Synchronization of Group-labelled Multi-graphs

Andrea Porfiri Dal Cin[1], Luca Magri[1], Federica Arrigoni[2], Andrea Fusiello[3] and Giacomo Boracchi[1]
[1]DEIB - Politecnico di Milano (Italy) [2]DISI - University of Trento (Italy) [3]DPIA - University of Udine (Italy)

Abstract

Synchronization refers to the problem of inferring the unknown values attached to vertices of a graph where edges are labelled with the ratio of the incident vertices, and labels belong to a group. This paper addresses the synchronization problem on multi-graphs, that are graphs with more than one edge connecting the same pair of nodes. The problem naturally arises when multiple measures are available to model the relationship between two vertices. This happens when different sensors measure the same quantity, or when the original graph is partitioned into sub-graphs that are solved independently. In this case, the relationships among sub-graphs give rise to multi-edges and the problem can be traced back to a multi-graph synchronization. The baseline solution reduces multi-graphs to simple ones by averaging their multi-edges, however this approach falls short because: i) averaging is well defined only for some groups and ii) the resulting estimator is less precise and accurate, as we prove empirically. Specifically, we present MULTISYNC*, a synchronization algorithm for multi-graphs that is based on a principled constrained eigenvalue optimization.* MULTISYNC *is a general solution that can cope with any linear group and we show to be profitably usable both on synthetic and real problems.*

1. Introduction

Many tasks in Computer Vision can be formulated as the *synchronization* [50] of group-labelled graphs: given a network of nodes labelled with unknown elements of a group Σ, it is required to estimate them from a collection of noisy relative measurements expressed as ratios (or differences) attached to edges. A prominent example is when the nodes of the graph are sensors and the goal is to recover the unknown attitude and location of each sensor in a common reference frame. In this case, labels are in the Special Euclidean Group $\Sigma = \mathrm{SE}(3)$, and the pairwise measurements are relative orientations. Depending on the group, the synchronization formulation can be exploited to model other relevant problems in Computer Vision, such as structure from motion, simultaneous localization and mapping (SLAM), multi-view matching and image mosaicking.

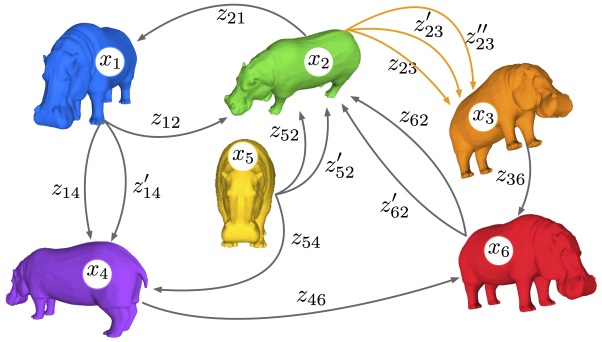


Figure 1: A multi-graph is a graph that admits multiple edges between its nodes. Nodes correspond to unknown group elements $x_i \in \mathrm{SO}(3)$ and edges correspond to known relative measures. A multi-edge with cardinality 3, resulting for instance from different estimates of relative transformations, is depicted in orange.

Traditionally, synchronization is defined on a *simple* graph, *i.e.*, a graph where vertex pairs are connected by at most one edge. In this work, instead, we deal with the case where *multiple* measurements are available for the same pair of nodes (multi-edge) and synchronization is performed on a *multi-graph* rather than on a simple graph (see Fig. 1). This general multi-graph synchronization framework allows us to naturally account for multiple measurements between the same pair of nodes, which often happens in many applications. In SLAM, for instance, multiple sensors (cameras, IMU, GPS, ...) can estimate the 6 d.o.f motion of a vehicle. Another scenario where multi-graphs naturally arise is in large-scale problems, where an original graph is partitioned into smaller sub-graphs that are solved independently to save computing time. In this context, the cut-edges connecting vertices from different sub-graphs give rise to multi-edges and yield a multi-graph synchronization problem, as will be clarified in Sec. 5. This partitioned approach not only reduces memory and processing time, but also enable multi-threading and parallelism. In addition, it has relevant implications in terms of privacy since it allows for data to be segregated so that each processing node sees only a portion of the data (see, *e.g.*, [25]).

The naive synchronization solution on multi-graphs –

henceforth named *edge averaging* – consists in reducing multi-edges to simple ones by averaging their measurements. This suffers from several shortcomings. First of all, averaging is well defined only for some groups: while it is possible to average rotations [32], there is not a principled solution for homographies. Secondly, even when it is possible to average multi-edges, the resulting estimator has sub-optimal statistical properties. As an example, consider the problem of estimating a scale factor s that links two matrices with noisy entries: $\mathsf{A} = s\mathsf{B}$. The optimal estimate is the least squares solution $s = \operatorname{tr}(\mathsf{B}^\top\mathsf{A})/\operatorname{tr}(\mathsf{B}^\top\mathsf{B})$ which is different from, *e.g.*, taking the average of the entry-wise division $\mathsf{A}./\mathsf{B}$. Our experiments confirm that this intuition holds also for synchronization.

Contributions. Our solution – named MULTISYNC– approaches synchronization of group-labelled multi-graphs from a new perspective: rather than averaging measures in order to collapse a multi-edge to a simple one, it expands the multi-graph replicating nodes involved in multi-edges and enforces identity constraints between replicated nodes. This leads to a constrained optimization problem for which we derive a general closed-form spectral solution, that can be applied to graphs labelled with any linear group. Our experiments, performed on both synthetic and real data sets, demonstrate that MULTISYNC outperforms edge averaging in terms of accuracy and precision. In the context of partitioned problems, our solution strikes a good balance between accuracy and complexity, as opposed to performing synchronization on the whole graph. To summarize, our contribution is three-fold:

- we present, for the first time, a formal definition of the synchronization of multi-graphs, which is a significant extension of the synchronization of simple graphs;
- we derive MULTISYNC, a practical algorithm for solving a synchronization problem on a multi-graph, which is based on an expansion algorithm coupled with a constrained spectral solution to deal with replicated nodes;
- we demonstrate how the multi-graph framework can be conveniently used to partition classical synchronization tasks, achieving a good trade-off between accuracy and complexity.

Outline. The paper is organized as follows. Sec. 2 reviews previous works. Sec. 3 provides the theoretical footing and presents our solution. Sec. 4 reports synthetic experimental results and Sec. 5 describes a possible application of multi-graph synchronization: partitioned synchronization. Conclusions are drawn in Sec. 6.

2. Related Work

The synchronization problem derives its name from clock synchronization [26], and has been extensively investigated in the Computer Vision community (see [3] for a recent survey). Depending on the chosen group, specific instances of the problem are obtained, which relate to different applications.

Synchronization over the Special Orthogonal Group (*i.e.*, $\Sigma = \mathrm{SO}(3)$) is referred to as *rotation synchronization*, *multiple rotation averaging* [32] or *rotation optimization* [57]. Existing techniques include least squares [37], spectral decomposition [50], the Weiszfeld algorithm [31], the Levenberg-Marquardt algorithm [19], Lie-group optimization [17], semi-definite programming [58, 22, 21], distributed optimization [56, 52], low-rank decomposition [6], Riemannian optimization [12], deep learning [42] and message passing [49]. When the Special Euclidean Group (i.e., $\Sigma = \mathrm{SE}(3)$) is considered, it results in *rigid-motion synchronization*, *motion averaging* or *pose-graph optimization*. Existing techniques include spectral decomposition [9, 7], Lie-group optimization [29], diffusion over dual quaternions [54], Riemannian optimization [55], semidefinite programming [44, 43], distributed optimization [53], group contraction [39], Bayesian optimization [14] and deep learning [33, 27]. Both rotation and rigid-motion synchronizations can be applied to structure from motion [40], registration of 3D point clouds [30] and SLAM [16, 23].

When considering the Symmetric Group (*i.e.*, $\Sigma = \mathrm{Sym}(d)$), we get *permutation synchronization*, which finds application in multi-view matching. Existing approaches include spectral decomposition [41, 47, 10], Gauss-Seidel relaxation [60], distributed optimization [35], and Riemannian optimization [13]. Other synchronization problems concern the Special Linear Group (*i.e.*, $\Sigma = \mathrm{SL}(3)$), which is used to represent homographies in image mosaicking [46], and the General Affine Group (*i.e.*, $\Sigma = \mathrm{GA}(3)$), which has been used to solve for global color matching [45].

All the aforementioned approaches, with few exceptions discussed in Remark 1, are limited to deal with *simple graphs*. There are cases, however, where *multi-graphs* naturally arise, the most prominent one being partitioned synchronization, which motivated our research. For large-scale graphs, synchronization approaches, in particular robust ones, may incur in severe computational issues as complexity increases with respect to the number of edges in the graph. An effective remedy is to partition the original measurement graph into smaller sub-graphs (called *patches* in this context) that can be easily synchronized, then one can combine the labeling of each patch to obtain a consistent labeling of the original graph. This last step can be seen as a synchronization on a multi-graph that resolves the ambiguity of the local patch-wise solutions (see Fig. 5).

The idea of partitioning a synchronization problem into smaller sub-problems (which are easier to solve) is present in a number of works in the context of structure from motion [11, 24], simultaneous localization and mapping [36] and motion segmentation [4]. Such techniques, however, do

not exploit the multi-graph formulation – that is introduced in this paper for the first time in the literature – but they (either implicitly or explicitly) turn the multi-graph into a simple one by averaging multi-edges. A few other partitioned pipelines are present in the literature (e.g., in the context of 3D reconstruction [62, 18, 61] or sensor network localization [20]), which, however, do not address synchronization problems, as they exploit additional information (such as coordinates of 3D points) alongside relative measurements.

3. Synchronization on multi-graphs

In this section we introduce the theoretical framework of synchronization on group-labelled multi-graphs (Sec. 3.1), then we present our algorithm (named MULTISYNC) for linear groups. At a high level, our method consists in the following main steps:

- **graph expansion:** the multi-graph is expanded to a simple graph by creating replicas of vertices (Sec. 3.2);
- **constrained optimization:** a constrained eigenvalue problem is solved, to address synchronization with duplicated vertices sharing the same label (Sec. 3.3).

3.1. Formulation

A *multi-edge* in a directed graph is a set of two or more edges with both the same tail vertex and the same head vertex. A graph that allows multi-edges is called a *multi-graph* (see Fig. 1 for an example), which extends the definition of graphs as follows:

Definition 1 (Multi-graph [15])**.** *A multi-graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, s, t)$ is a directed graph without loops, where $\mathcal{V}$ is the set of vertices, $\mathcal{E}$ is the set of edges, $s\colon \mathcal{E} \to \mathcal{V}$ is a function that maps an edge to its source vertex, and $t\colon \mathcal{E} \to \mathcal{V}$ is function mapping an edge to its target vertex.*

Definition 2 (Multi-edge [15])**.** *Given a multi-graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, s, t)$, the multi-edge $E(i, j)$ is the set:*

$$E(i,j) = \{e \in \mathcal{E} \colon s(e) = i \wedge t(e) = j\}. \tag{1}$$

Strictly speaking, every simple graph is also a multi-graph, but hereafter we consider them as different objects: a simple graph refers to a graph having cardinality $|E(i,j)| \leq 1$ for all $i, j \in \mathcal{V}$, whereas multi-graphs must have at least one pair of vertices (i, j) with $|E(i,j)| > 1$. The elements of a group $(\Sigma, *)$ can be used to label the edges of a multi-graph, yielding a group-labeled multi-graph:

Definition 3 (Group-labeled multi-graph)**.** *A Σ-labeled multi-graph is a tuple $\Gamma = (\mathcal{V}, \mathcal{E}, s, t, z)$ where $\mathcal{G} = (\mathcal{V}, \mathcal{E}, s, t)$ is a multi-graph and $z\colon \mathcal{E} \to \Sigma$ is the edge-labeling function. The edge set $\mathcal{E}$ satisfies the following property: if $e \in \mathcal{E}$ with $s(e) = u \wedge t(e) = v$, then $e' \in \mathcal{E}$ with $s(e') = v \wedge t(e') = u$, and z satisfies:*

$$z(e) = z(e')^{-1}. \tag{2}$$

Eq. (2) means that each edge connecting a pair of vertices (u, v) has a corresponding edge connecting (v, u), which is labelled with the inverse transformation.

Definition 4 (Consistent labeling)**.** *Let $\Gamma = (\mathcal{V}, \mathcal{E}, s, t, z)$ be a Σ-labelled multi-graph and let $x : \mathcal{V} \to \Sigma$ be a vertex labeling. We say that x is a consistent labeling if and only if the following condition holds:*

$$z(e) = x(i) * x(j)^{-1} \tag{3}$$

$\forall i, j \in \mathcal{V}$ and $\forall e \in \mathcal{E}$ such that $(s(e), t(e)) = (i, j)$.

Eq. (3) is also referred to as the *consistency constraint* since it means that, for any pair (i, j) of vertices, the labels of the edges connecting i and j must be equal to the ratio of the vertex labels $x(i)$ and $x(j)$. There is an inherent *ambiguity* in the synchronization problem: if $x : \mathcal{V} \to \Sigma$ satisfies Eq. (3), then also $y(i) = x(i) * w$ is a solution, for any (fixed) $w \in \Sigma$.

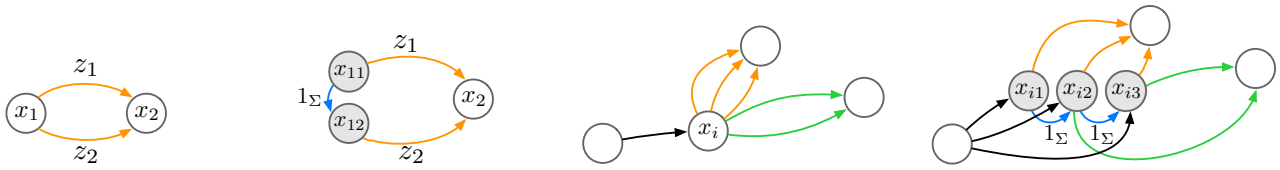
In the presence of noise the consistency constraint will not be satisfied exactly, thus the task of **multi-graph synchronization** is to find the unknown vertex labelling such that Eq. (3) is approximately satisfied, e.g., in the least-squares sense. In this case, multi-edges represents *redundant* relative measures to effectively compensate errors.

Remark 1. In this paper we concentrate on the spectral solution due to its generality, for it can be applied to any matrix group (e.g., rotations [50], rigid-motions [9, 7], homographies [8]) and also to semigroups (e.g., partial permutations [38]) and sets that have a poorer structure (e.g., binary matrices [5]). Observe that existing synchronization techniques based on spectral solutions cannot be applied to multi-graphs straightforwardly, because multiple edges cannot be represented in an adjacency matrix (see Sec. 3.3). Therefore the multi-graph has to be transformed into a simple graph, as will be clarified later. There are synchronization techniques [37] based on the following constraint:

$$z(e) * x(j) = x(i) \tag{4}$$

which gives rise to a trace-minimization problem [57] where multiple edges could be potentially taken into account as additional terms. These approaches, however, are limited to groups, since the above constraint is equivalent to Eq. (3) only in a group; by contrast, our method has the potential to be extended also to semi-groups, being based on the spectral solution. In addition, some techniques based on non-linear optimization (e.g., [19, 17, 55]) could be adapted to multi-graphs by letting edges represent sums of cost terms. These methods, however, are carefully designed for a chosen group, while our method offers an approximate closed-form solution for any matrix group.

Remark 2. The naive strategy of *edge averaging* – where a multi-edge is collapsed to a single edge by averaging labels



(a) Minimal multi-graph (b) Expansion (c) A node with two multi-edges (d) Expansion

Figure 2: Multi-graph expansion: the process of expanding a multi-graph into a simple graph without multi-edges by replicating specific vertices (shaded nodes) and by introducing additional constraints to preserve both absolute (global) and relative (pairwise) information between the nodes in the graph.

– produces a sub-optimal solution, as it does not exploit the redundancy of multiple measurements. In addition, averaging is not always well defined: while averages in $\mathrm{SO}(3)$ are well studied by [32], similar results are not known for homographies. For this reason, we follow a different strategy: instead of reducing the multi-graph by collapsing multi-edges, we first expand it by creating replicas of vertices, as illustrated in Sec. 3.2. Then, we recover a consistent labeling by a constrained spectral solution where duplicated vertices share *exactly* the same label, as described in Sec. 3.3.

3.2. Multi-graph expansion

We present an iterative greedy algorithm to expand a multi-graph into a simple graph retaining all the pairwise information. The rationale is that each multi-edge can be expanded into a number of simple edges equal to its cardinality, by properly replicating source or target vertices.

Fig. 2a represents a minimal multi-graph of two vertices x_1 and x_2 connected by a single multi-edge with cardinality 2. In order to expand this into a simple graph, it is sufficient to replicate only one of the two vertices, thus turning the multi-edge into 2 edges connecting the replicas to the non-replicated vertex. An additional edge labeled by 1_Σ (blue edge in Fig. 2b) is added to connect the replicas, since they share the same label. After this step, all vertices that are connected to a replicated node will have one less incoming multi-edge. This suggests that vertices having several multi-edges should be expanded first.

More in general, consider the expansion of a vertex x_i that is connected to more than one vertex by multi-edges and suppose, for the sake of illustration, that each multi-edge is characterized by a different cardinality. With reference to Fig. 2c, we have two multi-edges with 2 and 3 edges respectively coming from the same vertex x_i. Expansion of x_i consists in the following steps.

i) Replicate vertex: the number of replicas to be introduced for x_i is equal to the highest cardinality k of multi-edges outgoing from x_i, *i.e.*, 3 grey nodes in Fig. 2d.

ii) Add identity constraints: $k-1$ edges labeled by 1_Σ are added to enforce the identity between the k replicas (blue edges in Fig.2d).

iii) Distribute previous constraints among replicas: every node having an incoming edge in x_i is connected to all the replicas (black edges in Fig.2d). In addition, all the outgoing edges from x_i (even when belonging to different multi-edges *e.g.*, the orange and green multi-edges in Fig.2d) preserve their labels and simply change their source to one of the replicas. Distributing constraints does not significantly affect synchronization performance, and proves beneficial in terms of speed and memory, since less constraints need to be added to the expanded graph. The computational complexity of the expansion algorithm is $O(n^2 m)$ where m is the average multiplicity of multi-edges and n is the number of vertices. Please refer to the supplementary material for a more detailed description of the expansion algorithm.

3.3. Constrained eigenvalues optimization

Although both the theoretical framework of multi-graph synchronization (Sec. 3.1) and our expansion algorithm (Sec. 3.2) are general and hold for any group, for the sake of concreteness, we will henceforth focus on *linear groups*, i.e., groups admitting a matrix representation. A simple group-labelled graph, where the group is linear, can be synchronized using the *spectral method* [9, 7], that we briefly review in the following, before extending it to multi-graphs.

Let n denote the number of vertices of the **simple graph** and let us collect all the unknowns in a block matrix X of size $dn \times d$ and all the measures in another block-matrix Z of size $dn \times dn$, which are constructed as follows

$$\mathsf{X}=\begin{bmatrix} x(1) \\ x(2) \\ \vdots \\ x(n) \end{bmatrix}, \quad \mathsf{Z}=\begin{bmatrix} \mathsf{I}_d & z(1,2) & \dots & z(1,n) \\ z(2,1) & \mathsf{I}_d & \dots & z(2,n) \\ \vdots & & \ddots & \vdots \\ z(n,1) & z(n,2) & \dots & \mathsf{I}_d \end{bmatrix}. \quad (5)$$

Eq. (5) clearly refers to a complete graph. For incomplete graphs, Z is filled with zero blocks in correspondence of missing edges, *i.e.*, the available measures are given by

$$\mathsf{Z}_A = \mathsf{Z} \circ (\mathsf{A} \otimes 1_d)\,, \quad (6)$$

where A denotes the adjacency matrix of the graph, $\circ$ indicates the Hadamard (or entry-wise) product, $\otimes$ denotes the Kronecker product and 1_d is a $d \times d$ matrix filled by ones.

Proposition 1 ([9, 7]). *A consistent vertex labelling* X *satisfies the following equation:*

$$\mathsf{Z}_\mathsf{A}\mathsf{X} - (\mathsf{D} \otimes \mathsf{I}_d)\mathsf{X} = 0 \tag{7}$$

where D *denotes the degree matrix of the graph.*

This proposition is at the basis of the spectral solution for synchronization on a simple graph, which, due to noise, solves Eq. (7) by least squares:

$$\min_{\mathsf{X}^\top\mathsf{X}=\mathsf{I}_d} \|\mathsf{M}\mathsf{X}\|_F^2 \tag{8}$$

where $\mathsf{M} = \mathsf{Z}_\mathsf{A} - (\mathsf{D} \otimes \mathsf{I}_d)$ is defined from the matrix of incomplete relative measurements. It can be proved that the relaxed synchronization admits a closed-form solution as the null-space of M [9, 7], which in turn can be derived from the least eigenvectors of $\mathsf{M}^\top\mathsf{M}$.

Let us now consider the case of a group-labelled **multi-graph**. Synchronizing the *expanded* graph by solving Eq. (8) for the unknown labels X falls short, as it does not guarantee that replicated nodes have been assigned the same label. Indeed, the constraints given by edges labelled with the identity are treated as "soft" ones, like all the others. To obtain a labeling of the expanded graph consistent with the underlying multi-graph, it is hence necessary to enforce that replicated vertices share exactly the same labels. We are therefore led to a *constrained* version of Eq. (8):

$$\min_{\mathsf{X}} \|\mathsf{M}\mathsf{X}\|_F^2 \quad \text{subject to } \mathsf{X}^\top\mathsf{X} = \mathsf{I}_d, \quad \mathsf{C}^\top\mathsf{X} = 0, \tag{9}$$

where $\mathsf{C}^\top\mathsf{X} = 0$ enforces the equality between replicated vertices. Specifically, C is a $nd \times rd$ matrix composed by r column-blocks C_k of size $nd \times d$, that accommodate for r constraints between replicas: suppose the k-th constraint is of the form $\mathsf{X}_i - \mathsf{X}_j = 0$, hence

$$\begin{aligned} \mathsf{C} &= \begin{pmatrix} \mathsf{C}_1 & \cdots & \mathsf{C}_k & \cdots & \mathsf{C}_r \end{pmatrix}, \\ \mathsf{C}_k &= \begin{pmatrix} 0 & \cdots & \mathsf{I}_d & \cdots & -\mathsf{I}_d & \cdots & 0 \end{pmatrix}^\top. \end{aligned} \tag{10}$$

The constrained problem (9) admits a closed-form solution thanks to the following new result.

Theorem 1. *The stationary points of the cost function* (9) *are given by the eigenvectors of* $\left(\mathsf{I} - \mathsf{C}\mathsf{C}^\dagger\right)\mathsf{M}^\top\mathsf{M}$*, where* $\mathsf{C}^\dagger$ *is the pseudo-inverse of* C.

Proof. Problem (9) is equivalent to

$$\min_{\mathsf{X}} \operatorname{tr}\left(\mathsf{X}^\top(\mathsf{M}^\top\mathsf{M})\mathsf{X}\right) \text{ s. t. } \mathsf{X}^\top\mathsf{X} = \mathsf{I}_d,\ \mathsf{C}^\top\mathsf{X} = 0. \tag{11}$$

The Lagrangian of the cost function to this problem is

$$\begin{aligned} \mathcal{L} = \operatorname{tr}\left(\mathsf{X}^\top(\mathsf{M}^\top\mathsf{M})\mathsf{X}\right) + \operatorname{tr}\left(\Delta(\mathsf{X}^\top\mathsf{X} - \mathsf{I}_d)\right) + \\ \operatorname{tr}\left(\Gamma\mathsf{C}^\top\mathsf{X}\right), \end{aligned} \tag{12}$$

where Δ and Γ are matrices of unknown Lagrange multipliers, with Δ symmetric. Setting to zero the derivatives with respect to X we have

$$\frac{\partial\mathcal{L}}{\partial\mathsf{X}} = 2\mathsf{M}^\top\mathsf{M}\mathsf{X} + 2\mathsf{X}\Delta + \mathsf{C}\Gamma^\top = 0. \tag{13}$$

Revisiting the approach described in [28], we left-multiply by $\mathsf{C}^\top$, then using $\mathsf{C}^\top\mathsf{X} = 0$, we obtain

$$2\mathsf{C}^\top\mathsf{M}^\top\mathsf{M}\mathsf{X} + \mathsf{C}^\top\mathsf{C}\Gamma^\top = 0, \tag{14}$$

from which we get $\Gamma^\top = -2\mathsf{C}^\dagger\mathsf{M}^\top\mathsf{M}\mathsf{X}$. Plugging this closed-form expression for Γ into (13) yields

$$\left(\mathsf{I} - \mathsf{C}\mathsf{C}^\dagger\right)\mathsf{M}^\top\mathsf{M}\mathsf{X} = -\mathsf{X}\Delta, \tag{15}$$

where $\mathsf{C}^\dagger = (\mathsf{C}^\top\mathsf{C})^{-1}\mathsf{C}^\top$. Let $\mathsf{P} = \mathsf{I} - \mathsf{C}\mathsf{C}^\dagger$ and $\mathsf{X} = [\mathsf{x}_1, \ldots, \mathsf{x}_d]$, the last equation implies that the eigenvectors of $\mathsf{P}\mathsf{M}^\top\mathsf{M}$ are stationary points for (12), and the eigenvalues are the corresponding stationary values. Even if P and $\mathsf{M}^\top\mathsf{M}$ are symmetric, their product is not necessarily so. However, since $\mathsf{P}^2 = \mathsf{P}$ (so that it is a projection matrix), we get

$$\operatorname{eig}(\mathsf{P}\mathsf{M}^\top\mathsf{M}) = \operatorname{eig}(\mathsf{P}^2\mathsf{M}^\top\mathsf{M}) = \operatorname{eig}(\mathsf{P}\mathsf{M}^\top\mathsf{M}\mathsf{P}). \tag{16}$$

Hence, the stationary values of (12) are the eigenvalues of $\mathsf{P}\mathsf{M}^\top\mathsf{M}\mathsf{P}$, which are real (and the eigenvectors as well). □

Remark 3. If C has rank k, at least k eigenvalues will be zero, so the the solution to (9) is attained when x_i are the d orthogonal eigenvectors of $\mathsf{P}\mathsf{M}^\top\mathsf{M}$ corresponding to eigenvalues $\lambda_{k+1} \cdots \lambda_{k+d}$ in ascending order. Golub [28] suggests to get rid of the k zero eigenvalues due to the rank of C by using the rank-revealing QR factorization of C. More details on this are reported in the supplementary material.

Please note that Theorem 1 solves a *relaxed* problem, as the feasible set is given by $\mathsf{X} \in \mathbb{R}^{dn \times dn}$ with $\mathsf{X}^\top\mathsf{X} = I_d$, instead of $\mathsf{X} \in \Sigma^n$. In order to recover the block-wise structure of X in Eq. (5), it is hence necessary to project each $d \times d$ block of X onto the group Σ. When $\Sigma = \mathrm{SO}(3)$, for instance, the final projection can be done via Singular Value Decomposition. This produces our closed-form solution to multi-graph synchronization. In the presence of outliers, robustness can be easily gained via Iteratively Reweighted Least Squares (IRLS), as in [7].

To sum up, MULTISYNC takes in input a group-labelled multi-graph, and uses the expansion algorithm (Sec. 3.2) to dilate it to a simple graph with replicated nodes. The hard constraints between replicas are then integrated in the constrained Problem (9), for which a closed-form solution is derived using Theorem 1. Finally, each block of the solution is projected on Σ. It is worth noting that our approach is general and it can be applied to *any* linear group (and semi-groups), such as the ones mentioned in Sec. 2, which are relevant for a variety of vision applications.

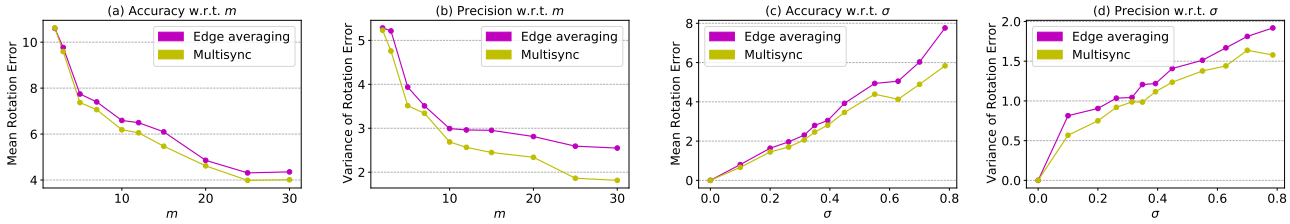


Figure 3: Experiments in $\mathrm{SO}(3)$: accuracy and precision of MULTISYNC and edge averaging on synthetic multi-graphs at various values of m (the average cardinality of multi-edges) and of noise level σ. The lower the curve the better.

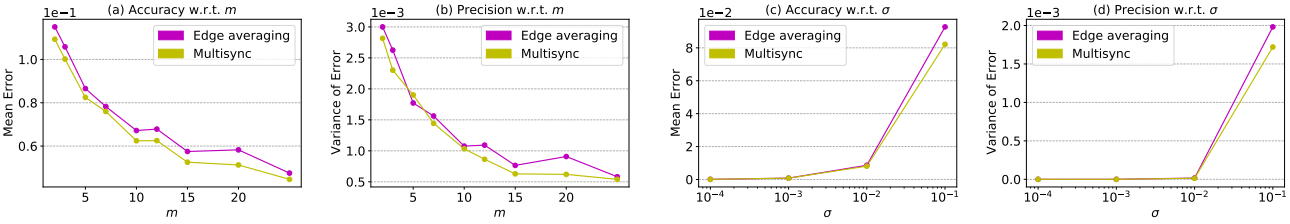


Figure 4: Experiments in $\mathrm{SL}(3)$: accuracy and precision of MULTISYNC and edge averaging on synthetic multi-graphs at various values of m (the average cardinality of multi-edges) and of noise level σ. The lower the curve the better.

4. Synthetic Experiments

We compared our solution (MULTISYNC) with the baseline approach of *edge averaging* on synthetic data, since, both these methods are general techniques explicitly developed for multi-graphs. Future work will consider alternative methods that can be easily adapted from simple graphs to multi-graphs (such as the ones discussed in Remark 1). Code is available at [1]. We considered synchronization problems in the group of rotations $\mathrm{SO}(3)$ and in the Special Linear Group $\mathrm{SL}(3)$. Random multi-graphs are generated according to parameters (n, p, m) where n indicates the number of vertices, p indicates the probability that any two vertices are connected by a multi-edge and m is the average cardinality of the multi-edges. We considered various levels of noise, which is defined in a different way for the two groups. No outliers are present in these synthetic data, therefore we concentrate on non-robust methods.

Synchronization in $\mathrm{SO}(3)$. We compare the estimated labels $\tilde{x}$ with the ground-truth ones x by the angular distance between rotations [34], which is defined as

$$\epsilon_i = \| \log(x(i)\tilde{x}(i)^\top) \| \tag{17}$$

and we report both the mean and the variance over all the nodes $i \in \mathcal{V}$. Edge averaging was performed using the (non-robust) chordal ℓ_2-mean [32].

In a first experiment, simulated multi-graphs have $n = 10$ vertices, $p = 0.75$ and a varying average cardinality m of multi-edges. For each run, n random ground truth rotation matrices $x(i)$ were instantiated by uniformly sampling Euler angles, hence relative rotations were generated as $z(i,j) = x(i)x(j)^\top \omega(i,j)$ with $\omega(i,j) \in \mathrm{SO}(3)$ being a small multiplicative noise. The Euler angles for the perturbation $\omega(i,j)$ were sampled from a Gaussian distribution with zero mean and a standard deviation $\sigma = \pi/8$. Fig. 3(a) reports the angular distance averaged on 100 trials, demonstrating the benefits of our MULTISYNC with respect to edge averaging in terms of accuracy, especially for higher values of m. Fig. 3(b) reports the variance of the error, showing that our approach delivers significantly more precise results for all values of m.

In a second experiment, we assess the response of the analysed methods when increasing the level of Gaussian noise σ, choosing a multi-graph where edge averaging and MULTISYNC showed comparable performance in the previous experiment, namely $n = 10$, $p = 0.75$ and $m = 5$. Fig. 3(c) reports the average angular distance as a function of σ, showing that MULTISYNC is more accurate than edge averaging, especially for large amounts of noise. From Fig. 3(d), we can appreciate that the variance increases less rapidly (with respect to σ) for our MULTISYNC as compared to edge averaging.

Synchronization in $\mathrm{SL}(3)$. We repeated the previous validation procedure on synthetic multi-graphs labeled in $\mathrm{SL}(3)$. In order to compare the estimated labels $\tilde{x}$ with the ground-truth ones x, the error was defined as the Frobenius norm of the deviation from the identity:

$$\epsilon_i = \| \mathbf{I}_3 - x(i)\tilde{x}(i)^{-1} \|_F \tag{18}$$

and we report both the mean and the variance over all the nodes $i \in \mathcal{V}$. Edge averaging was performed by computing

the element-wise average and then projecting the resulting matrix onto the $\mathrm{SL}(3)$ group.

Ground-truth transformations $x(i)$ were generated as random matrices with entries uniformly sampled in $[0, 1]$, which are eventually projected onto the group. The relative transformations were corrupted by an additive Gaussian noise with zero mean and standard deviation σ. Performances of the analysed methods (averaged on 100 random runs) are reported in Fig. 4, which confirms the findings attained in $\mathrm{SO}(3)$. With respect to the average multi-edge cardinality m, our method is both more accurate and more precise than edge averaging. The advantages of our solution are even more evident for large noise value σ, where MULTISYNC has a lower error and a smaller variance.

The rundown of these experiments is that edge averaging does not provide optimal results, as major approximations are introduced in the simplified measurement graph. MULTISYNC overcomes these limitations, and better combines the information provided by the multiple edge labels, especially when their multiplicity tends to be significant.

5. Application: partitioned synchronization

We illustrate how our multi-graph formulation can be used to deal with partitioned synchronization problems, demonstrating the advantages of our approach on real-data. The main steps can be detailed as follows:

A) Graph partitioning: The first step consists in partitioning the original graph Γ in multiple *patches* $\Phi_1, \dots \Phi_k$ (see the middle part of Fig. 5). To this end, we feed its adjacency matrix to spectral clustering. We choose the normalization of [48] because it tends to balance the number of edges among clusters[1], which is the relevant parameter for the computational complexity of the sparse matrix methods that we are using. Note that in some applications, *e.g.* where for privacy reasons information cannot be shared between all nodes, the partition of the graph is given and this clustering phase is sidestepped.

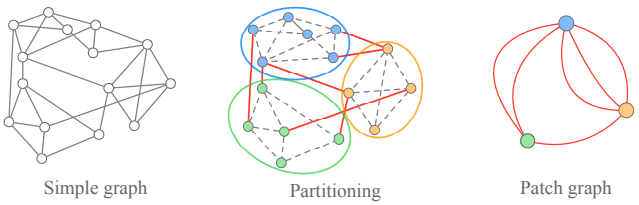


Figure 5: Partitioned synchronization: a simple graph (on the left) is partitioned in three patches. Each patch is synchronized individually. To solve for the local ambiguities, a patch-graph is built (on the right), whose nodes correspond to patches and multi-edges contain the cut edges (drawn in red). The patch-graph is synchronized with our multi-graph approach.

[1]This choice is not critical: other algorithms give similar results.

B) Synchronization on patches: Each patch Φ_u – which is a *simple* graph – is synchronized independently using standard synchronization techniques (see Sec. 2), obtaining a labeling $x^u \colon \Phi_u \to \Sigma$ for each patch Φ_u. Each local labeling comes with its own ambiguity as it is defined up to the action of an arbitrary group element. Hence, we need to remove the local ambiguities and lift the multiple labeling derived in this step to a single consistent labeling on the original full graph.

C) Patch-graph building: We build a multi-graph, termed the *patch-graph* as follows. The vertices of the patch-graph correspond to the patches $\Phi_1, \dots, \Phi_k$ and the multi-edge connecting Φ_u and Φ_v consists of all the *cut edges*, *i.e.*, the edges (if any) that have one endpoint in Φ_u and the other endpoint in Φ_v (see the right part of Fig. 5). Each node in the patch-graph is labelled with the *unknown* group element that represents the transformation that must be applied to the patch in order to fix its group ambiguity, let $w_u \in \Sigma$ be the label of node Φ_u. Measures on multi-edges of the patch graph are defined as follows. Let us consider a multi-edge connecting nodes Φ_u and Φ_v, that comprises multiple cut-edges between the two patches. Let us consider one cut-edge that connects node $i \in \Phi_u$ and node $j \in \Phi_v$, and let $x^u(i)$ and $x^v(j)$ be the respective labels found in the previous synchronization on patches. Such labels must be multiplied by the respective matrices representing local ambiguities, namely $w_u \in \Sigma$ and $w_v \in \Sigma$, hence the consistency constraint rewrites $z(i,j) = \big(x^u(i)w_u\big)\big(x^v(j)w_v\big)^{-1} = x^u(i)w_u w_v^{-1} x^v(j)^{-1}$. Therefore $w_u w_v^{-1} = x^u(i)^{-1} z(i,j) x^v(j)$ is the label of the multi-edge between Φ_u and Φ_v.

D) Synchronization on the patch-graph: Our MULTISYNC is applied to the patch-graph, in order to compute the unknown transformations $w_1 \dots w_k \in \Sigma$ (see Sec. 3). For each patch, the label assignments x^u are transformed accordingly by applying the respective transformation w_u. This produces a unique and globally consistent labeling up to a single global ambiguity.

For the experimental validation, we considered partitioned synchronization problems in $\mathrm{SO}(3)$. We used large-scale image data sets taken from [59, 19], which provide the input graph and estimates of relative rotations. The graphs are highly incomplete and affected by missing data. The output of *Bundler* [51] was taken as ground-truth, as customarily done in the literature. All the experiments were performed in MATLAB on an Intel Core i9 9900k at stock speeds coupled with 16 GB of 3200 MHz RAM.

As in Sec. 4, we contrast our method with *edge averaging*. Both MULTISYNC and edge averaging operates on the same patch graph constructed following steps from A to C, and they differ only in the way the patch graph is synchronized in Step D. In Step A, the number of clusters c in which the input graphs are partitioned is computed from

Table 1: Partitioned synchronization in $\mathrm{SO}(3)$ on real data sets from [59, 19]. The average error $\bar{\epsilon}$, median error $\hat{\epsilon}$, and running time t are reported for MULTISYNC and edge averaging. Full synchronization performances are included but are intended only as an ideal reference as it works on different assumptions, having at disposal the full graph instead of partitioning it. In Supplementary material Table is reported in full resolution.

			Edge averaging			MULTISYNC			Full sync		
Data set	n	c	$\bar{\epsilon}$	$\hat{\epsilon}$	t	$\bar{\epsilon}$	$\hat{\epsilon}$	t	$\bar{\epsilon}$	$\hat{\epsilon}$	t
Ellis Island	247	9	3.56	0.73	1.02	3.49	0.68	1.07	3	0.47	3.4
Piazza del Popolo	345	11	5.62	1.86	1.27	5.22	1.41	1.38	3.3	0.86	3.47
NYC Library	376	11	4.91	3.35	0.93	4.24	2.32	1.01	3.16	1.27	4.8
Madrid Metropolis	394	11	7.84	3.19	1.13	7.14	2.52	1.18	6.6	1.12	1.18
Yorkminster	458	12	5.96	3.98	1.33	4.98	2.91	1.37	3.5	1.58	4.83
Montreal N. Dame	474	12	2.67	1.02	1.32	2.11	0.87	1.41	1.12	0.5	10.1
Tower of London	489	13	6.43	3.51	1.07	5.54	2.74	1.12	4.21	2.33	3.91
Notre Dame	553	13	4.25	1.92	3.82	3.43	0.85	3.87	2.7	0.65	22
Alamo	627	14	6.89	1.63	4.21	6.42	1.57	4.28	3.7	1.02	26.5
Gendarmenmarkt	742	15	39.54	21.18	2.29	34.32	12.68	2.34	40.82	6.09	39.9
Vienna Cathedral	918	17	15.73	4.87	3.88	11.01	3.73	3.92	6.2	1.27	50.2
Union Square	930	17	7.71	3.88	3.65	7.25	3.67	3.71	6.18	3.6	6.61
Roman Forum	1102	17	10.03	9.12	4.95	6.91	3.39	5.01	2.81	1.4	12.1
Piccadilly	2508	21	19.89	10.21	17.45	17.47	8.21	17.61	4.42	1.94	241
Cornell Arts Quad	5530	41	8.64	3.29	17.88	6.25	2.71	18.02	3.2	1.71	191

the number of vertices n as $c = 0.54\sqrt{n}$, where the coefficient 0.54 has been tuned manually on the whole data set. In Step B, we synchronize the individual patches using MPLS (Message Passing Least Squares) [2, 49], which represents the state of the art in rotation synchronization. It is worth noting, however, that our framework allows to plug-in any method as far as the synchronization of single patches is concerned, as we shall see later on. For the sake of efficiency, in Step C, we considered multi-edges with cardinality at most 50 by randomly subsampling cut-edges. Finally, since these data sets are affected by outliers, multi-graph synchronization in Step D is performed by robust techniques:

- MULTISYNC is augmented with IRLS, as in [7], to provide robustness to outliers;
- in edge averaging, the labels of the multi-edges are averaged using the Weiszfeld algorithm [31], thus collapsing the multi-graph into a *simple* graph; then, the resulting graph is synchronized with MPLS [49].

For reference, we also include full synchronization on the entire graph (with MPLS), which ideally represents the best performance achievable in terms of accuracy.

Tab. 1 reports the mean/median angular error and the execution time of all the analysed methods. Please see the Supplementary Material for a full resolution version of the Table. For every data set, 50 independent runs were performed and average results were computed. MULTISYNC consistently outperforms edge averaging in terms of accuracy on all the cases. The worst accuracy on the Gendarmenmarkt data set is caused by the fact that the graph is relatively sparse and lacks cycle information, as already observed in [49]. With respect to the "gold standard" represented by full synchronization – that it is expected to outperforms partitioned approaches for it exploits all the information available on the full graph – the great advantage of MULTISYNC is that it trades relatively small loss in accuracy for shorter execution times. Some of the larger sequences, such as Piccadilly, show up to a $13\times$ improvement in this regard.

As already noted, MULTISYNC can be seen as a general framework that is agnostic about the synchronization technique used on sub-graphs (Step B). This aspect is investigated in Tab. 2, which shows the performance of MULTISYNC coupled with different synchronization methods, namely EIG-IRLS [7], L1-IRLS [17] and R-GODEC [6]. The state-of-the-art MPLS [49] is only reported in Tab. 1. EIG-IRLS results are not reported on Cornell Arts Quad because it did not reach convergence. It turns out that different performances of the various methods on the full graph reflect on the partitioned case: for instance, R-GoDec is the fastest solution whereas L1-IRLS is the most accurate. With respect to full synchronization, again we see that MULTISYNC strikes a good balance between accuracy and computational burden.

Table 2: Performances of MULTISYNC combined with different techniques, for synchronization in $\mathrm{SO}(3)$ on real data sets [59, 19]. The median error $\hat{\epsilon}$, and running time t are reported. In Supplementary material Table is reported in full resolution.

			EIG-IRLS				L1-IRLS				R-GoDec			
			MULTISYNC		Full sync		MULTISYNC		Full sync		MULTISYNC		Full sync	
Data set	n	c	$\hat{\epsilon}$	t	$\hat{\epsilon}$	t	$\hat{\epsilon}$	t	$\hat{\epsilon}$	t	$\hat{\epsilon}$	t	$\hat{\epsilon}$	t
Ellis Island	247	9	1.15	0.43	1.18	0.82	1.05	0.41	0.57	2.35	1.48	0.23	1.00	0.23
Piazza del Popolo	345	11	1.78	1.18	1.02	2.24	1.84	0.53	0.98	3.55	1.86	0.24	1.48	0.49
NYC Library	376	11	3.24	3.15	1.98	1.65	2.02	0.39	1.33	2.36	2.82	0.47	3.20	1.35
Madrid Metropolis	394	11	4.01	3.83	4.43	1.79	2.68	0.57	1.01	4.20	3.94	0.29	4.07	0.49
Yorkminster	458	12	2.91	2.85	1.81	3.18	2.86	1.07	1.69	2.29	2.85	0.43	2.69	2.03
Montreal N. Dame	474	12	2.12	6.92	0.59	4.09	1.01	3.67	0.58	7.10	1.02	0.52	0.85	1.05
Tower of London	489	13	2.98	3.74	2.79	2.43	2.83	1.20	2.63	1.94	2.89	0.46	3.28	2.11
Notre Dame	553	13	1.23	5.22	0.74	7.46	1.22	25.94	0.65	29.11	1.57	1.79	1.05	1.10
Alamo	627	14	1.99	2.01	1.19	11.05	1.87	1.84	1.09	32.22	1.61	0.79	1.48	1.67
Gendarmenmarkt	742	15	26.88	8.19	76.97	11.30	14.81	2.12	28.85	12.01	37.36	1.01	28.70	5.83
Vienna Cathedral	918	17	4.72	2.88	1.62	18.23	3.92	1.68	1.37	56.80	2.53	1.59	2.08	9.26
Union Square	930	17	18.95	4.13	4.93	6.48	4.33	1.05	3.97	4.82	20.17	1.57	7.16	10.58
Roman Forum	1102	17	7.89	6.11	1.86	15.46	3.85	2.20	2.27	12.46	5.54	1.12	7.54	14.77
Piccadilly	2508	21	39.55	54.05	24.87	284.87	9.01	8.93	1.89	287.23	11.79	12.24	13.36	47.13
Cornell Arts Quad	5530	41	-	-	-	-	5.49	30.10	1.98	73.51	17.13	28.24	13.21	586.6

6. Conclusion and future work

We studied the task of synchronization on multi-graphs. After formally introducing the theoretical framework, we derived an effective algorithm that works on any linear group. Our approach exploits the redundancy encapsulated in multi-edges and has better statistical properties with respect to the baseline approach of edge-averaging, improving both accuracy and precision. Applications of multi-graph synchronization are countless, including partitioned synchronization. Future work will include partial permutation synchronization and an analysis of alternative graph-expansion algorithms including optimality.

Acknowledgements. Federica Arrigoni was supported by the European Union's Horizon 2020 Research and Innovation Programme under the project SPRING (Grant Agreement No. 871245).

References

[1] https://github.com/andreadalcin/multisync. 6

[2] https://github.com/yunpeng-shi/mpls. 8

[3] F. Arrigoni and A. Fusiello. Synchronization problems in computer vision with closed-form solutions. *International Journal of Computer Vision*, 128:26–52, 2020. 2

[4] Federica Arrigoni, Luca Magri, and Tomas Pajdla. On the usage of the trifocal tensor in motion segmentation. In *Proceedings of the European Conference on Computer Vision*, 2020. 2

[5] Federica Arrigoni and Tomas Pajdla. Motion segmentation via synchronization. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2019. 3

[6] Federica Arrigoni, Beatrice Rossi, Pasqualina Fragneto, and Andrea Fusiello. Robust synchronization in SO(3) and SE(3) via low-rank and sparse matrix decomposition. *Computer Vision and Image Understanding*, 174:95–113, 2018. 2, 8

[7] F. Arrigoni, B. Rossi, and A. Fusiello. Spectral synchronization of multiple views in SE(3). *SIAM Journal on Imaging Sciences*, 9(4):1963 – 1990, 2016. 2, 3, 4, 5, 8

[8] A. Bartoli and P. Sturm. Constrained structure and motion from multiple uncalibrated views of a piecewise planar scene. *International Journal of Computer Vision*, 52(1):45–64, 2003. 3

[9] F. Bernard, J. Thunberg, P. Gemmar, F. Hertel, A. Husch, and J. Goncalves. A solution for multi-alignment by transformation synchronisation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 2, 3, 4, 5

[10] Florian Bernard, Johan Thunberg, Paul Swoboda, and Christian Theobalt. HiPPI: Higher-order projected power iterations for scalable multi-matching. In *Proceedings of the International Conference on Computer Vision*, 2019. 2

[11] Brojeshwar Bhowmick, Suvam Patra, Avishek Chatterjee, Venu Madhav Govindu, and Subhashis Banerjee. Divide and conquer: Efficient large-scale structure from motion using graph partitioning. In *12th Asian Conference on Computer Vision (ACCV 2014)*, 2014. 2

[12] T. Birdal, M. Arbel, U. Simsekli, and L. J. Guibas. Synchronizing probability measures on rotations via optimal transport. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1566–1576, 2020. 2

[13] Tolga Birdal and Umut Simsekli. Probabilistic permutation synchronization using the riemannian structure of the birkhoff polytope. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11105–11116, 2019. 2

[14] Tolga Birdal, Umut Simsekli, Mustafa Onur Eken, and Slobodan Ilic. Bayesian pose graph optimization via bingham distributions and tempered geodesic mcmc. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 2

[15] Béla Bollobás. *Modern graph theory*, volume 184. Springer Science & Business Media, 2013. 3

[16] Luca Carlone, Roberto Tron, Kostas Daniilidis, and Frank Dellaert. Initialization techniques for 3D SLAM: A survey on rotation estimation and its use in pose graph optimization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2015. 2

[17] A. Chatterjee and V. M. Govindu. Efficient and robust large-scale rotation averaging. In *Proceedings of the International Conference on Computer Vision*, 2013. 2, 3, 8

[18] Yu Chen, Shuhan Shen, Yisong Chen, and Guoping Wang. Graph-based parallel large scale structure from motion. *Pattern Recognition*, 107, 2020. 3

[19] David Crandall, Andrew Owens, Noah Snavely, and Daniel P. Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3001–3008, 2011. 2, 3, 7, 8

[20] M. Cucuringu, Y. Lipman, and A. Singer. Sensor network localization by eigenvector synchronization over the Euclidean group. *ACM Transactions on Sensor Networks*, 8(3):19:1 – 19:42, 2012. 3

[21] Frank Dellaert, David M. Rosen, Jing Wu, Robert Mahony, and Luca Carlone. Shonan rotation averaging: Global optimality by surfing $so(p)^n$. In *Computer Vision – ECCV 2020*, pages 292–308. Springer International Publishing, 2020. 2

[22] A. Eriksson, C. Olsson, F. Kahl, and T. Chin. Rotation averaging and strong duality. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 127–135, 2018. 2

[23] Taosha Fan, Hanlin Wang, Michael Rubenstein, and Todd Murphey. Cpl-slam: Efficient and certifiably correct planar graph-based slam using the complex number representation. *IEEE Transactions on Robotics*, 36(6):1719–1737, 2020. 2

[24] Meiling Fang, Thomas Pollok, and Chengchao Qu. Merge-sfm: Merging partial reconstructions. In *British Machine Vision Conference*, page 29, 2019. 2

[25] Marcel Geppert, Viktor Larsson, Pablo Speciale, Johannes L Schönberger, and Marc Pollefeys. Privacy preserving structure-from-motion. In *European Conference on Computer Vision*, pages 333–350. Springer, 2020. 1

[26] A. Giridhar and P.R. Kumar. Distributed clock synchronization over wireless networks: Algorithms and analysis. *Proceedings of the IEEE Conference on Decision and Control*, pages 4915–4920, 2006. 2

[27] Zan Gojcic, Caifa Zhou, Jan D. Wegner, Leonidas J. Guibas, and Tolga Birdal. Learning multiview 3D point cloud registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2

[28] Gene H Golub. Some modified matrix eigenvalue problems. *Siam Review*, 15(2):318–334, 1973. 5

[29] V. M. Govindu. Lie-algebraic averaging for globally consistent motion estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 684–691, 2004. 2

[30] V. M. Govindu and A. Pooja. On averaging multiview relations for 3D scan registration. *IEEE Transactions on Image Processing*, 23(3):1289–1302, 2014. 2

[31] R. Hartley, K. Aftab, and J. Trumpf. L1 rotation averaging using the Weiszfeld algorithm. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3041–3048, 2011. 2, 8

[32] R. I. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation averaging. *International Journal of Computer Vision*, 2013. 2, 4, 6

[33] Xiangru Huang, Zhenxiao Liang, Xiaowei Zhou, Yao Xie, Leonidas J. Guibas, and Qixing Huang. Learning transformation synchronization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2019. 2

[34] Du Q. Huynh. Metrics for 3D rotations: Comparison and analysis. *Journal of Mathematical Imaging and Vision*, 35(2):155–164, 2009. 6

[35] S. Leonardos, X. Zhou, and K. Daniilidis. Distributed consistent data association via permutation synchronization. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2645–2652, 2017. 2

[36] X. Li and H. Ling. Hybrid camera pose estimation with online partitioning for slam. *IEEE Robotics and Automation Letters*, 5(2):1453–1460, 2020. 2

[37] D. Martinec and T. Pajdla. Robust rotation and translation estimation in multiview reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007. 2, 3

[38] E. Maset, F. Arrigoni, and A. Fusiello. Practical and efficient multi-view matching. In *Proceedings of IEEE International Conference on Computer Vision*, pages 4568–4576, 2017. 3

[39] O. Ozyesil, N. Sharon, and A. Singer. Synchronization over cartan motion groups via contraction. *SIAM Journal on Applied Algebra and Geometry*, 2(2):207 – 241, 2018. 2

[40] Onur Ozyesil, Vladislav Voroninski, Ronen Basri, and Amit Singer. A survey of structure from motion. *Acta Numerica*, 26:305 – 364, 2017. 2

[41] Deepti Pachauri, Risi Kondor, and Vikas Singh. Solving the multi-way matching problem by permutation synchronization. In *Advances in Neural Information Processing Systems*, volume 26, pages 1860–1868. 2013. 2

[42] Pulak Purkait, Tat-Jun Chin, and Ian Reid. Neurora: Neural robust rotation averaging. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 137–154. Springer International Publishing, 2020. 2

[43] David M Rosen, Luca Carlone, Afonso S Bandeira, and John J Leonard. Se-sync: A certifiably correct algorithm for synchronization over the special euclidean group. *The International Journal of Robotics Research*, 38(2-3):95–125, 2019. 2

[44] D. M. Rosen, C. DuHadway, and J. J. Leonard. A convex relaxation for approximate global optimization in simultaneous localization and mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 5822 – 5829, 2015. 2

[45] E. Santellani, E. Maset, and A. Fusiello. Seamless image mosaicking via synchronization. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2:247–254, 2018. 2

[46] P. Schroeder, A. Bartoli, P. Georgel, and N. Navab. Closed-form solutions to multiple-view homography estimation. In *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, pages 650–657, 2011. 2

[47] Yanyao Shen, Qixing Huang, Nati Srebro, and Sujay Sanghavi. Normalized spectral map synchronization. In *Advances in Neural Information Processing Systems*, volume 29, pages 4925–4933. Curran Associates, Inc., 2016. 2

[48] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000. 7

[49] Yunpeng Shi and Gilad Lerman. Message passing least squares framework and its application to rotation synchronization. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8796–8806. PMLR, 2020. 2, 8

[50] A. Singer. Angular synchronization by eigenvectors and semidefinite programming. *Applied and Computational Harmonic Analysis*, 30(1):20 – 36, 2011. 1, 2, 3

[51] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D. In *SIGGRAPH: International Conference on Computer Graphics and Interactive Techniques*, pages 835–846, 2006. 7

[52] Johan Thunberg, Florian Bernard, and Jorge Goncalves. Distributed methods for synchronization of orthogonal matrices over graphs. *Automatica*, 80:243–252, 2017. 2

[53] J. Thunberg, F. Bernard, and J. Gonçalves. Distributed synchronization of euclidean transformations with guaranteed convergence. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 3757–3762, 2017. 2

[54] A. Torsello, E. Rodolà, and A. Albarelli. Multiview registration via graph diffusion of dual quaternions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2441 – 2448, 2011. 2

[55] R. Tron and K. Danilidis. Statistical pose averaging with varying and non-isotropic covariances. In *Proceedings of the European Conference on Computer Vision*, 2014. 2, 3

[56] R. Tron and R. Vidal. Distributed 3-D localization of camera sensor networks from 2-D image measurements. *IEEE Transactions on Automatic Control*, 59(12):3325–3340, 2014. 2

[57] Roberto Tron, Xiaowei Zhou, and Kostas Daniilidis. A survey on rotation optimization in structure from motion. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2016. 2, 3

[58] L. Wang and A. Singer. Exact and stable recovery of rotations for robust synchronization. *Information and Inference: a Journal of the IMA*, 2(2):145–193, 2013. 2

[59] K. Wilson and N. Snavely. Robust global translations with 1DSfM. In *Proceedings of the European Conference on Computer Vision*, pages 61–75, 2014. 7, 8

[60] Jin-Gang Yu, Gui-Song Xia, Ashok Samal, and Jinwen Tian. Globally consistent correspondence of multiple feature sets using proximal Gauss–Seidel relaxation. *Pattern Recognition*, 51:255 – 267, 2016. 2

[61] Lei Zhou, Zixin Luo, Mingmin Zhen, Tianwei Shen, Shiwei Li, Zhuofei Huang, Tian Fang, and Long Quan. Stochastic bundle adjustment for efficient and scalable 3d reconstruction. In *Computer Vision – ECCV 2020*, pages 364–379, 2020. 3

[62] S. Zhu, R. Zhang, L. Zhou, T. Shen, T. Fang, P. Tan, and L. Quan. Very large-scale global sfm by distributed motion averaging. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4568–4577, 2018. 3