Distributed Image-Based 3-D Localization of Camera Sensor Networks

Roberto Tron* and René Vidal*

Abstract— **We consider the problem of distributed estimation of the poses of N cameras in a camera sensor network using image measurements only. The relative rotation and translation (up to a scale factor) between pairs of neighboring cameras can be estimated using standard computer vision techniques. However, due to noise in the image measurements, these estimates may not be globally consistent. We address this problem by minimizing a cost function on $SE(3)^N$ in a distributed fashion using a generalization of the classical consensus algorithm for averaging Euclidean data. We also derive a condition for convergence, which relates the step-size of the consensus algorithm and the degree of the camera network graph. While our methods are designed with the camera sensor network application in mind, our results are applicable to other localization problems in a more general setting. We also provide synthetic simulations to test the validity of our approach.**

I. INTRODUCTION

Consider a set of low-power sensor nodes, each equipped with a camera, and assume that the nodes are able to communicate through a wireless interface. Networks of this kind could be used in a variety of applications, such as 3-D reconstruction of large environments or tracking of mobile targets. In many of these applications, the nodes are arbitrarily distributed in 3-D space, but they have to exchange information about a common quantity (*e.g.*, the position of the target). Therefore, there is a need to accurately *localize* the network, *i.e.*, to find the cameras' relative poses.

There is a large body of work on network localization. Most of these works can be categorized according to the assumptions made about the problem and the nature of the solution: dimension of the ambient space (planar versus tridimensional), type of algorithm (centralized versus distributed), type of measurements (distances, angles of arrival, bearing measurements, coordinate transformations between nodes; see [1] and reference therein), presence of *beacons* or *anchors* (*i.e.*, nodes with a known position, [2], [1]), and special assumptions about the environment (*e.g.*, dynamic environment [3], environment with markers [4], etc.).

In this paper, we assume that each camera can extract a set of 2-D points from each image, and that neighboring cameras (*i.e.*, cameras with intersecting field of view) can match these 2-D points to estimate (with noise) their relative rotation and their relative direction of translation. This can be done using standard two-view computer vision techniques [5]. These assumptions exclude most of the previous work on distributed localization, in particular all the vast literature that assumes distance measurements. We assume also that the scene is static (or that all the cameras are synchronized) and that the communications between cameras are lossless.

Under these assumptions, we consider the problem of recovering the camera locations and orientations (up to a global reference frame and scale factor) from image measurements only. There are many challenges to solving this problem in a distributed fashion. First, each node obtains its measurements with respect to its own reference frame, which is initially not known to any of the other nodes. Second, the measurements are subject to noise and are generally not *consistent* with each other (*i.e.*, combining the relative transformations between nodes arranged in a cycle does not necessarily give rise to the identity transformation). Third, due to limited hardware resources, even if the global network is connected as a whole, each node can only interact with nearby nodes.

Distributed consensus algorithms are a natural candidate for integrating noisy, local relative pose measurements into a global, consistent camera network localization. Traditional consensus algorithms were developed to work with low-dimensional, Euclidean quantities and typically assume that states and measurements coincide. Consensus algorithms have also been studied and used in a variety of situations, *e.g.*, distributed averaging, formation control, synchronization, rendez-vous in space, and with varying network topology, communication delays, etc. (see [6] for a review). Consensus-like algorithms have also been used for localization with range measurements (see, for instance, [7] and [1]).

Paper contributions. In this paper, we generalize existing consensus algorithms for estimating the pose of each node from noisy, inconsistent measurements. Specifically, we propose consensus algorithms that operate on non-Euclidean states (poses), which are different from the measurements (relative transformations extracted from images). We show how constraints that are global to the network (consistent localization, choice of a global reference frame and scale) can be enforced in a distributed way. Finally, we also generalize previous results on a distributed method for choosing the consensus step-size (in the estimation of the translations).

Related work. A distributed method for camera localization and calibration based on Belief Propagation (as opposed to consensus) was proposed in [8]. However, no conditions for a consistent solution were imposed and the non-Euclidean structure of the rotations was not rigorously exploited. Our approach is more closely related to [9], but with some important differences. First, we consider the more general 3-D case instead of 2-D. Second, our algorithms require communication among neighboring nodes only, as opposed to all nodes in each cycle. Third, we propose an algorithm to recover both the translations (up to a global scale) and

*Center for Imaging Science, Johns Hopkins University, Baltimore MD 21218. This work was supported by the grant NSF CNS-0834470.

the rotations (as opposed to the rotations only in [9]). Our approach is also related to averaging consensus on $SE(3)$ [10], except that we address the more challenging problem of camera localization, rather than pose averaging.

II. REVIEW, NOTATION AND DEFINITIONS

In this section, we will introduce the basic tools needed to develop our method. We will first review the geometry of the groups of rotations and rigid body transformations. We will then introduce some notions from graph theory and the basics of the localization problem. Finally, we will review the classical averaging consensus algorithm.

A. Geometry of $SO(3)$ and $SE(3)$

Our methods will use optimization tools on the group of rotations $SO(3) = \{R \in \mathbb{R}^{3\times 3} : R^\top R = I, \det(R) = +1\}$. Hence, we will first review the computation of distances and gradients in $SO(3)$. Given any point $R \in SO(3)$, the *tangent space at* R, $T_R SO(3)$, is given by $T_R(SO(3)) = \{R\hat{v} : \hat{v} \in so(3)\}$, where $so(3) = \{\hat{v} \in \mathbb{R}^{3\times 3} : \hat{v} = -\hat{v}^\top\}$ is the space of skew-symmetric matrices and $\hat{v}$ is the matrix generating the cross product by v, *i.e.*, $\hat{v}u = v \times u$, for all $u \in \mathbb{R}^3$.

A Riemannian metric is a continuous collection of inner products in the tangent space at each point. In $SO(3)$, the Riemannian metric is denoted as $\langle \Delta_1, \Delta_2 \rangle = \operatorname{trace}\{\Delta_1^\top \Delta_2\}$, $\Delta_1, \Delta_2 \in T_R SO(3)$. This metric can be used to measure the length of curves between two given points on the manifold. Curves with minimum length are called *geodesics* and their length defines the *geodesic distance* between the two points:

$$d^2_{SO(3)}(R, S) = -\frac{1}{2}\operatorname{trace}\big(\log(R^\top S)^2\big), R, S \in SO(3), \quad (1)$$

where $\log$ is the matrix logarithm.

The *exponential map* at a point $R \in SO(3)$, $\exp_R(\Delta) : T_R(SO(3)) \to SO(3)$ is a diffeomorphism that associates to each point Δ in a neighborhood of the origin of $T_R(SO(3))$ a point S on the (unique) geodesic passing through R in the direction Δ. The *logarithm map* $\log_R(S) : SO(3) \to T_R(SO(3))$ is the inverse of the exponential map. The exponential and logarithm map at any point in $SO(3)$ can be related to the same maps at the identity by *parallel transport*:

$$\exp_R(\Delta) = R\exp(R^\top \Delta), \quad (2)$$

$$\log_R(S) = R\log(R^\top S), \quad (3)$$

where $\exp$ is the matrix exponential.

Given a smooth function $f : SO(3) \to \mathbb{R}$, the *Riemannian gradient* $\operatorname{grad}_R f$ is given in [11] as

$$\operatorname{grad}_R f = f_R - R f_R^\top R, \quad (4)$$

where $f_R = \frac{\partial f}{\partial R}$ is the Euclidean derivative of f with respect to the matrix R. In particular, we will make use of the gradient of the squared distance function

$$\operatorname{grad}_R d^2_{SO(3)}(R, S) = -R\log(R^\top S) \in T_R SO(3). \quad (5)$$

Gradient descent iterations on $SO(3)$ take the form

$$R(l+1) = R(l)\exp_{R(l)}(-\varepsilon \operatorname{grad}_R f), \quad (6)$$

where $\varepsilon > 0$ is the step-size.

We will also perform optimization of functions on the group of rigid body transformations $SE(3) = \{g = (R, T) : R \in SO(3) \text{ and } T \in \mathbb{R}^3\}$. Unfortunately, unlike $SO(3)$, $SE(3)$ does not have a "natural" (bi-invariant) Riemannian metric. While one could use a left-invariant or right-invariant metric, for the sake of simplicity we will use the double geodesic distance (see [12]). In this case, the group structure is discarded and $SE(3)$ is considered as the product $SO(3) \times \mathbb{R}^3$. The distance between two transformation $g_1 = (R_1, T_1)$ and $g_2 = (R_2, T_2)$ is then given by:

$$d^2_g(g_1, g_2) = d^2_{SO(3)}(R_1, R_2) + \|T_1 - T_2\|^2. \quad (7)$$

The gradient on $SO(3) \times \mathbb{R}^3$ can then be computed by considering separately the rotation and translation components.

B. Graph Theory and Localization

We represent a network of N nodes as a directed, strongly connected graph $G = (V, E)$. The set $V = \{1, \dots, N\}$ represents the set of vertices in the graph and each $i \in V$ corresponds to one of the nodes. The set $E \subseteq V \times V$ represents the set of edges in the graph. An edge (i, j) belongs to E if node j can communicate with node i and measure their relative position (in the terminology of [8], G represents the *vision graph*). We assume that if an edge (i, j) belongs to E, then its reverse (j, i) is also in E (in other words, the adjacency matrix of the graph is symmetric).

Given a graph G, we define its *incidence matrix* $C \in \mathbb{R}^{|E|\times|V|}$. Each row c_e of C corresponds to a different edge $e \in E$. The vector c_e is zero except in the i-th entry (where it has a 1) and in the j-th entry (where it has a -1). We also define the *degree* of a node i, $\deg(i)$, as the number of outgoing edges, which by our assumptions is the same as the number of incoming edges. The *maximum degree* of G is denoted as $\Delta_G = \max_i \deg(i)$.

We represent the pose of each node i as $g_i = (R_i, T_i) \in SE(3)$. Given two rigid body transformation $g_1 = (R_1, T_1)$ and $g_2 = (R_2, T_2)$, the group multiplicative operation is the composition defined as $g_1 \circ g_2 = (R_1 R_2, R_1 T_2 + T_1)$. For each edge $(i, j) \in E$, we define the relative change of coordinates $g_{ij} = (R_{ij}, T_{ij}) \in SE(3)$ which are given by $g_{ij} = g_i^{-1} \circ g_j$ or, more explicitly,

$$\begin{aligned} R_{ij} &= R_i^\top R_j, \\ T_{ij} &= R_i^\top (T_j - T_i). \end{aligned} \quad (8)$$

We can obtain the pose of one camera from the pose of another as $g_j = g_i \circ g_{ij}$. Note that these relative transformations are invariant to the choice of a global reference frame. We define also the relative translation direction $t_{ij} = \frac{T_{ij}}{\|T_{ij}\|}$.

A *path* ℓ from node i to node j in G is defined as a sequence of nodes starting with i and ending in j such that there exist an edge in E between consecutive nodes. Formally $\ell = \{w_1, \dots, w_n\}$, $w_m \in V$, $w_1 = i$, $w_n = j$, $(w_m, w_{m+1}) \in E$, $m \in \{1, \dots, n-1\}$. A *cycle* is a path from node i to itself without repeated nodes (except for the initial and the final). Given a path ℓ, we define the *relative change of coordinates along* ℓ as $g_\ell = g_{w_n w_{n-1}} \circ \dots \circ g_{w_2 w_1}$.

We now give a formal definition of a *localized network*.

Definition 1 (Localized network): A network is said to be localized if there is a set of relative transformations $\{(R_{ij}, T_{ij})\}$ such that, when the reference frame of the first node is fixed to (R_1, T_1), the other absolute poses (R_i, T_i) are uniquely determined. That is, for any path ℓ from node 1 to node i, we have $g_i = g_\ell \circ g_1$, regardless of the chosen path.

C. Consensus Algorithms

We will now briefly review the derivation of the classical Euclidean consensus algorithm. We associate with each node i in the network a scalar *state* $x_i \in \mathbb{R}$. We say that the nodes are in a *consensus* configuration if $x_i = \alpha$, for all $i \in V$, where α is called the *collective decision* of the network.

A *distributed consensus algorithm* is a protocol that each node has to follow to update its state so that all the nodes reach a consensus. Moreover, the protocol must require communication with neighboring nodes only. A popular discrete time protocol for iteratively computing the mean of the states $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$ is given by

$$x_i(l+1) = x_i(l) + \varepsilon \sum_{i \rightsquigarrow j} (x_j(l) - x_i(l)), \quad x_i(0) = x_i, \tag{9}$$

where $x_i(l)$ is the state of the i-th node at the l-th iteration, $\varepsilon > 0$ is the step-size and $i \rightsquigarrow j$ implies that the sum is taken only on the terms $i, j \in V$, $(i, j) \in E$. It can be shown that if $\varepsilon < \frac{1}{\Delta_G}$ and the graph is connected, then the nodes converge to the mean $\bar{x}$, *i.e.*, $\lim_{l\to\infty} x_i(l) = \bar{x}$ [6].

Notice also that equation (9) is a gradient descent step to minimize the cost function

$$\varphi(x_1, x_2, \dots, x_N) = \frac{1}{2}\sum_{i \rightsquigarrow j} (x_i - x_j)^2, \tag{10}$$

and that the minimum of this cost function is attained when all states are in a consensus configuration. Since, in addition, the mean of the states is preserved at each iteration, *i.e.*,

$$\sum_{i=1}^{N} x_i(l) = \sum_{i=1}^{N} x_i(l+1) = \bar{x}, \tag{11}$$

the minimizer is given by the mean of the initial states $\bar{x}$.

III. DISTRIBUTED IMAGE-BASED 3-D LOCALIZATION

This section presents the main contribution of our work. We propose to solve the problem of finding a consistent localization for the network by defininig a suitable cost function and showing how it can be minimized on $SE(3)^N$ in a distributed way.

As stated in Section I, we use feature points extracted from the images of cameras i and j to obtain a noisy estimate $(\tilde{R}_{ij}, \tilde{t}_{ij})$ of their relative pose (R_{ij}, T_{ij}). Specifically, we assume that each camera can extract P_i feature points $x_i^{(p_i)}$, $p_i = 1, \dots, P_i$. These points are written in homogeneous coordinates, *i.e.*, they are points in $\mathbb{P}^2$, and correspond to the projections of points in 3-D space. We also assume that for each edge $(i, j) \in E$ the correspondences between points in images i and j can be established. Therefore, corresponding points $x_i^{(p_i)}$ and $x_j^{(p_j)}$ are related by

$$\mu_{i,p_i} x_i^{(p_i)} = \mu_{j,p_j} R_{ij} x_j^{(p_j)} + T_{ij}, \tag{12}$$

where the μ_{i,p_i} is the depth of the 3-D point p_i with respect to the i-th camera. It is a well known fact from computer vision that the corresponding images satisfy the *epipolar constraint*

$$x_i^{(p_i)\top} \hat{T}_{ij} R_{ij} x_j^{(p_j)} = 0. \tag{13}$$

Given enough point correspondences, it is possible to use (13) to estimate the *essential matrix* $\hat{T}_{ij} R_{ij}$ from which one can extract $t_{ij} = T_{ij}/\|T_{ij}\|$ and R_{ij} (see [13] for details).

Notice that (13) is a homogeneous equation in T_{ij}, thus the scale of the translation cannot be recovered. Intuitively, this corresponds to the fact that the global scale of a 3-D scene cannot be recovered from 2-D images alone due to perspective projection. In practice, due to noise, we will obtain $\tilde{t}_{ij} = \tilde{T}_{ij}/\|\tilde{T}_{ij}\|$ and $\tilde{R}_{ij}$, which are approximated versions of the true quantities. Moreover, due to communication constraints, the two cameras could perform their estimation from different sets of point correspondences, hence $\tilde{g}_{ij}$ could be different from $\tilde{g}_{ji}^{-1}$.

Our goal is to find a set of relative transformations g_{ij} that satisfy the consistency constraints given in Definition 1 and that, at the same time, are "as close as possible" to the relative measurements $\tilde{g}_{ij}$. An intuitive criterion is least squares, i.e., the sum of square distances. Since we are dealing with non-Euclidean quantities, we will use the distance d_g in $SO(3) \times \mathbb{R}^3$

$$\varphi = \frac{1}{2}\sum_{i \rightsquigarrow j} d_g^2(g_{ij}, \tilde{g}_{ij}) = \frac{1}{2}\sum_{i \rightsquigarrow j} d_{SO(3)}^2(R_{ij}, \tilde{R}_{ij}) + \|T_{ij} - \tilde{T}_{ij}\|^2. \tag{14}$$

In principle, we wish to minimize φ subject to the constraints in the definition of localized network. Unfortunately, these constraints involve the entire network and are not distributed. In [9] the authors propose a *cycle-distributed* solution where the constraints are enforced by sharing information between each node and all the cycles it belongs to.

In this paper, we propose to reparametrize each relative transformation g_{ij} with the absolute transformations g_i and g_j. In this way the consistency constraints will be automatically satisfied and each node will communicate with its neighboring nodes only. The following proposition shows the equivalence between Definition 1, the constraints considered in [9] and our parametrization of the solution.

Proposition 1: The following are equivalent:

1) The network is localized.
2) For any cycle $\ell = \{w_1, \dots, w_n, w_1\}$, $w_m \in V$, $n > 1$, the transformation along the cycle is $g_\ell = (I, 0)$.
3) There exist a set of absolute poses $g_i = (R_i, T_i)$ such that $g_{ij} = g_i^{-1} \circ g_j$.

A simple proof is given in [14]. Intuitively, given the absolute poses g_i, one can compute consistent relative transformations

g_{ij}. Conversely, if the relative transformations are given, there exist a set of g_i such that each g_{ij} factorizes as $g_i^{-1} \circ g_j$.

A downside of our formulation is that we have to give up the uniqueness of the solution: for any arbitrary global transformation $g_i' = g \circ g_i$, the value of the cost function φ will be the same. Nonetheless, Proposition 1 tells us also that, no matter what solution for the absolute poses g_i we compute, the relative transformations g_{ij} (which is what we are really interested in) will be uniquely determined.

Another challenge with the proposed formulation is that the translations can be estimated only up to a scale factor. Therefore, we need to add the unknown scales as new variables λ_{ij}, i.e. $\tilde{T}_{ij} = \lambda_{ij} \tilde{t}_{ij}$. After these considerations, we can rewrite (14) as

$$\begin{aligned} \varphi(\{R_i\}, \{T_i\}, \{\lambda_{ij}\}) &= \frac{1}{2} \sum_{i \rightsquigarrow j} d_g^2(g_i^{-1} g_j, \tilde{g}_{ij}) \\ = \frac{1}{2} \sum_{i \rightsquigarrow j} \big(d_{SO(3)}^2 (R_i^\top R_j, \tilde{R}_{ij}) &+ \| R_i^\top (T_j - T_i) - \lambda_{ij} \tilde{t}_{ij} \|^2 \big) \\ &= \varphi_R(\{R_i\}) + \varphi_T(\{R_i\}, \{T_i\}, \{\lambda_{ij}\}). \quad (15) \end{aligned}$$

Therefore, the cost φ is the sum of two terms: φ_R involving only the rotations and φ_T involving all the variables.

At this point, we observe that the unknown scales $\{\lambda_{ij}\}$ produce an undesired side-effect. If we substitute the trivial solution $T_i = T_j$, $i, j \in V$ and $\lambda_{ij} = 0$, $(i, j) \in E$ in (15), we achieve the global minimum $\varphi_T = 0$ regardless of the value of the rotations. Therefore, if we try to minimize (15) without any constraint on T_i or λ_{ij}, we could find solutions for the localization that are not meaningful (*e.g.*, all the translations collapsed to the same point). Moreover, we have to enforce the fact that the scales $\{\lambda_{ij}\}$ must be positive. One could think of different ways of adding constraints in order to settle both questions. In our case we propose to constraint the minimum scale, *i.e.*, $\lambda_{ij} \geq 1 \; \forall (i, j) \in E$. This is a global constraint because it involves all the nodes in the network. However, we will show in Section III-B that each node can enforce it separately, hence making it distributed.

In summary, we propose to find an optimal localization (in the least-squares sense) by solving the non-linear program

$$\begin{aligned} \min_{\{R_i\}, \{T_i\}, \{\lambda_{ij}\}} \; & \varphi(\{R_i\}, \{T_i\}, \{\lambda_{ij}\}) \\ \text{subject to } & \lambda_{ij} \geq 1 \quad (i, j) \in E \end{aligned} \qquad (16)$$

in a distributed fashion. To that end, in the following subsections we present the solution to two simpler problems. First, we ignore the translational part and find an initial set of rotations by optimizing φ_R only. Next, we assume that the rotations are fixed and find an initial set of translations and scales by optimizing φ_T only. Finally, we optimize φ over all the variables.

There are a number of reasons for this multi-step solution. First, the simpler problems can actually correspond to real localization problems (with relaxed assumptions with respect to what we assume here) and therefore are interesting in their own right (see Section V for a couple of examples). Second, by breaking our analysis into two parts, we can gain a better understanding of what the issues are in the solutions: specifically, how to choose a good initialization and what the relationships are between the network topology and the uniqueness of the solution. Third, we empirically noticed that the complete cost function φ has multiple local minima and is difficult to minimize if we start from a random configuration. On the other hand, minimizing the functions φ_R and φ_T is easier and gives a way to obtain a good initialization for the minimization of φ.

A. Estimation of the Rotations

In this section we will derive a distributed algorithm for estimating the rotational part R_i of the absolute pose g_i. The basic idea is to use the framework of consensus algorithms, but with a Riemannian gradient descent on the space of rotations as opposed to the simple Euclidean gradient descent. At a high level, each node k will first compute the gradient of φ_R with respect to its rotation R_k. This, in turn, will give a vector in the tangent space of R_k defining the geodesic along which node k will need to update its rotation. All the nodes will then compute the gradients and the updates at the same time and independently. Finally, the algorithm will iterate between these two steps until convergence.

More specifically, the gradient of φ_R with respect to one of the rotations R_k, can be computed as

$$\operatorname{grad}_{R_k} \varphi_R = -R_k \sum_{k \rightsquigarrow i} \log(R_k^\top R_i \tilde{R}_{ki}^\top) + \log(R_k^\top R_i \tilde{R}_{ik}). \quad (17)$$

Notice that the computation of $\operatorname{grad}_{R_k} \varphi_R$ involves a sum over nodes i which are neighbors of node k only, thus the gradient can be computed in a distributed fashion.

Now, let $R_k(l)$ denote the estimate of R_k at the l-th iteration. Then, $R_k(l)$ is updated by moving along the geodesic in the direction $-\operatorname{grad}_{R_k(l)}$ with a step-size of ε, *i.e.*,

$$R_k(l+1) = \exp_{R_k(l)}\big(-\varepsilon \operatorname{grad}_{R_k(l)}\big). \quad (18)$$

Initialization of the rotations. As with all non-linear optimization algorithms, we need a starting point, *i.e.*, some values for the $R_i(0)$'s. Since initially the localization of the network is unknown, we have to start with an arbitrary initialization. By following the gradient for minimizing the cost function we will reach a local minimum. The problem here is that φ_R has multiple local minima that do not correspond to the correct localization, even when the relative measurements are without noise (this can be easily verified empirically). The basic idea is then to find an initial set of rotations $\{R_i\}$ that are close enough to an optimal solution. We propose two options for this task:

1) Choose any node as a reference (say, node 1) and find a spanning tree that provides paths ℓ_{1i} from node 1 to any other node i. Set $R_1(0) = I$ and set $R_i(0) = \tilde{g}_{\ell_{1i}} R_1(0)$ (*i.e.*, propagate the noisy transformations along the branches of the spanning tree).
2) Modify the cost function φ_R by using an alternative distance in $SO(3)$ such that the optimal solution can be easily found by gradient descent. Then, find the $R_i(0)$ by minimizing this new cost function.

Notice that the first approach is faster, but it is not fully distributed. First, a reference node needs to be chosen in a distributed fashion. Second, the chosen reference node could fail. On the other hand, the second approach is completely distributed and there is no need of choosing a reference. In the following, we explore this second option in more detail.

We propose to first substitute the geodesic distance in $SO(3)$ by the Frobenius norm. This gives us the new cost function:

$$\varphi'_R = \frac{1}{2}\sum_{i\rightsquigarrow j}\|R_j - R_i\tilde{R}_{ij}\|_F^2. \tag{19}$$

We can then try to find a localization by minimizing φ'_R using Riemannian gradient descent as before. The gradient for the update equations for the rotation at a node k can be computed using (4) with the Euclidean gradient as

$$\frac{\partial\varphi'_R}{\partial R_k} = \sum_{k\rightsquigarrow i}(R_k - R_i\tilde{R}_{ki}^\top) + \sum_{i\rightsquigarrow k}(R_k - R_i\tilde{R}_{ik}). \tag{20}$$

It is easy to check that, in the case of perfect data, the global minimum of φ'_R corresponds to the correct value of the rotations. Note that the global minimum is not unique: there is an entire family of optimal solutions that can be obtained by multiplying from the left with the same rotation each R_i (since rotations are unitary matrices, the value of φ'_R does not change under this kind of transformations). This behavior is expected and it corresponds to the ambiguity in the choice of a global reference frame for the rotations.

We experimentally found that the function φ'_R is easier to minimize than φ_R. In all the cases we tried, the gradient descent algorithm found a solution extremely close to the ground truth. From this and from the fact that φ'_R is convex in $\mathbb{R}^{3\times 3}$ we conjecture that φ'_R does not have local minima. In our future research we aim to formally prove this conjecture.

The final strategy we propose is to use the algorithm with the linearized cost function in (19) to initialize the non-linear optimization. In this way, we have a fully distributed method for obtaining an initial estimate of the absolute rotations R_i given the noisy measurements $\tilde{R}_{ij}$.

B. Estimation of the Translations

For the estimation of the absolute translations T_i, we propose to use an approach similar to what we did for the rotations. The main idea is to solve the program

$$\begin{gathered}\min \varphi_T \\ \text{subject to } \lambda_{ij} \geq 1 \quad (i,j) \in E\end{gathered} \tag{21}$$

using projected gradient descent. For this purpose, we need to compute the gradient with respect to both T_k (for each node) and λ_{lk} (for each edge), as

$$\frac{\partial\varphi_T}{\partial T_k} = \sum_{k\rightsquigarrow i} 2(T_k - T_i) + \lambda_{ki}R_k\tilde{t}_{ki} - \lambda_{ik}R_i\tilde{t}_{ik}, \tag{22}$$

$$\frac{\partial\varphi_T}{\partial\lambda_{lk}} = \lambda_{lk} - (T_l - T_k)^\top R_k\tilde{t}_{kl}. \tag{23}$$

We can then update the $\{T_i\}$ and λ_{ij} using gradient descent, except that we need to enforce the constraint on the scales $\lambda_{ij} \geq 1$.

Notice that the global (unconstrained) optima of φ_T (solutions with $\lambda_{ij} = 0$, $(i,j) \in E$) are not in the feasible set of (21). Therefore, at least one of the constraints needs to be active at the optimal solution, hence the optimal λ_{ij}'s are such that $\min_{(i,j)\in E}\lambda_{ij} = 1$. Notice also that the inequalities in (21) define a convex set $\mathcal{S}$ (the intersection of half-spaces). It can be shown that the projection step of the gradient update on $\mathcal{S}$ can be computed separately for each edge, resulting in the new update equation

$$\lambda_{lk}(l+1) = \max\{1, \lambda_{lk}(l) - \varepsilon\frac{\partial\varphi_T}{\partial\lambda_{lk}}\}. \tag{24}$$

Since the function φ_T is convex and so is the feasible set $\mathcal{S}$, we are guaranteed to obtain a global minimum with any initialization and an appropriate choice of step-size ε.

Choice of the step-size ε. In any algorithm based on gradient descent the choice of the step-size is important. In this section we will show that the step-size for minimizing φ_T can be chosen by considering the degree of the nodes alone. Moreover, this step-size can be computed in a distributed way.

Let us begin by considering the minimization of a quadratic form $\varphi = \frac{1}{2}\|My\|^2$, where $M \in \mathbb{R}^{q\times p}$ is an arbitrary real valued matrix and $y \in \mathbb{R}^p$ is a vector of variables. A quadratic cost function restricted to a line (in the direction of the gradient) is a parabola and, intuitively, the maximum allowed step-size is determined by the maximum possible curvature of the parabola. This maximum curvature (and consequently the maximum step-size) is related to the the maximum eigenvalue of the matrix $M^\top M$. Unfortunately, eigenvalues are global quantities that are hard to estimate in a distributed way. For this reason we use the Geršgorin Discs and the Geršgorin Theorem (summarized below) to show that the maximum eigenvalue can be substituted with the maximum of the absolute row sum of $M^\top M$. Moreover, we will show that for our particular cost function φ_T, each row sum of $M^\top M$ can be computed independently at a different node. This will allow us to compute the step-size in a distributed way and in a small number of iterations. In addition, our results will generalize the upper bound for the step-size in the case of classical consensus, where it is known that $\varepsilon \in (0, \frac{1}{\Delta_G})$. We will now summarize the Geršgorin theorem and formally state our result.

Definition 2 (Geršgorin discs): Given a matrix $A \in \mathbb{R}^{p\times p}$, the Geršgorin discs S_i, $i = 1,\dots,p$ are defined as

$$S_i = \{y \in \mathbb{C} : |y - (A)_{i,i}| \leq \sum_{j=1,j\neq i}^{p} |(A)_{i,j}|\}, \tag{25}$$

where $(A)_{i,j}$ represents the i,j-th entry of the matrix A.

Theorem 1 (Geršgorin theorem): The eigenvalues of $A \in \mathbb{R}^{p\times p}$ are contained in the union of its Geršgorin discs.

Theorem 2: Let $\varphi = \frac{1}{2}\|My\|^2$, then $\nabla_y\varphi = M^\top My$. The gradient descent iteration $y(l+1) = y(l) - \varepsilon\nabla_y\varphi$ converges for $\varepsilon \in \Big(0, \min_i 2\big(\sum_j \big|(M^\top M)_{i,j}\big|\big)^{-1}\Big)$

The proof of this Theorem can be found in [14]. The basic idea is to first use the eigenvalue decomposition of

$M^\top M$ and show that the cost function decreases at each step only if $\varepsilon < \frac{2}{\sigma_{max}(M^\top M)}$, where $\sigma_{max}(M^\top M)$ represents the maximum eigenvalue of $M^\top M$. Using Geršgorin's Theorem we can bound the region where the eigenvalues of $M^\top M$ must lie, and hence find for which values of ε the algorithm converges.

In order to apply Theorem 2 to our algorithm for minimizing φ_T, we have to generalize it to the case where we add the projection step on the feasible set $\mathcal{S}$. This is done in Lemma 1 of [14], which shows that the same bounds apply. The proof of the lemma breaks the projected gradient step in two parts: one in the interior of the feasible set $\mathcal{S}$, the other along its boundary. Then it can be shown that the cost function will decrease after each part of each step.

For our particular function φ_T, M and y are defined as

$$M = \begin{bmatrix} & \operatorname{diag}\{(R_i t_{ij})_1\} \\ I \otimes C & \operatorname{diag}\{(R_i t_{ij})_2\} \\ & \operatorname{diag}\{(R_i t_{ij})_3\} \end{bmatrix} \in \mathbb{R}^{3|E|\times(3|V|+|E|)}, \quad (26)$$

$$y = \left[(T_1)_1 (T_2)_1 \cdots (T_{|V|-1})_3 (T_{|V|})_3 \cdots \lambda_{ij} \cdots\right]^\top, \quad (27)$$

where the notation $(v)_i$ indicates the i-th component of the vector v, "$\otimes$" is the Kronecker product and C is the incidence matrix of the graph G. If we explicitly write down the matrix $M^\top M$, the sum of each one of its rows involves quantities that can be locally computed by a single node. More formally, we have the following result.

Theorem 3: Define

$$\rho_{i,d} = 2\deg(i) + 2\sum_{i \rightsquigarrow j} \left|(R_i t_{ij})_d\right| \quad (28)$$

for all $i, j \in V$, $d \in \{1,2,3\}$ and

$$\rho_{ij} = \sum_{d=1}^{3} (R_i t_{ij})_d^2 + 2\sum_{d=1}^{3} \left|(R_i t_{ij})_d\right| \quad (29)$$

for all $(i,j) \in E$. From these, define also

$$\varepsilon_i = \frac{2}{\max_{j,d}\{\{\rho_{i,d}\},\{\rho_{ij}\}\}} \quad \forall i \in V \quad (30)$$

Then the projected gradient descent algorithm for minimizing φ_T is guaranteed to converge if $\varepsilon \in (0, \min_i \varepsilon_i)$.

Notice that the quantity ε_i can be computed locally at node i and the operation $\min_i \varepsilon_i$ can be computed with a consensus-like algorithm in a finite number of iterations. Hence we have a method that the nodes can use to automatically choose ε. The detailed steps for the proof of Theorem 3 can be found in [14].

As a final remark, the quantities $\rho_{i,d}$ and ρ_{ij} depend on the rotations R_i. By noticing that $|R_i t_{ij}| \leq 1$, we can also give a bound that is slightly less sharp, but depends only on the degrees of the nodes, as stated in Corollary 1. The proof of the corollary is straightforward by substitution.

Corollary 1: . The projected gradient descent algorithm converges if $\varepsilon \in (0, 2(\max\{9,\, 4\Delta_G\})^{-1})$

Translation ambiguity. From the analysis above, we can guarantee that the projected gradient descent algorithm will converge to a global minimum. However, we can have problems with the uniqueness of the solution. By the definition of the incidence matrix, we have that $C\mathbf{1} = 0$. Hence, the matrix M in (26) has a null space of dimension at least three, which is spanned by

$$v_1 = \begin{bmatrix} \mathbf{1}_{|V|} \\ \mathbf{0}_{|V|} \\ \mathbf{0}_{|V|} \\ \mathbf{0}_{|E|} \end{bmatrix}, \quad v_2 = \begin{bmatrix} \mathbf{0}_{|V|} \\ \mathbf{1}_{|V|} \\ \mathbf{0}_{|V|} \\ \mathbf{0}_{|E|} \end{bmatrix} \quad v_3 = \begin{bmatrix} \mathbf{0}_{|V|} \\ \mathbf{0}_{|V|} \\ \mathbf{1}_{|V|} \\ \mathbf{0}_{|E|} \end{bmatrix} \in \mathbb{R}^{3|V|+E}. \quad (31)$$

Here $\mathbf{0}_D$ and $\mathbf{1}_D$ are vectors in $\mathbb{R}^D$ with all the entries set to zero or one, respectively. The three vectors correspond to translations by the same amount of all the nodes along the three axes x, y and z. Note that these three vectors are orthogonal to the constraints on the λ_{ij}. This means that if $\hat{y}$ is a solution to the constrained minimization problem, then any linear combination of $\hat{y}$, v_1, v_2 and v_3 will also be a valid solution. Again, this behavior is expected, because it corresponds to the choice of a global reference frame for the translations. The question is how the translational ambiguity can be fixed by the distributed algorithm. It is not hard to check that, similarly to classical consensus, the average of the translations is preserved between iterations, *i.e.*,

$$\sum_{i=1}^{N} T_i(l) = \sum_{i=1}^{N} T_i(l+1). \quad (32)$$

This means that the translational ambiguity can be resolved by choosing the initial average of the translations to be zero, *e.g.*, by setting $T_i = 0\ \forall i \in V$.

In general, we assume that the null space of M is of dimension at most four, which guarantees that solving (21) will recover the desired solution. More details can be found in [14].

C. Complete Estimation

The final algorithm for minimizing φ in (10) is a straightforward combination of the methods presented in Sections III-A and III-B. Specifically, we proceed as follows:

1) Find consistent initial rotations by minimizing the linearized cost φ'_R and/or φ_R.
2) Find consistent initial translations and scales by minimizing φ_T while keeping the rotations fixed.
3) Refine all the estimates by minimizing φ.

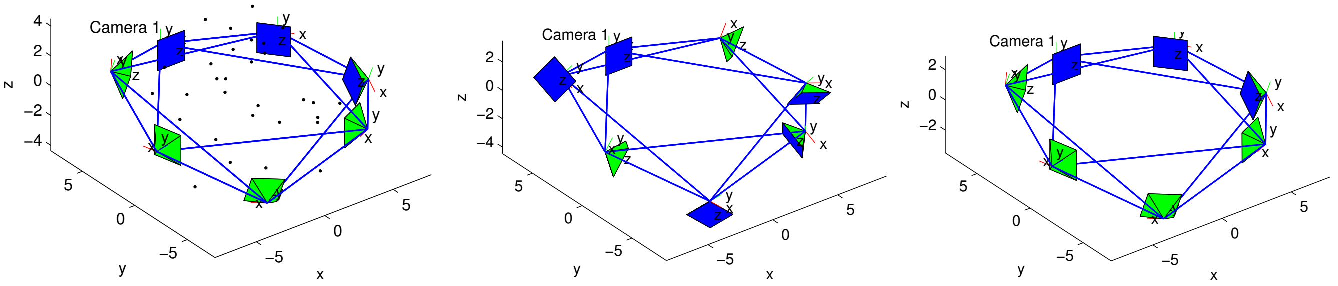
Notice that in step 3) the gradient of φ with respect to R_k contains an additional term coming from φ_T, *i.e.*,

$$\begin{aligned} \operatorname{grad}_{R_k} \varphi &= \operatorname{grad}_{R_k} \varphi_R \\ &+ \sum_{k \rightsquigarrow i} \lambda_{kj}\left((T_i - T_k)\tilde{t}_{kj}^\top - R_k \tilde{t}_{kj}(T_i - T_k)^\top R_k\right). \quad (33) \end{aligned}$$

The updates then follow the gradient as before.

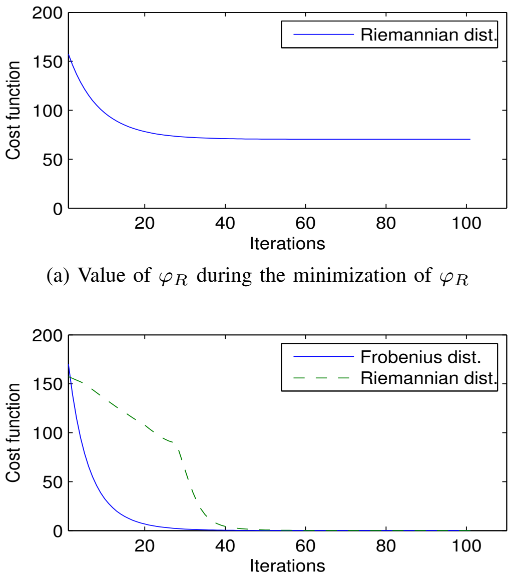
IV. EXPERIMENTS

In this section we will present some synthetic experiments to illustrate the behavior of our algorithms. We use a non-planar network with $N = 7$ camera nodes connected in



(a) The synthetic camera network setup (b) Solution found by minimizing φ_R (c) Solution found by minimizing φ'_R

Fig. 1: Pictorial comparison between the ground truth and the solutions found by minimizing the different cost functions



(a) Value of φ_R during the minimization of φ_R

(b) Value of φ'_R and φ_R during the minimization of φ'_R

Fig. 2: Experiment illustrating a situation where a local minimum is encountered while optimizing φ_R

Rotation errors (degrees, zero is the minimum)

Noise	0px	1px	2px	3px
Initial	0.000 (0.000)	2.776 (0.584)	3.926 (1.166)	4.809 (1.747)
Final	0.000 (0.000)	0.131 (0.004)	0.262 (0.015)	0.393 (0.035)

Normalized translation errors (degrees, zero is the minimum)

Noise	0px	1px	2px	3px
Initial	0.000 (0.000)	0.110 (0.006)	0.221 (0.025)	0.331 (0.057)
Final	0.000 (0.000)	0.097 (0.004)	0.194 (0.017)	0.291 (0.039)

Scale errors (geometric variance, one is the minimum)

Noise	0px	1px	2px	3px
Final	1.000	1.002	1.003	1.005

TABLE I: Average and variance (in parenthesis) of errors for all the edges starting from point correspondences and comparing with the ground truth.

a 4-regular graph, as pictorially represented in Figure 1a. All the cameras have a focal length of $f = 1$, are roughly distributed in a circle of radius $8f$ around the origin, and are facing toward the center of the scene. The 3-D structure is represented by 30 points randomly distributed in a cube with side $4.5f$. With these proportions, the projected points occupy about 75% of the normalized image frame in each camera. The projected points are corrupted with zero-mean Gaussian noise with standard deviation of 0, 1, 2, and 3 pixels on an image of 1000×1000 pixels. Given the point correspondeces, we use the eight-point algorithm (see [13] for details) to get an estimate of relative rotation and translation (up to scale).

In our first experiment, we show that approximating φ_R with φ'_R gives a better initialization of the rotations. We consider an ideal situation where the relative transformations (R_{ij}, T_{ij}) are noise free and, thus, we would expect to have a perfect recovery of the rotations. We randomly intialize the rotations R_i at each node and then we use the non-linear optimization algorithm given by (18). As we can see in Figures 1b and 2a the result is far from optimal and the function φ_R converges to a local minimum. Note that, for the sake of comparison, the global rotation ambiguity in the figure has been fixed by aligning the first camera with the ground-truth. On the other hand, the minimization of φ'_R avoids this problem and gives a good (typically close to optimal) solution, as we can see in Figures 1c and 2b. In Figure 2b we also plot the value of φ_R after each iteration of the gradient descent minimization of φ'_R. It is interesting to see that, though we are minimizing φ'_R, the value φ_R also decreases. Experimentally, we noticed that refining the solution from φ'_R by minimizing φ_R gives only marginal improvements. In other words, our conclusion is that minimizing φ'_R already gives a good initialization for the next step (initialization of translations and scales).

We now evaluate the performance of our algorithm as a function of noise. We optimize over the rotations with the Frobenius distance (600 iterations), then over the translations and scales (3000 iterations) and finally we optimize φ over all the variables (100 iterations). As we said before, optimizing the rotations using the Riemannian distance does not give significant benefits for the task of just finding a consistent initialization.

We repeated the experiment 100 times for each level of noise. For each trial and for each edge in the network, we collected the angle between the estimated and ground-truth rotations, the angle between the estimated and ground-truth translations and the geometric variance between estimated and ground-truth scales. For the scales, the idea is that all the ratios between the estimated and the ground-truth scale for each edge should be the same. To quantitatively determine how far these ratios are from being all equal to each other, the geometric variance (as opposed to arithmetic variance) is the most natural choice. The average and variance of the errors on rotation, translation and scales over all the edges before and after the optimization is reported in Table I. By looking at the results, we can notice that exploiting the network topology improves the estimates of the relative transformation at each edge (both in terms of the mean and the variance). Moreover, in the process we also recover the relative scales between the edges (which were unknown from the measurements only) and a localization consistent with Definition 1.

V. CONCLUSIONS AND FUTURE WORK

We presented distributed algorithms to optimally solve the localization problem for camera networks. Starting from image point correspondences, we used techniques from computer vision to get an initial estimate of the relative rotation and translation (up to scale) between cameras with intersecting fields of view. By minimizing a global cost function we derived a distributed algorithm to recover a better, consistent estimate of the relative poses. We were able to recover also the relative scales between the edges and to fix the global ambiguities inherent to the problem. We also analyzed the behavior and potential pitfalls of our approach (for instance, how to initialize the estimation of the rotations). In addition, we proposed an automatic, distributed method for choosing the step-size ε (in the estimation of the translations).

We developed our algorithms by considering the specific case of localization for camera networks, but their application could be broader. First, our results carry over to the 2-D case with the only modification that we need to consider rotations in $SO(2)$ and rigid body transformations in $SE(2)$ instead of the 3-D counterparts. Then, the problem of recovering only the rotations or only the translations corresponds to other practical instances of localization in sensor networks. For instance, if one has only bearing measurements, they can be converted to rotations and then our algorithm for optimizing over the rotations could be applied. Finally, one can easily incorporate additional information such as the presence of beacons (nodes with a known pose) and distance measurements. In this case some of the quantities (*e.g.*, the R_i, T_i of the beacons) are fixed and we would need to optimize only on the remaining unknowns. All these options, however, are out of the scope of this paper.

In future work, we plan to first analyze rigorously our initialization method (by finding the critical points of φ'_R) and the choice of the step-size for the optimization over the rotations. Then, we plan to extend our method to the problem of distributed bundle adjustment (*i.e.*, the minimization of the error between the noisy projected point images and the corrected images obtained from the estimated poses).

REFERENCES

[1] U.A. Khan, S. Kar, and J.M.F. Moura, "Distributed sensor localization in random environments using minimal number of anchor nodes," *IEEE Transactions on Signal Processing*, vol. 57, no. 5, pp. 2000–2016, May 2009.

[2] J. Aspnes, W. Whiteley, and Y. R. Yang, "A theory of network localization," *IEEE Transactions on Mobile Computing*, vol. 5, no. 12, pp. 1663–1678, 2006.

[3] S. Funiak, C. Guestrin, M. Paskin, and R. Sukthankar, "Distributed localization of networked cameras," in *International Conference on Information Processing in Sensor Networks*, 2006, pp. 34–42.

[4] 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.

[5] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge, 2nd edition, 2004.

[6] 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.

[7] Craig Gotsman and Yehuda Koren, "Distributed graph layout for sensor networks," in *Internat. Symposium on Graph Drawing*, 2004, pp. 273–284.

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

[9] G. Piovan, I. Shames, B. Fidan, F. Bullo, and B. Anderson, "On frame and orientation localization for relative sensing networks," in *IEEE Conference on Decision and Control*, 2008, pp. 2326–2331.

[10] R. Tron, R. Vidal, and A. Terzis, "Distributed pose averaging in camera networks via consensus on SE(3)," in *International Conference on Distributed Smart Cameras*, 2008.

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

[12] C. Altafini, "The De Casteljau algorithm on SE(3)," *Nonlinear control in the Year 2000*, vol. 258, pp. 23–34, 2000.

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

[14] R. Tron and R. Vidal, "Distributed image-based 3-D localization of camera sensor networks," Tech. Rep., Johns Hopkins University, 2009.