DISTRIBUTED POSE AVERAGING IN CAMERA NETWORKS VIA CONSENSUS ON SE(3)

Roberto Tron, René Vidal

Center for Imaging Science
Johns Hopkins University
Baltimore MD 21218, USA

Andreas Terzis

Computer Science Department
Johns Hopkins University
Baltimore MD 21218, USA

ABSTRACT

In this paper, we propose distributed algorithms for estimating the average pose of an object viewed by a localized network of camera motes. To this effect, we propose distributed averaging consensus algorithms on the group of 3-D rigid-body transformations, $SE(3)$. We rigorously analyze the convergence of the proposed algorithms, and show that naive generalizations of Euclidean consensus algorithms fail to converge to the correct solution. We also provide synthetic experiments that confirm our analysis and validate our approach.

Index Terms— camera sensor networks; pose estimation; consensus; optimization on manifolds.

1. INTRODUCTION

Recent hardware innovations have produced low-power, embedded computers (i.e., motes), equipped with small cameras that can self-organize into wireless mesh networks. These disruptive technologies provide the opportunity to devise compelling new applications at the intersection of sensor networks and computer vision. In particular, one can rethink many of the classical computer vision applications such as structure from motion (SfM), target tracking and object recognition, in the context of scenes observed by a large number of cameras.

A number of fundamental challenges must be addressed for this promise to be fulfilled. Specifically, most computer vision algorithms assume that all the data (i.e., the images) are available on a single computer where centralized processing is possible. However, this paradigm is inherently incompatible with sensor networks for two reasons. First, it requires the transmission of large volumes of raw data. Second, it demands processing resources not available in mote-class devices.

One is thereby naturally drawn to distributed algorithms, in which motes analyze the raw data locally and collaborate to reach a shared, global analysis of the scene. This approach is intuitively attractive, because it affords nodes to exchange only distilled information that is relevant to the collaboration. Because such information can be expressed in compact form for most applications –the pose of an object, for example, lives in a six-dimensional manifold– this approach can result in radical energy savings.

Prior work. Unfortunately, existing distributed sensing algorithms are not suited to our class of problems. For example, *distributed consensus algorithms*[1] that have been used extensively in sensor networks (see [1, 2, 3, 4, 5] and the references therein) operate on low-dimensional measurements (e.g., temperature) which lie in a Euclidean space. On the other hand, most of the quantities involved in the visual representation of a scene are not directly measurable (i.e., they do not correspond to the images that the nodes see) and belong to spaces with rich and complex non-Euclidean structures (e.g., Lie groups). Consequently, a principled treatment of such data requires the use of optimization on manifolds. While optimization on manifolds has been widely used in computer vision problems such as 3D reconstruction [6], 3D motion segmentation [7], and manifold clustering [8], relatively less work has been done in extending consensus algorithms to non-Euclidean data [9].

Multiple papers address the localization of camera networks [10, 11, 12, 13, 14, 15, 16]. Semi-automatic calibration methods use laser printed textures mounted on a board [10] or modulated LED emissions [11, 12]. Automatic methods perform local SfM via robust bundle adjustment [15], and integrate the information using belief propagation [16].

Paper contributions. We posit that a natural framework for obtaining a consistent estimate of the pose of an object is by using consensus. The key question is how to design a local SfM algorithm that (1) replaces the local average algorithm in the classical consensus problem and (2) converges to a consistent global estimate of the 3D pose when iteratively applied. To the best of our knowledge, there is no prior work on fully automatic and distributed object pose estimation using extensions of consensus algorithms. Also, there are no transcriptions of SfM constraints into distributed averaging.

This paper makes the following three contributions. First, we demonstrate that generalizing existing consensus algorithms from the Euclidean to the non-Euclidean setting is non-trivial. In fact, we show that naive extensions fail to converge to the correct solution. Second, we present distributed algorithms for solving a simple yet foundational computer vision problem: estimating the pose of an object viewed by

[1] Consensus algorithms are also known as information consensus, consensus averaging, agreement protocols, etc.

978-1-4244-2665-2/08/$25.00

all the cameras in a network. More specifically, we devise distributed algorithms for performing consensus on the group of 3-D rigid motions in order to average the pose of the object in question. Our algorithms make use of the geodesic distance in $SE(3)$, in contrast to the Euclidean distance adopted by traditional consensus approaches. As a result, our contribution is precisely to propose algorithms for estimating the average pose as the Karcher mean on this manifold in a distributed fashion. These algorithms lay the foundation necessary for future directions of research in this area. Finally, we evaluate the convergence and performances of the proposed algorithms both analytically and experimentally using simulations.

Why optimization on manifolds? As we will show later in the paper, averaging the pose of an object requires computing averages of rotations. To solve this problem, besides employing optimization on manifolds, other techniques have been previously used in the literature. Popular approaches are:

- *Normalized quaternions*: all the rotations are converted to quaternions which are subsequently averaged by considering them as Euclidean vectors. The resulting quaternion is then normalized and the corresponding rotation is treated as the desired average.
- *Euclidean average and projection*: the rotations are treated as 3×3 matrices and averaged in the Euclidean sense. Since, in general, the resulting matrix is not a proper rotation, an additional projection step is required.

As noted in [17], these approaches use approximations of the geodesic distance, which result in efficient ways for computing approximations of the geometric mean of rotations. However, applying these methods in a distributed setting is not straightforward. In the case of quaternions, this is because of a well-known ambiguity in which two quaternions q and $-q$ represent the same rotation. A centralized algorithm usually overcomes this difficulty by choosing one of the quaternions as a reference and then changing the signs of the others such that their scalar product with the reference is positive. If and how this may be done in a distributed fashion (i.e., without choosing a central node as reference) is an open question. On the other hand, using the projection of the Euclidean average of the rotation matrices does not take into account the curvature of the manifold. Hence, the convergence of distributed averaging to the desired mean is not guaranteed, as we will show in §3.1.

Paper outline. The rest of the paper is organized as follows. §2 presents a review of traditional consensus algorithms that deal with Euclidean data. It also reviews the basics of differential geometry with a focus on optimization on manifolds. Having introduced the necessary mathematical framework, in §3 we propose generalizations of classical consensus approaches to $SE(3)$ and study their convergence properties. §4 extends these algorithms to object pose estimation in a camera sensor network. Finally, §5 presents results on synthetic data that validate the different aspects of our proposed framework.

2. MATHEMATICAL BACKGROUND

In this section, we review some basic concepts related to Euclidean consensus algorithms and differential geometry that are relevant to our algorithms.

2.1. Review of Euclidean consensus algorithms

Consider a sensor network with N nodes. The communication in the sensor network can then be represented using an undirected graph $G = (V, E)$. The set $V = \{v_1, \ldots, v_N\}$ contains the vertices of the graph, where $v_i \in V$ represents the i-th camera sensor. The set $E \subset V \times V$ contains the edges of the graph and describes the communication links between these nodes. More specifically, if nodes i and j can directly communicate with each other, then the pair $\{i, j\} \in E$. Given a graph G, we define its *adjacency matrix* $A \in \mathbb{R}^{|V|\times|V|}$ as

$$[A]_{ij} = \begin{cases} 1 & \text{if } \{i,j\} \in E \\ 0 & \text{otherwise} \end{cases}. \tag{2.1}$$

Note that A is symmetric since G is undirected. We also define the *degree* of a node v_i as $\deg(v_i) = \sum_j [A]_{ij}$, i.e., the number of edges in which v_i appears. The *maximum degree* of G is defined as $\Delta_G = \max_i \deg(v_i)$.

In the Euclidean setting, we associate with each node $i = 1, \ldots, N$, a scalar *state* $x_i \in \mathbb{R}$. We then say that the nodes have reached *consensus* when the following condition holds

$$x_1 = x_2 = \ldots = x_N = \alpha. \tag{2.2}$$

The value α is called the *collective decision* of the network.

A *consensus algorithm* defines a *protocol*, i.e., a rule that each node has to follow to update its state so that all the nodes reach consensus. To this effect, the task of a *distributed consensus algorithm* is to define a protocol that uses only interactions between neighboring nodes.

A popular discrete time protocol for iteratively computing the mean of the initial states $\alpha = \frac{1}{n}\sum_{i=1}^{N} x_i$ is given by

$$x_i^{(k+1)} = x_i^{(k)} + \varepsilon \sum_{j \in N_i} a_{ij}(x_j^{(k)} - x_i^{(k)}), \tag{2.3}$$

where N_i is the set of neighbors of the i-th node, $x_i^{(k)}$ is the state of the i-th node at the k-th iteration, and $\varepsilon \leq \frac{1}{\Delta_G}$ is the learning rate. Note that the sum over N_i can be safely extended to $\{1, \ldots, N\}$, because $a_{ij} = 0$ if i and j are not neighbors. Eq. (2.3) can then be rewritten in terms of the updates $u_i^{(k)} = \varepsilon \sum_{j \in N_i} a_{ij}(x_j^{(k)} - x_i^{(k)})$, as

$$x_i^{(k+1)} = x_i^{(k)} + u_i^{(k)}. \tag{2.4}$$

It is easy to see that (2.3) is in fact a gradient descent algorithm that minimizes the function

$$\varphi(x_1, x_2, \ldots, x_N) = \frac{1}{2} \sum_{\{i,j\} \in E} a_{ij}(x_i - x_j)^2. \tag{2.5}$$

Also, it is easy to verify that the mean of the states is preserved at each iteration, i.e.,

$$\sum_{i=1}^{N} x_i^{(k)} = \sum_{i=1}^{N} x_i^{(k+1)} = N\alpha. \tag{2.6}$$

Equivalently, the sum of the updates over all the nodes is zero:

$$\sum_{i=1}^{N} u_i^{(k)} = 0. \tag{2.7}$$

When the graph G is connected, the minimum of (2.5) is achieved when the nodes reach a consensus, i.e., when all the states are equal to the average of the initial states. More information on this protocol can be found in [4]. Note that this algorithm can be readily extended to the case $x_i \in \mathbb{R}^n$ by applying it to each coordinate separately.

2.2. Review of Riemannian geometry

Given a smooth manifold $\mathcal{M}$, we define a smooth curve in the manifold as a smooth function $\gamma(t) : \mathbb{R} \to \mathcal{M}$. The *tangent space* of $\mathcal{M}$ at a point $x \in \mathcal{M}$, denoted as $T_x\mathcal{M}$, is then defined as the span of the tangent vectors for all the possible curves γ passing through x. A *Riemannian metric* is a continuous collection of dot products $\langle \cdot | \cdot \rangle_x$. Using this metric, we define the length of a curve between two points $x, y \in \mathcal{M}$ as

$$\mathcal{L}_a^b(\gamma) = \int_a^b \langle \dot\gamma(t) | \dot\gamma(t) \rangle_{\gamma(t)}^{\frac{1}{2}} dt, \tag{2.8}$$

where $\gamma(a) = x$ and $\gamma(b) = y$. A curve between x and y with minimum length is called a $geodesic$. The distance between two points in the manifold is subsequently defined as the length of the geodesic curve between them.

$$d(x, y) = \mathcal{L}_0^1(\gamma),\ \gamma(0) = x,\ \gamma(1) = y. \tag{2.9}$$

We can then define the *exponential map* $\exp_x(v) : T_x\mathcal{M} \to \mathcal{M}$, which maps each tangent vector $v \in T_x\mathcal{M}$ to the point in $\mathcal{M}$ obtained by following the geodesic passing through x with direction $\frac{v}{\|v\|_x}$ for a distance $\|v\|_x = \langle v, v \rangle_x$. The exponential map is a diffeomorphism between a sufficiently small neighborhood of 0 in $T_x\mathcal{M}$ and a neighborhood of x in $\mathcal{M}$. The *logarithm map* is the inverse of the exponential map and is denoted as $\log_x = \exp_x^{-1}$. Note that if $y = \exp_x(v)$ then

$$\|\log_x(y)\|_x = \|v\|_x = d(x, y). \tag{2.10}$$

2.3. Lie groups

A Lie group G is an algebraic group which can also be considered as a differentiable Riemannian manifold. In particular, the group is characterized by a unique identity element $e \in G$ and two group operations

$$\begin{aligned} \text{multiplication} \quad & g_1 g_2 : G \times G \to G, \text{ and} \\ \text{inversion} \quad & g^{-1} : G \to G, \end{aligned}$$

which are differentiable mappings. The tangent space at the identity is called the Lie algebra of the group. We denote as $\exp(\cdot)$ and $\log(\cdot)$, respectively, the exponential map and the logarithm map at the identity e. These mappings at a generic point $X \in G$ can be computed using *parallel transport* as:

$$\exp_X(A) = X \exp(X^{-1}A), \tag{2.11}$$

$$\log_X(B) = X \log(X^{-1}B), \tag{2.12}$$

with $A \in T_X G$ and $B \in G$.

In this paper, we are particularly interested in the special orthogonal group $SO(3) = \{R \in \mathbb{R}^{3\times 3} : R^\top R = I, \det(R) = 1\}$. This is the group of orthogonal 3×3 matrices having a determinant equal to one, and essentially describes all possible 3-D rotations. The Lie algebra for this group is $so(3)$, the space of 3×3 skew-symmetric matrices. In this case, the exponential map at the identity $\exp(\hat w)$, $\hat w \in so(3)$, can be defined using the well-known Rodrigues' formula [18], as

$$\exp(\hat w) = I + \frac{\hat w}{\|w\|}\sin(\|w\|) + \frac{\hat w^2}{\|w\|^2}\big(1 - \cos(\|w\|)\big), \tag{2.13}$$

where $\hat w \in so(3)$ is the matrix generating the cross product by $w \in \mathbb{R}^3$, i.e., $\hat w v = w \times u$ for all $u \in \mathbb{R}^3$. One can then derive the logarithm map at the identity as

$$\theta = \arccos\left(\frac{\mathrm{tr}(R) - 1}{2}\right), \tag{2.14}$$

$$\log(R) = \begin{cases} \dfrac{1}{2\sin\theta}(R - R^\top) & \text{if } \theta \neq 0 \\ 0 & \text{if } \theta = 0 \end{cases}, \tag{2.15}$$

where $\mathrm{tr}(R)$ is the trace of the matrix. It can then be shown that the distance in $SO(3)$ is given by

$$d^2(R, Q) = -\frac{1}{2}\,\mathrm{tr}\{[\log(R^\top Q)]^2\} \quad R, Q \in SO(3). \tag{2.16}$$

Let $v \in T_R SO(3)$ be a vector in the tangent space of $SO(3)$ at R and let $\gamma(t) = R\exp(tv)$ be the geodesic passing through R and having v as its tangent. The *covariant derivative* of a function $f : SO(3) \to \mathbb{R}$ at $R \in SO(3)$ (denoted as $\nabla_R f \in so(3)$) is the unique vector defined as

$$\mathrm{tr}(v^\top \nabla_R f) = \left.\frac{d}{dt} f(\gamma(t))\right|_{t=0}. \tag{2.17}$$

Note that the covariant derivative of a function on the manifold can be treated as the equivalent of the conventional derivative of a function defined on a Euclidean manifold. Therefore, it can be used for performing optimization on $\mathcal{M}$ using algorithms such as gradient descent. In particular, we will use the covariant derivative of the squared distance with respect to one of the arguments. This covariant derivative can be explicitly evaluated as (see [19] for a proof)

$$\nabla_R d^2(R, Q) = -R\log(R^\top Q). \tag{2.18}$$

2.4. Karcher mean

Let $\{x_i\}_{i=1}^N$ be a set of points in a smooth Riemannian manifold $\mathcal{M}$. The Riemannian metric can then be used to obtain the geometric distance $d(x, y)$ between any two points $x, y \in \mathcal{M}$. The Karcher mean of this set is defined as the point $\bar{x} \in \mathcal{M}$ for which the sum of squared distances

$$\sum_{i=1}^{N} d^2(x_i, \bar{x}) \tag{2.19}$$

is minimized. It can be shown (see [19]) that a necessary and sufficient condition for $\bar{x}$ to be the Karcher mean is that

$$\sum_{i=1}^{N} \log_{\bar{x}}(x_i) = 0. \tag{2.20}$$

This condition leads to the following iterative algorithm for computing the Karcher mean (see [20, 21, 22] for details)

1. Set initial mean in $\mathcal{M}$ as $\bar{x} = x_1$.
2. Compute the mean in $T_{\bar{x}}\mathcal{M}$ as $w = \frac{1}{N}\sum_{i=1}^{N} \log_{\bar{x}}(x_i)$.
3. While $\|w\| < \delta$, update $\bar{x}$ to $\bar{x} = \exp_{\bar{x}}(\varepsilon w)$, for $\varepsilon \leq 1$ and go to step 2.

Conditions for the convergence of this algorithm in the case of $SO(3)$ are given in [21]. Essentially, a sufficient condition for the Karcher mean to exist and be unique is that the N rotations are "close enough" to each other.

3. DISTRIBUTED OBJECT POSE ESTIMATION WITH RESPECT TO A FIXED REFERENCE

In what follows, we consider an object in a scene that is assumed to be visible to all the N cameras in the network. Furthermore, we assume that each of these $i = 1, \ldots, N$ nodes can individually estimate the pose of the object relative to a common reference frame as a pair of rotation and translation $g_i = (R_i, T_i) \in SE(3) = \{(R, T) : R \in SO(3), T \in \mathbb{R}^3\}$. The case where each camera estimates the pose of the object with respect to its own reference frame will be considered in the next section. Our goal then is to estimate an average pose $\bar{g}$ from all the measurements in a distributed fashion, i.e., by exchanging data only between neighboring pairs of nodes. We propose to do so by using a consensus algorithm on $SE(3)$.

Since any pose can be represented with a rotation and a translation, $SE(3)$ is homeomorphic to $SO(3) \times \mathbb{R}^3$. In such a scenario, one can use a standard result that when the data lie on a manifold which is the direct product of two spaces, one can separately consider the metrics and covariant derivatives associated with the two spaces [23]. Therefore, we perform our optimization on the individual spaces $SO(3)$ and $\mathbb{R}^3$. Note that in the case of $\mathbb{R}^3$, we can employ traditional Euclidean consensus algorithms as discussed in §3. Hence, we focus our attention on consensus algorithms for $SO(3)$, i.e., the rotation component of the pose of the object, using its Riemannian metric.

3.1. Consensus in the manifold ($SO(3)$)

Our first algorithm, which we refer to as *consensus in the manifold*, can be considered as a direct extension of the classic consensus algorithm to the non-Euclidean case. Essentially, we follow a Riemannian gradient descent scheme for minimizing the cost function in (3.1), as opposed to the traditional gradient descent method. In particular, we substitute the derivatives of the corresponding cost function in the Euclidean space (2.5), with covariant derivatives on the manifold. Consequently, the update step is modified so as to move along the geodesics defined by the covariant derivatives of the objective function. In this manner, we ensure that the solution estimated by the optimization scheme always lies on the required manifold.

Let us denote the measurement of the object's rotation from the i-th node as $R_i \in SO(3)$. We then define the potential function φ that is to be minimized in order to estimate the mean rotation, as

$$\varphi(R_1, \ldots, R_N) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij} d^2(R_i, R_j), \tag{3.1}$$

where $d(\cdot, \cdot)$ is the distance between two elements of $SO(3)$. The covariant derivative of φ with respect to the k-th rotation, can then be explicitly calculated as

$$\nabla_{R_k}\varphi = \nabla_{R_k}\sum_{i\neq k} a_{ik} d^2(R_i, R_k) = -\sum_{i=1}^{N} a_{ik} R_k \log(R_k^\top R_i). \tag{3.2}$$

In the simplification of the above equation, we used the facts that $a_{ij} = a_{ji}$, $d(\cdot, \cdot)$ is symmetric and $d(R_k, R_k) = 0$.

Our proposed consensus protocol on $SO(3)$ corresponds to using a Riemannian gradient descent search for minimizing the cost function φ. In what follows, we use $R_k^{(l)}$ to denote the estimate of R_k at the l-th iteration of our algorithm. Essentially, $R_k^{(l)}$ is updated along the geodesic corresponding to the covariant derivative direction $-\nabla_{R_k^{(l)}}\varphi$ with a step size ε, as

$$\begin{aligned} R_k^{(l+1)} &= \exp_{R_k^{(l)}}(-\varepsilon \nabla_{R_k^{(l)}}\varphi) \\ &= R_k^{(l)} \exp\left({R_k^{(l)}}^\top R_k^{(l)} \varepsilon \sum_{i=1}^{N} a_{ik} \log({R_k^{(l)}}^\top R_i^{(l)}\right) \\ &= R_k^{(l)} \exp\left(\varepsilon \sum_{i=1}^{N} a_{ik} \log\left({R_k^{(l)}}^\top R_i^{(l)}\right)\right). \end{aligned} \tag{3.3}$$

This update for $R_k^{(l)}$ can be rewritten more compactly as

$$R_k^{(l+1)} = R_k^{(l)} U_k^{(l)}, \tag{3.4}$$

where the update $U_k^{(l)}$ is defined as

$$U_k^{(l)} = \exp(\varepsilon \sum_{i=1}^{N} a_{ik} \log({R_k^{(l)}}^\top R_i^{(l)}). \tag{3.5}$$

The following proposition states the convergence properties of the consensus in the manifold protocol.

Proposition 1. *The consensus in the manifold protocol* (3.3) *converges to a local minimum of* φ.

Proof. Notice that all the terms in (3.1) are non-negative, i.e., $a_{ij} \geq 0$ and $d^2(R_i, R_j) \geq 0$. Therefore, the cost function φ is non-negative for all $R_1, \ldots, R_N \in SO(3)$ and bounded from below. Since our algorithm descends the potential function φ at each iteration, it is guaranteed to converge to a local minimum of the function φ. □

Although this proposition only guarantees convergence to a local minimum, in our experiments we have noticed that our protocol always converges to a global minimizer of φ. Our ongoing research aims at investigating whether the proposed protocol can converge to any of the potential local minimizers, and if so, under what conditions this happens. Nevertheless, we conjecture that, similarly to the centralized Karcher mean algorithm, the proposed protocol does converge to a global minimizer when the rotations R_i are "close enough" to each other. The following proposition analyzes the properties of such a global minimizer.

Proposition 2. *If* G *is connected, the global minimizer of* φ *is of the form* $(R_1, R_2, \ldots, R_N) = (R_c, R_c, \ldots, R_c)$ *for some* $R_c \in SO(3)$.

Proof. As we have seen before, φ is non-negative. In addition, we note that $\varphi = 0$ implies that for each pair $\{i, j\} \in E$, we have either $a_{ij} = 0$ or $d^2(R_i, R_j) = 0$. Also notice that the distance $d(R_i, R_j) = 0$ if and only if the associated rotations are equal, i.e., $R_i = R_j$. Since G is connected, we can conclude from the above that φ achieves its global minimum $\varphi = 0$, if and only if $\forall i, j : R_i = R_j = R_c$. □

It is important to note that the global minimizer of φ is not unique. This can be verified by observing that $\varphi = 0$ whenever $\forall i = 1, \ldots, N, R_i = R_c$ for any $R_c \in SO(3)$. In principle the same phenomena occurs in the Euclidean case, as can be seen by setting $x_i = x_j$ in Eq. (2.5). However, in the Euclidean case the average consensus protocol converges to the unique global mean, because the sum of the updates at each step preserves the mean of the data, as shown in Eq. (2.7).

In the case of $SO(3)$, by using the fact $a_{ij} = a_{ji}$ and $\log(R^\top Q) = -\log(Q^\top R)$ when $R, Q \in SO(3)$, we have that the sum of the logarithms of the updates over the nodes is zero:

$$\sum_{k=1}^{N} \log(U_k^{(l+1)}) = \varepsilon \sum_{k=1}^{N} \sum_{i=1}^{N} a_{ik} \log({R_k^{(l)}}^\top R_i^{(l)}) = 0. \quad (3.6)$$

However, this is not sufficient to make the protocol converge to the global Karcher mean of the N rotations. To see this, let us denote the Karcher mean of the rotations $R_k^{(l)}$ at the l-th iteration by $\bar{R}$. From the definition of the Karcher mean we have that

$$\sum_{k=1}^{N} \log(\bar{R}^\top R_k^{(l)}) = 0. \quad (3.7)$$

Now, recall the Campbell-Baker-Hausdorff (CBH) formula:

$$\exp(A)\exp(B) = \exp(C), \quad (3.8)$$

where A, B, and C are elements of the Lie algebra $so(3)$, and

$$\begin{aligned} C &= A + B + \frac{1}{2}[A, B] + \frac{1}{12}\big([A, [A, B]] + [B, [B, A]]\big) + \ldots] \\ &= A + B + \mathcal{O}(|(A, B)|) \end{aligned} \quad (3.9)$$

From our consensus protocol, we see that after one iteration:

$$\begin{aligned} &\sum_{k=1}^{N} \log(\bar{R}^\top R_k^{(l+1)}) = \sum_{k=1}^{N} \log(\bar{R}^\top R_k^{(l)} U_k^{(l)}) \\ &= \sum_{k=1}^{N} \log(\bar{R}^\top R_k^{(l)}) + \sum_{k=1}^{N} \log(U_k^{(l)}) \\ &+ \mathcal{O}\big(|(\bar{R}^\top R_k^{(l)}, \log(U_k^{(l)})|\big) \\ &= 0 + \mathcal{O}\big(|(\bar{R}^\top R_k^{(l)}, \log(U_k^{(l)})|\big) \end{aligned} \quad (3.10)$$

where we used (3.6) and (3.7) to arrive at the last equality. Hence, we see that the Karcher mean at iteration l is equal to the Karcher mean at iteration $l + 1$ only if the higher order terms can be ignored. We have shown the following result.

Proposition 3. *The consensus in the manifold protocol* (3.3) *does not generally converge to the Karcher mean.*

As a consequence, it is interesting to consider the particular case where all the rotations have a common axis and belong to a one-parameter family in $SO(3)$. In this particular situation, it is possible to write $R_i = \exp(\theta_i \hat{w})$, where $w \in \mathbb{R}^3$ represents the common rotation axis and $\theta_i \in \mathbb{R}$ are the rotation angles. Therefore, all the rotation updates $U_k^{(l)}$ also belong to the same family. Hence, they commute with each of the rotations, i.e., $R_i^{(l)} U_j^{(l)} = U_j^{(l)} R_i^{(l)}$, $i, j \in 1, \ldots, N$. When this happens, the Lie brackets and the other higher order terms of (3.10) are zero and the algorithm therefore converges to the correct mean.

3.2. Consensus in the tangent space ($so(3)$)

As we have seen above, the estimate of the global Karcher mean obtained by the consensus in the manifold algorithm does not necessarily remain constant at each iteration. To overcome this issue, we propose a second method called *consensus in the tangent space* that performs consensus on the tangent space to the manifold rather than in the manifold itself. This new method follows the same steps as the global iterative algorithm for estimating the Karcher mean described in §2.4. Therefore, it has the same convergence properties as the global algorithm, except that it requires a common initialization for all the nodes.

In the consensus in the tangent space protocol, all the nodes start with a common initialization $R_i^{(1)} = R_c$. At each iteration l, every node i computes the covariant derivative of the distance between the common estimate $R_i^{(l)}$ and its own rotation measurement R_i as

$$\nabla_{R_i^{(l)}} d^2(R_i^{(l)}, R_i) = R_i^{(l)} \log({R_i^{(l)}}^\top R_i). \tag{3.11}$$

The rotation estimate at each node is then updated as

$$R_i^{(l+1)} = R_i^{(l)} U^{(l)}, \tag{3.12}$$

where $U^{(l)}$ is a rotation matrix that is computed as

$$U^{(l)} = \exp\left(\frac{1}{N}\sum_{i=1}^{N} \log\left({R_i^{(l)}}^\top R_i\right)\right). \tag{3.13}$$

Notice that (3.13) requires the computation of the quantity $\bar{w} = \frac{1}{N}\sum_{i=1}^{N} \log({R_i^{(l)}}^\top R_i) \in so(3)$, which involves measurements from the entire network. In order to compute $\bar{w}$ in a distributed fashion, notice that each node can compute the matrix $w_i = \log({R_i^{(l)}}^\top R_i) \in so(3)$. Since $so(3)$ is isometric to $\mathbb{R}^3$, we can interpret each w_i as a vector in $\mathbb{R}^3$, and compute $\bar{w}$ using the Euclidean consensus algorithm. Upon convergence, all the nodes share the same value of $U^{(l)}$ and we can compute $R_i^{(l+1)}$ using (3.12). Notice that since the initial rotations $R_i^{(1)}$ were set to the same value, the estimates $R_i^{(l)}$ at the end of the l-th step, will be the same for all the nodes. Therefore, we can estimate the global Karcher mean by repeated application of the consensus algorithm in the tangent space, as follows:

1. Set initial rotations as $R_i^{(1)} = R_c \quad \forall i = 1, \ldots, N$.
2. Compute the mean $\bar{w} = \frac{1}{N}\sum_{i=1}^{N} \log({R_i^{(l)}}^\top R_i)$ across the network using an Euclidean consensus algorithm.
3. While $\|\bar{w}\|_F < \delta$, compute the updates as $R_i^{(l+1)} = R_i^{(l)} \exp(\bar{w})$, $i = 1, \ldots, N$ and goto step 2.

By comparing this procedure with the global algorithm used for estimating the Karcher mean in §2.4, it is easy to see that both algorithms modify the estimated mean through the same sequence of updates. Therefore, the consensus algorithm in the tangent space has the same convergence properties as the global algorithm.

3.3. Combined algorithm

In summary, we have presented two consensus algorithm on $SO(3)$ with complimentary properties. The consensus in the manifold algorithm is automatically initialized with the rotation R_i at each node, as in the classical consensus algorithm. Then, the rotations at each node are updated, and the estimates of the different nodes converge to a common value. Unfortunately, due to the curvature of $SO(3)$, this common value is only an approximation of the true global Karcher mean.

On the other hand, the consensus in the tangent space algorithm is provably convergent to the true Karcher mean. However, it requires a common initialization R_c, which is independent from the actual data R_i. This is potentially a problem, because R_c could be in the cut locus of one of the R_i's. In this case, the logarithm map is not defined and the algorithm's framework breaks down.

To overcome this issue, once could initialize the method using $R_c = R_k$ for some $k = 1, \ldots, N$. Alternatively, one can merge the two consensus algorithms in the following manner. First, we run the consensus in the manifold until it converges, thereby obtaining a set of rotations that are costant across the network and close to the true Karcher mean. This result is used to initialize the consensus in the tangent space, which then converges to the Karcher mean.

4. DISTRIBUTED OBJECT POSE ESTIMATION IN A LOCALIZED CAMERA NETWORK

In the discussion above, we have implicitly used the fact that all the poses $\{(R_i, T_i)\}_{i=1}^N$ are provided with respect to the same reference frame. In this section, we relax this condition and extend our algorithms to the case where each node measures the object pose with respect to its own reference frame.

Let $h_0 = (S_0, t_0) \in SE(3)$ be an arbitrary reference frame. For the sake of simplicity, we choose $h_0 = (I, \mathbf{0})$. Let $h_i = (S_i, t_i),\ i = 1, \ldots, N$ be the poses of the nodes with respect to the reference frame. We will assume that the network is *localized*, i.e., each node $i = \{1, \ldots, N\}$ knows the pose of node j (where $\{i, j\} \in E$) with respect to its frame of reference. We use $h_{ij} = (S_{ij}, t_{ij}) \in SE(3)$, with $i = \{1, \ldots, N\}$, $\{i, j\} \in E$, to indicate the pose of node i in the reference system of node j. It can easily be verified that

$$h_{ij} = h_j h_i^{-1}. \tag{4.1}$$

This change of coordinates allows us to transform pose estimates from coordinate frame i to coordinate frame j as

$$(R_j, T_j) = (S_{ij} R_i, S_{ij} T_i + t_{ij}). \tag{4.2}$$

These simple considerations will enable us to modify the consensus algorithms presented in the previous section in order to relax the constraint that all the rotations need to be expressed in the same coordinate system.

4.1. Consensus in the manifold

When the rotations are written with respect to different coordinate systems, the cost function in (3.1) minimized by the consensus in the manifold algorithm needs to be modified to

$$\varphi(R_1, \ldots, R_N) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij} d^2(S_{i0} R_i, S_{j0} R_j). \tag{4.3}$$

This is essentially the same cost function as the one in (3.1), except that the R_i's are transformed from frame i to a common reference frame by the S_{i0}'s. Notice that the distance $d(\cdot,\cdot)$ is left-invariant with respect to the action of the rotation group, i.e., $d(R', R'') = d(RR', RR''), \forall R, R', R'' \in SO(3)$. As a consequence, each summand in the cost function φ does not depend on the choice of its coordinate system. In fact, since $S_{ij} = S_{j0}^{-1} S_{i0}$, we can rewrite φ as

$$\varphi(R_1, \ldots, R_N) = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij} d^2(S_{ij} R_i, R_j), \qquad (4.4)$$

which brings the rotation at node i into the coordinate system of node j as $S_{ij} R_i$. It follows that the gradient of φ does not depend on the choice of the coordinate system either.

Therefore, we can implement the consensus in the manifold algorithm as before, i.e., $R_k^{(l+1)} = R_k^{(l)} U_k^{(l)}$, except that the update matrices $U_k^{(l)}$ are modified to

$$U_k^{(l)} = \exp(\varepsilon \sum_{i=1}^{N} a_{ik} \log({R_k^{(l)}}^\top S_{ik} R_i^{(l)})). \qquad (4.5)$$

Notice that Eq. (4.5) is the same as Eq. (3.5), except that $R_i^{(l)}$ is now written with respect to the coordinate frame of node k.

4.2. Consensus in the tangent space

Adapting the consensus in the tangent space to the case where the rotations are in different reference frames requires only a modification in the initialization of the nodes. To see this, recall that this algorithm proceeds by applying a common update $U^{(l)}$ to all the nodes, i.e., $R_i^{(l+1)} = R_i^{(l)} U^{(l)}$. The update is computed as in (3.13) by performing Euclidean consensus on the quantities $\log\left({R_i^{(l)}}^\top R_i\right)$. Since $\log({R_i^{(l)}}^\top S_i^\top S_i R_i) = \log({R_i^{(l)}}^\top R_i)$ for any $S_i \in SO(3)$, the update does not depend on the choice of the coordinate system at each node. Therefore, $U^{(l)}$ can be still computed as in (3.13), as long as R_i and $R_i^{(l)}$ are expressed in the same coordinate system. In order for this to be the case, we only need to modify the initialization of the nodes. Specifically, the initial values $R_i^{(1)}$ have to be consistent with the localization of the network, so that $R_i^{(1)}$ and R_i are in the same reference frame.

In conclusion, we require a method to generate this consistent initialization, bearing in mind the assumption that only the relative transformations h_{ij} between neighbors are known. This motivates our final algorithm which we propose below.

4.3. Combined algorithm

In order to initialize the consensus in the tangent space method, we can set $R_1^{(1)} = R_1$ and then propagate its estimate to the entire network as $R_j^{(1)} = S_{1j} R_1$, $j = 2, \ldots, N$. Alternatively, we can use the consensus in the manifold as an initialization for the consensus in the tangent space, similarly to what we did in Section 3.3. The latter solution has two advantages: first, it provides a fully distributed way to obtain an initialization consistent with the network localization (which is more robust to the failure of single nodes). Second, the solution found by the first stage is close to the true Karcher mean, therefore fewer iterations are required for the convergence of the consensus in the tangent space algorithm.

5. EXPERIMENTS

We tested our algorithms on synthetic data by simulating a network with $N = 20$ nodes connected with a k-regular graph, $k = 6$. We generated the measurements for the nodes by adding noise to the ground truth pose

$$g = \{I, \begin{bmatrix} 5 & 5 & 5 \end{bmatrix}^\top\} \in SE(3). \qquad (5.1)$$

In $SO(3)$ we used as noise term rotations around a random axis (drawn from a uniform distribution on $\mathbb{S}^2$) by an angle generated from a Gaussian distribution with zero mean and variance 20°. However, we rejected angles drawn outside the interval $[-90^\circ, 90^\circ]$ in order to ensure that the Karcher mean is actually unique [21]. For the noise in the translation space, we simply used Gaussian noise with a covariance matrix $0.35I$.

For the consensus in the manifold, we used 70 iterations. For the consensus in the tangent space, we used 4 iterations for the main loop and 140 iterations for the Euclidean consensus. We initialized this algorithm with $R_c = I$. For the combined algorithm, we took the result of the consensus in the manifold and added two iterations of the consensus in the tangent space, again with 140 iterations for the Euclidean consensus.

In our experiments the number of iterations was fixed, but a better implementation would stop when a given tolerance for the convergence is satisfied. Note that the consensus in the tangent space requires very few iterations for the main loop, but it requires a really good convergence of the inner Euclidean consensus (hence the high number of iterations) in order to keep all the nodes aligned with high precision.

We have run two sets of experiments. In Experiment 1, all the measurements are in a common reference frame. In Experiment 2, we use a localized network where the reference frames are randomly generated within a cube of dimension $20 \times 20 \times 20$ and with uniform random orientations. We assume that the transformations between neighboring nodes h_{ij} are known without noise.

We repeated the experiments 100 times and collected the initial errors for the rotations and translations, the final error for the translations (common to all the algorithms) and the final error for the rotations for the three algorithm. Figures 1 and 2 show histograms of these results, while Table 1 reports the maximum errors. For the rotations, we measured the angle of rotation in degrees, while for the translation we used the Euclidean norm of the error. In both cases, we used the result of the corresponding centralized algorithm as reference.

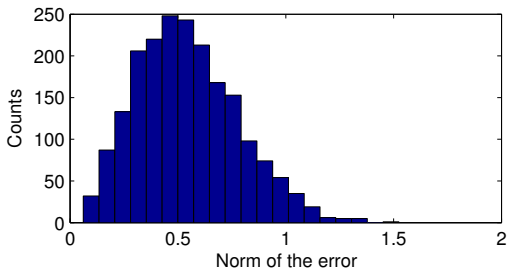
We see that the rotation errors for the consensus in the manifold are on the order of 10^{-2}. As expected they are not zero but, as a first order approximation, they can be considered sufficiently low for many applications. The errors for the consensus in the tangent space and the combined algorithm are on the order of 10^{-6} and 10^{-5}, respectively, and are actually zero for many of the trials, as one can see from the histograms. In this case, the errors are mostly due to numerical issues. The combined algorithm performs slightly worse than the consensus in the tangent space algorithm because the residual errors at the end of the first step (consensus in the manifold) carry over to the second step (consensus in the tangent space).

We will now examine in more detail the trial for which the consensus in the manifold algorithm gave the worst estimate. Fig. 3 plots the angle between the estimate at each node and the global Karcher mean as a function of the number of iterations. Note that the final error is not zero, because the algorithm does not preserve the mean of the data from one iteration to the next. To better illustrate this point, Fig. 4 plots the error between the Karcher mean of the $R_i^{(l)}$'s and the global Karcher mean at the beginning and after each iteration. As we can see, at the beginning such error is zero, but it rapidly increases after few iterations. This is not the case for the consensus in the tangent space algorithm. Fig. 5 shows the error between the estimate at each node and the global Karcher mean, with the same input data as before. As expected, all the nodes have the same error, because their estimates are aligned. The error decreases in a small number of iterations (but remember that each iteration requires to run the Euclidean consensus in the tangent space until convergence) and the final error is zero.

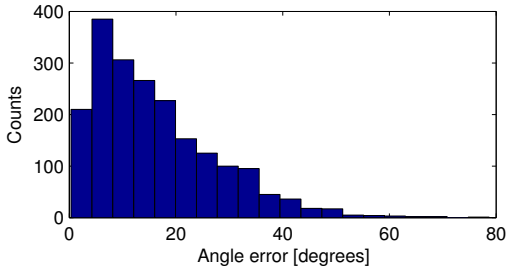
6. CONCLUSION AND FUTURE WORK

We have presented a distributed consensus algorithm for computing the average pose of an object from relative pose measurements acquired by a localized camera sensor network. A key contribution of our work was to demonstrate that naive extensions of Euclidean consensus algorithms to $SE(3)$ may fail to converge to the correct average. To correct this issue, we proposed a distributed consensus algorithm on $SE(3)$, which combines classical consensus on Euclidean spaces with optimization on $SE(3)$. We also assessed the validity of our approach through synthetic simulations.

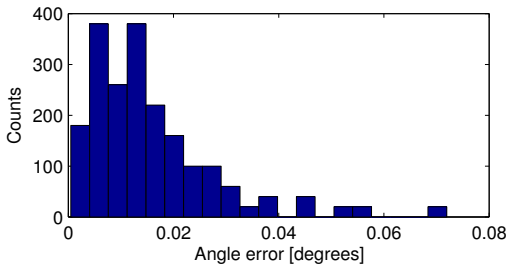
The are a number of interesting directions in which our work could be extended. A first direction is to apply our approach to real data, e.g., for estimating the pose of a face for face recognition purposes. A second direction is to study conditions for initializing our consensus algorithm that guarantee convergence to the global minimum. We would like also to modify our algorithm in order to improve its robustness to errors in the localization of the network. Moreover, we would like to extend our framework to the problem of network localization itself and distributed bundle adjustment, by removing the estimation of the pose at each node as an intermediate step



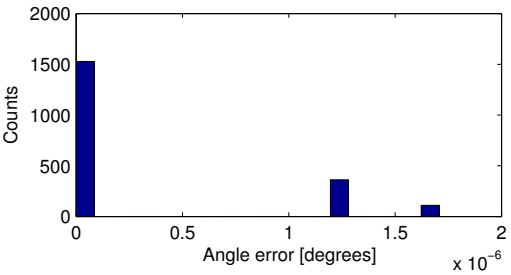
(a) Initial translation errors.



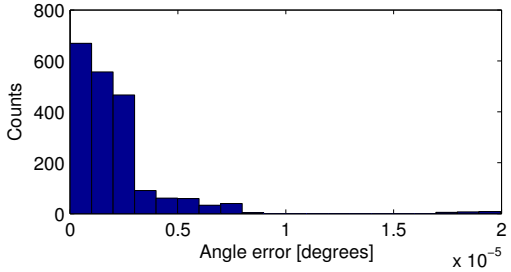
(b) Initial rotation errors.



(c) Final rotation errors for the consensus in the manifold.

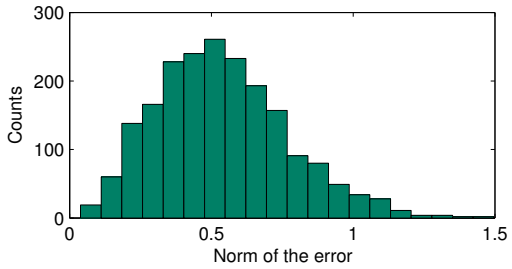


(d) Final rotation errors for the consensus in the tangent space.

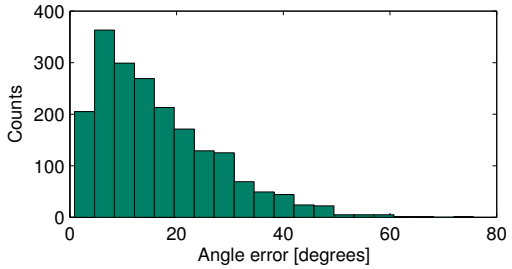


(e) Final rotation errors for the combined algorithm.

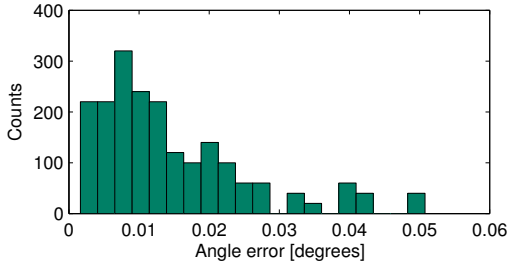
Fig. 1. Experiment 1: initial and final translation and rotation errors for all the proposed algorithms with measurements in a common reference frame.



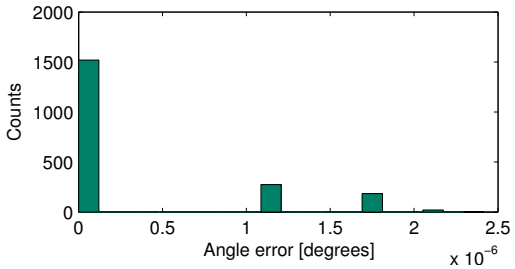
(a) Initial translation errors.



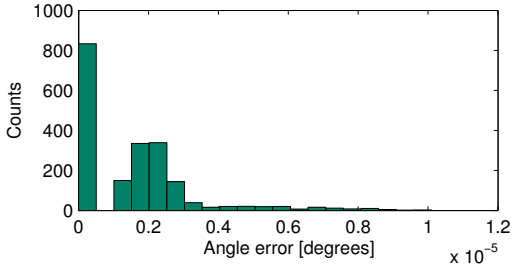
(b) Initial rotation errors.



(c) Final rotation errors for the consensus in the manifold.



(d) Final rotation errors for the consensus in the tangent space.



(e) Final rotation errors for the combined algorithm.

Fig. 2. Experiment 2: initial and final translation and rotation errors for all the proposed algorithms with a localized network.

	Experiment 1	Experiment 2
Consensus in $SO(3)$ (rot)	$7.19 \cdot 10^{-2}$	$5.08 \cdot 10^{-2}$
Consensus in $so(3)$ (rot)	$1.71 \cdot 10^{-6}$	$2.41 \cdot 10^{-6}$
Combined consensus (rot)	$1.99 \cdot 10^{-5}$	$1.01 \cdot 10^{-5}$
Translation	$1.87 \cdot 10^{-8}$	$2.04 \cdot 10^{-8}$

Table 1. Maximum errors for rotations and translations with measurements in the same reference frame (Experiment 1) and with localized network (Experiment 2).

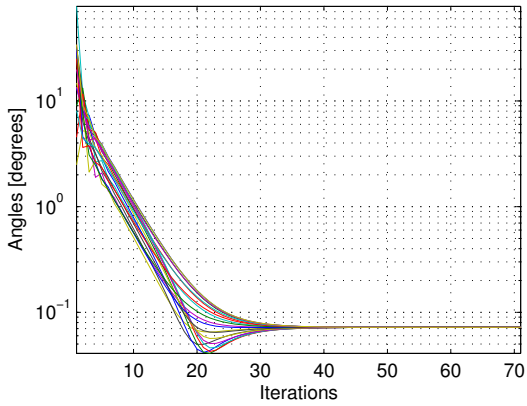


Fig. 3. Rotation errors between the estimate of each node and the global Karcher mean for the worst trial of the consensus in the manifold algorithm.

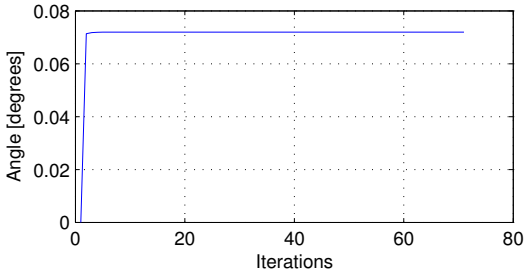


Fig. 4. Rotation errors between the Karcher mean of the estimates after each iteration and the global Karcher mean for the worst trial of the consensus in the manifold algorithm.

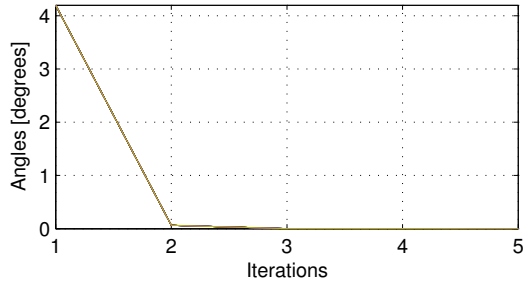


Fig. 5. Rotation errors between the estimate of each node and the global Karcher mean for the consensus in the tangent space algorithm.

and working directly with feature coordinates in the images.

7. ACKNOWLEDGEMENTS

The authors thank Avinash Ravichandran and Dheeraj Singaraju for their valuable help during the writing of this paper. The authors also thank the anonymous reviewers for their comments and suggestions. This work was partially supported by grants NSF CAREER 0447739, NSF EHS-0509101, ONR N00014-05-1083 and WSE/APL Contract: Information Fusion & Localization in Distributed Sensor Systems.

8. REFERENCES

[1] M. DeGroot, "Reaching a consensus," *Journal of the American Statistical Association*, vol. 69, no. 345, pp. 118–121, 1974.

[2] A. Jadbabaie, J. Lin, and A. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 988–1001, 2003.

[3] R. Olfati-Saber and R. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 3, pp. 1520–1533, 2004.

[4] R. Olfati-Saber, J. Fax, and R. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[5] W. Ren, R. Beard, and E. Atkins, "Information consensus in multivehicle cooperative control," *IEEE Control Systems Magazine*, pp. 71–82, 2007.

[6] R. Vidal, Y. Ma, S. Hsu, and S. Sastry, "Optimal motion estimation from the multiview normalized epipolar constraint," in *IEEE International Conference on Computer Vision*, 2001, vol. 1, pp. 34–41.

[7] R. Vidal and S. Sastry, "Optimal segmentation of dynamic scenes from two perspective views," in *Conference on Computer Vision and Pattern Recognition*, 2003, vol. 2, pp. 281–286.

[8] A. Goh and R. Vidal, "Clustering and dimensionality reduction on Riemannian manifolds," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

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

[10] P. Baker and Y. Aloimonos, "Calibration of a multicamera network," in *CVPR Workshop on Omnidirectional Vision and Camera Networks*, 2000, pp. 134–141.

[11] A. Barton-Sweeney, D. Lymberopoulos, and A. Savvides, "Sensor localization and camera calibration in distributed camera sensor networks," in *International Conference on Broadband Communications, Networks and Systems*, 2006, pp. 1–10.

[12] R. Farrell, R. Garcia, D. Lucarelli, A. Terzis, and I-J. Wang, "Localization in multi-modal sensor networks," in *Third International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, 2007.

[13] S. Sinha, M. Pollefeys, and L. McMillan, "Camera network calibration from dynamic silhouettes," in *International Conference on Computer Vision and Pattern Recognition*, 2004, vol. 1, pp. 195–202.

[14] S. Sinha and M. Pollefeys, "Synchronization and calibration of a camera network for 3D event reconstruction from live video," in *International Conference on Computer Vision and Pattern Recognition*, 2005, vol. 2, p. 1196.

[15] D. Devarajan, R. Radke, and H. Chung, "Distributed metric calibration of ad hoc camera networks," *ACM Transactions on Sensor Networks*, vol. 2, no. 3, pp. 380–403, 2006.

[16] D. Devarajan and R. Radke, "Calibrating distributed camera networks using belief propagation," *EURASIP Journal of Applied Signal Processing*, pp. 221–221, 2007.

[17] C. Gramkow, "On averaging rotations," *Journal of Mathematical Imaging and Vision*, vol. 15, no. 1-2, pp. 7–16, 2002.

[18] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry, *An Invitation to 3D Vision: From Images to Geometric Models*, Springer Verlag, 2003.

[19] Maher Moakher, "Means and averaging in the group of rotations," *SIAM Journal on Matrix Analysis and Applications*, vol. 24, no. 1, pp. 1–16, 2002.

[20] H. Karcher, "Riemannian center of mass and mollifier smoothing," *Communications on Pure and Applied Mathematics*, vol. 30, no. 5, pp. 509–541, 1977.

[21] J. Manton, "A globally convergent numerical algorithm for computing the centre of mass on compact Lie groups," in *International Conference on Automation, Robotics, Control and Vision*, 2004, vol. 3, pp. 2211–2216.

[22] M. Frechet, "Les elements aleatoires de nature quelconque dans un espace distancie," *Annales De L'Institut Henri Poincare*, vol. 10, pp. 235–310, 1948.

[23] Y. Ma, J. Košecká, and S. Sastry, "Optimization criteria and geometric algorithms for motion and structure estimation," *International Journal of Computer Vision*, vol. 44, no. 3, pp. 219–249, 2001.