Distributed Consensus on Camera Pose

Anne Jorstad, *Student Member, IEEE*, Daniel DeMenthon, *Senior Member, IEEE*, I-Jeng Wang, *Member, IEEE*, and Philippe Burlina, *Senior Member, IEEE*

***Abstract*—Our work addresses pose estimation in a distributed camera framework. We examine how processing cameras can best reach a consensus about the pose of an object when they are each given a model of the object, defined by a set of point coordinates in the object frame of reference. The cameras can only see a subset of the object feature points in the midst of background clutter points, not knowing which image points match with which object points, nor which points are object points or background points. The cameras individually recover a prediction of the object's pose using their knowledge of the model, and then exchange information with their neighbors, performing consensus updates locally to obtain a single estimate consistent across all cameras, without requiring a common centralized processor. Our main contributions are: 1) we present a novel algorithm performing consensus updates in 3-D world coordinates penalized by a 3-D model, and 2) we perform a thorough comparison of our method with other current consensus methods. Our method is consistently the most accurate, and we confirm that the existing consensus method based upon calculating the Karcher mean of rotations is also reliable and fast. Experiments on simulated and real imagery are reported.**

***Index Terms*— Computer vision, distributed computing, iterative pose estimation, Karcher mean, multiple view geometry, multiview geometry, object model, penalized consensus, sensor fusion, SoftPOSIT, video surveillance.**

I. Introduction

COMPUTER vision research has made significant progress in estimating motion and structure information in a scene observed by a set of cameras [7], [9]. This topic has seen the development of many algorithms for binocular, trinocular, or multiview geometries, all of which must make the assumption that a central computing entity (a collector node) has access to all cameras' input. We consider the alternate paradigm that has been adopted by researchers in distributed sensor networks whereby connectivity between camera nodes is limited, and there is no centralized processing unit to collect data from each node and perform computations. Instead, each camera is responsible for its own computation (feature detection and pose estimation) and must then share and fuse data with its neighbors. When the pose estimate from each camera agrees with those from its immediate neighbors, then by induction the entire connected network has arrived at a single common estimate of the scene pose information. A possible scenario motivating our work is for search and rescue missions where several small and agile robots with limited communication and processing power are tasked to roam through a disaster zone and collaborate to find injured persons and assess damage to buildings and infrastructure. The algorithm we present would also be useful for a number of other applications such as distributed video surveillance, SLAM, space exploration (many robots collaborating to dock into or repair a spacecraft), or assistive environments.

We address the problem of recovering the pose (position and orientation) of an object as viewed by a distributed camera network, see Fig. 1, extending our previous work in [10]. It is assumed that each camera is given a model of the object as a collection of 3-D vectors expressed in the object frame of reference, that the cameras are internally and externally calibrated, and that not all points on the object are necessarily visible to each camera. The assumption we make about the presence of a known object model can be justified as follows: this object could be the model of one of the collaborating robots seen by all robots, or could come from a database of previously acquired 3-D models such as from Google 3-D Warehouse. Each camera is able to detect a subset of the model points in its image, but the correspondence between the image points and the model points is not known. The object is embedded in a scene with clutter points that may compromise establishing correspondences. 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, returning a rotation matrix and translation vector from the camera origin. The cameras then share local pose estimates in a common reference frame with their neighboring cameras in an iterative fashion, and information is dispersed through the network using variations of the consensus algorithm [15]. The object's pose information exchanged between the cameras can be fully represented using only six values: the three elements of its translation vector in the world coordinates, and the three angles of its rotation.

The process of having all nodes in a network iteratively share information with their neighbors to arrive at a common estimate has been addressed by several message passing algorithms including the consensus algorithm [15]. Consensus algorithms iteratively share information between nodes, and return an estimate at each node that can be shown to be the average of the

Manuscript received June 30, 2009; revised January 28, 2010. First published April 01, 2010; current version published August 18, 2010. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Amy R. Reibman.

A. Jorstad is with the Applied Mathematics and Statistics, and Scientific Computation Department, University of Maryland, College Park, MD 20742 USA. She is also with the Applied Physics Laboratory, The Johns Hopkins University, Baltimore, MD 21210 USA (e-mail: jorstad@math.umd.edu).

D. DeMenthon is with the Applied Physics Laboratory, The Johns Hopkins University, Baltimore, MD 21210 USA (e-mail: Daniel.DeMenthon@jhuapl.edu).

I.-J. Wang and P. Burlina are with the Applied Physics Laboratory and the Department of Computer Science, The Johns Hopkins University, Baltimore, MD 21210 USA (e-mail: I-Jeng.Wang@jhuapl.edu; Philippe.Burlina@jhuapl.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2010.2047167

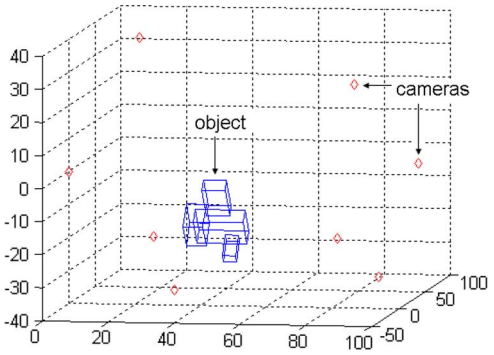


Fig. 1. Virtual setup of eight cameras looking at an object with 32 model points.

values at input. We chose to work in the consensus paradigm because theoretical convergence can be established even when the network topology has loops and, hence, the overheads incurred in constructing and maintaining a tree for communication, as required for most message passing algorithms to ensure theoretical convergence, are avoided. Also, consensus can handle time-varying topologies, which will be particularly relevant when we extend the techniques to operate in dynamic environments, when the object, the camera, or both, are moving. Since its introduction, consensus has been applied and extended in several ways. It has been used for cooperative distributed control to navigate swarms by exchanging local configuration information [17], [21]. The authors in [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], [16] where each sensor estimate is based upon its own and its neighbors' information. A distributed optimization method was described in [19] 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, see for example [4], [5]. Gossip algorithms have been developed for asynchronous systems where information is not shared between consistent pairs of nodes throughout the iterations. Our proposed method can be easily modified to fit into the gossip algorithm structure to enable asynchronous operations, and in this setting would inherit the convergence properties of traditional gossip algorithms, for example, [20] and [24].

Since it relies on linear operations, the traditional consensus algorithm can only correctly perform averaging in Euclidean spaces. One challenge of applying consensus to our distributed pose estimation problem formulation is that we have to handle rotations. For the case of rotations, consensus is not directly applicable since the special orthogonal group SO(3) of rotations is not Euclidean, and averaging rotation matrices does not yield a rotation matrix. Recently, new methods have been proposed in the literature to address this issue, including [18], [22], and [25]. In [25], consensus on rotations was performed using the Karcher mean previously introduced in [11], which finds the true mean of a set of rotations on the manifold of SO(3). Alternatively, in [18], averaging on rotations is performed using axis-angle representations. In [22], an extension of the definition of the mean was proposed to handle rotation averaging. These approaches rely on approximating the averages for rotations using axis-angle representations or quaternions, or require that the initial data be "close" in some sense, to ensure convergence to the actual Karcher mean.

The principal contribution of our paper is the development of a novel method based upon penalized averaging. Our method works directly in Euclidean space by averaging common estimates of the 3-D positions of the object model's salient points, which are indeed Euclidean quantities. The method obviates the need for special handling of rotations by introducing a rigidity constraint (a deformation penalty) that forces our point set to conform to the known model. This rigidity penalty steers the solution towards point estimates which strike a compromise between pure consensus estimates and consistency with the model. To allow for the computation of average rotations, our penalty replaces the constraints and approximations used in prior rotation-based methods, by a penalty expressed as a model conformance constraint: this penalty constrains the consensus estimate to preserve distances and in essence forces it to satisfy the requirement that it be an isometry. The benefit of our approach is that it is directly performed in the Euclidean space for which consensus was originally designed, meant to be applied, and for which extensive analysis results have been derived.

The second main contribution of our paper is to perform a thorough comparison of our method with other existing consensus methods. Two classes of consensus approaches are compared in this work: those that perform consensus on the 3-D world coordinates of the object points, and those that perform consensus in the space of rotations and translations. We also look at the effect of interleaving pose estimation with the consensus iterations. Consensus is reached when every camera agrees on the same pose as its neighbors', and the results of synthetic Monte Carlo simulations and real images for each consensus variation are presented. Traditional consensus algorithms converge to the average of the input data, and we see that the variations on the consensus method presented here generally result in slightly smaller but statistically equivalent error for small image noise, and in noticeably smaller error for larger image noise. We show through experimental results that our approach, while simpler, correctly computes and seamlessly disperses the average pose to the entire network of nodes.

The rest of the paper is organized as follows: Section II reviews the SoftPOSIT pose acquisition algorithm, Section III describes in detail each proposed consensus method; experimental results and discussion are presented in Section IV, and we conclude in Section V.

II. Pose Acquisition

We first address the problem of computing object pose. A pose computation algorithm is used to generate an initial pose estimation for each camera, and later is interleaved within the consensus iterations. The SoftPOSIT algorithm [2] takes in a model of an object, specified by a set of 3-D 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. These can then be used to generate 3-D world coordinates for all model points as estimated by

any given camera. This simultaneous pose and correspondence problem is formulated as a minimization of a global objective function to which a deterministic annealing schedule is applied

$$E = \sum_{j=1}^{J} \sum_{k=1}^{K} c_{jk} \left(d_{jk}^2 - \alpha \right) \tag{1}$$

where there are J image points, K model points, c_{jk} are elements of the correspondence matrix, d_{jk} is the distance between the observed image point j and the projected model point k, and α is the error allowed in matching points, which depends upon 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 upon updating perspective projections so that they become scaled orthographic projections to iteratively solve for the pose. A single Posit step uses the homogeneous coordinate transformation equation to estimate the homogeneous scale w from the current rotation matrix and translation vector, then uses this new w to obtain a better estimate for a new rotation and translation

$$R = \begin{bmatrix} \overrightarrow{R_1^{\mathrm{T}}} \\ \overrightarrow{R_2^{\mathrm{T}}} \\ \overrightarrow{R_3^{\mathrm{T}}} \end{bmatrix} \quad \overrightarrow{T} = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \tag{2}$$

$$w = \frac{\overrightarrow{R_3^{\mathrm{T}}} \cdot \overrightarrow{P_0 P_i}}{T_z} + 1 \tag{3}$$

$$\begin{bmatrix} wx_i \\ wy_i \\ w \end{bmatrix} = \left[\begin{array}{c|c} f\overrightarrow{R_1^{\mathrm{T}}} & fT_x \\ f\overrightarrow{R_2^{\mathrm{T}}} & fT_y \\ \overrightarrow{R_3^{\mathrm{T}}} & T_z \end{array}\right] \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} \tag{4}$$

where in the third equation, $[X_i\, Y_i\, Z_i\, 1]^T$ are the homogeneous coordinates of the vector $\overrightarrow{P_0 P_i}$, which is the vector in the model from the reference point P_0 to point P_i, and f is the focal length of the camera. The method of least squares is used to solve for R and $\overrightarrow{T}$ using the information from all points.

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} between image point j and the projection of model point k into the image under the current pose estimation. These distances are then used to predict a new best correspondence c_{jk} with respect to an annealing variable β

$$c_{jk} = \exp\left[-\beta \left(d_{jk}^2 - \alpha \right) \right]. \tag{5}$$

The full matrix C is normalized by alternately normalizing its rows and columns, a procedure known as Sinkhorn balancing [23]. This matrix defines the new prediction of correspondences. Given the current correspondences, a new prediction for the pose (R, $\overrightarrow{T}$ and w) is found using Posit, as defined previously. The next set of distance values can then be determined from the updated correspondences and pose.

III. 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 [15], consensus is reached when all nodes agree on a single scalar value, that is, when

$$x_1 = x_2 = \ldots = x_N = x_{\text{consensus}}. \tag{6}$$

If all connections between nodes are equally weighted, this value will be equal to the average of all initial values in the network

$$x_{\text{consensus}} = \frac{1}{N} \sum_{j=1}^{N} x_j. \tag{7}$$

To iteratively update each node in the network, the following update scheme can be applied:

$$x_i^{(k+1)} = x_i^{(k)} + \epsilon \sum_{j=1}^{N} w_{ij} \left(x_j^{(k)} - x_i^{(k)} \right) \tag{8}$$

where k is the iteration number, ϵ is the step size, w_{ij} is the weight given to the connection between node i and node j, and consensus is initialized with node values $x_i^{(0)}$. In our algorithm, we take the w_{ij} to be either 0 or 1, equally weighting all neighbors of each node to simplify the setup. Techniques have been proposed to optimize the weights used in consensus algorithms to speed up convergence or directly minimize the mean squared error. These techniques require explicit knowledge of the global topology and can be applied to our problem as long as the topology is stationary and the topological information is available at all nodes. In future work, we intend to investigate how to weight the importance of each node connection as an inverse function of its error, such as in [27]. The basic consensus update equation used here is

$$x_i^{(k+1)} = x_i^{(k)} + \epsilon \sum_{j \in \mathcal{N}_i} \left(x_j^{(k)} - x_i^{(k)} \right) \tag{9}$$

where $\mathcal{N}_i$ is the index set of the neighbors of node x_i. 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 [15]. It can be seen that this update is in fact a gradient descent method, minimizing the following objective function:

$$\phi(x_1, \ldots, x_N) = \frac{1}{2} \sum_{i,j} a_{ij} (x_i - x_j)^2 \tag{10}$$

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 3-D world coordinates, and consensus over

rotations. The consensus methods presented here generally result in values which are potentially different from the basic Euclidean average of (7), because the consensus updates are not performed in Euclidean space. Our results are seen to have a slightly smaller error on average than a pure Euclidean averaging of the SoftPOSIT output pose predictions.

Notation and Assumptions: In the next sections, we use the following notation: M is the number of model points; N is the number of cameras; $\overrightarrow{X}_{i,m}^{(k)}$ are the 3-D coordinates of object point m estimated by camera i after k consensus iterations, expressed in the *world* reference frame (estimated by camera i); $\overrightarrow{X}_m$ are the actual 3-D coordinates of object point m, expressed in the world reference frame (ground truth); $\overrightarrow{Q}_m$ are the 3-D coordinates of model point m, where model point $\overrightarrow{Q}_0$ is the origin, in the *object* reference frame (assumed known); $R_{O2W,i}^{(k)}$ is the rotation matrix transforming coordinates from the object to the world reference frame estimated by camera i after k consensus iterations; R_{O2W} is the rotation matrix transforming coordinates from the object to the world reference frame (ground truth); $\overrightarrow{T}_{O2W}$ is the translation vector placing the object in the world reference frame (ground truth).

From this notation, the object to world transformation is given by

$$\overrightarrow{X}_m = R_{O2W}\overrightarrow{Q}_m + T_{O2W}. \tag{11}$$

A. Consensus on World Coordinates

1) Direct Consensus on World Coordinates: We first present a direct application of the consensus algorithm to estimate the 3-D world coordinates of the object points. The object model can be rotated and translated by the pose computed from SoftPOSIT to provide 3-D world coordinates for every model point of the initial estimate from each camera. This first consensus algorithm performs its averaging over these coordinates directly, as in (9). The update equation is

$$\overrightarrow{X}_{i,m}^{(k+1)} = \overrightarrow{X}_{i,m}^{(k)} + \epsilon \sum_{j\in\mathcal{N}_i}\left(\overrightarrow{X}_{j,m}^{(k)} - \overrightarrow{X}_{i,m}^{(k)}\right) \tag{12}$$

where the algorithm is initialized by the SoftPOSIT output points $\overrightarrow{X}_{i,m}^{(0)}$. 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 (12) makes no guarantee that the final solution is consistent with the object model. To rectify this, we propose to add to the minimization (10) for vectors a term which penalizes any discrepancies with the model. For each model point we want to incorporate the rigidity constraint

$$\left(\overrightarrow{X}_{i,m}^{(k)} - \overrightarrow{X}_{i,0}^{(k)}\right) = R_{O2W}(\overrightarrow{Q}_m - \overrightarrow{Q}_0). \tag{13}$$

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 usually the most accurate point of the SoftPOSIT estimation. The estimated 3-D world coordinate vector from point $\overrightarrow{X}_{i,0}^{(k)}$ to point $\overrightarrow{X}_{i,m}^{(k)}$ is aligned through R_{O2W} to the frame of the model vector from $\overrightarrow{Q}_0$ to $\overrightarrow{Q}_m$. Therefore, the norm of the difference between these two vectors measures how much the estimate adheres to the model. The objective function introduces this penalty weighted by penalty constant $\gamma \in [0,1]$, and the resulting objective function with regard to object point m becomes

$$\phi\left(\overrightarrow{X}_{1,m}^{(k)},\ldots,\overrightarrow{X}_{N,m}^{(k)}\right) = \frac{1}{2}\sum_{i,j} a_{ij}\left\|\overrightarrow{X}_{i,m}^{(k)} - \overrightarrow{X}_{j,m}^{(k)}\right\|^2 + \frac{1}{2}\gamma\sum_i\left\|\left(\overrightarrow{X}_{i,m}^{(k)} - \overrightarrow{X}_{i,0}^{(k)}\right) - R_{O2W,i}^{(k)}(\overrightarrow{Q}_m - \overrightarrow{Q}_0)\right\|^2. \tag{14}$$

The first order optimality constraint requires that the derivative of this objective function be equal to zero, yielding a gradient-descent-like algorithm. Setting the derivatives to zero yields the penalized gradient descent update equation

$$\overrightarrow{X}_{i,m}^{(k+1)} = \overrightarrow{X}_{i,m}^{(k)} + \epsilon\sum_{j\in\mathcal{N}_i}\left(\overrightarrow{X}_{j,m}^{(k)} - \overrightarrow{X}_{i,m}^{(k)}\right) - \epsilon\gamma\left[\left(\overrightarrow{X}_{i,m}^{(k)} - \overrightarrow{X}_{i,0}^{(k)}\right) - R_{O2W,i}^{(k)}(\overrightarrow{Q}_m - \overrightarrow{Q}_0)\right]. \tag{15}$$

The rotation $R_{O2W,i}^{(k)}$ aligning the object and world reference frames is computed by considering the set of model points $\overrightarrow{Q}_m$ expressed in the object frame and the last consensus estimates of these points positions $\overrightarrow{X}_{i,m}^{(k)}$ expressed in the world coordinate system, so that the constraint in (13) is satisfied. This is solved using SVD as explained in [26]. The proposed algorithm then alternates between the computation of the penalized consensus step and the update of $R_{O2W,i}^{(k)}$.

If $\gamma = 0$, we have the original consensus as in (12), and as γ gets larger the consensus result conforms more to the model. The penalty constant γ is therefore a measure of confidence in the model. If $\gamma = 1$, the consensus algorithm is equivalent to performing consensus over the locations of the model reference points $\overrightarrow{X}_{i,0}^{(k)}$ alone, and then simply adding in the rest of the input model as defined by the internal model vectors $\overrightarrow{Q}_m$. With $\gamma = 1$, (15) becomes

$$\overrightarrow{X}_{i,m}^{(k+1)} = \overrightarrow{X}_{i,m}^{(k)} + \epsilon\left[\sum_{j\in\mathcal{N}_i}\left(\overrightarrow{X}_{j,m}^{(k)} - \overrightarrow{X}_{i,m}^{(k)}\right) + \overrightarrow{X}_{i,0}^{(k)} + R_{O2W,i}^{(k)}(\overrightarrow{Q}_m - \overrightarrow{Q}_0) - \overrightarrow{X}_{i,m}^{(k)}\right]. \tag{16}$$

For the origin of the object $(m = 0)$, the terms $\overrightarrow{X}_{i,0}^{(k)} - \overrightarrow{X}_{i,m}^{(k)} = 0$ and $\overrightarrow{Q}_m - \overrightarrow{Q}_0 = 0$ in (16), and the update is equivalent to the first method (12)

$$\overrightarrow{X}_{i,0}^{(k+1)} = \overrightarrow{X}_{i,0}^{(k)} + \epsilon\sum_{j\in\mathcal{N}_i}\left(\overrightarrow{X}_{j,0}^{(k)} - \overrightarrow{X}_{i,0}^{(k)}\right). \tag{17}$$

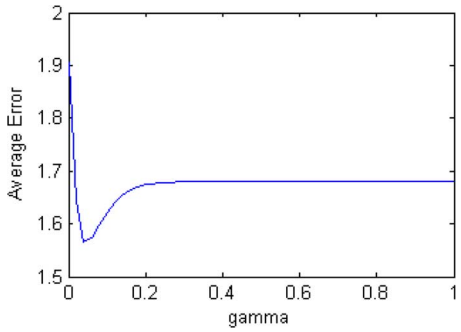


Fig. 2. As the penalty constant γ varies from 0 to 1, the average resulting error at convergence is presented for an image noise standard deviation of 8.

When $m \neq 0$, let $\overrightarrow{X}_{i_0,m}^{(k)} = \overrightarrow{X}_{i,0}^{(k)} + R_{O2W,i}(\overrightarrow{Q}_m - \overrightarrow{Q}_0)$. Then from (16)

$$\overrightarrow{X}_{i,m}^{(k+1)} = \overrightarrow{X}_{i,m}^{(k)} + \epsilon \sum_{j \in \mathcal{N}_i, i_0} \left(\overrightarrow{X}_{j,m}^{(k)} - \overrightarrow{X}_{i,m}^{(k)} \right). \quad (18)$$

This is now in the form of a regular consensus update equation. In the new term added to the sum, the points $\overrightarrow{X}_{i,0}^{(k)}$ are updated to reach consensus from (17), and $(\overrightarrow{Q}_m - \overrightarrow{Q}_0)$ is a constant vector provided in the input model. Because all point coordinates must be in agreement when consensus is reached, as in (6), all points $\overrightarrow{X}_{i,m}^{(k)}$ must converge to the value of $\overrightarrow{X}_{i_0,m}^{(k)}$, so for all m

$$\lim_{k \to \infty} \overrightarrow{X}_{i,m}^{(k+1)} = \overrightarrow{X}_{i,0}^{(0)} + R_{O2W,i}^{(k)}(\overrightarrow{Q}_m - \overrightarrow{Q}_0). \quad (19)$$

This means that when $\gamma = 1$, the values of all $\overrightarrow{X}_{i,m}$ at consensus simply add the input model vectors to the value determined for $\overrightarrow{X}_{i,0}$ at consensus.

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. A plot demonstrating how the final error at consensus changes depending upon γ is presented in Fig. 2. In this study, $\gamma = 1/10$.

B. Consensus Over Rotations

We compare our proposed penalized consensus approach for pose estimation to other methods relying on special handling of rotations. 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 3-D coordinates of M model points, and in general, $6 \ll 3\,\mathrm{M}$. It has been shown [13] that rotations and translations can be averaged separately to achieve the correct final result. It is assumed that each camera is able to convert its own rotation and translation into the coordinate frame of a single reference camera. If instead each camera only knows how to convert its own reference frame to those of its immediate neighbors, then at each iteration each rotation and translation must be adjusted for each camera, but this is a simple calculation that involves two matrix multiplications and one vector addition [25], and does not noticeably affect the consensus results.

Each element of the translation vector can be averaged just like the world coordinates in (12)

$$\overrightarrow{T}_i^{(k+1)} = \overrightarrow{T}_i^{(k)} + \epsilon \sum_{j \in \mathcal{N}_i} \left(\overrightarrow{T}_j^{(k)} - \overrightarrow{T}_i^{(k)} \right). \quad (20)$$

Elements of a rotation matrix cannot be directly averaged. Direct averaging 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 $\vec{u}$ [18]

$$\theta = cos^{-1}\left(\frac{trace(R) - 1}{2}\right) \quad (21)$$

$$\vec{u} = \frac{1}{2sin(\theta)} \begin{bmatrix} R_{3,2} - R_{2,3} \\ R_{1,3} - R_{3,1} \\ R_{2,1} - R_{1,2} \end{bmatrix} \quad (22)$$

and from Rodrigues' rotation formula

$$R = \begin{bmatrix} 0 & -\vec{u}(3) & \vec{u}(2) \\ \vec{u}(3) & 0 & -\vec{u}(1) \\ -\vec{u}(2) & \vec{u}(1) & 0 \end{bmatrix} sin(\theta) + [I - \vec{u}\vec{u}^T]cos(\theta) + \vec{u}\vec{u}^T. \quad (23)$$

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

$$\vec{u}_{\mathrm{sum}} = \sum_{i=1}^{N} \theta_i \vec{u}_i \quad (24)$$

$$\theta_{\mathrm{ave}} = \frac{1}{N} ||\vec{u}_{\mathrm{sum}}|| \quad (25)$$

$$\vec{u}_{\mathrm{ave}} = \frac{1}{\theta_{\mathrm{ave}} N} \vec{u}_{\mathrm{sum}}. \quad (26)$$

The consensus update step then becomes

$$(\theta_i \vec{u}_i)^{(k+1)} = (\theta_i \vec{u}_i)^{(k)} + \epsilon \sum_{j \in \mathcal{N}_i} (\theta_j \vec{u}_j)^{(k)} \quad (27)$$

where θ and $\vec{u}$ at consensus can be determined by normalizing the result, and these can be converted back to a traditional rotation matrix using (23). Consensus here is reached when the differences between any pair of axes and their corresponding angles are sufficiently small. This method produces an exact barycentric average in the space of rotations, but it does not effectively handle the continuous equivalence of θ across $0 = 2\pi$, so boundary conditions skew the result away from the true average [18].

2) Consensus in SE(3) Using the Karcher Mean: We compare also to the rotation consensus proposed in both [22] and [25] and based upon the work from [14], which exploits the Karcher mean as a way of performing averaging in the manifold of rotation matrices, known as SO(3). The main ideas of the Karcher mean-based consensus are summarized next. The Riemannian distance between a pair of rotations is the length of the shortest geodesic curve between the 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 [11] of the set. SO(3) is a Lie group, and so(3), the space of skew symmetric matrices, is its corresponding Lie algebra, and is tangent to SO(3) at the identity. 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 3-D vectors in $\mathbb{R}^3$ is referred to jointly as SE(3). Elements in SO(3) will be denoted as R. Elements in so(3) can be considered either as a matrix, $\hat{w}$, or as a vector, w

$$\hat{w} = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix} \tag{28}$$

$$w = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}. \tag{29}$$

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

$$\theta = cos^{-1}\left(\frac{\text{trace}(R)-1}{2}\right) \tag{30}$$

$$\log(R) = w = \frac{\theta}{2\sin(\theta)}\begin{bmatrix} R_{3,2}-R_{2,3} \\ R_{1,3}-R_{3,1} \\ R_{2,1}-R_{1,2} \end{bmatrix}. \tag{31}$$

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

$$\theta = \|w\|_2 \tag{32}$$

$$\exp(w) = R = I + \hat{w}\sin(\theta) + \hat{w}^2\left(1 - cos(\theta)\right). \tag{33}$$

It is straightforward to verify that these are exactly equivalent to the definitions for θ, $\vec{u}$ and R in the Axis-Angle representation, given $\|w\| = 1$. But here the consensus updates will be performed on the manifold.

As in [25], one version of the consensus update is calculated as

$$R_i^{(k+1)} = R_i^{(k)} \exp\left(\epsilon \sum_{j\in\mathcal{N}_i} \log\left[\left(R_i^T\right)^{(k)} \cdot R_j^{(k)}\right]\right) \tag{34}$$

but the result of these iterations is just an approximation to the Karcher mean. To calculate the true mean, we follow the method in [25] using the results of the previously shown iterations to start each camera's approximation with the same initial value that is close to the true average. The second set of iterations are

$$R_i^{(k+1)} = R_i^{(k)} \exp\left(\frac{1}{N} \sum_{j\in\mathcal{N}_i} \log\left[\left(R_i^T\right)^{(k)} \cdot R_j^{(0)}\right]\right). \tag{35}$$

The differences here are that instead of multiplying by $R_j^{(k)}$ inside the sum at each iteration, we multiply by the initial input, $R_j^{(0)}$, and instead of scaling the sum by step size ϵ, we multiply by $1/N$. 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.

3) 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, especially when the amount of error in the overall system is small, and quaternions will not be discussed further.

C. Interleaving Posit Iterations

Depending upon 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, as calculated using (3) and (4). 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. Each Posit iteration updates the rotation and translation by solving a system of linear equations, and so because Posit is a linear algorithm, inserting Posit steps maintains the linear properties of the consensus averaging.

```
Algorithm
 1) SoftPOSIT on each camera → R_i^(0), T_i^(0)
 2) Iterate for each camera:
    a) Consensus: update from neighbors
       → R_i^(k_a), T_i^(k_a)
    b) Posit: update own estimate from image
       → R_i^(k_b), T_i^(k_b)
```

Posit iterations pull individual camera solutions towards their own individual solution and away from the general consensus, so Posit iterations must be halted before the entire system can converge. Empirically it was decided that when Posit has converged to within a pixel for a single camera, this camera ceases to perform Posit steps after three more iterations.

Interleaving Posit steps within the consensus algorithm might help 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. We will see, however, that this is often not helpful.

IV. Results and Discussion

A. Network Topology

Three different network topologies were considered. Depending upon the amount of connectivity between the cameras, the system will reach consensus at different rates, and the values at convergence will differ. In traditional consensus algorithms, only the number of required iterations, not the final result, will change depending upon the topology of the system. Here, as described in Section III, we have considered variations on the traditional consensus method, and so the results we achieve are not just a pure average of the initial data, and instead vary somewhat depending upon the network topology. The first consensus method we described results in the standard consensus average results. Consensus theory tells us that λ_2, the second smallest eigenvalue of the graph Laplacian L, is the

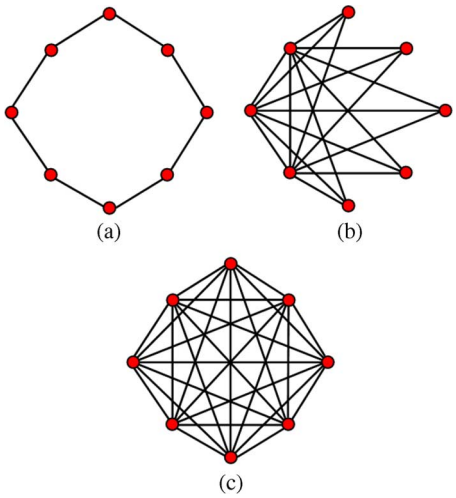


Fig. 3. Three network topologies considered. (a) Single cycle; (b) three hubs; (c) fully connected.

algebraic connectivity of the graph, and a larger λ_2 will lead to faster convergence [15]. 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. 3(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. 3(b). In the third, the network is fully connected, $\lambda_2 = 8$, Fig. 3(c).

B. Experiments on Synthetic Data

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 internally and externally calibrated, facing the object, and the object fits completely within each camera's field of view. In many applications, some points may not be visible to all cameras. In this case, it is straightforward to use only the information from those cameras able to see each point to perform consensus on that point, bypassing the other cameras, and as long as the entire network is connected and the object reference point is visible to all cameras, a consistent consensus output will result. For each camera, mean zero random noise with standard deviation $\sigma = 1, 2, 4, 8$ was added, and 50 SoftPOSIT trials were run for each case, with the output then being fed into each consensus algorithm. In the SoftPOSIT initial pose estimation, the annealing variable is updated using the relation $\beta = 1.05\,\beta$. The algorithm is initialized with a small $\beta \approx 0.0004$, and proceeds until $\beta = 0.5$, or convergence has been reached. In general, the SoftPOSIT algorithm must be restarted many times with randomly generated starting values in order to converge to a valid solution and not get caught in local minima. In this work, we require that SoftPOSIT match 80% of the model points in order to declare that a valid pose has been found. If no such pose has been found after 100 random restarts, the algorithm then reduces the number of required point matches by one and tries again, reducing the required number by one every 100 restarts until a match is found. This procedure ensures that the algorithm will terminate in a reasonable amount of time.

At first, each camera was allowed to communicate with only two neighbors, forming a single cycle network topology, as in Fig. 3(a). The consensus step size was taken to be $\epsilon = 0.65\,(1/\Delta)$, where Δ is the maximum degree of any node in the network. 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. Trials were repeated again, removing 20% of the image points from the nearer cameras at random, and adding 20% more new clutter points in the viscinity of the object. All errors are computed as the difference between the coordinates of each estimated object point at consensus, $X_m^{(\infty)}$, and the true coordinates in 3-D world space, X_m. We compare the average error and the maximum error for each class of trials

$$E_{ave} = \frac{1}{M}\sum_{m=1}^{M}\left\|\overrightarrow{X}_m^{(\infty)} - \overrightarrow{X}_m\right\|_2 \tag{36}$$

$$E_{max} = \max_m \left\|\overrightarrow{X}_m^{(\infty)} - \overrightarrow{X}_m\right\|_2. \tag{37}$$

As a baseline, we consider the average of the errors of each individual camera, before consensus. Consensus theory [15] for Euclidean spaces tells us that this will be the same as the error of the direct consensus method on world coordinates.

The results for the closer cameras are presented in Fig. 4. The virtual object in space has a bounding box of roughly $20 \times 20 \times 20$ artificial units, and the errors are reported in these same units. Similar results were obtained for the further cameras, but with larger overall errors as expected. In Fig. 5, the errors from all methods are plotted together for comparison. We see that for small image noise, all methods produce results statistically equivalent to what would be found by a centralized computer performing an average, but here this averaging is computed in a distributed fashion and seamlessly dispersed over a decentralized network. Performing consensus using penalized world coordinates produces the lowest error on average for every class of trials. For large image noise, the winning performance of the penalized consensus method is statistically significant. The methods performing consensus on rotations are statistically equivalent for low noise, especially when considering the overlap of the distributions of errors, but as image noise and clutter increases, the Karcher mean is also seen to perform well.

We see that interleaving Posit iterations is only occasionally helpful, and generally significantly increases the error in the final result. For full numerical results presented for synthetic data, see [10]. Pose estimations are updated, but the correspondences are not changed, so this addition has limited effectiveness. Only in the case when the noise in the original image was large, but no clutter points were added, did adding Posit iterations improve the accuracy of the final result. In future work, we intend to update the point correspondences in addition to the pose from each camera, and we believe that adding Posit iterations will then help to reduce error.

If communications between cameras have limited bandwidth, then it is desirable to pass as little information as possible

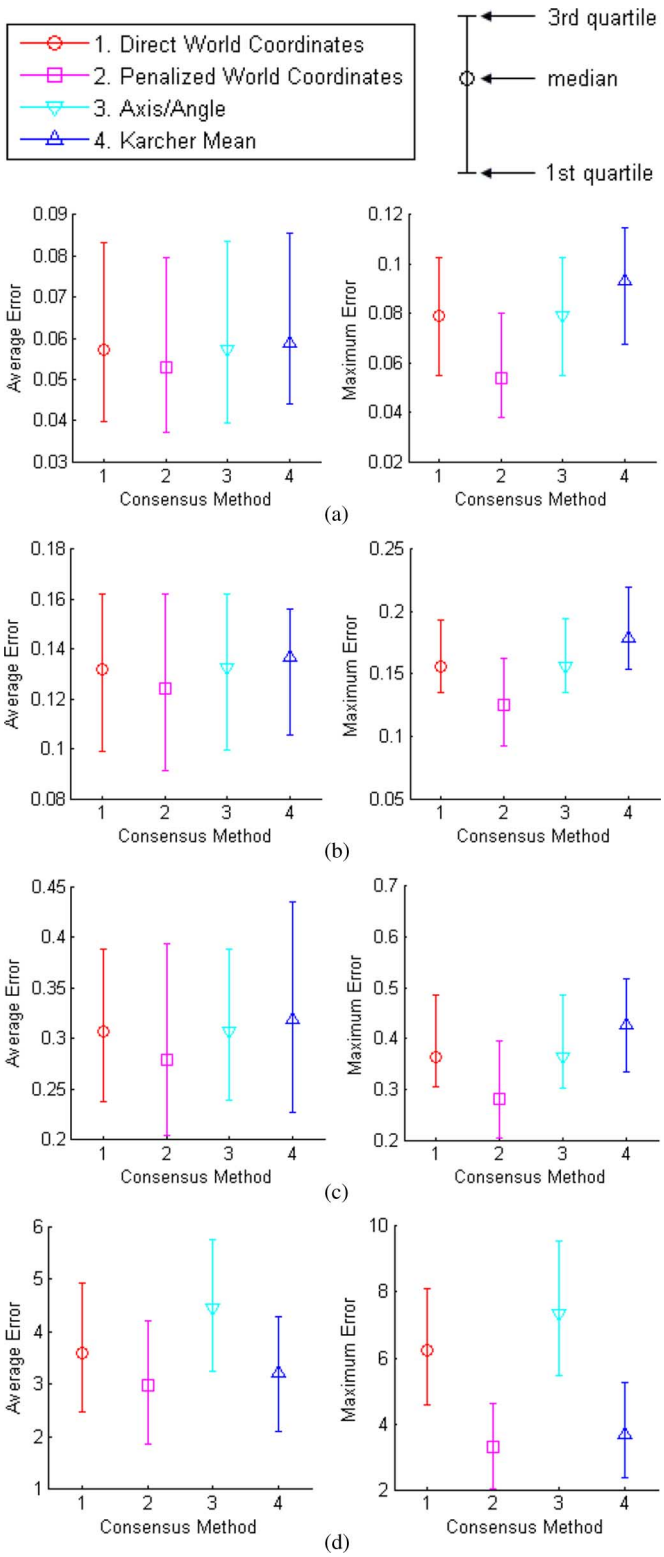


Fig. 4. Average errors (left plot) and maximum errors (right plot) in pixels are presented for each noise standard deviation, with an error bar denoting the 1st quartile, median, 3rd quartile, with cameras placed 3 to 7 times the size of the object away from the object. (a) std of image noise $= 1$; (b) std of image noise $= 2$; (c) std of image noise $= 4$; (d) std of image noise $= 8$.

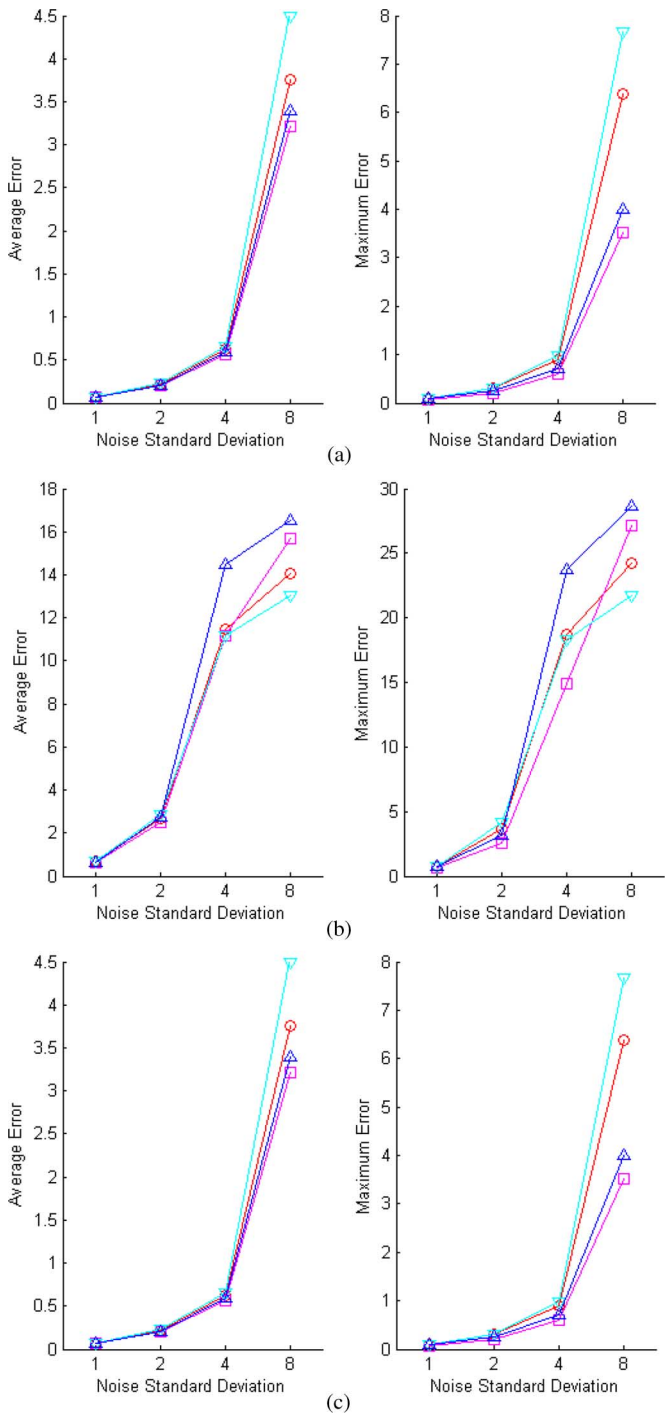


Fig. 5. Average and maximum errors from each consensus method for each noise standard deviation plotted together, with methods marked and colored as in Fig. 4. (a) Cameras placed 3 to 7 times the size of the object away from the object. (b) Cameras placed 10 to 20 times the size of the object away from the object. (c) Cameras placed 3 to 7 times the size of the object away from the object, with each point being removed with a probability of 20%, and 20% additional clutter points added to the image.

between the cameras. Sharing pose information requires that six values be exchanged (a 3-D translation vector plus three angles of rotation). Using penalized world coordinates, $3M$ values are used for computation for a model with M model points, but it is always possible to convert the current pose estimate into its corresponding rotation and translation representation before exchanging data, and then back to world coordinates to complete the next consensus step, so the communications cost is similar to that required by the methods performing consensus

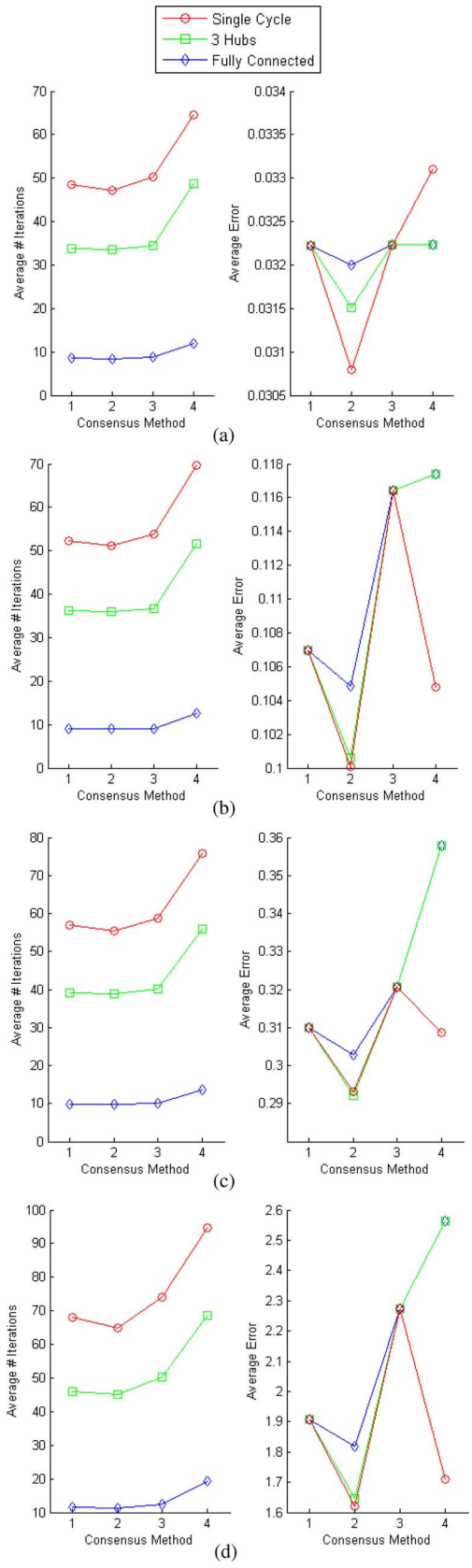


Fig. 6. For each network topology, each consensus method is plotted against the average number of iterations (left plot), and the average errors (right plot). The consensus methods are numbered along the x-axis as defined in Fig. 4. (a) std of image noise $= 1$; (b) std of image noise $= 2$; (c) std of image noise $= 4$; (d) std of image noise $= 8$.

TABLE I
AVERAGE NUMBER OF ITERATIONS AND TIMES TO CONVERGENCE FOR EACH METHOD WITH NO POSIT STEPS, AS RUN IN MATLAB ON A DESKTOP PC

	World Coords	*Penalized WCs*	*Axis-Angle*	*Karcher Mean*
# Iterations	56.43	54.54	59.23	76.20
Total time	0.40 sec	0.93 sec	0.30 sec	0.22 sec

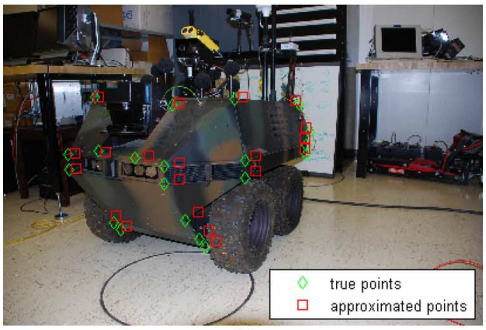
over rotations. Each such conversion requires only a few simple linear algebra computations, but using either of the methods that perform consensus over rotations, these extra conversion steps are not necessary. In this case, using the Karcher mean method might be preferable, but the method of penalized world coordinates most closely follows the spirit of consensus by applying it to Euclidean quantity and replacing the rotation constraints or approximations with a rigidity penalty.

When comparing the three topologies from Fig. 3, 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. As discussed in Section IV-A, performing consensus directly in world coordinates results in the average of the input data, which is the standard consensus result, but the other methods are variations on the traditional algorithm which do not perform consensus in pure Euclidean space, and this is why the number of iterations and the final error both change depending upon the network topology. See Fig. 6 for results. If the network topology does not form a cycle, the second set of iterations in the Karcher mean method (35) will not converge. In this case, the approximate solution provided by the first set of iterations (34) must be taken as the final result, with some loss of accuracy expected.

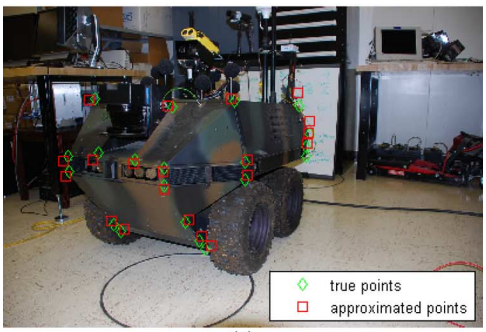
Convergence is guaranteed for the traditional consensus algorithm [15], and so all the methods presented here will return a solution. However, the accuracy of the output is dependent upon the overall accuracy of the input. If the initial point detections are poor, then SoftPOSIT might take a very long time to arrive at an initial prediction of the pose, and this prediction might be very erroneous. This in turn will affect the consensus output of the entire system. Our algorithm is affected by the same limitations as the original SoftPOSIT algorithm. The use of robust estimation would help minimize these effects. A primary benefit of using consensus for pose estimation is that it works for ad-hoc networks no matter how the network topology changes through time. Each node exchanges information with whoever its neighbors happen to be for each complete consensus calculation, the only requirement is that the network be connected.

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 I. Performing consensus over rotations is faster than over world coordinates, because calculations are only performed on

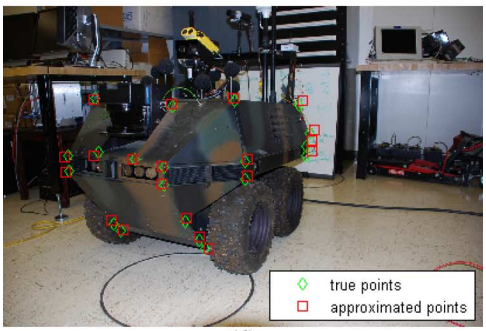
(a)



(b)



(c)



(d)

Fig. 7. Example robot image, and the estimated point locations at different stages of the method. Red points are the estimated locations and green points are the true locations. (a) Input image. (b) SoftPOSIT output before consensus. (c) After consensus using penalized world coordinates. (d) After consensus using the Karcher mean.

six data points at each step, and because no extra conversions are required to exchange these six values.

C. Experiments on Real Data

Pictures of a robot were taken from eight distinct views by an uncalibrated camera, for example see Fig. 7(a). The internal

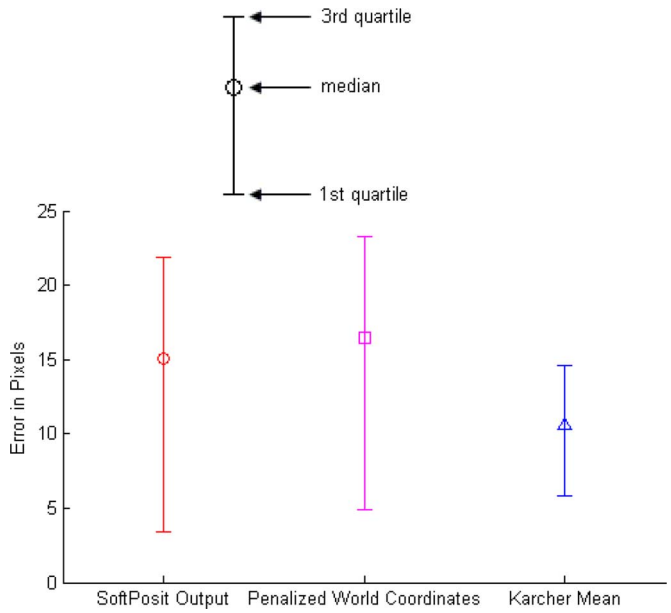


Fig. 8. Average error over all points in pixels. Results are calculated on the SoftPOSIT output before any consensus method is performed (red, left), on consensus by penalized world coordinates (magenta, center), and on consensus by the Karcher mean (blue, right). The error bars denote the 1st quartile, median, and 3rd quartile.

camera calibration and camera locations were computed using the knowledge of the size of the floor tiles. The robot model consisted of 36 corner points, and was obtained from a VRML description of the robot. Between 17 and 23 model points were visible in each image, and these points were identified by hand. The points were selected manually in order to validly demonstrate the abilities of the algorithm presented; in practice an automatic feature detector would be used. This collection of 2-D points was made 50% larger by the addition of uniformly distributed clutter points centered at the center of the image points, with a possible range of 150% of the range of the image points. The initial poses provided as starting conditions to the SoftPOSIT algorithm were random but known to be within 45° of the true rotation in each of the x-, y-, and z-axes, and between ± 2 times the size of the true translation in each dimension. The algorithm was restarted 10 times, using 10 different sets of random clutter and starting conditions. The error at each point was calculated as the distance in pixels from the true point location in the image. For comparison, the average of all point errors was calculated on the output from SoftPOSIT before any consensus methods were performed. This error was then calculated on the output from the two most promising consensus methods: using penalized world coordinates and using the Karcher mean. These results are presented in Fig. 8. It is seen that by only sharing information with their immediate neighbors, all cameras were able to agree on a single pose whose error was similar in size to the overall input error in the system, which was unknown to the cameras. This agrees with the results we saw previously, that the variations on the consensus method presented here generally result in statistically equivalent error for small image noise, and in noticeably smaller error for larger image noise. Complete results from one of the 8 robot images is presented in Fig. 7. The average number of iterations and time to consensus is seen in Table II.

TABLE II
AVERAGE NUMBER OF ITERATIONS AND TIMES TO CONVERGENCE FOR EACH METHOD PERFORMED ON REAL DATA, AS RUN IN MATLAB ON A DESKTOP PC

	Penalized World Coordinates	*Karcher Mean*
# Iterations	79.30	104.30
Total time	0.70 sec	0.18 sec

V. CONCLUSION

We have presented a set of algorithms for determining the pose of a known object as seen from several cameras in a decentralized network. Performing consensus over world coordinates penalized by an object model, and performing consensus in SE(3) using the Karcher mean were found to result in the lowest overall error. Comparing errors in a synthetic 3-D world domain, the method of penalized world coordinates was seen to perform the best overall, and actual experiments show similar errors computed in the image domain. The penalized world coordinates method is a simpler algorithm allowing consensus to be performed in the traditional Euclidean space of 3-D point coordinates, but requires performing computations on three values for every model point at every iteration, and requires an additional step to convert the world coordinates to and from the compact pose representation requiring only six values in order to minimize the bandwidth required to exchange data. Performing consensus using the Karcher mean results in error 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. When image noise is very high, we showed that it was helpful to interleave Posit and consensus iterations. The number of iterations required for convergence in all cases can be decreased by increasing the connectivity of the camera network. The amount of computation is small enough to make this method practical in real-world situations. The algorithms presented here return a final object pose that has less average error than the average of the input pose predictions, and arrive at this result without requiring a centralized computer capable of communicating with all cameras simultaneously.

ACKNOWLEDGMENT

The authors would like to thank R. Tron, Prof. R. Vidal and Prof. A. Terzis (JHU) for kindly making their code available for the algorithm described in [25]. They would also like to thank Dr. D. Lucarelli (JHU/APL) for his helpful input, and Dr. P. David (ARL) for providing the pictures and VRML model of the robot used in the experiments.

REFERENCES

[1] R. Carli, A. Chiuso, L. Schenato, and S. Zampieri, "Distributed Kalman filtering based on consensus strategies," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 4, May 2008.

[2] P. David, D. DeMenthon, R. Duraiswami, and H. Samet, "SoftPOSIT: Simultaneous pose and correspondence determination," *Int. J. Comput. Vis.*, vol. 59, no. 3, pp. 259–284, Sep.-Oct. 2004.

[3] D. DeMenthon and L. S. Davis, "Model-based object pose in 25 lines of code," *Int. J. Comput. Vis.*, vol. 15, pp. 123–141, Jun. 1995.

[4] D. Devarajan and R. Radke, "Calibrating distributed camera networks using belief propagation," *EURASIP J. Appl. Signal Process.*, p. 221, 2007.

[5] D. Devarajan, R. Radke, and H. Chung, "Distributed Metric Calibration of Ad-Hoc Camera Networks," *ACM Trans. Sensor Netw.*, 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," in *Proc. 3rd Int. Conf. Intelligent Sensors, Sensor Networks and Information Processing*, 2007, pp. 37–42.

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

[8] C. Gramkow, "On averaging rotations," *J. Math. Imag. Vis.*, vol. 15, pp. 7–16, 2001.

[9] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[10] A. Jorstad, P. Burlina, I.-J. Wang, D. DeMenthon, and D. Lucarelli, "Model-based pose estimation by consensus," in *Proc. Int. Conf. Intelligent Sensors, Sensor Networks and Information Processing*, 2008, pp. 569–574.

[11] H. Karcher, "Riemannian center of mass and mollifier smoothing," *Commun. Pure Appl. Math.*, vol. 30, pp. 509–541, 1977.

[12] D. Lowe, "Object recognition from local scale invariant features," in *Proc. Int. Conf. Computer Vision*, Sep. 1999, vol. 2, p. 1150.

[13] Y. Ma, J. Košecká, and S. Sastry, "Optimization criteria and geometric algorithms for motion and structure estimation," *Int. J. Comput. Vis.*, vol. 44, no. 3, pp. 219–249, Sep. 2001.

[14] M. Moakher, "Means and averaging in the group of rotations," *SIAM J. Matrix Anal. Appl.*, vol. 24, no. 1, pp. 1–16, 2002.

[15] R. Olfati-Saber, J. A. Fax, and R. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[16] R. Olfati-Saber, "Distributed Kalman filter with embedded consensus filters," in *Proc. IEEE Conf. Decision and Control*, 2005, pp. 8179–8184.

[17] R. Olfati-Saber, "Swarms on sphere: A programmable swarm with synchronous behaviors like oscillator networks," in *Proc. IEEE Conf. Decision and Control*, 2006, pp. 5060–5066.

[18] X. Pennec and N. Ayache, "Uniform distribution, distance and expectation problems for geometric features processing," *J. Math. Imag. Vis.*, vol. 9, pp. 49–67, 1998.

[19] M. Rabbat and R. Nowak, "Distributed Optimization in sensor networks," in *Proc. 3rd Int. Symp. Information Processing in Sensor Networks*, Apr. 2004, pp. 20–27.

[20] M. Rabbat, "On spatial gossip algorithms for average consensus," in *Proc. IEEE/SP 14th Workshop on Statistical Signal Processing*, 2007, pp. 705–709.

[21] W. Ren, R. Beard, and E. Atkins, "Information consensus in multivehicle cooperative control," *IEEE Control Syst. Mag.*, vol. 27, no. 2, pp. 71–82, Apr. 2007.

[22] A. Sarlette and R. Sepulchre, "Consensus optimization on manifolds," *SIAM J. Control Optim.*, pp. 56–76, 2009.

[23] R. Sinkhorn, "A relationship between arbitrary positive matrices and doubly stochastic matrices," *Ann. Math. Statist.*, vol. 35, pp. 876–879, Jun. 1964.

[24] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Trans. Autom. Control*, vol. AC–31, no. 9, pp. 803–812, Sep. 1986.

[25] R. Tron, R. Vidal, and A. Terzis, "Distributed pose averaging in camera networks via consensus on SE(3)," in *Proc. IEEE Int. Conf. Distributed Smart Cameras*, Sep. 2008, pp. 1–10.

[26] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 4, Apr. 1991.

[27] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proc. 4th Int. Symp. Information Processing in Sensor Networks*, Apr. 2005, pp. 63–70.

Anne Jorstad (S'09) received the B.A. degree in mathematics from Cornell University, Ithaca, NY, in 2005, the M.A. degree in mathematics from The University of Wisconsin-Madison in 2007, and is currently pursuing the Ph.D. degree at the University of Maryland-College Park under the supervision of Professor David Jacobs.

She has worked with the Mathematics and Computing Technology division of The Boeing Company and with the Johns Hopkins University Applied Physics Laboratory Milton S. Eisenhower Research Center. Her research interests include shape representation, comparison and optimization in computer vision and computational geometry.

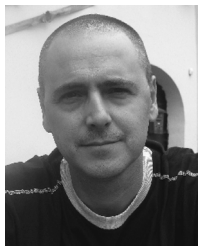
Daniel DeMenthon (SM'10) graduated from Ecole Centrale de Lyon, France, in 1972, received the M.S. degree in applied mathematics from Université Claude Bernard, Lyon, France, in 1973, the M.S. degree in naval architecture and offshore engineering from the University of California, Berkeley, in 1980, and the Ph.D. degree in computer science from Université Joseph Fourier, Grenoble, France in 1993.

He is currently a Senior Research Scientist at the Johns Hopkins University's Applied Physics Laboratory, Baltimore, MD. From 1985 to 2008, he was a Research Engineer, then an Associate Research Professor at the Institute for Advanced Computer Studies at the University of Maryland, College Park. From 2005 to 2008, he served as Program Director for computer vision research at the National Science Foundation. He is known for his work in pose estimation and video analysis. He has published over 80 papers in journals and conferences, and was awarded six U.S. patents and two European patents. His research interests include object recognition and real time computer vision.

I-Jeng Wang (S'89–M'91) received the M.S. degree in electrical engineering from the Pennsylvania State University, University Park, in 1991, and the Ph.D. degree in electrical engineering from Purdue University, West Lafayette, IN, in 1996.

He had a Postdoctoral position with the Institute for Systems Research at University of Maryland, College Park from 1996 to 1997. Since October 1997, he has been with the Johns Hopkins Applied Physics Laboratory, Baltimore, MD, where he is a Senior Research Scientist with the Milton S. Eisenhower Research Center. He is also an Assistant Research professor at the Johns Hopkins University Department of Computer Science. His current research interests include stochastic control and optimization, graphical models and Bayesian learning, and distributed information processing in sensor networks.

Dr. Wang served on the Editorial Board of the IEEE TRANSACTIONS ON AUTOMATIC CONTROL from 2006 to 2009. He is a member of ACM.



Philippe Burlina (S'92–M'93–SM'08) received the engineering degree from Université de Technologie de Compiègne, Compiègne, France, in 1988 and the M.S. and Ph.D. degrees in electrical engineering from University of Maryland, College Park, in 1991 and 1994, respectively.

He is currently an Assistant Research Professor at the Johns Hopkins University Department of Computer Science, Baltimore, MD, and heads the Machine Vision Section in Johns Hopkins Applied Physics Laboratory Milton S. Eisenhower Research Center. Prior to that, he held various industry R&D positions including, most recently, as a Director of Software Development at FileNet (now part of IBM). His interests include computer vision, hyperspectral video processing, medical image analysis, distributed visual sensors, machine learning, and enterprise software development.