translations requires cameras to exchange only the six values specifying the six degrees of freedom of the object pose, and performing consensus in SE(3) using the Karcher mean is generally the best choice. We show further that interleaving pose calculation with the consensus iterations improves the final result when the image noise is large.

1. INTRODUCTION

We address the problem of recovering the pose (position and orientation) of an object as viewed by a camera sensor network. It is assumed that the object fits entirely within the field of view of each camera, and each camera is given a model of the object. Each camera is able to detect all model points in its image, but the correspondence between the image points and the model points is not known. Each camera determines its own estimate of the object pose using the SoftPosit algorithm [2], which determines the best fit of a known model to a set of feature points in an image, and returns a rotation matrix and translation vector from the camera origin. The cameras then share local pose estimates with their neighboring cameras in an iterative fashion, and information is dispersed through the network using variations of the consensus algorithm [13]. We consider two classes of consensus approaches: those that pass the 3D world coordinates of the object between the cameras, and those that pass the rotation matrices and translation vectors themselves and perform consensus in the space of rotations. We also look at the effect of interleaving pose estimation with the consensus iterations. Consensus is reached when all cameras agree on a single pose, and the results of Monte Carlo simulations for each consensus variation are presented. We see that each consensus method converges to a more accurate result than do the individual cameras alone on average, and performing consensus over penalized world coordinates and using the Karcher mean in SE(3) offer the best overall results.

The idea of nodes in a network iteratively sharing information with their neighbors to arrive at a common estimate has been addressed by several message passing algorithms including the consensus algorithm [13]. Since its introduction, consensus has been applied and extended in several ways. The traditional consensus algorithm performs averaging in Euclidean spaces, but we also want to consider the space of rotations, and an extension using a special definition of the mean on a Riemannian manifolds was used in [17]. An alternate approach [18] was developed for performing consensus on the 3D special orthogonal group, based on Karcher averaging [10] applied to pose estimation. Reference [6] looked at the problem of network self-localization using cameras and magnetometers in a distributed fashion. A Kalman filter was cast into a consensus algorithm in [1] [14] where each sensor estimate is based on its own and its neighbors' information. A distributed optimization method was described in [16] which can also potentially be used for robust estimation. Alternate message passing methods such as belief propagation have also been widely used for distributed fusion [4] [5].

Computer vision research has made significant progress in estimating pose information in a scene observed by a set of cameras [7] [9]. This problem has seen the development of many algorithms, most of which make the assumption that a central computing entity has access to all cameras' input. Instead, we consider the same problem for a distributed camera sensor network where connectivity between camera nodes is limited, and there is no centralized processing unit to collect data from each node and perform computations. Each camera is responsible for its own computation (feature detection and pose estimation) and must then share and fuse data with its neighbors to collectively arrive at a common estimate of the scene pose information. This type of algorithm is useful for a number of applications, including space exploration (many robots collaborating to repair a spacecraft), disaster recovery (robots performing reconnaissance and damage assessment after an earthquake), and video surveillance.

978-1-4244-2957-8/08/$25.00 © 2008 IEEE

2. CAMERA AND NETWORK CONFIGURATION

The camera network considered here consists of eight cameras pointed at a known object, see Fig. 1. The object fits completely within the field of view of each camera, and each camera is given a model of the object as collection of 3D points. All object points are visible to all cameras, and no clutter points are added. The SoftPosit algorithm used to find the initial pose estimation performs well when these assumptions are relaxed, but adding such extra unknowns complicates the comparison of camera network consensus algorithms. Each camera is able to convert its pose estimation to a common reference frame.

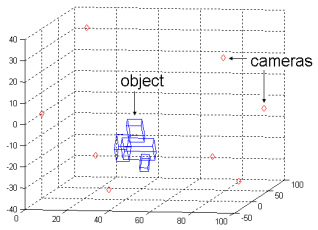
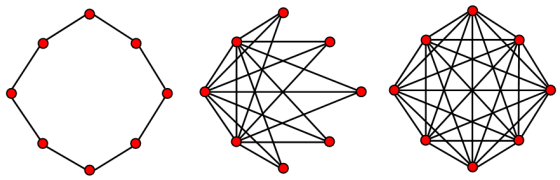


Fig. 1: Virtual setup of 8 cameras looking at object with 32 model points.

Three different network topologies were considered. Depending on the amount of connectivity between the cameras, the system will reach consensus at different rates, and the values at convergence will differ. Consensus theory tells us that λ_2, the second smallest eigenvalue of the graph Laplacian L, is the algebraic connectivity of the graph, and a larger λ_2 will lead to faster convergence [13]. The system will converge as long as the network is connected. In the first topology considered, each camera is connected to two neighbors, forming one single graph cycle, and $\lambda_2 = 0.5858$, see Fig. 2(a). In the second, three cameras act as hubs, connected to every other camera, while the other five cameras are only connected to the hubs, $\lambda_2 = 3$, Fig. 2(b). In the third, the network is fully connected, $\lambda_2 = 8$, Fig. 2(c).



(a) Single cycle (b) Three hubs (c) Fully connected

Fig. 2: The three network topologies considered.

3. INITIAL POSE ACQUISITION

The SoftPosit algorithm [2] takes in a model of an object, specified by a set of 3D object points, and a set of potential feature points in an image, and determines the best image-to-model feature correspondence and pose of the object that fits a subset of the feature points. A rotation matrix R and a translation vector $\overrightarrow{T}$ from the camera origin are returned, along with a correspondence matrix C matching model points to feature points in the image. This simultaneous pose and correspondence problem is formulated as a minimization of a global objective function:

$$E = \sum_{j=1}^{N} \sum_{k=1}^{M} c_{jk}(d_{jk}^2 - \alpha) \tag{1}$$

where there are M model points, N image points, c_{jk} are elements of the correspondence matrix, d_{jk} is the distance between the observed image point j and the potential projected model point k, and α is the error allowed in matching points, which depends on σ, the assumed standard deviation of the image noise.

The traditional Posit algorithm [3] assumes that point correspondences between the image and model are known, and uses a scheme based on updating scaled orthographic projections to iteratively solve for the pose. A single Posit step uses the homogeneous coordinate transformation equation to estimate the homogeneous scale from the current rotation matrix and translation vector, then uses this scale to obtain a better estimate for the rotation and translation. The full SoftPosit algorithm iterates between Posit, to determine pose, and deterministic annealing, to determine correspondences. Given an initial guess for pose, the algorithm determines the distances d_{jk} which are then used to predict the best correspondence c_{jk} with respect to a slowly increasing annealing variable β:

$$c_{jk} = \exp\left[-\beta(d_{jk}^2 - \alpha)\right]. \tag{2}$$

Given the current correspondences, a new prediction for the pose is found using Posit, which then allows a new set of distance values to be calculated.

4. CONSENSUS ALGORITHMS

Once each camera has obtained its own prediction of the pose of the object, these estimates must be combined to generate a system-wide pose consensus. In a distributed network, there is no centralized processor. Instead, each camera can only communicate with its immediate neighbors, and information is propagated through the nodes. In the traditional consensus algorithm over N nodes [13], consensus is reached when all nodes agree on a single value, which will be equal to the average of all values in the network if the connections between all nodes are equally weighted. To iteratively update each node in the network, the following update scheme can be applied:

$$x_i^{(k+1)} = x_i^{(k)} + \epsilon \sum_{j \in \mathcal{N}_i} (x_j^{(k)} - x_i^{(k)}), \tag{3}$$

where k is the iteration number, ϵ is the step size, $\mathcal{N}_i$ is the index set of the neighbors of node x_i, and the algorithm is initialized with nodes $x_i^{(0)}$. To guarantee convergence, the positive step size ϵ must be less than $1/\Delta$, where Δ is the maximum degree of any node in the network [13]. It can be seen that this update is a gradient descent method, minimizing the following objective function:

$$\phi(x_1, ..., x_N) = \frac{1}{2} \sum_{i,j} a_{ij}(x_i - x_j)^2, \tag{4}$$

where a_{ij} are elements of the adjacency matrix defining the network connectivity. The minimum is attained when all nodes agree on one consensus value.

We study several variations of the consensus algorithm for pose estimation. Two classes of consensus schemes are considered: consensus on 3D world coordinates, and consensus over rotations.

In the next sections we use the following notation: M = number of model points; N = number of cameras; R_i = rotation matrix from camera i to the object; $\overrightarrow{T}_i$ = translation vector from camera i to the object; $\overrightarrow{X}_{i,m}$ = estimated 3D world coordinates of object point m from camera i, in a common reference frame; $\overrightarrow{P}_m$ = actual 3D world coordinates of object point m of the true solution, in a common reference frame with origin $\overrightarrow{P}_0$; $\overrightarrow{Q}_m$ = input 3D coordinates of model point m, in a reference frame parallel to the common frame with origin $\overrightarrow{Q}_0$.

A. Consensus on World Coordinates

1) Unpenalized Consensus on World Coordinates: The object model can be rotated and translated by the pose computed from SoftPosit to provide 3D world coordinates for every model point of the initial estimate from each camera. This consensus algorithm performs its averaging over these coordinates directly, as in (3). The update equation is:

$$\overrightarrow{X}_{i,m}^{(k+1)} = \overrightarrow{X}_{i,m}^{(k)} + \epsilon \sum_{j \in \mathcal{N}_i} (\overrightarrow{X}_{j,m}^{(k)} - \overrightarrow{X}_{i,m}^{(k)}), \tag{5}$$

where the algorithm is initialized by the SoftPosit output points $X_{i,m}^{(0)}$. Consensus is reached when the difference between estimates of neighboring connected cameras is sufficiently small.

2) Penalized Consensus on World Coordinates Using the Model: The first method (5) makes no guarantee that the final solution is consistent with the object model. To rectify this, we add to the minimization equation (4) a term which penalizes any discrepancies with the model, weighted by penalty constant $\gamma \in [0, 1]$. The penalty is measured in relation to model point $\overrightarrow{Q}_0$, the origin of the model coordinate system. This is the point that the Posit algorithm uses to construct the translation from the camera to the object, and is generally the most accurate point of the SoftPosit estimation. The objective function with regard to object point m becomes:

$$\begin{aligned}\phi(\overrightarrow{X}_{1,m},\ldots,\overrightarrow{X}_{N,m}) = \frac{1}{2}\sum_{i,j} a_{ij}\|\overrightarrow{X}_{i,m} - \overrightarrow{X}_{j,m}\|^2 \\ + \gamma\frac{1}{2}\sum_i \|(\overrightarrow{X}_{i,m} - \overrightarrow{X}_{i,0}) - (\overrightarrow{Q}_m - \overrightarrow{Q}_0)\|^2.\end{aligned} \tag{6}$$

If $\gamma = 0$, we have the original consensus as in (5), and as γ gets larger the consensus result conforms more to the model. If $\gamma = 1$, the consensus algorithm is equivalent to performing consensus over the locations of the model reference points $\overrightarrow{X}_{i,0}$ alone, and then simply adding in the rest of the input model as defined by the internal model vectors $\overrightarrow{Q}_m$. The penalty constant γ is therefore a measure of the user's confidence in the model. Convergence will be reached for all $\gamma \in [0, 1]$, as both endpoints follow the traditional consensus algorithm, and all values in between result in a quadratic programming problem. In this study, $\gamma = \frac{1}{10}$.

Applying the gradient descend method to (6), the penalized update equation is:

$$\begin{aligned}\overrightarrow{X}_{i,m}^{(k+1)} =& \overrightarrow{X}_{i,m}^{(k)} + \epsilon \sum_{j \in \mathcal{N}_i} (\overrightarrow{X}_{j,m}^{(k)} - \overrightarrow{X}_{i,m}^{(k)}) \\ & - \epsilon\gamma[(\overrightarrow{X}_{i,m}^{(k)} - \overrightarrow{X}_{i,0}^{(k)}) - (\overrightarrow{Q}_m - \overrightarrow{Q}_0)].\end{aligned} \tag{7}$$

B. Consensus over Rotations

Instead of averaging the 3D world space coordinates, the consensus can take place over the rotations and translations themselves. A rotation and a translation each have three degrees of freedom, so performing consensus iterations over the rotations and translations consists of passing six arguments, instead of the 3D coordinates of M model points, and in general, $6 \ll 3M$. It can be shown [11] that rotations and translations can be averaged separately to achieve the correct final result. Each element of the translation vector can be averaged just like the world coordinates in (5):

$$\overrightarrow{T}_i^{(k+1)} = \overrightarrow{T}_i^{(k)} + \epsilon \sum_{j \in \mathcal{N}_i} (\overrightarrow{T}_j^{(k)} - \overrightarrow{T}_i^{(k)}). \tag{8}$$

Elements of a rotation matrix cannot be averaged so directly. Doing so does not produce meaningful values, and generally does not return a valid rotation matrix. Instead, rotation calculations must be completed in an appropriate, non-Euclidean space.

1) Consensus in Axis-Angle Representation: Any rotation induced by a rotation matrix R can be represented by an angle θ and a normalized axis of rotation $\vec{u}$ [15]:

$$\theta = cos^{-1}\left(\frac{trace(R) - 1}{2}\right), \tag{9}$$

$$\vec{u} = \frac{1}{2sin(\theta)}\begin{bmatrix} R_{3,2} - R_{2,3} \\ R_{1,3} - R_{3,1} \\ R_{2,1} - R_{1,2} \end{bmatrix}, \tag{10}$$

and from Rodrigues' rotation formula:

$$R = \begin{bmatrix} 0 & -\vec{u}(3) & \vec{u}(2) \\ \vec{u}(3) & 0 & -\vec{u}(1) \\ -\vec{u}(2) & \vec{u}(1) & 0 \end{bmatrix} sin(\theta) + [I - \vec{u}\vec{u}^T]cos(\theta) + \vec{u}\vec{u}^T. \tag{11}$$

In this sense rotations can be summed as follows, using scaled rotation vectors $\theta_i \cdot \vec{u}_i$:

$$\vec{u}_{sum} = \sum_{i=1}^{N} \theta_i \cdot \vec{u}_i, \tag{12}$$

$$\theta_{ave} = \frac{1}{N}\|\vec{u}_{sum}\|, \tag{13}$$

$$\vec{u}_{ave} = \frac{1}{\theta_{ave} \cdot N} \vec{u}_{sum}. \quad (14)$$

The consensus update step then becomes:

$$(\theta_i \cdot \vec{u}_i)^{(k+1)} = (\theta_i \cdot \vec{u}_i)^{(k)} + \epsilon \sum_{j \in \mathcal{N}_i} (\theta_j \cdot \vec{u}_j)^{(k)} \quad (15)$$

where θ and $\vec{u}$ at consensus can be determined by normalizing the resulting vector, and these can be converted back to a traditional rotation matrix using equation (11).

This method produces an exact barycentric average in the space of rotations, but it cannot take into account the continuous equivalence of θ across $0 = 2\pi$, so boundary conditions skew the result away from the true average [15].

2) Consensus in SE(3) using the Karcher Mean: In order to calculate the true mean of a set of rotations in the manifold of rotation matrices, known as SO(3), we consider the Riemannian distance between rotations [12]. This distance is the length of the shortest geodesic curve connecting rotations which lies entirely within SO(3). The element of SO(3) which minimizes the sum of squared distances between itself and all members of a set of rotations is called the Karcher mean [10] of the set. SO(3) is a Lie group, and so(3), the space of skew symmetric matrices, is its corresponding Lie algebra. This means that a metric can be defined over elements in each, and the Karcher mean can be calculated with the help of these algebraic constructs. The space of rotation matrices in SO(3) plus 3D vectors in $\mathbb{R}^3$ is referred to jointly as SE(3). Elements in so(3) can be considered as a matrix, $\hat{w}$, or as a vector, w:

$$\hat{w} = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix}, \quad (16)$$

$$w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}. \quad (17)$$

The $\log()$ function converts elements of SO(3) to so(3). With θ defined in (9), $\log(R)$ is equal to $\vec{u}$ in (10):

$$\log(R) = \vec{u}. \quad (18)$$

The $\exp()$ function converts elements of so(3) to SO(3):

$$\theta = \|w\|_2, \quad (19)$$

$$\exp(w) = I + \hat{w} sin(\theta) + \hat{w}^2(1 - cos(\theta)). \quad (20)$$

It is straightforward to verify that $\exp(w)$ in (20) is equivalent to the definition for R in (11), given $\|w\| = 1$. But here the consensus updates will be performed on the manifold.

As in [18], one version of the consensus update is:

$$R_i^{(k+1)} = R_i^{(k)} \cdot \exp\left(\epsilon \sum_{j \in \mathcal{N}_i} \log\left[(R_i^T)^{(k)} \cdot R_j^{(k)}\right]\right), \quad (21)$$

but the result of these iterations is just an approximation to the Karcher mean. To calculate the true mean, we use the output of the above iterations to start each camera's approximation with the same initial value that is close to the true average. The second set of iterations use the input $R_j^{(0)}$:

$$R_i^{(k+1)} = R_i^{(k)} \cdot \exp\left(\frac{1}{N} \sum_{j \in \mathcal{N}_i} \log\left[(R_i^T)^{(k)} \cdot R_j^{(0)}\right]\right). \quad (22)$$

These iterations converge to the true Karcher mean in the manifold. Consensus in both cases is reached when the sum of the terms inside the $\exp()$ is sufficiently small.

Remark: Consensus in Quaternions: Taking a direct linear average of quaternions is fundamentally an approximation [8]. Although the approximation is good for small angles, the inaccuracy can be significant for large angles. Our experiments demonstrated that consensus in quaternions is significantly less accurate than all other methods presented here, and quaternions will not be discussed further.

C. Interleaving Posit Iterations

Depending on the input and constraints imposed on SoftPosit, the pose or correspondences returned might not be close to the true solution. Such an incorrect solution negatively influences the consensus average, and the pose of these solutions should be further modified during the consensus iterations. This is achieved by inserting a Posit step between each consensus step. Posit is the part of the SoftPosit algorithm that adjusts the pose, assuming point correspondences are known. For each camera, the Posit iteration realigns the pose to better match the object point locations from the original input image, to locally rectify some of the error generated by the consensus process.

Algorithm:

1) SoftPosit on each camera $\rightarrow R_i^{(0)}, T_i^{(0)}$
2) Iterate for each camera:
   a) Consensus: update from neighbors $\rightarrow R_i^{(k_a)}, T_i^{(k_a)}$
   b) Posit: update own estimate from image $\rightarrow R_i^{(k_b)}, T_i^{(k_b)}$

Interleaving Posit steps within the consensus algorithm helps rectify faulty pose estimates, as the incoming information from other cameras can pull a bad estimate out of a local minimum and encourage it to converge to a more globally accurate solution. However, as correspondences are not changed, this addition has only limited effectiveness. In fact, we will see that adding Posit iterations is only a good idea when the amount of noise in the original images is large.

5. EXPERIMENTS

A virtual system was constructed in which a known object with 32 model points was viewed by eight arbitrarily placed cameras between three and seven times the size of the object away from the object, see Fig. 1. Each camera is calibrated, facing the object, and the object fits completely within each

camera's field of view. For each camera, mean zero random noise with standard deviation $\sigma = 2, 4, 8, 12$ was added, and 50 SoftPosit trials were run for each case, with the output then being fed into each consensus algorithm. At first, each camera was allowed to communicate with only two neighbors, forming a single cycle network topology, as in Fig. 2(a). Each of the described consensus methods was run on its own, and again interleaving Posit steps. Trials were repeated with the distances from the cameras to the object multiplied by three. All errors are computed as the difference between the coordinates of each estimated object point at consensus, X_m, and the true coordinates in 3D world space, P_m. We compare the average error and the maximum error for each class of trials:

$$E_{ave} = \frac{1}{M} \sum_{m=1}^{M} \|\overrightarrow{X}_m - \overrightarrow{P}_m\|, \tag{23}$$

$$E_{max} = \max_m \|\overrightarrow{X}_m - \overrightarrow{P}_m\|. \tag{24}$$

As a baseline, we consider the average error from an individual camera, calculated by taking the average and maximum of the above errors over all points from each camera's SoftPosit output, and referred to in the plots as Direct Averaging:

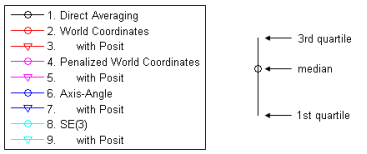
$$E_{DirectAve} = \frac{1}{MN} \sum_{m=1}^{M} \sum_{i=1}^{N} \|\overrightarrow{X}_{i,m} - \overrightarrow{P}_m\|. \tag{25}$$

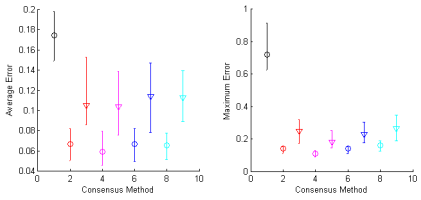
The consensus result will return the average error of each data point over all cameras, which is in general smaller than (25):

$$E_{ConsensusAve} = \frac{1}{M} \sum_{m=1}^{M} \|\frac{1}{N} \sum_{i=1}^{N} \overrightarrow{X}_{i,m} - \overrightarrow{P}_m\|. \tag{26}$$

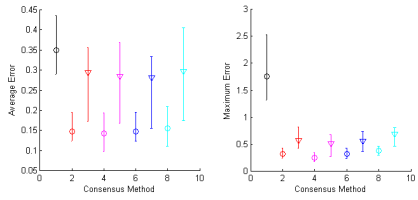
The results for the closer cameras are presented in Fig. 3. Similar results were obtained for the further cameras, but with larger overall errors as expected. In Fig. 4, the errors from all methods are plotted together for comparison. We see that all consensus methods result in less error on average than the direct average error from the individual cameras. Performing consensus using penalized world coordinates produces the lowest error on average for every class of trials. However, the results are not very distinct from the other methods when the image noise level is low or the cameras are close to the object, especially when considering the overlap of the distributions of errors. Adding Posit steps appears to be a bad idea when there is not much image noise, but a better idea as the noise increases. In the trials with large noise standard deviation, the translation output from SoftPosit is reasonable, but the rotations are often quite far off, so adding Posit iterations benefits some of these cases.

If communications between cameras have limited bandwidth, then it is desirable to pass as little information as possible between the cameras. Using penalized world coordinates, $3M$ values must be exchanged for a model with M model points. However, using either of the methods that perform consensus over rotations, only 6 values need to be exchanged (a 3D translation vector plus 3 angles of rotation). Consensus

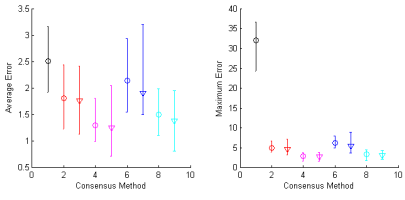




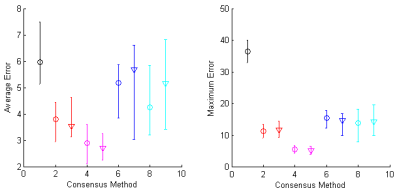
(a) std of image noise = 2



(b) std of image noise = 4



(c) std of image noise = 8



(d) std of image noise = 12

Fig. 3: The average errors (left plot) and maximum errors (right plot) in pixels are presented for each noise standard deviation, with an error bar denoting the 1st quartile, median, 3rd quartile, with cameras placed 3 to 7 times the size of the object away from the object.

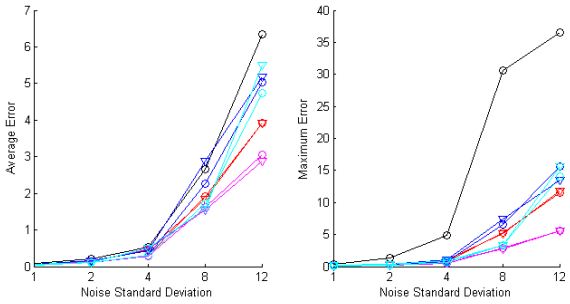
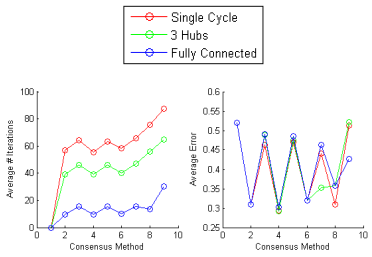
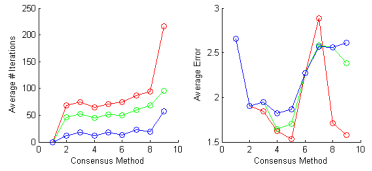


Fig. 4: Average and maximum errors from each consensus method plotted together, with cameras placed 3 to 7 times the size of the object away.



(a) std of image noise = 4



(b) std of image noise = 8

Fig. 5: For each network topology, each consensus method is plotted against the average number of iterations (left plot), and the average errors (right plot), for two image noise standard deviations. The consensus methods are ordered along the x-axis as defined in Fig. 3.

in SE(3) using the Karcher mean is seen to perform slightly better on average than the Axis-Angle method.

When comparing the three topologies from Fig. 2, all show similar errors, but the fully connected network with the largest λ_2 converges in the fewest number of iterations, while the single cycle network with significantly fewer edges converges the slowest. The error between the true solution and the final result from the single cycle is on average slightly less than the others for the nonstandard consensus methods. See Fig. 5 for results.

The bottleneck of the proposed algorithm is the SoftPosit step, which is really preprocessing for the consensus methods. Each consensus method takes very little time to converge, see Table 1. Adding Posit steps takes approximately 30% longer to converge. Performing consensus over rotations is faster than over world coordinates, because only six data points need to be exchanged at each step.

TABLE 1: AVERAGE TIMES TO CONVERGENCE FOR EACH METHOD WITH NO POSIT STEPS, AS RUN IN MATLAB ON A DESKTOP PC.

	World Coordinates	Penalized WCs	Axis-Angle	SE(3)
Time (seconds)	0.40	0.93	0.30	0.22

6. CONCLUSIONS

We have presented a set of algorithms for determining the pose of a known object as seen from several cameras in a decentralized network. All consensus methods presented converge to a more accurate result than do the individual cameras alone on average. Performing consensus over world coordinates penalized by an object model was found to result in the lowest overall error, but this method requires exchanging three values for every model point at every iteration, which can be slow and require significant bandwidth. If the system is bandwidth-limited, it is instead desirable to perform consensus over the rotation and translation of the object in a common reference frame, as this requires exchanging only six values in total. We see that performing consensus in SE(3) using the Karcher mean produces good results in this limited bandwidth setting. The average error of this method is comparable to that of the penalized world coordinates method for low and midrange image qualities, and this method performs the fastest out of all tested. As the noise level of the input images increases, it is beneficial to add Posit steps which reference the original input images between consensus steps, to help rectify some of the error returned from the original SoftPosit pose estimate. The number of iterations required for convergence can be decreased by increasing the connectivity of the camera network.

ACKNOWLEDGMENTS

We wish to thank R. Tron, R. Vidal and A. Terzis for kindly making their code available as described in [18].

REFERENCES

[1] R. Carli, A. Chiuso, L. Schenato, S. Zampieri. “Distributed Kalman Filtering Based on Consensus Strategies.” IEEE Journal on Selected Areas in Communications, vol. 26, May 2008.

[2] P. David, D. DeMenthon, R. Duraiswami, H. Samet. “Softposit: Simultaneous Pose and Correspondence Determination.” International Journal of Computer Vision, 59, No. 3, pp. 259-284, September-October 2004.

[3] D. DeMenthon, L.S. Davis. “Model-Based Object Pose in 25 Lines of Code.” International Journal of Computer Vision, pp. 123-141, June 1995.

[4] D. Devarajan, R. Radke. “Calibrating Distributed Camera Networks Using Belief Propagation.” EURASIP Journal of Applied Signal Processing, 2007.

[5] D. Devarajan, R. Radke, H. Chung. “Distributed Metric Calibration of Ad-Hoc Camera Networks.” ACM Transactions on Sensor Networks, vol. 2, no. 3, pp. 380-403, 2006.

[6] R. Farrell, R. Garcia, D. Lucarelli, A. Terzis, and I-J. Wang. “Localization in Multi-Modal Sensor Networks.” Third International Conference on Intelligent Sensors, Sensor Networks and Information Processing, 2007.

[7] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint.* MIT press, 1993.

[8] C. Gramkow. “On Averaging Rotations.” Journal of Mathematical Imaging and Vision, vol. 15, pp. 7-16, 2001.

[9] R. Hartley, A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2004.

[10] H. Karcher. “Riemannian Center of Mass and Mollifier Smoothing.” Communications on Pure and Applied Mathematics, vol. 30, 1977.

[11] Y. Ma, J. Košecká, S. Sastry. “Optimization Criteria and Geometric Algorithms for Motion and Structure Estimation.” International Journal of Computer Vision, vol. 44, No. 3, pp. 219-249, September 2001.

[12] M. Moakher. “Means and Averaging in the Group of Rotations.” Siam Journal on Matrix Analysis and Applications, vol. 24, No. 1, 2002.

[13] R. Olfati-Saber, J.A. Fax, R. Murray. “Consensus and Cooperation in Networked Multi-Agent Systems.” Proceedings of the IEEE, Vol. 95, No. 1, pp. 215-233, January 2007.

[14] R. Olfati-Saber. “Distributed Kalman Filter with Embedded Consensus Filters.” IEEE Conference on Decision and Control, 2005.

[15] X. Pennec, N. Ayache. “Uniform Distribution, Distance and Expectation Problems for Geometric Features Processing.” Journal of Mathematical Imaging and Vision, vol. 9, pp. 4967, 1998.

[16] M. Rabbat, R. Nowak. “Distributed Optimization in Sensor Networks.” Proceedings of the Third International Symposium on Information Processing in Sensor Networks, April 2004.

[17] A. Sarlette, R. Sepulchre. “Consensus Optimization on Manifolds.” SIAM Journal of Control and Optimization, 2008.

[18] R. Tron, R. Vidal, A. Terzis. “Distributed Pose Averaging in Camera Networks Via Consensus on SE(3).” IEEE International Conference on Distributed Smart Cameras, September 2008.