"© 2020 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works."

Efficient two step optimization for large embedded deformation graph based SLAM

Jingwei Song[1], Fang Bai[1], Liang Zhao[1], Shoudong Huang[1] and Rong Xiong[2]

***Abstract*— Embedded deformation graph is a widely used technique in deformable geometry and graphical problems. Although the technique has been transmitted to stereo (or RGB-D) camera based SLAM applications, it remains challenging to compromise the computational cost as the model grows. In practice, the processing time grows rapidly in accordance with the expansion of maps. In this paper, we propose an approach to decouple the nodes of deformation graph in large scale dense deformable SLAM and keep the estimation time to be constant. We observe that only partial deformable nodes in the graph are connected to visible points. Based on this fact, the sparsity of the original Hessian matrix is utilized to split the parameter estimation into two independent steps. With this new technique, we achieve faster parameter estimation with amortized computation complexity reduced from $O(n^2)$ to almost $O(1)$. As a result, the computational cost barely increases as the map keeps growing. Based on our strategy, the computational bottleneck in large scale embedded deformation graph based applications will be greatly mitigated. The effectiveness is validated by experiments, featuring large scale deformation scenarios.**

I. INTRODUCTION

Simultaneous Localization and Mapping (SLAM) is a technique widely used in robotics for pose estimation and environment mapping. While SLAM in the rigid scenario is quite mature, SLAM with deformations is still new to the community. Several attempts in this direction include RGB-D camera based deformable human body reconstruction and RGB camera based soft-tissue SLAM in Minimally Invasive Surgery (MIS). The key challenge in developing deformable SLAM is how to define and formulate the deformation.

As an early stage, researchers assume deformation scenario is technically similar to SLAM in dynamic environment. By assuming a large amount of target features to be static, approaches like [1] and [2] were proposed, adopting conventional extended Kalman filter (EKF) and Parallel Tracking and Mapping (PTAM) by applying some thresholds to separate rigid and non-rigid feature points. Recent modification based on ORB-SLAM [3] have been proposed and modified in [4] [5] [6] [7]. The above works show that extending traditional rigid SLAM to non-rigid scenarios is possible. However, map reconstruction based on the rigid assumption inevitably leads to gaps within rigid patches from different perspectives. The more extensive deformation of the environment, the larger gaps exist in the recovered map.

A better solution is to model the deformation. Recognizing map deformation modeling is vital and indeed many researches turn to the computer vision society for efficient deformation description. One of the most well-known and well-studied techniques is the Embedded Deformation (ED) [8], which was initially proposed for designing smooth character motions in cartoons. ED graph makes use of a sparsely interconnected scattered nodes with attributes of local rigid transformation. By mixing these local rigid transformations with weights, a global deformation is simulated in a discrete but smooth form. ED graph has been widely applied in RGB-D and stereo SLAM [9] [10] and 3D human reconstructions [11] [12] (discrete hierarchical warp grid but very similar to ED graph) [13] [14] [15]. The ED graph based SLAM formulation is able to describe the deformation of the environment in addition to camera pose, which blossoms with the application like [11] [12] [13] [15] for human motion reconstruction and [10] for large scale SLAM in MIS.

In the meantime, for more challenging monocular equipments, Finite Element Method (FEM) [16] [17] and Structure from Template (SfT) [18] [19] were proposed to simulate deformation. Both methods use a predefined grid or triangular mesh template to describe the soft-tissue. However, to the best of our knowledge, it is hard to apply these methods when the map is incrementally built and no complete real-time implementation has demonstrated how it will effectively be applied in large scenarios.

Overall, ED graph based formulation is the most applicable and widely used approach for modeling deformations. Numerous real-time dense SLAM systems have validated its effectiveness, particularly in batch scenarios, with fast sequential or parallel implementation. However, ED graph based formulation also comes with disadvantages and limitations. One major issue in ED graph is that when a new observation is incorporated, the number of nodes increases dramatically, posing heavy computational burden. Little attention has been paid in the field of truncated signed distance function based 3D human reconstruction because the target size, as well as map extent, are predefined. State estimation and map updating are all confined within a volume. In more general cases, however, as reported in [10], when reconstructing geometry without a predefined volume, the environment size is unbounded due to the sharp growth of

[1] Centre for Autonomous Systems, University of Technology, Sydney, P.O. Box 123, Broadway, NSW 2007, Australia

[2] State Key Laboratory of Industrial Control Technology and Institute of Cyber-Systems and Control, Zhejiang University, 38 Zheda Road, 310027 Zhejiang, China

This work is supported in part by the Australian Governments International Postgraduate Research Scholarship and Endeavour Research Fellowship and in part by the Joint Centre for Robotics Research between University of Technology Sydney and Zhejiang University.

Email: {jingwei.song; fang.bai}@student.uts.edu.au, rxiong@zju.edu.cn, {Liang.Zhao; Shoudong.Huang}@uts.edu.au

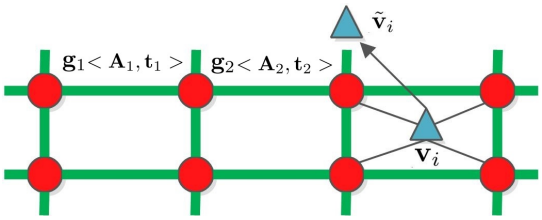


Fig. 1: An example of ED graph. The red circles are ED nodes, say node j, encoding a geometric position $\mathbf{g}_j$, and an affine transformation given by $\mathbf{A}_j$ and $\mathbf{t}_j$. The blue triangle is a vertex, that can be deformed from $\mathbf{v}_i$ to $\tilde{\mathbf{v}}_i$, through the impact of its neighboring ED nodes.

the graph, which eventually implies an amortized $O(n^2)$ complexity with respect to the number of nodes in the graph. In a word, optimizing an expanding ED graph in an unconstrained space significantly limits the performance of the system.

Even though little is known in speeding up ED graph based SLAM systems, numerous publications are dedicated to the graph based optimization in rigid scenarios. A typical graph optimization problem models robot poses (or landmark positions) as nodes in the graph, while the edges encode the relative measurement between connected nodes [20]. Graph sparsification is the most widely applied technique to marginalize subsets of nodes [21] [22] [23]. The key process in this topic is to sparsify edges and marginalize nodes based on indicators like divergence [24]. We will show in this paper that marginalization methods are of great value to enhance efficiency in ED based SLAM. In this paper, we will mitigate the $O(n^2)$ computational bottleneck encountered in the expanding environment, for ED graph based SLAM as reported in [10]. We analyze the spatial relationship between ED nodes and observed features and reveal the inherent sparsity pattern of the Hessian matrix. With this discovery, we classify ED nodes into points relevant (**PR**) and points irrelevant (**PI**) nodes and propose a decoupled optimization strategy.

In this work we convert the ED graph based formulation into a matrix form, which unveils a sparsity pattern that can be further exploited by restructuring the Jacobian matrix to benefit efficient marginalization. Based on this insight, we split ED graph optimization into two steps and a lossy decoupled optimization strategy is proposed. The accuracy and effectiveness are tested on both the ED based geometry deformation as well as ED based SLAM application. The proposed strategy is validated on the MIS-SLAM framework [10] with and without the proposed strategy. For more details on the MIS-SLAM framework, please refer to our previous contribution in [10].

II. Revisit General ED Graph Based SLAM

An ED graph, as other graph representations, comprises nodes and edges. See Fig. 1 for an example. We use the term 'node' to describe an ED node, and 'vertex' to describe a feature point. Each ED node j defines a local rigid motion modeled by $\mathbf{A}_j, \mathbf{t}_j$, where $\mathbf{A}_j \in \mathbb{R}^{3\times3}$ is an affine matrix and $\mathbf{t}_j \in \mathbb{R}^3$ is a translation vector. Given a vertex $\mathbf{v}_i$, neighbouring ED nodes define a deformation field indicating how $\mathbf{v}_i$ is deformed to target position $\tilde{\mathbf{v}}_i$. For a mesh based geometry, shape deformation is equivalent to transform vertices in the ED graph.

In practice, to avoid over-parameterization, the ED node $\mathbf{g}_j$ are deliberately positioned to minimize the size of parameters. For example in [8] [11] [13] [25], ED nodes are generated by uniformly downsampling the original model. The deformation between the point $\mathbf{v}_i$ and $\tilde{\mathbf{v}}_i$ is modeled as a transformation in the following form:

$$\tilde{\mathbf{v}}_i = \mathbf{R}_c \sum_{j=1}^{m} \omega_j(\mathbf{v}_i)[\mathbf{A}_j(\mathbf{v}_i - \mathbf{g}_j) + \mathbf{g}_j + \mathbf{t}_j] + \mathbf{T}_c, \quad (1)$$

where $\mathbf{R}_c$ and $\mathbf{T}_c$ are global rotation and translation relating to camera motion, m is the number of neighboring nodes. The global camera pose $[\mathbf{R}_c, \mathbf{T}_c]$ is not considered in the original ED work [8], however added later on in [11] [13] [25]. $\omega_j(\mathbf{v}_i)$ is a weight that quantifies the contribution of the ED node j to the vertex $\mathbf{v}_i$. We choose the number of neighboring nodes to $m = 4$ and define the weight:

$$\omega_j(\mathbf{v}_i) = 1 - ||\mathbf{v}_i - \mathbf{g}_j||/d_{max}, \quad (2)$$

where d_{max} is the maximum distance from the vertex $\mathbf{v}_i$ to the neighbouring $k+1$ nearest ED node.

Eq. (1) shows how one vertex is transformed with a given ED graph. Conversely, if the source ($\mathbf{v}_i$) and target ($\tilde{\mathbf{v}}_i$) point pairs are given, the parameters of ED graph ($\mathbf{A}_j$ and $\mathbf{t}_j$) can also be inferred. Thus, the ED graph based SLAM formulation is to infer the ED graph parameter from a cluster of arbitrary source and target points pairs. At last, ED graph is applied to deform the whole model. To the best of our knowledge, all ED graph based SLAM formulation consist of at least three terms: Rotation constraint, regularization constraint and a penalty term on the distance of deformed source and target point pairs. Some methods introduce more innovating terms, like visual hull term in [13], key features in [12] which are beyond the scope. An ED graph based SLAM is to minimize:

$$\underset{\mathbf{R}_c, \mathbf{T}_c, \mathbf{A}_1, \mathbf{t}_1 \ldots \mathbf{A}_m, \mathbf{t}_m}{\operatorname{argmin}} \quad \omega_{rot} E_{rot} + \omega_{reg} E_{reg} + \omega_{data} E_{data}. \quad (3)$$

The major key point pairs matching term is E_{data}, while two soft constraints E_{rot} and E_{reg} regulate deformation. E_{rot} hauls the affine matrix close to $SO(3)$ by minimizing the column vectors $\mathbf{c}_1$, $\mathbf{c}_2$ and $\mathbf{c}_3$ in:

$$E_{rot} = \sum_{j=1}^{m} Rot(\mathbf{A}_j), \quad (4)$$

$$\begin{aligned} Rot(\mathbf{A}) = &(\mathbf{c}_1^T \cdot \mathbf{c}_2)^2 + (\mathbf{c}_1^T \cdot \mathbf{c}_3)^2 + (\mathbf{c}_2^T \cdot \mathbf{c}_3)^2 + \\ &(\mathbf{c}_1^T \cdot \mathbf{c}_1 - 1)^2 + (\mathbf{c}_2^T \cdot \mathbf{c}_2 - 1)^2 + (\mathbf{c}_3^T \cdot \mathbf{c}_3 - 1)^2. \end{aligned} \quad (5)$$

Regularization is to ensure smoothness of the deformed surface, the motion field generated from ED nodes should be consistent; otherwise, the deformed shape will be fragmented and distorted. This is a general practice named 'as-rigid-as-possible' and defined in following form:

$$E_{reg} = \sum_{j=1}^{m} \sum_{k \in \mathbb{N}(j)} \alpha_{jk} ||\mathbf{A}_j(\mathbf{g}_k - \mathbf{g}_j) + \mathbf{g}_j + \mathbf{t}_j - (\mathbf{g}_k + \mathbf{t}_k)||^2. \tag{6}$$

Similar to [8], α_{jk} is the weight quantifying the influences of the neighboring ED nodes. $\mathbb{N}(j)$ is the set of all connected nodes to node j.

To solve geometrical model to frame registration, 'back-projection' formulation is adopted as a substitution to iterative closest point (ICP). Readers may refer to [13] [11] [25]. For simplicity, we introduce the basic source and target key point pairs described by [8]. Let $\tilde{\mathbf{v}}_i$ and $\mathbf{v}_i$ be the pairs of source-target points (defined in Eq. (1)). Normally predefined in interactive phase, these key points define how model is to be deformed. These key points define a data term which minimizes a distance as:

$$E_{data} = \sum_{i=1}^{n} ||\tilde{\mathbf{v}}_i - \mathbf{v}_i||^2. \tag{7}$$

III. Efficient two step optimization

A. Matrix form of ED graph deformation

To fully exploit the structure we rewrite the data term Eq. (7) in a matrix form for the convenience of sparsity analysis. Let us consider a group of predefined key source points $\mathbf{P} = [\mathbf{v}_1 ... \mathbf{v}_n]$ and the key target points $\tilde{\mathbf{P}} = [\tilde{\mathbf{v}}_1 ... \tilde{\mathbf{v}}_n]$ in the matrix form. According to Eq. (1), each point $\mathbf{v}_i$ is deformed by its 4 neighboring nodes. Thus we define two matrices $\mathbf{M}$ and $\mathbf{C}$ (Eq. (8)):

$$\underset{3m \times n}{\mathbf{M}} = \begin{bmatrix} \cdots & & \cdots \\ \cdots & \mathbf{e}_{n,1} & \cdots \\ \cdots & & \cdots \\ \cdots & \mathbf{e}_{n,2} & \cdots \\ \cdots & & \cdots \\ \cdots & \mathbf{e}_{n,3} & \cdots \\ \cdots & & \cdots \\ \cdots & \mathbf{e}_{n,4} & \cdots \\ \cdots & & \cdots \end{bmatrix} \underset{m \times n}{\mathbf{C}} = \begin{bmatrix} \cdots & & \cdots \\ \cdots & \omega_{\mathbf{N}(n,1)} & \cdots \\ \cdots & & \cdots \\ \cdots & \omega_{\mathbf{N}(n,2)} & \cdots \\ \cdots & & \cdots \\ \cdots & \omega_{\mathbf{N}(n,3)} & \cdots \\ \cdots & & \cdots \\ \cdots & \omega_{\mathbf{N}(n,4)} & \cdots \\ \cdots & & \cdots \end{bmatrix}, \tag{8}$$

where $\mathbf{e}_{n,k} = \omega_{\mathbf{N}(n,k)} * (\mathbf{v}_i - \mathbf{g}_{\mathbf{N}(n,k)})$ and $\mathbf{N}(i,j)(j = 1,2,3,4)$ is the set of neighboring node to point $\mathbf{v}_i$. Note that different source points have different topology, thus the location of non-zero elements are not aligned well in each column. The sum of each column in matrix $\mathbf{C}$ is 1 due to the point to node topology. In Eq. (7), a source point $\mathbf{v}_i$ is transformed by its 4 neighboring node which yields 4 non-zero elements in $\mathbf{M}$ and $\mathbf{C}$. The sum of all weight $\omega_j(\mathbf{v}_i)$ is 1. We arrange the parameters of ED nodes $\mathbf{A}_i$ and $\mathbf{t}_i$ in the following form:

$$\mathbf{\Lambda} = \begin{pmatrix} \mathbf{A}_1 & \cdots & \mathbf{A}_m. \end{pmatrix} \tag{9}$$

$$\mathbf{T} = \begin{pmatrix} \mathbf{t}_1 + \mathbf{g}_1 & \cdots & \mathbf{t}_m + \mathbf{g}_m \end{pmatrix}. \tag{10}$$

Then Eq. (7) takes the following form:

$$E_{data} = ||\mathbf{R}_c \cdot [\mathbf{\Lambda} \cdot \mathbf{M} + \mathbf{T} \cdot \mathbf{C}] + \mathbf{T}_c \otimes \mathbf{1} - \tilde{\mathbf{P}}||_F^2. \tag{11}$$

where $\otimes$ is the kronecker product. $\mathbf{1}$ is $1 \times n$ vector of ones. And $||\cdot||_F^2$ is the Frobenius norm. Eq. (11) can be written compactly in the following form:

$$E_{data} = ||\mathbf{R}_c \begin{pmatrix} \mathbf{\Lambda} & \mathbf{T} \end{pmatrix} \begin{pmatrix} \mathbf{M} \\ \mathbf{C} \end{pmatrix} + \mathbf{T}_c \otimes \mathbf{1} - \tilde{\mathbf{P}}||_F^2. \tag{12}$$

If we further define $\mathbf{\Pi} = [\mathbf{M}^T \ \mathbf{C}^T]$ and $\mathbf{\Phi} = [\mathbf{\Lambda} \ \mathbf{T}]^T$, Eq. (12) can be formed as follows:

$$E_{data} = ||\mathbf{R}_c[\mathbf{\Pi}\mathbf{\Phi}]^T + \mathbf{T}_c \otimes \mathbf{1} - \tilde{\mathbf{P}}||_F^2. \tag{13}$$

The property of Jacobian of E_{data} is determined by $\mathbf{\Pi}$.

B. Sparsity in ED graph formulation

It is natural to solve Eq. (13) in batch. As the number of vertices increases, the dimension of Jacobian relating to state $\mathbf{\Phi}$ increase dramatically. Fortunately, we can explore the fact that only partial ED nodes are connected to currently visible points, due to the limited Field Of View (FOV) of the camera. See Fig. 2 as an example of constant FOV of depth image. The model keeps expanding while the target depth remains in small size. A typical ED node and target depth relationship is illustrated in Fig. 2(d); 2/3 of the nodes are not within depth FOV resulting no contribution to E_{data}. Fig. 3 shows a typical Jacobian of the cost function. In E_{data} block, the shadow region indicates nodes connected to points (**PR** nodes) while zero block shows the nodes (**PI** nodes) connecting to inactive points. In this paper, we will exploit the sparity of PR nodes in the zero bocks.

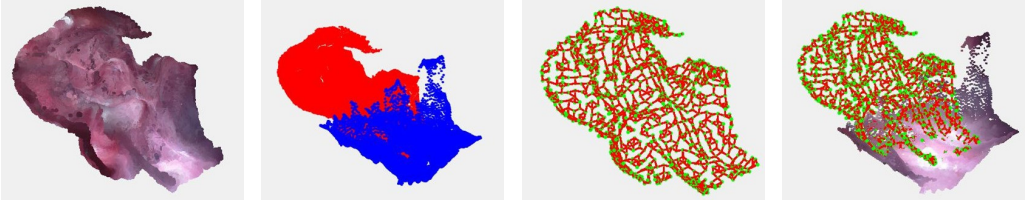
The same sparsity also applies to Eq. (8). By rearranging matrix $\mathbf{\Pi}$ from Fig. 3(a) to Fig. 3(b), we achieve a new Jacobian with zero block. Using this new matrix, Eq. (13) writes:

$$\begin{aligned} E_{data} &= ||\mathbf{R}_c[\mathbf{\Pi}\mathbf{\Phi}]^T + \mathbf{T}_c \otimes \mathbf{1} - \tilde{\mathbf{P}}||_F^2 \\ &= ||\mathbf{R}_c[\begin{pmatrix} \mathbf{\Pi}' & \mathbb{O} \end{pmatrix} \begin{pmatrix} \mathbf{\Phi}_1 \\ \mathbf{\Phi}_2 \end{pmatrix}]^T + \mathbf{T}_c \otimes \mathbf{1} - \tilde{\mathbf{P}}||_F^2, \end{aligned} \tag{14}$$

where $\mathbf{\Phi}_1$ contains $\mathbf{A}_j$ and $\mathbf{t}_j$ of **PR** node set and $\mathbf{\Phi}_2$ contains $\mathbf{A}_j$ and $\mathbf{t}_j$ of **PI** node set. $\mathbf{\Pi}'$ is the subset of $\mathbf{\Pi}$ relating to **PR** nodes (the shadow region in Fig. 3).

C. Lossy two level optimization

Explained in Section III-B, the size of the **PR** nodes is almost constant due to the limited size of depth image in expanding scenario. For instance, the point cloud generated from Hamlyn dataset [26] (grabbed from monitor) is $320 \times 240 = 76800$ at most. **As the model grows, the total number of nodes in ED graph is increasing but the number of nodes is almost constant**. Taking advantage of Eq. (14), the optimization can be divided into two levels: the optimization of **PR** nodes $\mathbf{\Phi}_1$ and the optimization of the rest **PI** nodes $\mathbf{\Phi}_2$.



(a) (b) (c) (d)

Fig. 2: Illustrated are the visible points and the node graph. (a) is the latest reconstruction. (b) shows both the model (red) and the target depth (blue). (c) is the ED nodes and the edges. (d) presents the ED nodes and the target depth.

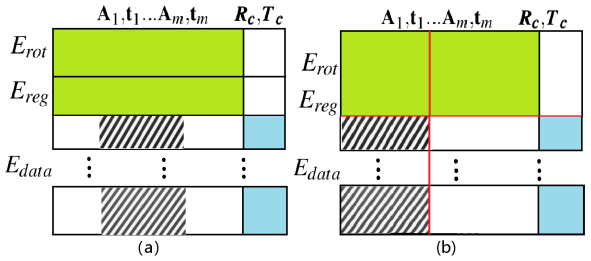


Fig. 3: (a) is the Jacobian matrix while (b) is the re-ordered Jacobian. Empty blocks are consisted of zero elements.

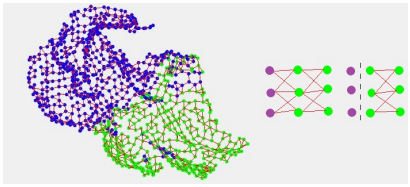


Fig. 4: Left shows two types of nodes and edges. Green nodes are the **PR** nodes and purple nodes are **PI** nodes. The right indicates the connections of **PR** and **PI** nodes. Left is the full connection while the **PR** and **PI** connections are cut in our **Level I** optimization

Specifically, we first optimize ($\mathbf{\Phi}_1$, $\mathbf{R}_c$ and $\mathbf{T}_c$) by fixing $\mathbf{\Phi}_2$ in (**Level I**) optimization, to obtain an estimation of $\mathbf{\Phi}_1$, $\mathbf{R}_c$ and $\mathbf{T}_c$. Then the value of the parameters obtained from **Level I**, will be fixed in **Level II**, together with the two soft constraints E_{rot} and E_{reg} to optimize the parameter $\mathbf{\Phi}_2$. We explicitly enforce this idea by formulating following energy function:

$$\underset{\mathbf{R}_c,\mathbf{T}_c,\mathbf{\Phi_1}}{\operatorname{argmin}} \ \omega_{rot}\tilde{E}_{rot} + \omega_{reg}\tilde{E}_{reg} + \omega_{data}E_{data} \tag{15}$$

$$\underset{\mathbf{\Phi}_2}{\operatorname{argmin}} \ \omega_{rot}E_{rot} + \omega_{reg}E_{reg} \tag{16}$$

Eq. (15) and Eq. (16) are the **Level I** and **Level II** energy functions, where $\tilde{E}_{rot}$ and $\tilde{E}_{reg}$ are the curtailed energy function of E_{rot} and E_{reg} containing Φ_1. In other words, the size of Eq. (15) is only related to the size of **PR** nodes. Therefore, the computational complexity in **Level I** is reduced from $O(n^2)$ to constant $O(1)$ thanks to the constant size of $\mathbf{\Phi}_1$. Although optimizing **Level II** is still $O(n^2)$, considering the scale of data term E_{data} is far larger than the rest, the computational cost in **Level II** is relatively low. Note that the new strategy keeps the time consuming step **Level I** constant while **Level II** still $O(n^2)$. **However, this won't cause any issue since the size of Level II is almost negligible compared with Level I**.

D. Connection with marginalization and information loss

In this section, we will draw the connection of the proposed two level optimization method with an exact marginalization based method. The analysis will show that the information loss is very low, illustrating the feasibility of the decoupled optimization Eq. (15) and Eq. (16).

When generating Eq. (14), the Jacobian shown in Fig. 3 is re-ordered by classifying $[\mathbf{A}_1, \mathbf{t}_1...\mathbf{A}_m, \mathbf{t}_m]$ into **PR** nodes $\mathbf{\Phi}_1$ and **PI** nodes $\mathbf{\Phi}_2$. The state $[\mathbf{R}_c, \mathbf{T}_c, \mathbf{A}_1, \mathbf{t}_1...\mathbf{A}_m, \mathbf{t}_m]$ are classified as $\mathbf{x}_c(\mathbf{R}_c, \mathbf{T}_c, \mathbf{A}_1, \mathbf{t}_1...\mathbf{A}_k, \mathbf{t}_k)$ and $\mathbf{x}_f(\mathbf{A}_{k+1}, \mathbf{t}_{k+1}...\mathbf{A}_m, \mathbf{t}_m)$. Fig. 3 shows the Jacobian in the new order. The first two term are combined due to their sparsity because E_{rot} and E_{reg} are constraints between ED nodes, i.e. unrelated to feature points. The only full block in Fig. 3 is E_{data} w.r.t $\mathbf{\Phi}_1$ (shadow region), in specific $\frac{\partial \mathbf{J}_2}{\partial \mathbf{x}_c}$. Let us write down the Jacobian and Hessian as,

$$\mathcal{J} = \begin{bmatrix} \frac{\partial \mathbf{J}_1}{\partial \mathbf{x}_c} & \frac{\partial \mathbf{J}_1}{\partial \mathbf{x}_f} \\ \frac{\partial \mathbf{J}_2}{\partial \mathbf{x}_c} & \mathbb{O} \end{bmatrix} \overset{\text{def}}{=} \begin{bmatrix} \mathbf{J}_{1c} & \mathbf{J}_{1f} \\ \mathbf{J}_{2c} & \mathbb{O} \end{bmatrix} \tag{17}$$

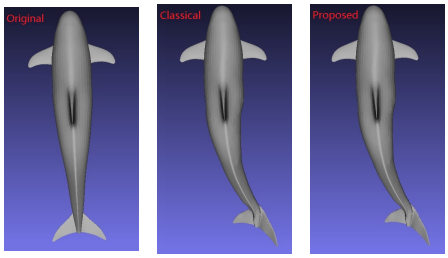
$$\mathcal{H} = \begin{bmatrix} \mathbf{J}_{1c}^T\mathbf{J}_{1c} + \mathbf{J}_{2c}^T\mathbf{J}_{2c} & \mathbf{J}_{1c}^T\mathbf{J}_{1f} \\ \mathbf{J}_{1f}^T\mathbf{J}_{1c} & \mathbf{J}_{1f}^T\mathbf{J}_{1f} \end{bmatrix} \overset{\text{def}}{=} \begin{bmatrix} \mathbf{\Lambda}_{cc} & \mathbf{\Lambda}_{cf} \\ \mathbf{\Lambda}_{cf}^T & \mathbf{\Lambda}_{ff} \end{bmatrix} \tag{18}$$

$\frac{\partial \mathbf{J}_2}{\partial \mathbf{x}_c}$ is the only dense block in Hessian $\mathcal{H}$. Let us use Schur complement [20] and separate the optimization as follows:

$$\begin{bmatrix} \mathbf{\Lambda}_{cc} & \mathbf{\Lambda}_{cf} \\ \mathbf{\Lambda}_{cf}^T & \mathbf{\Lambda}_{ff} \end{bmatrix} \begin{bmatrix} \mathbf{x}_c \\ \mathbf{x}_f \end{bmatrix} \overset{\text{def}}{=} \begin{bmatrix} \mathbf{y}_c \\ \mathbf{y}_f \end{bmatrix} \tag{19}$$

$$\begin{bmatrix} \mathbf{\Lambda}_{cc} - \mathbf{\Lambda}_{cf}\mathbf{\Lambda}_{ff}^{-1}\mathbf{\Lambda}_{cf}^T & \mathbb{O} \\ \mathbf{\Lambda}_{cf}^T & \mathbf{\Lambda}_{ff} \end{bmatrix} \begin{bmatrix} \mathbf{x}_c \\ \mathbf{x}_f \end{bmatrix} = \begin{bmatrix} \mathbf{y}_c - \mathbf{\Lambda}_{cf}\mathbf{\Lambda}_{ff}^{-1}\mathbf{y}_f \\ \mathbf{y}_f \end{bmatrix} \tag{20}$$

Using the Schur complement, we can solve $\mathbf{x}_c$ independent of $\mathbf{x}_f$. The computation of $\mathbf{x}_c$ (including $\mathbf{R}_c$, $\mathbf{T}_c$ and $\mathbf{\Phi}_1$) is constant (explained in Section III-C). After solving $\mathbf{x}_c$, the optimization of $\mathbf{x}_f$ is quite cheap as the $\mathbf{\Lambda}_{ff} = (\frac{\mathbf{J}_1}{\partial \mathbf{x}_f})^T(\frac{\mathbf{J}_1}{\partial \mathbf{x}_f})$ is only related to E_{rot} and E_{reg}. The sparsity of Hessian and



(a) (b) (c)

Fig. 5: Qualitative comparisons of our strategy and original ED based deformation. (a) is the original dolphin mesh. We show the result of deformed shape (b) along with the result of classical ED deformation (c).

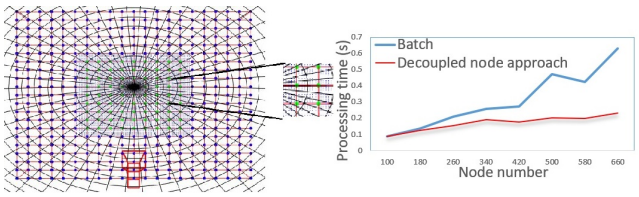


Fig. 6: An example model with static camera. One step optimization from plain surface model (tiny dots) to warped surface (grid). **PI** nodes are in blue and **PR** nodes are in green.

small number of nodes ($n \gg m$) makes the time of solving $\mathbf{x}_f$ much less.

The standard lossless formulation in Eq. (6) can be converted to the the two-step lossy method with reasonable approximation. In Eq. (3), the first term E_{rot} is the sum of error of affine transformation (Eq. (4)) making the Jacobian strictly diagonal, while the second term E_{reg} defines the transformation error among ED nodes (Eq. (6)). The major part $\mathbf{A}_j(\mathbf{g}_k - \mathbf{g}_j) + \mathbf{g}_j + \mathbf{t}_j$ is also within one node j except the very last $-(\mathbf{g}_k + \mathbf{t}_k)$. The last variable $\mathbf{t}_k$ makes the Jacobian not strictly diagonal. We come up with an idea that by ignoring Jacobian of $\mathbf{t}_k$, and reordering two diagonal Jacobians, $[\frac{\partial \mathbf{J}_1}{\partial \mathbf{x}_c} \frac{\partial \mathbf{J}_1}{\partial \mathbf{x}_f}]$ becomes diagonal and $\mathbf{\Lambda}_{cf} = (\frac{\partial \mathbf{J}_1}{\partial \mathbf{x}_c})^T(\frac{\partial \mathbf{J}_1}{\partial \mathbf{x}_f}) = 0$. In this case, Eq. (20) can be separated into two equations, with respect to $\mathbf{x}_c$ and $\mathbf{x}_f$ respectively, which also correspond to the Level I and Level II of the proposed method.

Fig. 4 visualizes the feasibility of the lossy decoupled optimization approach in geometry. **PR** nodes (green) are the only nodes connected to visible points and contribute to E_{data}. All **PI** nodes (purple) merely share edges with **PR** nodes and are passively deformed according to the behaviors of **PR** nodes. Equivalently, the inter-nodes relation in the Jacobian of E_{reg} shows these connections. In view of this, our lossy decoupled optimization approach first optimize **PR** nodes and then estimate **PI** nodes.

In conclusion, solving energy function Eq. (3) by ignoring Jacobian of $\mathbf{t}_k$ is equivalent to the proposed decoupled optimization in Eq. (15) and Eq. (16).

The information loss of the proposed approach is relatively low. Fig. 4 illustrates the connection between **PR** nodes and **PI** nodes is weak on the boundary, in contrast with the dense connections among **PR** nodes. It also demonstrates how the connection between **PR** and **PI** nodes are removed in **Level I**, and the connection between $\frac{\partial \mathbf{J}_1}{\partial \mathbf{x}_c}$ and $\frac{\partial \mathbf{J}_1}{\partial \mathbf{x}_f}$ are removed resulting in $\mathbf{\Lambda}_{cf} = (\frac{\partial \mathbf{J}_1}{\partial \mathbf{x}_c})^T(\frac{\partial \mathbf{J}_1}{\partial \mathbf{x}_f}) = \mathbb{O}$. The information between two **PR** nodes is strong while that among the **PI** nodes is weak. The weak information is neglected in **Level I**, attributing to relatively low information loss in optimization process.

IV. Results and Discussion

Our goal is to have both qualitative assessments and quantitative comparisons between the original ED graph optimization and the proposed method. For qualitative comparison, we show the sacrificed accuracy has few impacts on final reconstructed map. A dolphin model is downloaded from turbosquid (https://www.turbosquid.com). w.r.t quantitative test in SLAM, both methods are compared on a synthetic dataset and datasets from [26], where we choose three in-vivo stereo videos with deformation and rigid scope movement. The experiments are conducted on the same hardware and sorfware setting of MIS-SLAM [25]. The module of state estimation in MIS-SLAM is modified to the proposed approach. Note that an iterative solver, i.e. the preconditioned conjugate gradient method, is employed to solve the resulting linear systems, as it provides a way of parallel computing on GPU.

A. Qualitative ED deformation comparisons

Fig. 5 shows a qualitative comparison. It is not aiming as a proof of the superiority of the proposed method over the original ED graph method. [8] has already claimed real-time implementation on CPU as well as very nice results. Aiming at speeding up deformable SLAM application, the qualitative result of our lossy decoupled approach is not comparable to the batch estimation of ED. However, we want to illustrate that the proposed method can achieve a similar result and the difference is not visible to the naked eye or difficult to discern. Fig. 5 confirms that the deformed shape performed by our approach do actually have similar result. It looks similar to the original ED graph partially due to simple topology of dolphin. Other complex models like human results in a visible but not very apparent difference. Although the decoupled optimization works well, the results can be much worse than ED when the nodes are too sparse. The deformation is dependent on **PR** nodes and the insufficiency of **PR** nodes (or nodes in conventional one step ED) causes vertex deformation that the expected target is not reached.

B. Time complexity comparisons

Fig. 6 show a tiny one-step simulation and the result. Tractable time consumption remains small because **PR** node does not change.

We compare the original MIS-SLAM [25] with the improved version. Fig. 7 illustrates the running time for three

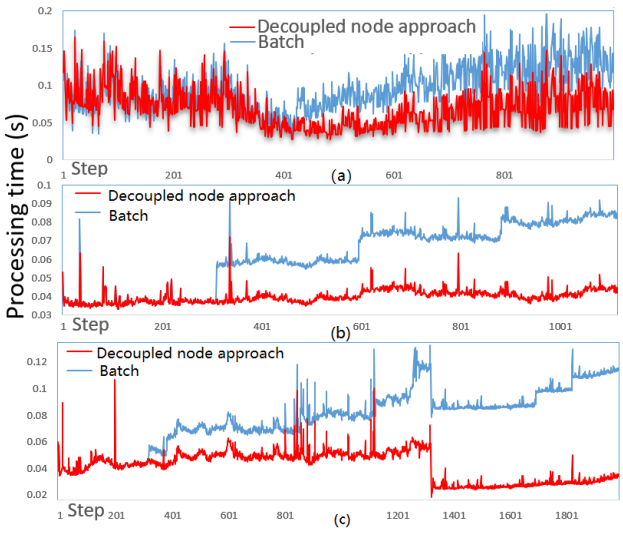


Fig. 7: Processing time comparsions of model 6 (a), 20 (b) and 21 (c) in Hamlyn dataset. Blue lines are the batch optimization and red lines are our nodes decoupled optimization. We cannot illustrate **Level I** and **Level II** separately due to time consumption of **Level II** is extremely low.

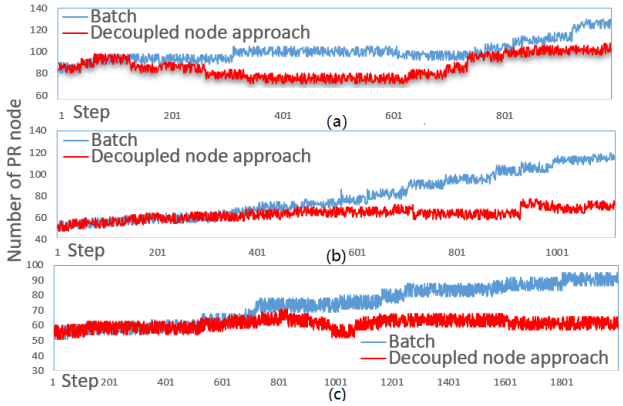


Fig. 8: Optimizing nodes comparisons in first level computation of model 6 (a), 20 (b) and 21 (c) in Hamlyn dataset. The red lines are the result of our decoupled optimization strategy while the blue lines are the original batch strategy.

Hamlyn datasets (model 6, 20 and 21). In all scenarios decoupled optimization yields better efficiency than batch processing especially in case of long term process (the last dataset in Fig. 7). In the first few steps, the robot is steady and ED graph is not expanding significantly. This attributes to the similar processing time in the first few hundred steps. When the robot moves, the ED graph expands intensively and processing time increases abruptly in state optimization. In contrast, by limiting the node graph, decoupled optimization keeps time consumption stable due to the constant **PR** node scale. The attached video shows the range of movement in model 6 is much smaller than model 20 and 21. That's the main reason the proposed method does not improve model 6 too much. Besides, note that the time consumption in 1300 of Fig. 7 (c) drops down abruptly, it is because the number of visible points (in Eq. (7)) decreases and the computation reduces significantly. Overall, as the environment gets larger, our approach keeps much lower time consumption.

C. Accuracy comparisons

The lossy decoupled optimization strategy inevitably attributes the loss of accuracy. Section IV-A shows the quality of the deformed map is well preserved in ED deformation process. Moreover, we compare the proposed method and the original one on the same parameters and weights of terms in SLAM application (MIS-SLAM). Different from arbitrary key points matching in ED deformation formulation (Eq. (7)), in SLAM application the E_{data} is in the form of model-to-depth scan matching like point-to-plane ICP. For quantitative validation, we measure the point-to-point distance of deforming map and target scan. For direct validation to ground truth, three synthetic data (heart, right kidney, and stomach) are generated by deliberately deforming models from CT scanned phantom. As a compliment, the three laparoscopy datasets from Hamlyn are tested (shown in Fig. 8), but only the back-projection error in each iteration is available since there's no ground truth. In the batch approach, the average distance of back-projection registration of the three simulation scenarios is 0.18mm (model 6), 0.13mm (model 20) and 0.12mm (model 21). While dataset with ground truth (Hamlyn dataset 10/11) achieves 0.08mm, 0.21mm (average errors). With our decoupled optimization approach, we achieve 0.31mm (model 6), 0.26mm (model 20), 0.22mm (model 21) and 0.14mm, 0.29mm errors. The attached video shows no big difference in terms of structure and texture.

We also test the average error. On top of the in-vivo dataset, synthetic data are generated. Please refer to the attached video for more details. Roughly speaking, the proposed method sacrifices 32% accuracy in exchange for 50% speed gains.

V. Conclusion

We propose a decoupled approach for ED graph optimization that reduces computational complexity from $O(n^2)$ to near $O(1)$. The decoupled optimization structure achieves faster computation in scenario of expanding environment. Our strategy sacrifices small amount of accuracy in exchange for near-constant processing speed. The constant computation complexity of the lossy strategy should have great potential in ED graph based SLAM in unbounded map scenario.

The node marginalization strategy in this paper, however, is straightforward, in the sense that it only classifies nodes based on the node-vertex connectivity. It's reasonable because different from the pose graph, ED graph is parallelized in GPU since the time consumption requirement is more strict. However, it remains to be of great interest to test if pose graph pruning method like Kullback–Liebler divergence outperforms the proposed work while remains acceptable consumption in GPU environment.

REFERENCES

[1] O. G. Grasa, J. Civera, and J. Montiel, "Ekf monocular slam with relocalization for laparoscopic sequences," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pp. 4816–4821, IEEE, 2011.

[2] B. Lin, A. Johnson, X. Qian, J. Sanchez, and Y. Sun, "Simultaneous tracking, 3d reconstruction and deforming point detection for stereoscope guided surgery," in *Augmented Reality Environments for Medical Imaging and Computer-Assisted Interventions*, pp. 35–44, Springer, 2013.

[3] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[4] N. Mahmoud, I. Cirauqui, A. Hostettler, C. Doignon, L. Soler, J. Marescaux, and J. Montiel, "Orbslam-based endoscope tracking and 3d reconstruction," in *International Workshop on Computer-Assisted and Robotic Endoscopy*, pp. 72–83, Springer, 2016.

[5] N. Mahmoud, A. Hostettler, T. Collins, L. Soler, C. Doignon, and J. Montiel, "Slam based quasi dense reconstruction for minimally invasive surgery scenes," *arXiv preprint arXiv:1705.09107*, 2017.

[6] L. Chen, W. Tang, N. W. John, T. R. Wan, and J. J. Zhang, "Slam-based dense surface reconstruction in monocular minimally invasive surgery and its application to augmented reality," *Computer methods and programs in biomedicine*, vol. 158, pp. 135–146, 2018.

[7] A. Marmol, A. Banach, and T. Peynot, "Dense-arthroslam: dense intra-articular 3d reconstruction with robust localization prior for arthroscopy," *IEEE Robotics and Automation Letters*, 2019.

[8] R. W. Sumner, J. Schmid, and M. Pauly, "Embedded deformation for shape manipulation," *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3, p. 80, 2007.

[9] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "Elasticfusion: Real-time dense slam and light source estimation," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1697–1716, 2016.

[10] J. Song, J. Wang, L. Zhao, S. Huang, and G. Dissanayake, "Dynamic reconstruction of deformable soft-tissue with stereo scope in minimal invasive surgery," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 155–162, 2018.

[11] R. A. Newcombe, D. Fox, and S. M. Seitz, "Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 343–352, 2015.

[12] M. Innmann, M. Zollhöfer, M. Nießner, C. Theobalt, and M. Stamminger, "Volumedeform: Real-time volumetric non-rigid reconstruction," in *European Conference on Computer Vision*, pp. 362–379, Springer, 2016.

[13] M. Dou, S. Khamis, *et al.*, "Fusion4d: real-time performance capture of challenging scenes," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, p. 114, 2016.

[14] K. Guo, F. Xu, T. Yu, X. Liu, Q. Dai, and Y. Liu, "Real-time geometry, albedo, and motion reconstruction using a single rgb-d camera," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 3, p. 32, 2017.

[15] M. Dou, P. Davidson, S. R. Fanello, S. Khamis, A. Kowdle, C. Rhemann, V. Tankovich, and S. Izadi, "Motion2fusion: real-time volumetric performance capture," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 6, p. 246, 2017.

[16] A. Agudo, B. Calvo, and J. Montiel, "3d reconstruction of non-rigid surfaces in real-time using wedge elements," in *European Conference on Computer Vision*, pp. 113–122, Springer, 2012.

[17] A. Petit and S. Cotin, "Environment-aware non-rigid registration in surgery using physics-based simulation," in *Asian Conference on Computer Vision (ACCV) Workshops*, 2018.

[18] J. Lamarca and J. Montiel, "Camera tracking for slam in deformable maps," in *The European Conference on Computer Vision (ECCV) Workshops*, September 2018.

[19] J. Lamarca, S. Parashar, A. Bartoli, and J. Montiel, "Defslam: Tracking and mapping of deforming scenes from monocular sequences," *arXiv preprint arXiv:1908.08918*, 2019.

[20] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[21] N. Carlevaris-Bianco, M. Kaess, and R. M. Eustice, "Generic node removal for factor-graph slam," *IEEE Transactions on Robotics*, vol. 30, no. 6, pp. 1371–1385, 2014.

[22] K. Eckenhoff, L. Paull, and G. Huang, "Decoupled, consistent node removal and edge sparsification for graph-based slam," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3275–3282, IEEE, 2016.

[23] J. Vallvé, J. Solà, and J. Andrade-Cetto, "Graph slam sparsification with populated topologies using factor descent optimization," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1322–1329, 2018.

[24] D. J. MacKay and D. J. Mac Kay, *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[25] J. Song, J. Wang, L. Zhao, S. Huang, and G. Dissanayake, "Mis-slam: Real-time large-scale dense deformable slam system in minimal invasive surgery based on heterogeneous computing," *IEEE Robotics and Automation Letters*, vol. 3, pp. 4068–4075, Oct 2018.

[26] S. Giannarou, M. Visentini-Scarzanella, and G.-Z. Yang, "Probabilistic tracking of affine-invariant anisotropic regions," *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 130–143, 2013.