Non-Rigid Structure-from-Motion via Differential Geometry with Recoverable Conformal Scale

Yongbo Chen, Yanhao Zhang, Shaifali Parashar, Liang Zhao, Shoudong Huang

Non-Rigid Structure-from-Motion via Differential Geometry with Recoverable Conformal Scale

Yongbo Chen, *Member, IEEE*, Yanhao Zhang, Shaifali Parashar, Liang Zhao, *Member, IEEE*, Shoudong Huang, *Senior Member, IEEE*

***Abstract*—Non-rigid structure-from-motion (NRSfM), a promising technique for addressing the mapping challenges in monocular visual deformable simultaneous localization and mapping (SLAM), has attracted growing attention. We introduce a novel method, called Con-NRSfM, for NRSfM under conformal deformations, encompassing isometric deformations as a subset. Our approach performs point-wise reconstruction using 2D selected image warps optimized through a graph-based framework. Unlike existing methods that rely on strict assumptions, such as locally planar surfaces or locally linear deformations, and fail to recover the conformal scale, our method eliminates these constraints and accurately computes the local conformal scale. Additionally, our framework decouples constraints on depth and conformal scale, which are inseparable in other approaches, enabling more precise depth estimation. To address the sensitivity of the formulated problem, we employ a parallel separable iterative optimization strategy. Furthermore, a self-supervised learning framework, utilizing an encoder-decoder network, is incorporated to generate dense 3D point clouds with texture. Simulation and experimental results using both synthetic and real datasets demonstrate that our method surpasses existing approaches in terms of reconstruction accuracy and robustness. The code for the proposed method will be made publicly available on the project website: https://sites.google.com/view/con-nrsfm.**

***Index Terms*—Deformable SLAM, NRSfM, conformal deformations, parallel separable iterative optimization, encoder and decoder network.**

I. Introduction

Manuscript received January 14, 2025; revised May 26, 2025; accepted September 18, 2025. Date of publication; date of current version. This work was supported by the Australia Research Council (ARC) Discovery grant Project DP120102786. This paper was recommended for publication by Editor Civera, Javier and Editor-in-Chief Burgard, Wolfram upon evaluation of the reviewers' comments. (Corresponding author: Yanhao Zhang.)

Yongbo Chen was with the Robotics Institute, University of Technology Sydney, Australia, and is now with the School of Automation and Intelligent Sensing, Shanghai Jiao Tong University, Shanghai, 200240, People's Republic of China (e-mail: shjtdx_cyb@sjtu.edu.cn).

Yanhao Zhang and Shoudong Huang are with the Robotics Institute, University of Technology Sydney, Ultimo, NSW 2007, Australia (e-mail: yanhaozhang1991@gmail.com, Shoudong.Huang@uts.edu.au).

Shaifali Parashar is now with Institut National des Sciences Appliquées de Lyon (LIRIS, INSA-Lyon), 69100 Villeurbanne, France. (e-mail: shaifali.parashar@liris.cnrs.fr)

Liang Zhao is with the School of Informatics, University of Edinburgh, Edinburgh, UK (e-mail: Liang.Zhao@ed.ac.uk).

This work has been accepted for publication in the IEEE Transactions on Robotics (T-RO).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org.

Digital Object Identifier.

NON-RIGID structure-from-motion (NRSfM) addresses the challenge of reconstructing the 3D shapes of deforming objects from multiple calibrated monocular images. It is a fundamental problem in 3D computer vision, with applications ranging from entertainment [1] to modern surgery [2]. The NRSfM algorithm reconstructs 3D shapes in local camera coordinates, inherently intertwining camera motion with object deformations. This concept closely aligns with deformable visual SLAM, which aims to localize a robot and map its environment, even in dynamically deforming scenarios. NRSfM has the potential to play a pivotal role in overcoming the mapping challenges associated with deformable visual SLAM [2]. By integrating tools for robot pose estimation, NRSfM can enhance mapping consistency through information fusion, paving the way for significant advancements in deformable visual SLAM research [3], [4].

Naturally, the NRSfM problem is unsolvable without introducing constraints or assumptions on deformations. Common approaches to deformation modeling include statistical constraints, such as low-rank shapes [5] or trajectory basis [6], and physical constraints, such as isometry (preserving local geodesics) [7], inextensibility [8], and local rigidity [10]. Among these, physical methods generally outperform statistical ones in reconstructing highly deformable objects.

Recent advancements have introduced local formulations of physical constraints, including isometry (distance-preserving) [11], conformality (angle-preserving) [12], and smoothness [13]. These methods assume surfaces to be locally planar (LP) and deformations to be locally linear (LL), expressing physical constraints in terms of local depth derivatives as unknowns. This approach offers several advantages: it is often more accurate, computationally efficient, and robust to missing data. However, there are two major limitations: *1) indirect depth computation:* all constraints are derived on local normals (expressed with depth derivatives) and the depth is only obtained by interpolating them. Such a formulation leads to weaker constraints on the surface geometry which affects their performance. *2) LP and LL assumptions:* While LP and LL assumptions are valid for continuous surfaces, these methods often operate on sparsely sampled points (100–200 points) on the surface. This sparse sampling compromises the accuracy of local derivative computations, leading to degraded performance, especially in scenarios involving strong deformations, such as surface bending. These limitations emphasize the need for further refinement to improve accuracy and robustness in handling complex deformations.

In this paper, we extend the existing local formulations of isometry/conformality to overcome these limitations and

develop a novel NRSfM method tailored for conformal deformations. The main contributions of this paper are:

- **Rotational invariance under conformal deformation**: We establish that connections and moving frames across surfaces preserve distinct invariance properties under different types of deformations. Critically, we prove that connections under conformal deformation preserve rotational invariance, enabling the decoupling of conformal scale and depth estimation. This introduces a novel and strict theoretical constraint on the geometry of deformable surfaces. As a result, we derive physical constraints in terms of conformal scale, depth, and normals (expressed as depth derivatives).
- **Relaxation of LL and LP assumptions:** Unlike prior approaches, we do not impose LL or LP assumptions. Instead, we define physical constraints that hold up to the second-order derivatives of local depth, introducing additional variables to better align the formulation with real-world scenarios. To solve this, we propose a parallel, separable, iterative optimization algorithm that independently recovers depth and normals (depth derivatives). The algorithm is robust to initialization and offers acceptable computational complexity.

Meanwhile, we develop an encoder-decoder neural network trained using a self-supervised learning approach on simulated datasets. This network reconstructs 3D dense point clouds with texture from the normal field, offering a more comprehensive representation of deformable surfaces. Our experiments show that our proposed method, Con-NRSfM, outperforms the existing state-of-the-art (SOTA) methods, especially when the deformations are non-isometric and/or cause strong bending.

II. Related work

A. NRSfM methods

We review the existing NRSfM methods in terms of the deformation modeling they impose.

i) Statistical constraints: Starting from [10], [15], the researches focus on the factorization-based methods with low-rank assumptions. They consider instantaneous 3D shape of a deforming object to be a linear combination of a small size shape [16], [17], trajectory [18], or force basis [19], [20]. Assuming that deforming shapes are viewed under affine projections [8], the additional low-dimensional priors include the trajectory basis priors [18], non-linear modeling [21], spatial smoothness [22], spatio-temporal smoothness [23], temporal smoothness [24], and a quadratic deformation model [25]. Recently, [28], [69], [70] extend this framework to neural networks which could learn the shape-basis from a small set of images. However, these methods work on sparse data such as simple wireframe objects. [14] introduced the first unsupervised, end-to-end NRSfM approach which uses the low-rank constraints to reconstruct dense data. These methods perform effectively on objects with simple deformations, but struggle to handle more complex or severe deformations.

ii) Global physical constraints: Global physical methods aim to preserve the physical properties of the entire point cloud and its corresponding surface. For instance, inextensibility—a convex relaxation of isometry—has been used to develop a convex second-order cone programming (SOCP) formulation for isometric NRSfM using point parameterization [8]. Similarly, the NRSfM problem has been formulated as a convex semi-definite program (SDP) [29], incorporating constraints based on the cosine law with edge parameterization to minimize non-rigidity. While these global physical methods can achieve satisfactory accuracy in certain scenarios, they often suffer from high computational complexity and limited robustness, making them less practical for real-world applications.

iii) Piece-wise physical constraints: Piece-wise physical methods approximate the shape of small surface regions using simple models, recovering depth for each region and subsequently stitching the reconstructions together to maintain surface continuity. For example, in [30] and [31], NRSfM problems are addressed using orthographic and perspective camera models, respectively, based on piece-wise rigidity. However, these methods face significant challenges due to the high computational cost of region segmentation. Additionally, defining optimal segmentation for generic surfaces is difficult, leading to inefficiencies and reduced accuracy [8]. More recently, [9] introduced a method that utilizes specular highlights in images as geometric and photometric cues, adding constraints to the previous isometric-inextensible NRSfM model [8]. This approach enhances the robustness of the reconstruction process by incorporating additional visual information.

iv) Local physical constraints: Local physical methods represent an object's 3D deformable shape in each image as a smooth Riemannian manifold, assuming inter-image registration (image warp) is available. These methods enforce physical constraints, including isometry [32], [33], conformality, equiareality [12], and diffeomorphism [13], to restrict deformations between surfaces. They leverage metric tensors (MT) to measure local distances, angles, and areas, and Cartan's connections (CC) to assess local curvatures. Using linear relationships between local normals across surfaces (LL and LP), these methods reduce the number of variables, enabling rapid computation of local normals. The surfaces are then reconstructed by integrating the local normals, which is computationally more intensive. The local formulation also efficiently handles noisy or missing data, reconstructing various deformable objects from videos or sparse images. Con-NRSfM extends this approach by imposing conformal constraints on local depth in addition to normals, without relying on the surface assumptions of LL and LP. A recent work [37] addresses the conformal case by relying on LP and LL assumptions using existing connection constraints to derive surface normals via a closed-form formulation. In contrast, our method theoretically relaxes these assumptions, introduces a novel connection constraint, and employs a more robust and scalable optimization framework. Table I highlights the differences between Con-NRSfM and other methods. Furthermore, Con-NRSfM introduces a parallel, separable optimization framework and extends this strategy using a deep learning network capable of recovering 3D shapes with texture.

TABLE I: Comparison of our method with existing local methods

Method	Deformation modeling	Variables	Assumptions	Constraints	Surface representation
[11]	Isometry	Normals	LP	MT, CC	Inverse-depth
[12]	Isometry, Conformality, Equiareality	Normals	LP+LL	MT, CC	Inverse-depth
[13]	Diffeomorphism	Normals	LL	CC	Depth
[37]	Isometry, Conformality	Normals	LP+LL	MT, CC	Inverse-depth
Con-NRSfM	Isometry, Conformality	Depth, Normals, Conformal scale	None	MT, CC	Depth

B. Related applications

In this section, we review and discuss the challenges and recent adaptations of NRSfM in practical scenarios, such as monocular deformable SLAM and visual endoluminal robotic navigation, to highlight its real-world significance. Monocular deformable SLAM is a novel and challenging problem where most classical SLAM methods exhibit low accuracy or even fail. This is primarily due to the absence of direct depth measurements and the violation of the rigid and static scene assumptions traditionally used for observed features. Def-SLAM [2] is the first monocular SLAM system explicitly designed for deformable environments. It constructs and incrementally updates a deformable map by incorporating key modifications into the ORB-SLAM framework [34] to handle non-rigid scenes. An isometric NRSfM approach [11] is embedded within the mapping module to generate surface templates at keyframe intervals. These templates are then aligned in the tracking module using a Shape-from-Template (SfT) method and a deformation energy function [35], enabling simultaneous estimation of both camera trajectory and surface deformations in real time. SfT methods, which reconstruct the deformed shape of an object from a single monocular image using a known 3D template in its rest configuration, are conceptually aligned with NRSfM and often achieve faster performance. As demonstrated in Def-SLAM, SfT can serve as a complementary tool to NRSfM by refining deformation estimates when the template is available through NRSfM over time. Building on similar techniques, SD-DefSLAM [33] incorporates optical flow to address the data association problem in the SLAM pipeline, significantly improving robustness and accuracy. This enhancement makes it particularly suitable for visual endoluminal robotic navigation in monocular medical imaging. More recently, in minimally invasive procedures, the work in [36] combines an embedded deformation graph (EDG) with isometric NRSfM to enable centralized optimization of both map points and camera motion over time. This approach further improves scene reconstruction accuracy in deformable soft tissue using monocular endoscopy.

III. Mathematical Modeling Preliminary

A. Notations

Throughout this paper, unless otherwise stated, bold lowercase, and bold uppercase letters are reserved for vectors and matrices, respectively. Sets and spaces are shown by uppercase letters with mathcal font. $\boldsymbol{I}_{n\times n}$ is the identity matrix with n dimension. $SO(n)$ (special orthogonal group) is defined as: $SO(n) \triangleq \{\boldsymbol{R} \in \mathbb{R}^{n\times n} : \boldsymbol{R}^\top \boldsymbol{R} = \boldsymbol{I}_{n\times n},\ \det(\boldsymbol{R}) = 1\}$. $(\star)^\top$ means the transpose of matrix and vector $\star$. $\det(\star)$ represents the determinant of the matrix $\star$. $\star \times \bullet$ means the direct product group of $\star$ and $\bullet$. For connection Γ, its superscript, first subscript, and second subscript respectively represent the row, column, and block numbers. If any of the indices is written as a variable (e.g., $i,\ j,\ k$), it indicates that we are considering the entire sub-vector or sub-matrix corresponding to all possible values of that index (all choices along this script), like $\Gamma^1_{j1} \in \mathbb{R}^{3\times 1} \subset \Gamma^i_{j1} \in \mathbb{R}^{3\times 3} \subset \Gamma^i_{jk} = \Gamma \in \mathbb{R}^{3\times 6}$. If the indices are specified as integers (e.g., 1, 2), they indicate the specific, fixed part of the group or matrix being referenced. For a mapping $\bullet$, $\star(\bullet)$ means the corresponding concepts, like connection Γ and moving frame E, applied on mapping $\bullet$.

B. Perspective Projection and Image Embedding

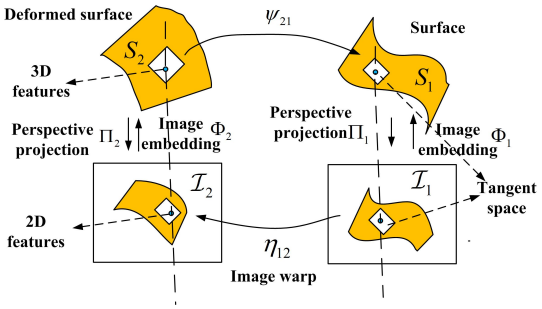


Fig. 1: A 2-view model for NRSfM.

As shown in Fig. 1, assuming that the considered multiple deformation surfaces are Riemannian manifolds, based on the calibrated monocular camera, we are considering the mappings between i-th and j-th images $\mathcal{I}_i,\ \mathcal{I}_j$ and their corresponding surfaces $\mathcal{S}_i,\ \mathcal{S}_j$. The camera mapping from the surfaces to the images is the perspective projection $\Pi_i :\ \mathcal{S}_i \in \mathbb{R}^3 \to \mathcal{I}_i \in \mathbb{R}^2$ and its inverse mapping is the image embedding, which can be assumed as the smooth function of which the variables are the pixels $(u, v)^\top$ on the images $\Phi_i :\ \mathcal{I}_i \to \mathcal{S}_i$. For the perspective camera, the perspective projection Π and the image embedding Φ are respectively denoted as[1]:

$$\begin{aligned} \boldsymbol{x} &= (u, v)^\top = \Pi(\boldsymbol{z}) = (z_1/z_3, z_2/z_3)^\top, \\ \boldsymbol{z} &= (z_1, z_2, z_3)^\top = \Phi(\boldsymbol{x}) = \beta(u, v)(u, v, 1)^\top, \end{aligned} \tag{1}$$

where $\boldsymbol{z} = (z_1, z_2, z_3)^\top,\ z_3 > 0$ is a 3D feature on the surface, $\beta(u, v)$ means the smooth depth function of the pixel $(u, v)^\top$.[2] In this paper, the mappings Π and Φ are represented using B-splines [38], which allows us to accurately obtain interpolation value, first- and second-order derivatives of these functions.

[1]For brevity, we have ignored the subscript i which means the index of the surface and the image.

[2]For brevity, we will ignore the variable (u, v) and $\beta(u, v)$ is written as β in the following text.

C. Deformation Mapping

The partial goal of the NRSfM method is to obtain the mappings $\Psi_{ij} : \mathcal{S}_i \rightarrow \mathcal{S}_j$ among the deformable surfaces. For the generic deformation, the NRSfM problem is not solvable. So as to limit the solution, the typical local assumptions on the deformation are restricted in the diffeomorphic (isometry, conformality, and equiareality). According to [13], for the sparse matched features, if Ψ_{ij} results in the diffeomorphic deformation of the surface, its approximated Jacobian matrix follows: $\mathbf{J}_{\Psi_{ij}} = \text{diag}(\lambda_1, \lambda_2, \lambda_3)\mathbf{R}$, where $\text{diag}(\lambda_1, \lambda_2, \lambda_3)$ means the scale matrix, λ_i are scalars, and $\mathbf{R} \in SO(3)$ is a rotation matrix.[3] Following the same approximation, the Jacobian matrices of the isometric and conformal deformations will be degenerated to $\mathbf{J}_{\Psi_{ij}} = \text{diag}(1,\ 1,\ 1)\mathbf{R} = \mathbf{R}$ and $\mathbf{J}_{\Psi_{ij}} = \text{diag}(\lambda,\ \lambda,\ \lambda)\mathbf{R} = \lambda\mathbf{R}$ respectively, where $\lambda > 0$ is a conformal scale factor. It is noted that the conformal scale λ is not the translation scale between different frames for monocular images corresponding to scale ambiguity. The scale ambiguity is solved by Step 3 in Section V-B.

D. Selected Image Warp

Based on the matched features obtained by the standard image matching methods, such as optical flow [39], Scale Invariant Feature Transform (SIFT) [40], and Speeded Up Robust Features (SURF) [41], we can define a dense geometric transformation functions $\eta_{ij} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ between the pair of images $\mathcal{I}_i$ and $\mathcal{I}_j$, called image warp. Our local physical method belongs to the point-wise method. The main part to share the information connection between different features is the image warp, so its accuracy and robustness are very important for the whole NRSfM framework. Each pair of images is possible to be used to compute the image warp. Considering the large computational cost of the image warp, a complete weighted undirected graph $\mathcal{G}_c = (\mathcal{V}_c,\ \mathcal{E}_c,\ w_c)$, of which the nodes, the edges, and the weighted values are the images, the image warp, and the number of the matched features, is introduced to select the well-connected sub-graph and only the image warps corresponding to the edges of the selected sub-graph are computed. The tree-connectivity $t_w(\mathcal{G}) = \det(\boldsymbol{\mathcal{L}}_w^{\mathcal{G}})$ of the connected weighted undirected sub-graph $\mathcal{G} = (\mathcal{V},\ \mathcal{E},\ w) \subseteq \mathcal{G}_c$, which is the log-determinant function of the weighted Laplacian matrix, is used as the objective function of the selection method. Given the limited number of the edges N_e, the selection problem can be considered as the problem to select maximum spanning tree,[4] if $N_e = n_c - 1$, and the famous k-edge selection problem (k-ESP, $k = N_e - n_c + 1$), if $N_e > n_c - 1$ [42]. The corresponding edge selection problem is formulated as:

$$\max_{\mathcal{G}} t_w(\mathcal{G}),\ s.t.\ |\mathcal{G}| = N_e,\ \mathcal{E} \subseteq \mathcal{E}_c,\ \mathcal{V} = \mathcal{V}_c. \tag{2}$$

This sub-graph selection problem is solved by the combination of Kruskal's maximum spanning tree algorithm (for maximum spanning tree problem, obtain $\mathcal{T}_{opt}$) and the greedy-based method with the rank-1 update (for k-ESP, obtain $\mathcal{G}_{opt}$) [7], [43]. As a sub-modular maximization problem with a cardinality constraint, the greedy-based method for the k-ESP has some performance guarantee. Then, a well-connected sub-graph $\mathcal{G}_{opt}$ is obtained and its corresponding edges are used to calculate the image warps based on 2D Schwarzian derivatives under the infinitesimal planarity assumption [10]. The obtained results provide us the first-/second-order derivatives of the image warps and their inverse mappings.

E. Moving Frames and Connections

For the image embedding, a local frame of reference, *moving frame*, $E = (\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3)$, at surface $\mathcal{S}$ expressed in terms of Φ is given by

$$E = \left(\boldsymbol{e}_1 = \frac{\partial\ \Phi}{\partial\ u},\ \boldsymbol{e}_2 = \frac{\partial\ \Phi}{\partial\ v},\ \boldsymbol{e}_3 = \boldsymbol{e}_1 \times \boldsymbol{e}_2\right). \tag{3}$$

Introducing the definition (1) of the image embedding Φ into the above equation, we have:

$$\begin{aligned} \boldsymbol{e}_1 &= \beta(y_1 u + 1, y_1 v, y_1)^\top, \\ \boldsymbol{e}_2 &= \beta(y_2 u, y_2 v + 1, y_2)^\top, \\ \boldsymbol{e}_3 &= \beta^2(-y_1, -y_2, y_1 u + y_2 v + 1)^\top, \end{aligned} \tag{4}$$

where $y_1 = \frac{1}{\beta}\frac{\partial\ \beta}{\partial\ u}$ and $y_2 = \frac{1}{\beta}\frac{\partial\ \beta}{\partial\ v}$.

The local change in *moving frames* is a linear function. The coefficients Γ^i_{jk}, known as *connections*, describe the fundamental properties of the surfaces.

$$\begin{aligned} \frac{\partial\ \boldsymbol{e}_j}{\partial\ u} &= \Gamma^1_{j1}\boldsymbol{e}_1 + \Gamma^2_{j1}\boldsymbol{e}_2 + \Gamma^3_{j1}\boldsymbol{e}_3,\ j = 1, 2, 3 \\ \frac{\partial\ \boldsymbol{e}_j}{\partial\ v} &= \Gamma^1_{j2}\boldsymbol{e}_1 + \Gamma^2_{j2}\boldsymbol{e}_2 + \Gamma^3_{j2}\boldsymbol{e}_3. \end{aligned} \tag{5}$$

F. Metric Preservation under Conformal Deformation

In Fig. 1, considering two features $\bar{\boldsymbol{x}} \in \mathcal{I}_1$ and $\boldsymbol{x} \in \mathcal{I}_2$ on two images $\mathcal{I}_1$ and $\mathcal{I}_2$, their corresponding 3D features $\bar{\boldsymbol{z}} \in \mathcal{S}_1$ and $\boldsymbol{z} \in \mathcal{S}_2$ on surfaces $\mathcal{S}_1$ and $\mathcal{S}_2$, and the mappings (image embedding $\Phi_1 :\ \bar{\boldsymbol{x}} \rightarrow \bar{\boldsymbol{z}}$ and $\Phi_2 :\ \boldsymbol{x} \rightarrow \boldsymbol{z}$; image warp $\eta_{12} :\ \bar{\boldsymbol{x}} \rightarrow \boldsymbol{x}$; deformation mapping $\Psi_{21} :\ \boldsymbol{z} \rightarrow \bar{\boldsymbol{z}}$) between them. We can get the maps and their Jacobian matrices following:

$$\Phi_1 = \Psi_{21} \circ \Phi_2 \circ \eta_{12}, \quad \mathbf{J}_{\Phi_1} = \mathbf{J}_{\Psi_{21}}\mathbf{J}_{\Phi_2}\mathbf{J}_{\eta_{12}}. \tag{6}$$

Under a local conformal deformation, we have $\mathbf{J}_{\Psi_{21}} = \lambda\mathbf{R}$. If $\lambda = 1$, the deformations are isometric. The 2D part of the moving frames of the mappings Φ_1 and $\Psi_{21} \circ \Phi_2 \circ \eta_{12}$ satisfy the following metric preservation equation [13]:

$$\mathbf{J}_{\Phi_1}{}^\top\mathbf{J}_{\Phi_1} = \lambda^2\mathbf{J}_{\eta_{12}}{}^\top\mathbf{J}_{\Phi_2}{}^\top\mathbf{J}_{\Phi_2}\mathbf{J}_{\eta_{12}}. \tag{7}$$

IV. Connections under Conformal Deformation

In this section, we present all the novel conclusions, which are different from the ones in the literature, about the connection under conformal deformation.

[3]This property follows an approximation definition of the diffeomorphic mapping assuming the off-diagonal elements to be zero. The more general diffeomorphic mapping has six degrees of freedom for the scale matrix.

[4]It is noted that, in order to recover the depth of the features in all the images, N_e needs to satisfy $N_e \geq n_c - 1$.

Assuming that the depth function β is second-order differentiable at every pixel, based on the definitions (4) and (5), we have:

$$\begin{pmatrix} \Gamma^1_{11} & \Gamma^1_{21} & \Gamma^1_{31} & \Gamma^1_{12} & \Gamma^1_{22} & \Gamma^1_{32} \\ \Gamma^2_{11} & \Gamma^2_{21} & \Gamma^2_{31} & \Gamma^2_{12} & \Gamma^2_{22} & \Gamma^2_{32} \\ \Gamma^3_{11} & \Gamma^3_{21} & \Gamma^3_{31} & \Gamma^3_{12} & \Gamma^3_{22} & \Gamma^3_{32} \end{pmatrix} = \begin{pmatrix} \beta_1 u+\beta & \beta_2 u & -\beta\beta_1 \\ \beta_1 v & \beta+\beta_2 v & -\beta\beta_2 \\ \beta_1 & \beta_2 & \beta\beta_1 u+\beta\beta_2 v+\beta^2 \end{pmatrix}^{-1} \begin{pmatrix} \beta_{11}u+2\beta_1 & \beta_2+\beta_{12}u & -\beta_1^2-\beta\beta_{11} \\ \beta_{11}v & \beta_{12}v+\beta_1 & -\beta_1\beta_2-\beta\beta_{12} \\ \beta_{11} & \beta_{12} & T_1 \\ \beta_{12}u+\beta_2 & \beta_{22}u & -\beta\beta_{12}-\beta_1\beta_2 \\ \beta_{12}v+\beta_1 & \beta_{22}v+2\beta_2 & -\beta\beta_{22}-\beta_2^2 \\ \beta_{12} & \beta_{22} & T_2 \end{pmatrix}, \tag{8}$$

where

$$\begin{aligned} T_1 &= 3\beta\beta_1+u\beta_1^2+u\beta\beta_{11}+v\beta_{12}\beta+v\beta_1\beta_2, \\ T_2 &= u\beta_2\beta_1+u\beta\beta_{12}+3\beta\beta_2+v\beta_2^2+v\beta\beta_{22}, \\ \beta_1 &= \frac{\partial\ \beta}{\partial\ u}=\beta y_1,\ \beta_2=\frac{\partial\ \beta}{\partial\ v}=\beta y_2, \\ \beta_{11} &= \frac{\partial^2\ \beta}{\partial^2\ u}=\beta y_{11},\ \beta_{22}=\frac{\partial^2\ \beta}{\partial^2\ v}=\beta y_{22}, \\ \beta_{12} &= \frac{\partial^2\ \beta}{\partial\ u\ \partial\ v}=\frac{\partial^2\ \beta}{\partial\ v\ \partial\ u}=\beta y_{12}=\beta y_{21}. \end{aligned} \tag{9}$$

Unlike the approaches in [13] and [30], which assume infinitesimally planar surfaces and simplify their formulations by neglecting second-order derivatives, our method retains these terms to enable exact computation.

Given the moving frames $\overline{E}(\Phi_1) = E(\Psi_{21}\circ\Phi_2\circ\eta_{12}) = (\bar{\boldsymbol{e}}_1^*, \bar{\boldsymbol{e}}_2^*, \bar{\boldsymbol{e}}_3^*)$, $E(\Phi_2\circ\eta_{12}) = (\boldsymbol{e}_1^*, \boldsymbol{e}_2^*, \boldsymbol{e}_3^*)$, $E(\Phi_2) = (\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3)$ on $\mathcal{S}_1$ and $\mathcal{S}_2$ with image coordinates $\bar{\boldsymbol{x}} = (\overline{u}, \overline{v})$ and $\boldsymbol{x} = (u, v)$ respectively, we write $\mathbf{J}_{\Phi_1} = (\bar{\boldsymbol{e}}_1^*, \bar{\boldsymbol{e}}_2^*)$ and $\mathbf{J}_{\Phi_2} = (\boldsymbol{e}_1, \boldsymbol{e}_2)$. Using (3), the relation between moving frames is:

$$\begin{aligned} &(\boldsymbol{e}_1^*, \boldsymbol{e}_2^*) = \mathbf{J}_{\Phi_2}\mathbf{J}_{\eta_{12}},\ \boldsymbol{e}_3^* = \boldsymbol{e}_1^*\times\boldsymbol{e}_2^* = \det(\mathbf{J}_{\eta_{12}})\boldsymbol{e}_3, \\ &E(\Phi_2\circ\eta_{12}) = (\mathbf{J}_{\Phi_2}\mathbf{J}_{\eta_{12}}, \boldsymbol{e}_3\det(\mathbf{J}_{\eta_{12}})) = E(\Phi_2)\mathbf{J}_{\eta_{3\times3}}, \\ &(\bar{\boldsymbol{e}}_1^*, \bar{\boldsymbol{e}}_2^*) = \lambda\mathbf{R}(\boldsymbol{e}_1^*, \boldsymbol{e}_2^*),\ \bar{\boldsymbol{e}}_3^* = \bar{\boldsymbol{e}}_1^*\times\bar{\boldsymbol{e}}_2^* = \mathbf{R}\lambda^2\boldsymbol{e}_3^*, \\ &\overline{E}(\Phi_1) = \mathbf{R}E(\Phi_2\circ\eta_{12})\Lambda = \mathbf{R}E(\Phi_2)\mathbf{J}_{\eta_{3\times3}}\Lambda, \end{aligned} \tag{10}$$

where $\mathbf{J}_{\eta_{3\times3}} = \mathrm{diag}(\mathbf{J}_{\eta_{12}}, \det(\mathbf{J}_{\eta_{12}}))$ and $\Lambda = \mathrm{diag}(\lambda, \lambda, \lambda^2)$. *It is noted that these constraints (especially last equation in* (10)*) are different from the constraints (6) in the related work [13].*[5]

Under the different assumptions on the deformations ($\mathbf{J}_{\Psi_{21}} = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3)\ \mathbf{R}$), we investigate the invariant relationship of the connections of the mapping Φ_1 and the composite mapping $\Phi_2\circ\eta_{12}$ and obtain the following results which are different from [13]. We write the connections in equation (8) as $\Gamma = (\Gamma^i_{j1}, \Gamma^i_{j2})$.

[5] [13] assumes deformations to be infinitesimally linear. In the formulation, $\mathbf{J}_{\Psi_{21}} = \Upsilon\mathbf{R}$, $\Upsilon = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3)$, $\lambda_1 \neq \lambda_2 \neq \lambda_3$ and $\lambda_1, \lambda_2, \lambda_3 \approx 1$. This can be understood as as-rigid-as-possible transformation. This assumption causes the important conclusions in [13]: $\bar{e}_3^* = \bar{e}_1^*\times\bar{e}_2^* = \Upsilon\boldsymbol{R}e_3^*$ and $\overline{E}(\Phi_1) = \Upsilon\boldsymbol{R}E(\Phi_2\circ\eta_{12})$, which is different from our formula derivation.

Claim 1. If the deformation mapping Ψ_{21} is more general, such as diffeomorphic, the connection of the mapping Φ_1 is not equal to the one of the mapping $\Phi_2\circ\eta_{12}$ unless additional special conditions hold.

$$\Gamma(\Phi_1) \neq \Gamma(\Phi_2\circ\eta_{12}), \tag{11}$$

where $\Gamma(\star)$ means the connections of the mapping $\star$. For the proof process, please refer to Appendix VIII.

Claim 1 shows that the connections of the mapping Φ_1 and $\Phi_2\circ\eta_{12}$ are non-invariant under the local generic diffeomorphic deformation without any assumptions. Based on the moving frame relations (10), we now derive the relation for connections under conformal deformations. We next show that like moving frames, connections across surfaces are related by image warps (up to first and second order) and conformal scale, λ. They are, however, invariant to the rotation $\mathbf{R}$.

Theorem 1. *For a conformal deformation, $\mathbf{J}_{\Psi_{21}} = \lambda\mathbf{R}$, the relation between connections $\Gamma(\Phi_1)$ and $\Gamma(\Phi_2\circ\eta_{12})$ is invariant to rotation $\mathbf{R}$ and is not invariant to λ, given by*

$$\begin{aligned} &\mathbf{J}_{\eta_{3\times3}}\Lambda\Gamma^i_{j1}(\Phi_1)\Lambda^{-1} = \\ &\left(\Gamma^i_{j1}(\Phi_2)\frac{\partial u}{\partial\overline{u}}+\Gamma^i_{j2}(\Phi_2)\frac{\partial v}{\partial\overline{u}}\right)\mathbf{J}_{\eta_{3\times3}}+\frac{\partial\mathbf{J}_{\eta_{3\times3}}}{\partial\overline{u}}, \\ &\mathbf{J}_{\eta_{3\times3}}\Lambda\Gamma^i_{j2}(\Phi_1)\Lambda^{-1} = \\ &\left(\Gamma^i_{j1}(\Phi_2)\frac{\partial u}{\partial\overline{v}}+\Gamma^i_{j2}(\Phi_2)\frac{\partial v}{\partial\overline{v}}\right)\mathbf{J}_{\eta_{3\times3}}+\frac{\partial\mathbf{J}_{\eta_{3\times3}}}{\partial\overline{v}}. \end{aligned} \tag{12}$$

Proof. For the proof process, please refer to Appendix IX. □

Theorem 1 shows that the relationship of the connections $\Gamma^i_{jk}(\Phi_1)$ and $\Gamma^i_{jk}(\Phi_2\circ\eta_{12})$ is rotational invariant and is only related to the conformal scale λ. We can compute the connections and the conformal scale of $\mathcal{S}_1$ from those of $\mathcal{S}_2$ using η_{12}. [11] used the connections in 2D case to find a joint conformal NRSfM under the local infinitesimally planar approximation without considering the rotation scale λ. Our goal is, however, to recover λ and get a better estimate on the local depth using the full information of the connections in 3D case and without any approximation in the differentiable order. We will exploit the result in the metric preservation (7) and Theorem 1 to develop our algorithm.

Corollary 1. *If the deformation mapping Ψ_{21} is local conformal, the second-order leading principal minors and the last elements of the connections $\Gamma^i_{jk}(\Phi_1)$ and $\Gamma^i_{jk}(\Phi_2\circ\eta_{12})$ of the mapping Φ_1 and $\Phi_2\circ\eta_{12}$ are invariant.*

Proof. For the proof process, please refer to Appendix X. □

Corollary 2. *For the conformal deformation Ψ_{21}, we can write $\Gamma^i_{j1}(\Phi_1) = (\bar{\boldsymbol{T}}^1_{kl})$, $\Gamma^i_{j2}(\Phi_1) = (\bar{\boldsymbol{T}}^2_{kl})$, $\Gamma^i_{j1}(\Phi_2) = (\boldsymbol{T}^1_{kl})$, and $\Gamma^i_{j2}(\Phi_2) = (\boldsymbol{T}^2_{kl})$, $k, l = 1, 2$ as 2×2 block matrices. Then, for two frames, only considering the rotational invari-*

ance of the connection in (12)*, the conformal scale has a closed-form solution based on the sum of squares formulation:*

$$\begin{aligned}\lambda &= \frac{1}{8}\sum_{i=1}^{4}\sum_{j=1}^{2} b_j^i \\ b_j^1 &= \frac{(\frac{\partial u}{\partial \bar{u}}\boldsymbol{T}_{21}^1 + \frac{\partial v}{\partial \bar{u}}\boldsymbol{T}_{21}^2)\mathbf{J}_{\eta_{12}}\boldsymbol{a}_j}{\det(\mathbf{J}_{\eta_{12}})\bar{\boldsymbol{T}}_{21}^1\boldsymbol{a}_j}, \\ b_j^2 &= \frac{\boldsymbol{a}_j{}^\top\mathbf{J}_{\eta_{12}}\bar{\boldsymbol{T}}_{12}^1}{\boldsymbol{a}_j{}^\top(\frac{\partial u}{\partial \bar{u}}\boldsymbol{T}_{12}^1 + \frac{\partial v}{\partial \bar{u}}\boldsymbol{T}_{12}^2)\det(\mathbf{J}_{\eta_{12}})}, \\ b_j^3 &= \frac{(\frac{\partial u}{\partial \bar{v}}\boldsymbol{T}_{21}^1 + \frac{\partial v}{\partial \bar{v}}\boldsymbol{T}_{21}^2)\mathbf{J}_{\eta_{12}}\boldsymbol{a}_j}{\det(\mathbf{J}_{\eta_{12}})\bar{\boldsymbol{T}}_{21}^2\boldsymbol{a}_j}, \\ b_j^4 &= \frac{\boldsymbol{a}_j{}^\top\mathbf{J}_{\eta_{12}}\bar{\boldsymbol{T}}_{12}^2}{\boldsymbol{a}_j{}^\top(\frac{\partial u}{\partial \bar{v}}\boldsymbol{T}_{12}^1 + \frac{\partial v}{\partial \bar{v}}\boldsymbol{T}_{12}^2)\det(\mathbf{J}_{\eta_{12}})},\end{aligned} \tag{13}$$

where $b_j^i \in \mathbb{R},\ \boldsymbol{a}_1 = (1,\ 0)^\top,\ \boldsymbol{a}_2 = (0,\ 1)^\top$.

Proof. For the proof process, please refer to Appendix XI. □

In the later section, we will consider both the invariance of the connection in (12) and metric preservation in (7) to recover the conformal instead of using the connection invariance only, which leads to a non-linear equation with a higher dimension.

V. Con-NRSfM Algorithm

Our algorithm is built based on the rotational invariance property of the connections and the metric preservation of the moving frame (Theorem 1 and (7)) using the parallel separable iterative framework and self-supervised convolutional network.

A. Point-wise Solution

For the i-th feature in the j-th shape, we can define several variables $y_1^{(i,j)}$, $y_2^{(i,j)}$, $y_{11}^{(i,j)}$, $y_{12}^{(i,j)}$, $y_{22}^{(i,j)}$, $\beta^{(i,j)}$, and $\lambda^{(i,j)}$, corresponding to the first-order terms y_1, y_2, the second-order terms y_{11}, y_{12}, y_{22}, the depth β, and the conformal scale λ. Based on Section III-D and a given edge number N_e, we can generate a well-connected graph $\mathcal{G}_{opt}$ [7]. Assuming that all the features are detected and tracked in all the images (this is only for simplification of the equation, later in Section VI we will show the proposed algorithm is fine for data missing), the NRSfM problem aims to compute the depths of N_p 3D points from N_m monocular images. This problem can be formulated as a graph optimization problem of which the variable dimension is $7N_pN_m$. For the e-th edge $(i_e,\ j_e) \in \mathcal{G}_{opt}$, considering the i-th feature in the i_e-th and the j_e-th shape, the factors $\bar{f}_{j'}(i, i_e, j_e)$ of this graph optimization problem are the virtual measurements written by the invariance relationships (7) and (12). As shown in Fig. 2, we obtain a weighted non-linear least squares (NLLS) problem:

$$\min \sum_{(i_e,j_e)\in\mathcal{G}_{opt}} \sum_{i=1}^{N_p} \omega_e \sum_{j'=1}^{21} \|\bar{f}_{j'}(i, i_e, j_e)\|^2, \tag{14}$$

where

$$\begin{aligned}\bar{f}_{j'}(i, i_e, j_e) &= f_{j'}(\widetilde{\boldsymbol{x}}_n^{(i,i_e)}, \widetilde{\boldsymbol{x}}_n^{(i,j_e)}, \lambda^{(i,i_e,j_e)}), \\ \widetilde{\boldsymbol{x}}_n^{(i,i_e)} &= (y_1^{(i,i_e)}, y_2^{(i,i_e)}, y_{11}^{(i,i_e)}, y_{12}^{(i,i_e)}, y_{22}^{(i,i_e)}, \beta^{(i,i_e)})^\top, \\ \widetilde{\boldsymbol{x}}_n^{(i,j_e)} &= (y_1^{(i,j_e)}, y_2^{(i,j_e)}, y_{11}^{(i,j_e)}, y_{12}^{(i,j_e)}, y_{22}^{(i,j_e)}, \beta^{(i,j_e)})^\top\end{aligned} \tag{15}$$

are the virtual measurements, ω_e is the weight of the edge (i_e, j_e) in the selected graph $\mathcal{G}_{opt}$, j' is the equation number of the virtual measurements. Because (12) corresponds to two 3×3 matrices and (7) related to three independent elements in a 2×2 matrix, the total number of the virtual measurements for one feature within two images is $2 \times 9 + 3 = 21$.

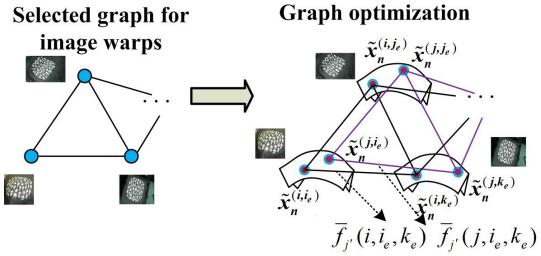


Fig. 2: Point-wise solution using graph optimization.

B. Separable Framework using Parallel Strategy

Typically, a weighted NLLS problem is solved using gradient-based methods [7]. To mitigate optimization sensitivity, reduce the impact of multiple local minima, and address issues related to unbounded scaling, we propose a separable framework. It decouples the sub-variables and leverages a parallel strategy to enhance efficiency and robustness.

Pre-Step. Variable optimization using isometric assumption: Under the isometric assumption (a special case of the conformal assumption) and the IP assumption [10], for two images, the invariance properties of Christoffel Symbols (CS) [44] and MT from related work [11] are utilized to construct four virtual measurements. The first-order terms, $y_1^{(i,j)}$ and $y_2^{(i,j)}$, are estimated by solving a weighted NLLS problem, where the virtual measurements act as factors. Since the measurements between 3D points are independent of feature IDs, the overall NLLS problem involving N_p 3D points can be decomposed into N_p independent sub-NLLS problems, each corresponding to specific features. These sub-problems are solved in parallel using multiple cores, as illustrated in Fig. 3. Each core independently addresses a sub-problem, and the final solution is obtained by concatenating the results from all sub-problems. The first-order terms are efficiently computed using the trust-region-reflective (TRR) algorithm, aided by the given sparse Jacobian matrix.

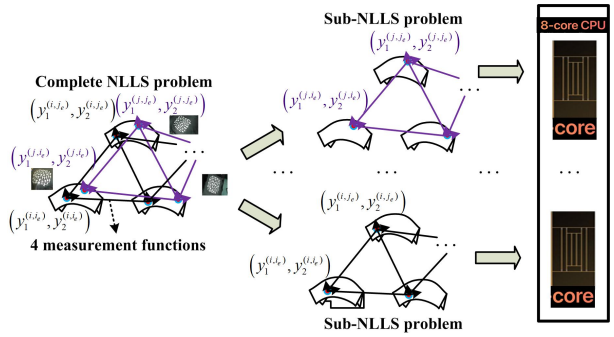


Fig. 3: Parallel method to solve the point-based problems.

By introducing the obtained first-order terms into the Jacobian matrix $\mathbf{J}_{\Phi_i}$ of the image embedding Φ_i, the normal

fields are obtained by normalizing the cross-product of two columns of the Jacobian matrix. Then, by integrating the normal fields [10], the whole surfaces with the depth β are recovered. Different from the parallel strategy in the first-order terms optimization, the depths of the points belonging to one frame are recovered together, which means that dividing the complete problem based on the points is impossible. Hence, each core runs the depth recovery corresponding to one frame, instead of a feature, as shown in Fig. 4. Then, the conformal scales λ is given by the average of 11 equations in (7) and (13).

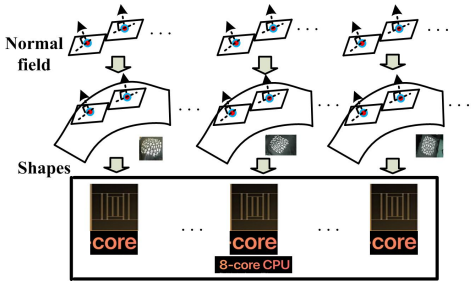


Fig. 4: Parallel method to solve the depth recovery.

Step 1. Second-order terms optimization: At this stage, the first-order terms y_1, y_2, the depth β, and the conformal scale λ as known constant values. The second-order terms $y_{11}^{(i,j)}$, $y_{12}^{(i,j)}$, and $y_{22}^{(i,j)}$ are considered as variables. Initially, these second-order terms are set to zero in the first iteration, while in subsequent iterations, the solution from the previous iteration is used as the starting value. Since metric preservation does not involve second-order information, only the rotational invariance property is applied here. For every pair of features corresponding to edges in the selected tree $\mathcal{G}_{opt}$, the connection (8) is incorporated into the rotational invariance property (12), yielding 18 virtual measurements. These measurements depend on the second-order terms as variables and the elements of equation (12) as factors. Using a parallel approach similar to that in Fig. 3, point-based sub-NLLS problems are solved efficiently. The second-order terms are computed using the TRR algorithm, with the optimization iterations capped at $N_t = 3$. Subsequent steps for solving problems use the same optimization strategy and parameters.

Step 2. First-order terms optimization using conformal assumption: In this step, only the first-order terms $y_1^{(i,j)}$, $y_2^{(i,j)}$ are regarded as the variables in problem (14) and the other variables are considered as the known constant values based on the conformal assumption. The variables $y_1^{(i,j)}$, $y_2^{(i,j)}$ are obtained based the point-based sub-NLLS problems.

Step 3. Depth computation: We express local normals using first-order terms and integrate the normal field along the surface to recover depth up to a scale factor. This process is performed in parallel across different frames, similar to Fig. 4.

Step 4. Conformal scale optimization: In the Pre-Step, the conformal scale is calculated using an approximate formulation. However, this solution is unstable and only valid when all factors in (12) and (7) are perfectly satisfied, which is unrealistic in real-world cases. In this step, similar to Steps 2 and 3, all previously computed terms are treated as constants, while the conformal scale $\lambda^{(i,i_e,j_e)}$ becomes the variable. Multiple sub-NLLS problems[6] are constructed using different features and solved efficiently.

Terminal conditions: At each iteration, we go through Step 1 to Step 4 and get a set of optimized parameters $\boldsymbol{x}_n^{(i,i_e)}(k)$ and $\lambda^{(i,i_e,j_e)}(k)$, where k is the iteration number. Then, we will check its convergence index $Terminal(k) = \sum_i \sum_{i_e} \|\boldsymbol{x}_n^{(i,i_e)}(k) - \boldsymbol{x}_n^{(i,i_e)}(k-1)\| + \sum_i \sum_{(i_e,j_e)\in\mathcal{G}_{opt}} \|\lambda^{(i,i_e,j_e)}(k) - \lambda^{(i,i_e,j_e)}(k-1)\| < \sigma$ using the results from the last two iterations. If the condition is met, the depth $\beta^{(i,i_e)}$ is finalized as the solution.

Remark 1. *Our separable optimization framework is inspired by the principle of alternating optimization [45]. Given that the underlying problem is a non-convex NLLS formulation, we cannot theoretically guarantee the convergency of using separable framework. However, in practice, the framework exhibits favorable numerical properties, including better conditioning of individual sub-problems, reduced memory usage, and empirically improved convergence behavior. We now consider the applied factors $\bar{f}_{j'}(i,i_e,j_e)$ in the NLLS formulation. The fixed coefficients associated with these factors are precomputed from image warping and encapsulate shared surface information. Once computed, these fixed coefficients enable each factor to impose only the connection constraint* (7) *and the metric tensor constraint* (12) *for the same spatial point observed across different frames, without introducing measurement constraints between different points within the same frame. Consequently, several key optimization steps in our framework, including the initialization in the Pre-step, second-order term optimization in Step 1, first-order term optimization in Step 2, and conformal scale adjustment in Step 4, are inherently independent across different points. This natural independence permits efficient parallelization across multiple cores without affecting convergence or final estimation accuracy. With the given derivative terms $y_1^{(i,j_e)}, y_2^{(i,j_e)}, y_{11}^{(i,j_e)}, y_{12}^{(i,j_e)}, y_{22}^{(i,j_e)}$ fused along the surface, the normal field of the whole surface is obtained. The subsequent depth recovery, performed up to a scale factor using the estimated normal field, is also frame-independent. Thus, parallelization in Step 3 across different frames is theoretically sound and empirically valid. Overall, our parallel, separable framework achieves high accuracy, efficiency, and robustness. As shown in Section VI, it solves the problem reliably, even with zero or random initialization.*

C. Dense 3D point cloud with texture

In certain NRSfM tasks, the desired output is a dense 3D point cloud with texture. To achieve this, a self-supervised convolutional neural network can be employed to generate the dense point cloud from a sparse normal field and the corresponding input images[7].

To apply the self-supervised approach for automatic data generation, we randomly sample sparse 3D point clouds with a specified number of points, n_p, within a predefined range,

[6]Due to Corollary 1, second-order leading principal minors and the last elements of the connections are excluded.

[7]Alternatively, if only a sparse point cloud is required, the classical integration method described in Step 3 can be used to directly recover the sparse point cloud.

including normalized pixel coordinates and depth values. Tangent vectors are computed along pixel directions for all sampled 3D features, and their corresponding normal vectors are derived, forming the normal field using the BBS tool[8]. This process allows the generation of numerous surfaces with their normal fields as inputs and depth values as outputs. To create smoother surfaces, we further employ fifth-degree polynomial surfaces to fit the randomly sampled point cloud and extract new points along the fitted surface.

Convolutional neural networks are commonly used with image datasets, so the next step involves encoding the normal field inputs and depth outputs into RGB and depth images, respectively. Consider the obtained normal field $\{\boldsymbol{n}_i \in \mathbb{R}^3\}$, $i = 1, 2, \cdots, n_p$, corresponding to the 2D features (u_i, v_i), $i = 1, 2, \cdots, n_p$. By combining the three components of the normal vector with the 2D features in a single frame (u_i, v_i) $i = 1, 2, \cdots, n_p$, we can construct 3D point clouds. These point clouds can then be integrated into a dense 3D surface using the BBS tool. The three resulting surfaces corresponding to one frame are treated as the RGB channels of an image by interpolating all pixel positions (u_i, v_i) $i = 1, 2, \cdots, n_p$. In this way, the normal fields are encoded as multiple RGB images, which serve as inputs for the convolutional neural network. Similarly, the depth data is encoded into RGB images by assigning the same values to all three channels. Alternatively, the depth values corresponding to the 2D features can also be encoded into depth-specific images. To improve robustness, data augmentation is applied by adding Gaussian noise to the components of the input normal vectors. This improves the model's ability to generalize to noisy data during practical applications.

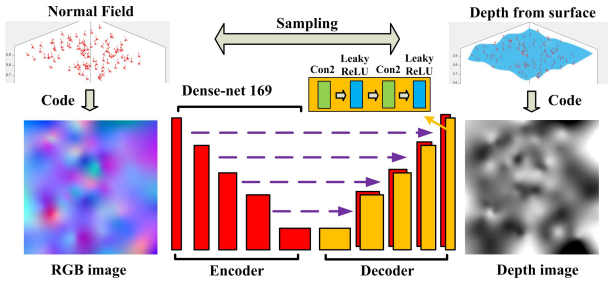


Fig. 5: Self-supervised network for depth recovery.

Fig. 5 provides an overview of our framework for depth recovery from the normal field using a straightforward encoder-decoder network with skip connections. The architecture is a slightly modified version of a well-known network for monocular image-based depth estimation [46]. The encoder is based on DenseNet-169 [47], but unlike [46], which employs transfer learning, our network does not use pre-trained weights from ImageNet [48]. This decision stems from the fact that our input images, derived from normal fields, differ significantly from traditional monocular images of real-world objects, rendering ImageNet pre-trained weights unsuitable for feature extraction in our case. The decoder consists of five basic blocks, each comprising convolutional layers, a leaky rectified linear unit (Leaky ReLU), and a $2\times$ bilinear upsampling layer with concatenation. Our network adopts the same simple architecture approach as [46], avoiding Batch Normalization [49] or other advanced layers described in [50]. Additionally, we enhance the architecture by appending an upsampling layer, a 2D convolutional layer, and a Leaky ReLU to align the output depth image with the input RGB size.

For the loss function, we only use the point-wise L1 loss defined as the differences between the ground truth depth values and the predicted depth values:

$$L(\boldsymbol{p}, \bar{\boldsymbol{p}}) = \frac{1}{n_a} \sum_{i=1}^{n_a} |\boldsymbol{p}_i - \bar{\boldsymbol{p}}_i|_1, \tag{16}$$

where n_a is the number of depth image pixels, $\boldsymbol{p}_i$ and $\bar{\boldsymbol{p}}_i$ are respectively the ground truth and prediction of the i-th pixel.

The primary advantage of the trained network is its ability to directly recover a dense point cloud, rather than being limited to a sparse solution. This makes our self-supervised convolutional neural network significantly faster and more robust compared to traditional integration and texturing methods [7].

D. Algorithm Summary

Our Con-NRSfM method integrates differential geometry, a separable parallel framework, and a learning-based neural network. Using the edge selection approach described in Section III-D, we identify a well-connected subgraph from a complete weighted graph. For each pair of connected images, the image warp is computed and used to formulate a point-wise weighted NLLS problem based on virtual measurements derived from metric preservation (7) and rotational invariance of the connection (12). To address this large-scale NLLS problem, we employ a separable framework that decouples the first-order terms y_1, y_2, second-order terms y_{11}, y_{12}, y_{22}, depth β, and conformal scale λ, improving both robustness and solution accuracy. The process begins with simplifying assumptions to solve an isometric NRSfM efficiently, providing the first-order terms. Depth values are then recovered using an integration approach, and conformal scales are quickly estimated using an approximate closed-form solution. Next, we apply a point-based parallel iterative strategy to optimize the first-order terms, second-order terms, and conformal scale sequentially using the proposed NLLS formulation (14). Finally, the depth and corresponding 3D dense point cloud with texture are recovered from the normal field, leveraging the first-order terms and a pre-trained self-supervised network described in Section V-C. The complete pipeline is summarized in Algorithm 1 and illustrated in the flow chart in Fig. 6.

VI. Simulation and Experiments

In this part, we present simulations and experiments with synthetic and real datasets using C++ (Section VI-D) and MATLAB (other subsections) on a Dell G5-5500 laptop with an Intel Core i7-10870H 2.20 GHz processor. Our network is pre-trained using Tensorflow 2.0. Our method is compared with some state-of-the-art NRSfM methods by the shape error and % 3D error, which are commonly used in the NRSfM

[8]The BBS tool is a toolbox for efficiently handling bicubic B-splines, enabling interpolation of 3D point clouds [46].

Algorithm 1 Con-NRSfM Algorithm

Input: 2D sparse point clouds (u, v) with feature correspondences in different frames or multiple monocular images $\mathcal{I}_i$, the given edge number $N_e = n_c - 1 + k$.

Output: The 3D point clouds $\boldsymbol{z}$ with depth or dense surfaces.

Pre-Step: Variable optim. using isometric assumption (line 1-7)

1: Build a complete graph $\mathcal{G}_c = (\mathcal{V}_c, \mathcal{E}_c, w_c)$ using feature matching for every image pair;
2: Select a well-connected subgraph $\mathcal{G}_{opt}$ using the Kruskal's maximum spanning tree algorithm and the greedy-based method;
3: Compute the image warps based on 2D Schwarzian derivatives;
4: Compute first-order terms using the isometric assumption;
5: Obtain the normal field $\{\boldsymbol{n}_i \in \mathbb{R}^3\}$ by Jacobian matrix $\mathbf{J}_{\Phi_i}$;
6: Recover β by integrating the normal field $\{\boldsymbol{n}_i\}$ in parallel;
7: Compute conformal scale λ_c using average value;
8: **while** $Terminal(k) \geq \sigma$ **do** ▷ Terminal conditions

Step 1: Second-order terms optim. (line 9-10)

9: Get the second-order terms y_{11}, y_{12}, and y_{22} in parallel given the constant values: the first-order terms $y_{1,c}$, $y_{2,c}$, the depth β_c, and the conformal scale λ_c;
10: $y_{11} \rightarrow y_{11,c}$, $y_{12} \rightarrow y_{12,c}$, $y_{22} \rightarrow y_{22,c}$

Step 2: First-order optim. using conformal assumption (line 11-12)

11: Get the first-order terms y_1 and y_2 in parallel by solving (14) given the constant values: the second-order terms $y_{11,c}$, $y_{12,c}$, $y_{22,c}$, the depth β_c, and the conformal scale λ_c;
12: $y_1 \rightarrow y_{1,c}$, $y_2 \rightarrow y_{2,c}$

Step 3: Depth computation (line 13-14)

13: Recover β by integrating the normal field $\{\boldsymbol{n}_i\}$ in parallel;
14: $\beta \rightarrow \beta_c$;

Step 4: Conformal scale optim. (line 15-16)

15: Get the conformal scale λ in parallel by solving (14) given the constant values: the first-order terms $y_{1,c}$, $y_{2,c}$, the second-order terms $y_{11,c}$, $y_{12,c}$, $y_{22,c}$, and the depth β_c;
16: $\lambda \rightarrow \lambda_c$;
17: **end while**
18: **if** Images $\mathcal{I}_i$ are available $\bigcap$ require dense point clouds **then** Recover the dense point clouds using network in Section V-C;
19: **end if**
20: **return** Sparse point cloud or dense surface with texture.

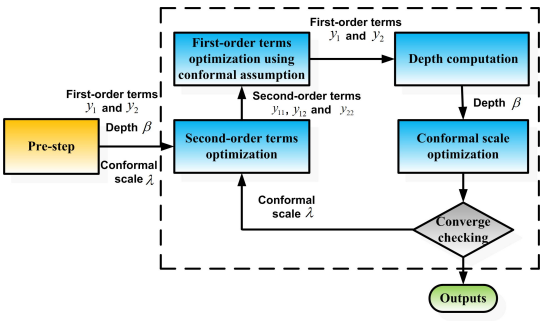


Fig. 6: Flow chart for the core of Con-NRSfM algorithm.

literature [7], [32], computed as RMSEs between the reconstructed and ground-truth results. All error computation of the recovered shapes is performed after using ABSOR function by finding optimal rotation, scale, and translation [51].

A. Simulation Dataset

The theorem proposed in Section IV will be verified and then the performance of the proposed method will be evaluated using two synthetic datasets against other methods, including **infP** [11][9], **Diff** [13], **Ch17** [8], **SDP17** [38][10], and **Go20** [7][11].

1) Theorem Verification: The proposed theorems and corollaries show the important invariance property of the connections undergoing different deformations. Here we mainly focus on the numerical verification of the relationship (12) in Theorem 1. The conformal deformation is simulated using 11 balls with different radii R_i and central coordinates $(x_i, y_i, z_i)^\top$, where i is the index of the balls. In the local coordinate system with the camera center as origin, all the features $(x_k^f, y_i^f, z_i^f)^\top$ on the partial surface of the ball, the depths $\beta_i = z_i$, and their first/second-order derivatives are written as the function of the pixels $(u_k, v_k)^\top$ using the perspective projection. For the other balls, the matched features $(\bar{x}_i^f, \bar{y}_i^f, \bar{z}_i^f)^\top$ and their corresponding pixels $(\bar{u}_k, \bar{v}_k)^\top$ are computed as:

$$\begin{pmatrix} \bar{x}_i^f \\ \bar{y}_i^f \\ \bar{z}_i^f \end{pmatrix} = \begin{pmatrix} R_j/R_i(x_k^f - x_i) + x_j \\ R_j/R_i(y_k^f - y_i) + y_j \\ R_j/R_i(z_k^f - z_i) + z_j \end{pmatrix}, \tag{17}$$

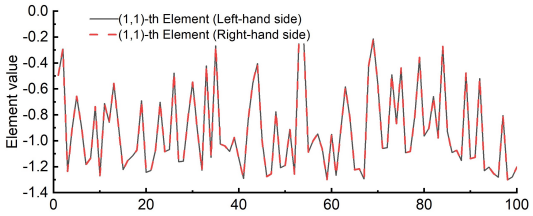
and $(\bar{u}_k, \bar{v}_k)^\top = (\bar{x}_i^f/\bar{z}_i^f, \bar{y}_i^f/\bar{z}_i^f)^\top$. Similarly, we can get their depths $\bar{\beta}_i = \bar{z}_i$, first/second-order derivatives, and the image warps using the analytical relationship between the different pixels $(\bar{u}_k, \bar{v}_k)^\top$ and $(u_k, v_k)^\top$. The conformal scale between two balls is computed as $\lambda_{ij} = R_i/R_j$.

Based on these analytical equations, 100 matched features are generated for each ball. All the obtained parameters are introduced to the rotational invariance property as shown in (12). As an example, the (1, 1)-th elements at both sides of (12) corresponding to 100 features and two balls are presented in Fig. 7a. In some references [10], [13], the assumption of the infinitesimal planarity is used, which means that only the first-order terms are considered. Hence, as a comparison, connection (8) ignoring the second-order terms is also introduced to the invariance property (12). The similar results are also shown in Fig. 7b. We can find that, when second-order terms in connections are considered, the simulation results completely satisfy the rotational invariance shown in Theorem 1. The simulation results only considering the first-order terms only roughly follow the rotational invariance in Theorem 1. We define an index for results using only first-order terms:

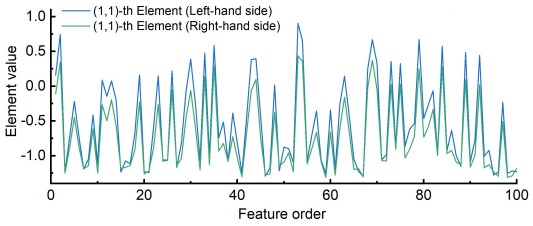
$$Index = \sum_{k=1}^{100} \sum_{i=1}^{3} \sum_{j=1}^{3} \frac{1}{900} \frac{|\Theta_{ij}^l(k) - \Theta_{ij}^r(k)|}{\max_k \Theta_{ij}^l(k) - \min_k \Theta_{ij}^l(k)}, \tag{18}$$

where $\Theta_{ij}^l(k)$ and $\Theta_{ij}^r(k)$ respectively mean the (i, j)-th element of the left-hand and right-hand sides of (12) for the k-th feature; $\max_k$ and $\min_k$ mean the maximal and minimal values for all the features. Using this definition. we test multiple simulated balls (1 ball with 10 matched balls) with different centers and radii. For the case with first-order terms only, the indexes of different balls are respectively 6.82%, 7.03%, 9.69%, 8.07%, 9.89%, 11.6%, 7.70%, 6.01%, 8.98%, and 12.20%. For the case using both first-order and second-order terms, all the indexes are equal to 0% consistently, which validates the correctness of Theorem 1.

[9]It shows a similar performance with its extended version **iso** [12].
[10]Due to the lack of the open source, we re-implement this algorithm.
[11]The edge number needs to be set in this method. We use the default edge number with $n_c - 1$ for **Go20** and our framework with no additional edges.



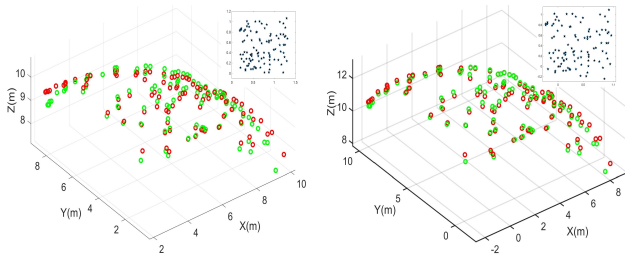
(a) First and second-order terms. The result shows that the left-hand side is exactly the same as the right-hand side, when both the first and second-order terms are used.



(b) First-order terms only. The result shows that the left-hand side is approximately equal to the right-hand side, when only the first terms are used.

Fig. 7: The (1, 1)-th element of the first equation of (12) shows the invariance property.

2) Synthetic Datasets: We simulate a synthetic conformal dataset with 7 deforming scenes and 100 features using the perspective projection model to test the performance of our method. This conformal dataset is built based on 7 ball surfaces with different radii and central coordinates. For this dataset, because the calibrated features beyond the vision range $\mathcal{R}_v$[12], which is set as $[-0.5, 0.5]$ to u-axis and $[-0.5, 0.5]$ to v-axis for our network, we do not use the deep learning network in Section V-C. As an example, the reconstructed result for two deforming shapes using our method is shown in Fig. 8. The mean %3D errors are respectively **0.8052**%(**Ours**), 4.0454%(**Diff**), 1.0643%(**infP**), 1.4486% (**Ch17**), 1.4545%(**SDP17**), and 0.9792%(**Go20**). The average shape errors are respectively **6.0822**°(**Ours**), 24.9474°(**Diff**), 7.2102°(**infP**), 13.0426°(**Ch17**), 12.9205°(**SDP17**), and 7.4458°(**Go20**). Based on the obtained results for these datasets, our method shows the best performance.



(a) Shape 2 (b) Shape 4

Fig. 8: Ground truth (green) and reconstructed results (red) for the synthetic dataset. Right-up sub-graphs are 2D input scenes.

[12]Because we only sample the random data in a given range, if the vision range $\mathcal{R}_v$ and the deformation range $\mathcal{R}_d$ of the feature measurements are outside the given range, our trained network cannot offer accurate depth estimation results.

In order to show the robustness of our proposed method, the comparison results with the changing data are presented. Table II shows the mean %3D error (in %) and the mean shape error (in °) when 0-70% point data are missing from every image, including the first image. A feature, which is missing in the first image but is included in others will be regarded as an appearing feature[13]. The final results in Table II show the high accuracy and good robustness of our proposed method.

TABLE II: Comparison using missing and appearing data

Missing	**infP***	**Ours**	**Diff****
0%	1.06%-7.21°	**0.81%-6.08°**	4.05%-24.95°
10%	1.23%-8.62°	**1.10%-8.38°**	-
20%	1.47%-10.25°	**1.18%-9.78°**	-
30%	2.01%-15.39°	1.74%-**13.59°**	-
40%	2.04%-15.97°	**1.66%-12.98°**	-
50%	3.07%-20.53°	**1.86%-13.21°**	-
60%	3.48%-19.85°	**1.87%-13.94°**	-
70%	4.05%-24.73°	**2.39%-18.78°**	-
Missing	**Ch17**	**SDP17**	**Go20**
0%	1.45%-13.04°	1.45%-12.92°	0.98%-7.45°
10%	1.53%-14.05°	1.49%-14.01°	1.15%-8.44°
20%	1.48 %-14.01°	1.51%-14.21°	1.28%-10.63°
30%	**1.63**%-15.24°	1.71%-15.52°	1.74%-13.67°
40%	2.05%-16.49°	2.09%-16.37°	1.77%-14.02°
50%	10.76%-34.13°	8.97%-31.78°	1.99%-14.67°
60%	14.06%-58.55°	13.22%-49.31°	1.97%-14.98°
70%	16.93%-75.82°	16.11%-68.99°	2.44%-19.04°

* Due to limitations in the Gloptipoly 3 toolbox [52] used in **infP**, errors arise when the synthetic dataset contains one or more features observed fewer than twice, causing the open-source code to fail in producing valid results. This issue does not affect the other compared methods. In cases with a high missing rate (e.g., 70%), such errors occur frequently, and no valid output is returned. Therefore, we select a specific dataset in which each feature is observed at least twice to get the readable results and ensure a fair comparison.
**Diff is not very stable for the synthetic dataset with missing features and no correct result is obtained using the open source code.

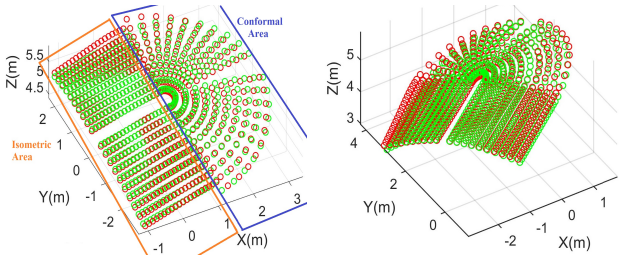
We also simulate a challenging synthetic dataset with 10 deforming scenes and 600 features under both isometric and conformal deformations using perspective projection, as clearly indicated in Fig. 9a. Here we only consider the sparse result. Therefore, the dense depth recovery network is not deployed. The reconstruct results for two deforming shapes using our method are shown in Fig. 9. Their mean %3D errors are **1.1241**%(**Ours**), 2.9611%(**Diff**), 1.2130%(**infP**), 1.1891% (**Ch17**), 1.3871%(**SDP17**), and 1.2303%(**Go20**). The average shape errors are respectively **9.6309**°(**Ours**), 16.2807°(**Diff**), 11.5302°(**infP**), 9.7148°(**Ch17**), 14.7723°(**SDP17**), and 11.9024° (**Go20**). Results confirm our method's superiority.

B. Isometric Real Datasets

In this part, we perform experiments using both short-term and long-term real datasets to compare with the other five methods. The T-shirt dataset with 10 images and the Flag dataset with 30 images are the short-term datasets and the long-term datasets include Rug and KinectPaper datasets.

T-shirt dataset: The T-shirt dataset [13] consists of 85 manually set features correspondences across 10 different images

[13]The **Diff** and **infP** methods cannot deal with the case with appearing features, so we ignore these appearing features in the recovered shape and the error computation. Our method and the **Ch17** method can easily solve the problem with the appearing features, so these features remain.



(a) Shape 1 (b) Shape 5

Fig. 9: NRSfM ground truth (green) data and reconstructed results (red) for the synthetic dataset 2.

of a T-shirt deforming isometrically. The features belong to the vision range $\mathcal{R}_v$ and the dataset includes the corresponding RGB images, so we implement our proposed method with the network in Section V-C. Using the full datasets and all the features, we recover the shape, generate, and texture the dense point cloud, as shown in Fig. 10.

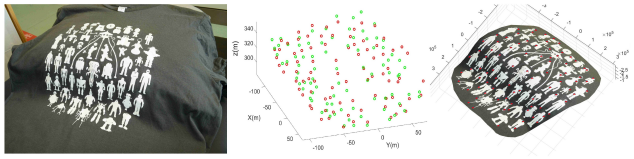
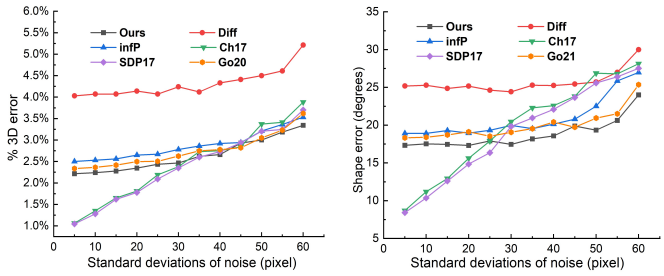


Fig. 10: The reconstructed dense point cloud with texture.



(a) % 3D error (b) Shape error (degrees)

Fig. 11: Comparison results of the mean shape error and % 3D error using the T-shirt dataset with different levels of noise.

In T-shirt dataset, we also show the accuracy of the obtained sparse point cloud using the integrating way. The mean %3D errors of these methods are respectively 2.2401%(**Ours**), 4.0234%(**Diff**), 2.4932%(**infP**), 0.8789%(**Ch17**), **0.7979**% (**SDP17**), and 2.3296%(**Go20**). The average shape errors are respectively 17.5239°(**Ours**), 25.0743°(**Diff**), 7.2102°(**infP**), 6.9904°(**Ch17**), **6.4235**°(**SDP17**), and 18.3721°(**Go20**). For this dataset, the **SDP17** and **Ch17** methods show the best performance. Our method, **Go20**, and **infP** show a stable and fair performance. The **Diff** method does not work well on this dataset. T-shirt dataset is almost isometric (a special case of conformal). **Ch17** and **SDP17** use only point correspondences whereas **Ours** also uses their first and second-order derivatives. T-shirt has only 85 manually clicked point correspondences on 10 images, thus the computation of the first- and (especially) second-order derivatives is inaccurate. There is practically no noise on this dataset. This is why **Ch17** and **SDP17** perform better than **Ours** on T-shirt.

When 50% point data are missing from every image, including the first image, the mean %3D errors (in %) of these methods are respectively 3.05% (**infP**), **2.77**% (**Ours**), 3.82% (**Diff**), 7.02% (**Ch17**), 6.68% (**SDP17**), and 2.78% (**Go20**). The mean shape error are 21.05° (**infP**), **20.95**° (**Ours**), 25.67° (**Diff**), 34.20° (**Ch17**), 30.12° (**SDP17**), and 20.98° (**Go20**). Our results indicate that the **Ch17** and **SDP17** methods are not reliable in the presence of many missing points, which is very common in cases of the tracking lost in the real applications of NRSfM, such as deformable SLAM [11].

We add Gaussian noises, of which the standard deviations range from 5 pixels to 50 pixels, to this dataset. The image size in this dataset is 4800×3200, so these levels of noise are reasonable. These six methods are compared with each other. The changes of the mean %3D error (in %) and the mean shape error (in °) are shown in Fig. 11. We can find that our method is robust to noisy data and the **SDP17** and **Ch17** methods are sensitive to the noises.

Flag dataset: Based on a real flag and a virtual perspective camera, the Flag dataset [53] is a self-synthetic data with 250 tracking features. We randomly select 30 images from the whole dataset with 450 frames and add Gaussian noises (with a standard deviation of 4 pixels) to each feature. The comparison results of the % 3D error are shown in Fig. 12. The examples of the reconstructed results are shown in Fig. 13.

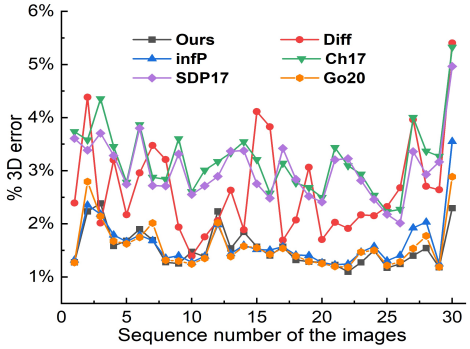


Fig. 12: Comparison results of the mean % 3D error using the Flag dataset.

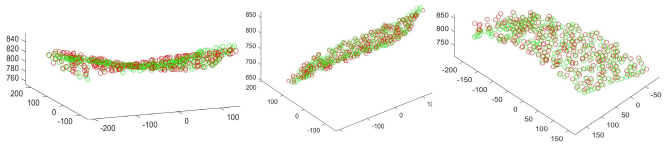


Fig. 13: The ground truth (green) and reconstructed results (red) of the Flag dataset.

The mean % 3D errors are respectively **1.5364**% (**Ours**), 2.6631% (**Diff**), 1.6289% (**infP**), 3.0996% (**Ch17**), 3.0309% (**SDP17**), and 1.5710% (**Go20**). These results show that our method has the best performance on this dataset. **Ch17** and **SDP17** have the largest errors as compared to other methods.

The following two long-term datasets, including Rug and KinectPaper datasets, are used to further verify the performance of our method in the isometric cases. Because the SOCP and SDP formulations are solved using the inner-point method, **Ch17** and **SDP17** take more than 3 hours for 60 images with 300 features. Therefore, we split the long-term sequences to sets of 30 images (Rug dataset) and 15 images

(KinectPaper dataset), and then evaluated these two methods. The other methods, including **infP**, **Diff**, **Go20**, and **Ours**, are based on the full datasets and they can be solved within 1 hour given the image warps. The analysis of the computational time will be further discussed in Section VI-F2.

Rug dataset: The Rug dataset [11] is a public isometric data with 159 images and 300 features showing the deformed rug from different views. This is a challenging dataset with outliers and poor given correspondences, due to the low frame-rate of the recorded sequences. With no missing features, the comparison results of the % 3D error are shown in Fig. 14. Reconstructed results by **Ours** are illustrated in Fig. 15.

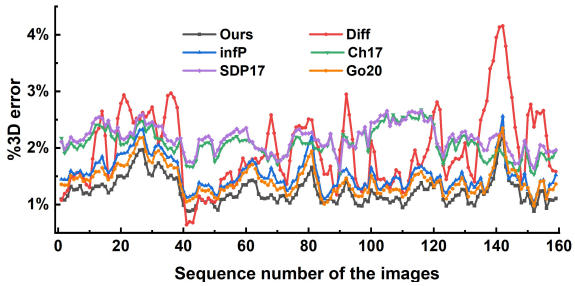


Fig. 14: The comparison results of the Rug dataset.

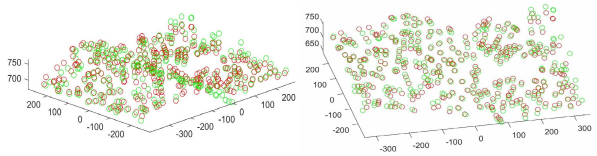


Fig. 15: The ground truth (green) and reconstructed results (red) of the Rug dataset.

The mean % 3D errors of all the images are respectively **1.2511**% (**Ours**), 1.970% (**Diff**), 1.5286% (**infP**), 2.082% (**Ch17**), 2.1748% (**SDP17**), and 1.4296% (**Go20**). It is clearly visible that our method performs better than the other methods.

KinectPaper dataset [11]: It is a long sequence with 191 images and 1503 features. It shows the isometric deformations of a paper. Similar to the Rug dataset, this sequence also contains outliers. Six methods are applied to this dataset in order to compare their performance. Based on all the visible features, the comparison results of the % 3D error are shown in Fig. 16. The ground truth and the reconstructed results for the KinectPaper dataset are presented in Fig. 17.

The mean % 3D errors are respectively **0.7011**% (**Ours**), 3.1577% (**Diff**), 1.2106% (**infP**), 1.9164% (**Ch17**), 2.0804% (**SDP17**), and 0.9294% (**Go20**). These results demonstrate that our method evidently outperforms the others.

C. Real Datasets with More Generic Deformations

A key advantage of our method is its applicability to both isometric and conformal deformations. We therefore test it on the NRSfM Challenge Dataset [54] and a self-collected conformal dataset.

NRSfM Challenge Dataset. We apply our method in the NRSfM Challenge Dataset, which includes 5 image sequences called *Articulated Joints*, *Balloon Deflation*, *Paper Bending*, *Rubber Stretching*, and *Paper Tearing*. They represent 5 different classical non-rigid deformations: articulated (piecewise-

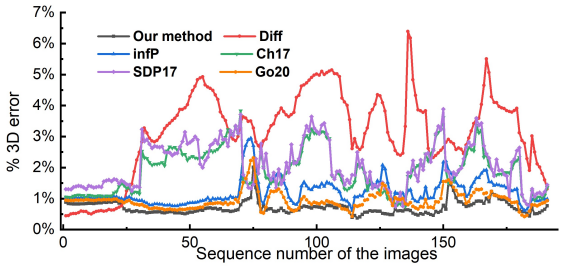


Fig. 16: The comparison results of the KinectPaper dataset.

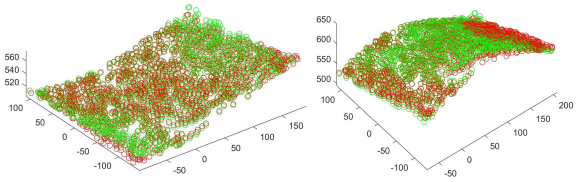


Fig. 17: The reconstructed results of the KinectPaper dataset.

rigid), balloon (conformal), paper bending (isometric), rubber (elastic), and paper being torn. Because of the virtual camera, the vision range is commonly outside the vision range $\mathcal{R}_v$ and only the integrating method is applied in this part. The image features are obtained using 6 different camera motions with orthogonal and perspective projections and the ground-truth for one frame is provided for each sequence. Our method is based on the assumption that the deforming surface is a manifold, for the stereoscopic datasets (like balloon), the back-side features will break the one-by-one mapping (image embedding) between our image and the visible surface. Therefore, we test the balloon and two Paper datasets without using missing (invisible) features and the Articulated and Stretching datasets using full features. The comparison results, including score, ground truth (green), and reconstructed shapes (red), among **Ours**, **Ch17**, **Pa21-R** [55], **Pa21-S** [55], **An17** [56], **Lee16** [57], **Diff**, **Closed** [37], and **Best** (**Best** means the one that does best as reported in the benchmark statistics provided on the website [37]) are presented in Table III. The reconstructed results are shown in Fig. 18.

Results show our method performs robustly and competitively with SOTA in both perspective and orthographic cases on datasets with generic deformations.

Self-Collected Conformal Dataset. Existing real-world datasets capturing conformal deformation are limited. To evaluate the performance of our framework under elastic deformation, we collected a custom dataset using the Intel RealSense D435i depth camera. This dataset consists of 13 images of a deforming balloon. To enhance texture and facilitate feature detection, visual markers were affixed to the balloon's surface. Sparse 2D features were extracted using the SIFT, and feature correspondences were established across frames. The corresponding depth values from the depth images were used as ground truth[14]. In this dataset, the mean 3D reconstruction errors for each method are as follows: **2.2634**% (**Ours**), 3.2822% (**infP**), N/A (**Ch17**), 2.9578% (**Pa21-S**), and 3.4588% (**Go20**) and the corresponding average shape errors

[14]The D435i depth camera offers about 5–40 mm accuracy within 1–2 meter range, limiting ground truth precision.

TABLE III: Results on the NRSfM challenge datasets.

Camera	NRSfM Challenge Dataset (Perspective projection)				
Method	Articulated	Balloon	Bending	Rubber	Tearing
Case	full	missing	missing	full	missing
Ch17	91.6	53.5	63.8	62.5	51.9
An17	65.1	55.2	64.7	48.1	50.8
Lee16	105.5	-	-	70.3	-
Pa21-R	25.1	41.6	40.5	20.6	28.7
Pa21-S	26.0	40.7	40.4	20.6	27.8
Diff	**21.3**	37.8	39.0	30.3	23.8
Best	40.7	35.7	39.0	30.3	24.9
Closed	21.8	38.1	38.2	17.2	23.1
Ours	22.8	**17.9**	**21.8**	**16.9**	**15.3**
Camera	NRSfM Challenge Dataset (Orthographic projection)				
Ch17	88.7	52.6	65.0	66.3	57.2
An17	58.1	46.4	50.3	38.9	38.4
Lee16	105.3	-	-	69.2	-
Pa21-R	21.8	27.2	32.3	23.1	20.7
Pa21-S	22.0	27.3	32.2	33.0	21.0
Diff	18.7	33.6	34.0	**17.1**	18.8
Best	35.5	33.8	37.0	22.9	18.3
Closed	20.1	26.8	32.1	17.2	18.3
Ours	**13.2**	**17.1**	**21.3**	19.3	**17.0**

– indicates that method failed to return a result due to missing data. Most of these compared results are obtained from Table 5 in Reference [37] considering full or missing cases.

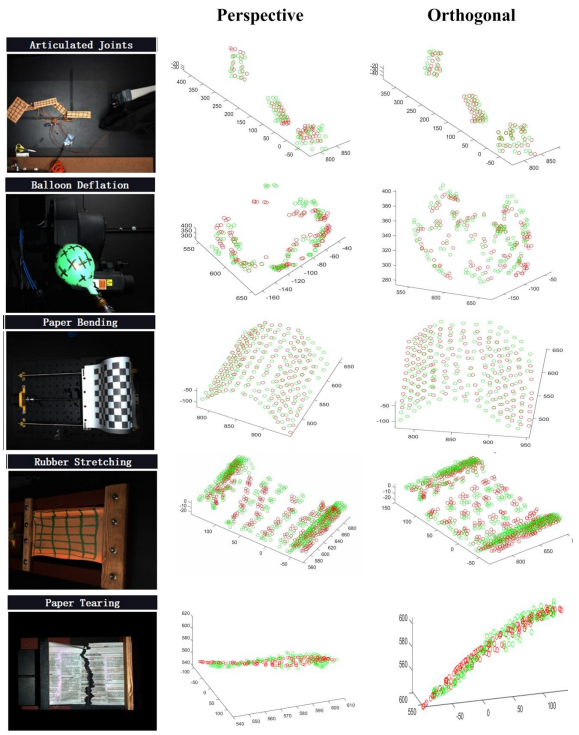
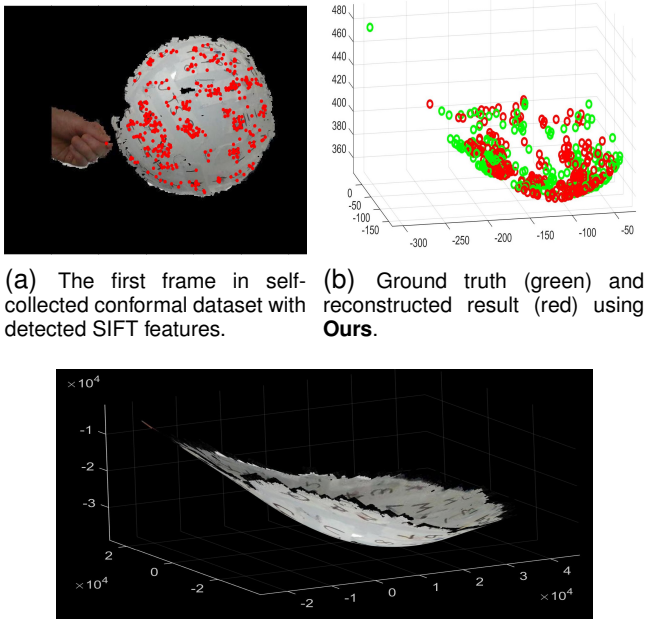


Fig. 18: NRSfM challenge dataset, ground truth (green), and reconstructed results (red) using **Ours**.

are: **8.0563**° (**Ours**), 26.3880° (**infP**), N/A (**Ch17**), 23.4654° (**Pa21-S**), and 37.1828° (**Go20**)[15]. Our method achieves the best performance on this dataset. Fig. 19 shows the reconstructed results for the first image alongside the ground truth.

[15]**Ch17** method encounters memory limitations on a desktop with 32 GB RAM, and thus its results are unavailable. The result for **infP** is obtained by excluding features that are observed fewer than two times.



(a) The first frame in self-collected conformal dataset with detected SIFT features.

(b) Ground truth (green) and reconstructed result (red) using **Ours**.

(c) Textured result using network

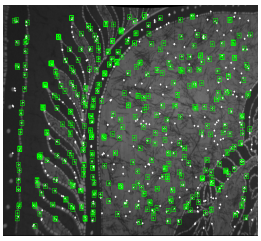
Fig. 19: Reconstructed results of the self-collected dataset.

D. Deformable SLAM Datasets

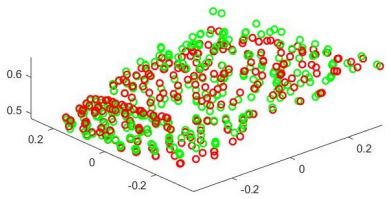
The deformable SLAM is currently considered as an important application of the NRSfM method. In [2], the NRSfM method is used to build the sparse local map fusing the tracking camera and the dense map combining with the template mesh. In this part, in order to further compare with other methods, the datasets, including Mandala dataset [2] and Hamlyn dataset [58], [59], are introduced using our proposed method. We use the front-end of the Def-SLAM [2] with ORB descriptor [60] to pick out and track the 2D features in the deformable environment. The ground truth depths of both datasets are obtained from the corresponding stereo sequence.

Mandala dataset The Mandala dataset is composed of 5 sequences (640×480 pixels at 30 fps) with exploratory trajectories observing a textured kerchief deforming near-isometrically. Using the Mandala3 dataset, we generate an NRSfM dataset based on the 9-th keyframe consisting of 9 images with 411 tracking normalized features. Some features are missing during the deforming process and the example image with tracking features is shown in Fig. 20a. The corresponding reconstructed results for the 9-th keyframe are shown in Fig. 20b and Fig. 20c. This dataset is very challenging with much missing data and unreliable data association. The mean % 3D errors are respectively **1.2768**% (**Ours**), 1.7442% (**Diff**), 1.7389% (**infP**), and 1.4955% (**Go20**). The average shape errors are **16.14972**° (**Ours**), 20.0113° (**Diff**), 19.2663° (**infP**), and 17.2663° (**Go20**). The **Ch17** and **SDP17** methods generate very poor results with more than 10% 3D error and 30° shape errors and we can consider them as a failure to deal with this dataset. These experimental results demonstrate that our method evidently outperforms the others for this dataset.

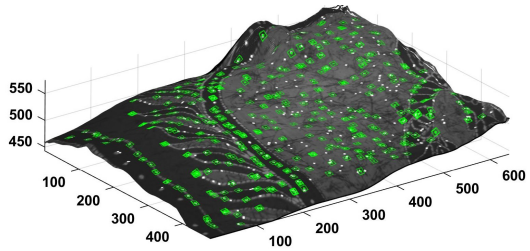
Hamlyn dataset The Hamlyn dataset includes six different sequences and we pick out the SeqHeart sequence to evaluate



(a) The 9-th keyframe in Mandala3 dataset.

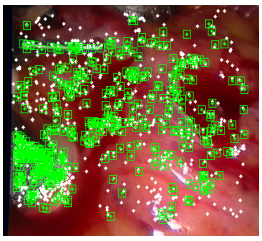


(b) Ground truth (green) and reconstructed result (red) using **Ours**.

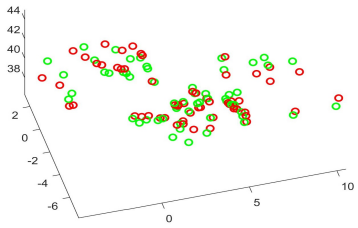


(c) Dense result with texture using network

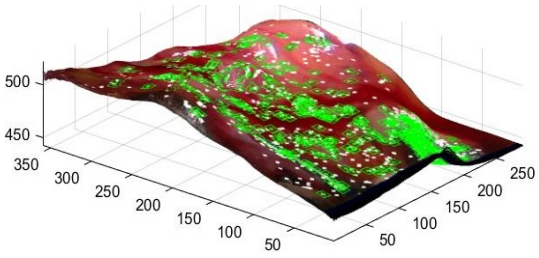
Fig. 20: Reconstructed results of Mandala3 dataset.



(a) The 4-th keyframe in SeqHeart dataset.



(b) Ground truth (green) and reconstructed result (red) using **Ours**.



(c) Dense point cloud with texture using network

Fig. 21: Reconstructed results of the 4-th keyframe of the SeqHeart sequence in the Hamlyn dataset.

our algorithm [2]. This sequence shows a non-rigid beating heart observed by a fixed camera. Using the deformable SLAM front-end, we generate an NRSfM dataset consisting of 7 images with 325 tracking normalized features. A lot of features are missing during the deforming process and the 4-th keyframe image with tracking features is shown in Fig. 20a. This dataset is also very challenging with a lot of missing data and unreliable data association. The mean % 3D errors are respectively 2.5431% (**Ours**), 2.5415% (**infP**), 3.4061% (**Ch17**), 3.1733% (**SDPP7**), and 2.5322% (**Go20**). The average shape errors are 27.9817° (**Ours**), 27.9921° (**infP**), 30.4817° (**Ch17**), 29.4817° (**SDP17**), and 28.1123° (**Go20**). The results of this image sequence show that our method, the **infP** method, and the **Go20** method perform better than the others.

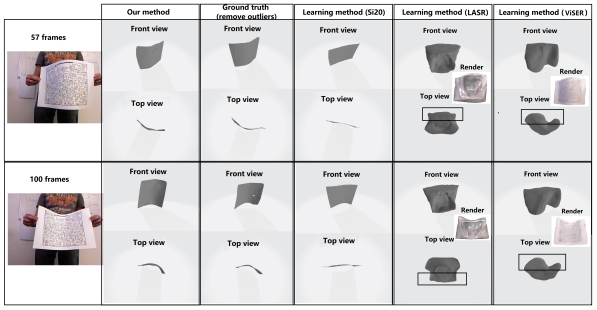
E. Dense datasets

Recently, unsupervised neural networks such as **Si20** [14], **LASR** [61], and **ViSER** [62] have demonstrated superior performance compared with several classical dense NRSfM methods, including **TB** [63], **MP** [64], **DSTA** [65], **GM** [66], and **JM** [67]. To demonstrate the efficiency of the proposed method, we compare its performance with these state-of-the-art learning-based approaches using two publicly available dense datasets: Paper (Dense) and Tshirt (Dense) [14]. Both datasets contain a large number of feature correspondences or images, which leads to high computational complexity. We evaluate the performance of **Ours** against **An17** [56], **Pa19** [11], **Pa21-R** [55], **Pa21-S** [55], **Si20**, **LASR**, and **ViSER**, and report the mean 3D error[16] in Table IV. Fig. 22 visually illustrates the reconstruction results.

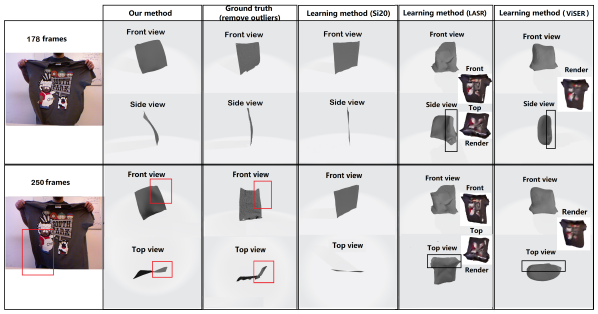
[16]Note that the mean 3D error used here differs from the previously used mean % 3D error. Its definition is given in [14]. The reported results of **LASR** and **ViSER** are obtained using their default configurations without additional tuning, and we exclude some non-front, backside points for a fair comparison. If we consider the normal field, which represents the shape, the disadvantage of these two methods becomes even more pronounced. Our method is evaluated on a downsampled dataset with 2000 features.

TABLE IV: Mean 3D error results on the dense datasets (%).

	Paper	Tshirt		Paper	Tshirt
An17	4.48	2.76	**Pa19**	5.47	3.91
Pa21-R	5.33	3.99	**Pa21-S**	5.52	4.02
Si20	3.32	3.09	**LASR**	5.56	8.49
ViSER	3.74	6.33	**Ours**	**3.01**	**2.33**



(a) Obtained meshes (example) for Paper dataset.

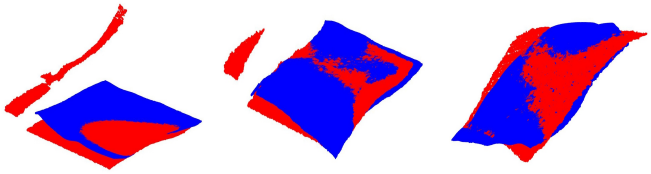


(b) Obtained meshes (example) for Tshirt dataset.

Fig. 22: Recovered results of two datasets.

The results show that our method achieves the best overall performance. Based on the reported results in their papers and our findings in Table IV, **ViSER** and **LASR** are better

suited for spatially convex, deforming objects (e.g., animals and humans) that undergo large motions and are observed from diverse viewpoints, as they reconstruct shapes from a spherical prior. In other words, the target shape is topologically equivalent to a sphere. However, these methods are ill-suited for reconstructing thin Riemannian manifolds or surfaces, particularly when camera motion is small and restricted, which is the typical setting considered in this paper. In fact, in the Paper (Dense) dataset, its ground truth includes a few outliers. All the above results in Table IV only remove the outliers with too large distance and the threshold is set as 100 [17]. We find that this threshold is not enough to remove all outliers. When the threshold is reduced to 70, which helps to remove more outliers, the mean 3D error of our method will be 2.82%. Fig. 23 shows an example of outliers in the 75th frame.



(a) remain outliers (b) threshold= 100 (c) threshold= 70

Fig. 23: An example in 75-th frame (Paper dataset). Ground truth (red), including outliers, and reconstructed results (blue).

We note that these unsupervised methods differ from conventional learning, where a network is trained once and applied to new datasets. They use the networks to represent the deformation, which is more similar to the parametrization. Hence, they need to compute again for each new dataset, and the computational time will be discussed, shown in Section VI-F2.

F. Further Analysis

1) Network Result: In this part, we evaluate the performance of our DenseNet-169-based U-Net with data augmentation against two alternatives: (1) the same network without data augmentation, and (2) a similar U-Net with a ResNet-50 encoder [68]. In our approach, we augment the normalized normal field by injecting Gaussian noise with a mean of 0.2 and a variance of 0.1—equivalent to approximately 20% noise. The noise-free subset is identical for both the augmented and non-augmented models. We report the training losses over 100 epochs in Fig. 24, illustrating the convergence behavior of each model. Corresponding prediction results for a testing data point are shown in Fig. 25. For the clean testing dataset, the relative prediction error ratios (with our method as baseline) are 0.9571 for the non-augmented DenseNet model and 2.6069 for the ResNet-50 U-Net, where a smaller ratio indicates better accuracy. For the noisy testing dataset, the error ratios are 6.7416 (non-augmented) and 2.3806 (ResNet-50). Our trained network achieves comparable prediction accuracy to the version without data augmentation and significantly outperforms U-Nets with ResNet encoders on noise-free data. Under noisy conditions, our method demonstrates superior robustness compared to the other approaches.

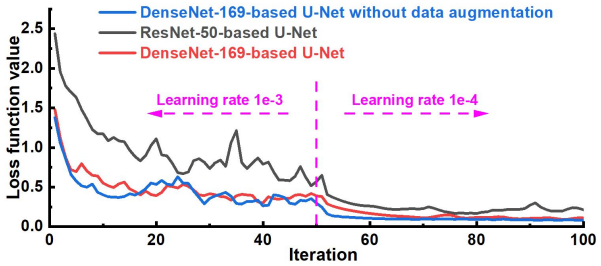


Fig. 24: Training loss over epochs.

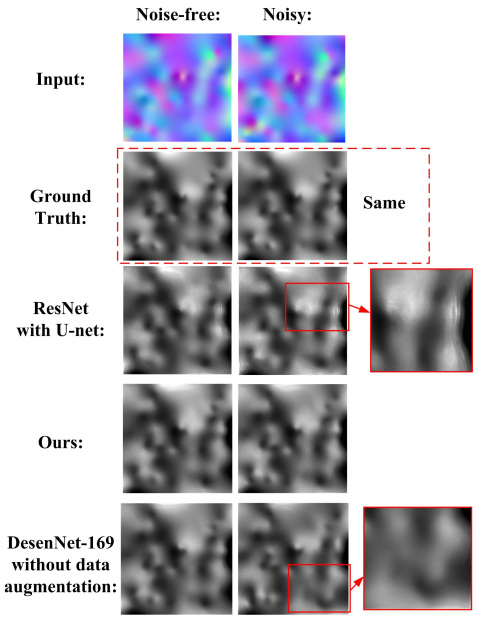


Fig. 25: Network prediction results.

2) Computational Time Study: Based on the hardware setting (Dell G5-5500 laptop with NVIDIA GeForce RTX 2070 MQ), we compare our method with the **infP** method, the **Diff** method, the **Ch17** method, and the **Go20** method in the running ability. The running times with the given image warps (without considering the texturing) using MATLAB R2021a are shown in Table V. The result shows that our method has an acceptable computational ability. The **infP** method is the fastest one and the **Ch17** method shows a much slower running ability compared with the others. In fact, in our implementation, Our results indicate that the **SDP17** commonly takes 2-3 times longer than the computational time of the **Ch17** method. For the learning method **Si20**, it costs more than 200 seconds/frame to recover the Paper (Dense) dataset within 5×10^4 iterations. **LASR** and **ViSER** cost about 45.10 seconds/frame and 50.36 seconds/frame to recover the Paper (Dense) dataset, respectively. **Ours** costs 11.50 seconds/frame using a downsample Paper (Dense) dataset with 2000 features/frame and reaching a better performance.

All these results reported in Table V limit the recovered result to be sparse point features, instead of a dense point cloud. If we would like to perform the colorful 3D dense reconstruction, using our network combining the depth recovery

[17]We reuse datasets, evaluation Matlab function, and results offered by [14], so the threshold is officially given.

TABLE V: Computational ability comparison (s/frame)

Datasets	N_m/N_p	**Ours**	**infP**	**Diff**	**Ch17**	**Go20**
Flag	30/250	2.08	0.41	3.41	101.10	0.33
Rug	159/300	1.69	0.31	4.50	> 200	1.48
Rug$\times 5^{**}$	791/300	1.61	0.20	4.66	$\gg 200$	2.06
Rug$\times 10$	1581/300	1.62	0.18	4.67	$\gg 200$	2.89

**Rug$\times\star$ means to directly combine $\star$ Rug datasets with the same feature tracking as the normal Rug dataset. Our results indicate that our code can deal with the middle size dataset, like Rug$\times 50$ with 7901 frames and 300 feature/frame, and keep the similar running speed (1.398 s/frame) based on common desktop configuration with i7-13700k and 32 GB RAM.

and the texturing, our method will show some computational advantages. It commonly takes about 0.6s to recover the 3D dense point cloud with texture for one frame. For the other sparse methods, the depth recovery from the normal field and the texturing cost about 1.4s to operate one frame [7] with almost the same accuracy.

To further evaluate the effectiveness of the proposed parallel strategy, we compare our framework with a baseline version in which all parallel operations are disabled. The comparison is conducted using the Rug$\times\star$ datasets, where $\star = 2,\ 4,\ 6,\ 8, 10$. Since the parallelization does not affect the estimation results, we report only the ratio of computational time between the parallel and non-parallel versions, as illustrated in Fig. 26. All experiments were conducted on a desktop with an Intel Core i7-13700K CPU, 32 GB of RAM, and MATLAB R2021a. The results clearly demonstrate that the parallel strategy significantly enhances computational efficiency on a multi-core CPU platform.

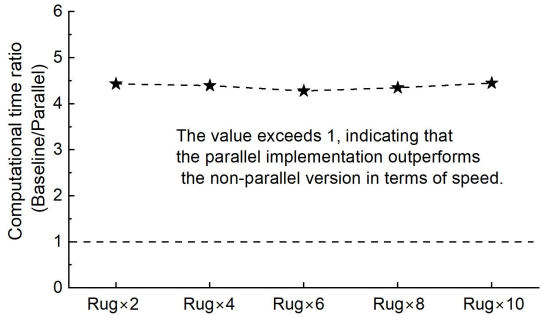


Fig. 26: Computational time ratio between two versions.

3) Ablation Study: In order to verify the importance of each component of our proposed method, we complete an ablation study in this sub-section. In our method, the computation of second-order derivatives helps to improve the reconstruction quality of surfaces, especially with high curvature. The application of conformal scale helps to estimate a good scale of reconstruction with wider application scope. The separable framework improves the robustness and avoids the local minima. Table VI shows the ablation study of each component.

TABLE VI: Ablation study of each component

% 3D and Shape errors	Synthetic 1		T-shirt	
Our framework	0.81%	6.08°	2.24%	17.52°
Without Pre-Step	7.82%	41.35°	4.52%	44.83°
Without separable opt.	5.54%	33.50°	3.41%	27.87°
Without conformal	4.53%	29.06°	2.73%	21.42°
Without second	1.22%	10.47°	2.60%	20.48°

In the table, "Our framework" means to use the complete Con-NRSfM method shown in Algorithm 1. "Without Pre-Step" means to ignore the operations Pre-Step in Section V-B and the others following the complete method. "Without separable opt." means joint optimization of all the variables, instead of the operations Step 1-4 in Section V-B. "Without conformal" means the method setting all the conformal scales as 1. "Without second" represents the method setting all the second-order derivatives as 0. Our experiments show that the introduction of the separable optimization framework, the conformal scale, and second-order term all help a lot in improving the result accuracy. Synthetic 1 dataset benefits more from the conformal scale than the near-isometric T-shirt dataset, as it follows the conformal deformation.

4) Parameter Sensitivity: In this section, we evaluate the parameter sensitivity of our algorithm to demonstrate its robustness with respect to key parameter settings. As a representative example, we examine the impact of the optimization iteration count N_t used in Step 1, 2, and 4. Since our method follows the idea of alternating optimization, it is not necessary to fully optimize each variable in every iteration. Instead, the design encourages updating variables in a small number of steps while keeping others fixed. We vary N_t from 1 to 5 and report the corresponding results on the T-shirt and Flag datasets in Fig. 27. The results indicate that our method maintains stable performance across different values of N_t, highlighting its robustness to this parameter. To strike a balance between computational efficiency and accuracy, we set the default value of N_t to 3 in our experiments[18].

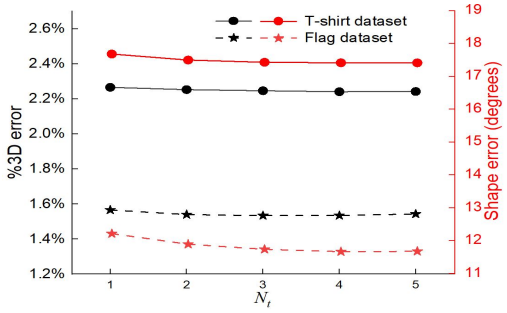


Fig. 27: Estimation results for two datasets with different N_t.

5) Stability and Failure Cases: Because of the accurate image warp, the point-wise graph optimization, and the separable optimization framework, our complete method shown in Algorithm 1 is very stable and robust with the initial value (Pure zeros/ones or random values), noises (Fig. 11), and missing data (Table II). In all our experiments, the initial values of the variables are all zeros/ones without any specific prior. Our method can handle most NRSfM scenarios and the failure case did not happen in all our experiments. Because our method relies on image correspondences to formulate reconstruction constraints, our method will fail when the existing image matching methods fail to detect the image feature correspondences, which is also challenging for other feature-based approaches.

[18]This experiment was conducted on a new desktop equipped with an Intel Core i7-13700K CPU. As a result, very small numerical differences may exist compared to the results presented in Section VI-B.

6) Limitations: As mentioned in Section III-B, a key assumption of this work is that the deforming object is modeled as a Riemannian manifold, implying a continuous and smooth surface without discontinuities or self-intersections. Consequently, this method is not well-suited for reconstructing highly non-surface or large-scale deforming objects, such as complex human limb motions (Human3.6M [69]) or articulated hand movements (InterHand2.6M [70]). It is important to note that, although the point-wise formulation improves flexibility and robustness in handling sparse correspondences and moderate deformations, it still fundamentally assumes that all features lie on a continuous surface. That is, each feature is expected to have a valid mapping to a local patch of the underlying manifold. As a result, if the features originate from regions with large-area self-intersections, occluded folds, or detached components, this assumption breaks down, and the reconstruction may become very inaccurate. Another limitation stems from the current implementation, which relies heavily on MATLAB's TRR-based optimization tools. This reliance results in memory constraints when processing large datasets such as Rug$\times$300. The implementation handles medium-sized datasets like Rug$\times$50 effectively, achieving comparable computational performance to other SOTA methods (Table V). However, for larger datasets, a practical way is to divide the data into smaller subsets and perform reconstruction separately on each to avoid memory overflow.

VII. Conclusion and future work

This paper introduces a novel theoretical framework, Con-NRSfM, for addressing conformal NRSfM problems. The framework leverages the rotational invariance of connections to express conformal constraints without relying on assumptions about surface geometry. The proposed formulation is solved using selected image warps, a separable parallel graph optimization strategy, and a self-supervised convolutional network. Our method has been extensively tested on a wide range of synthetic and real datasets featuring diverse baseline viewpoints and deformations. The results demonstrate that it outperforms SOTA methods in both accuracy and robustness.

This paper marks an initial step toward applying differential geometric modeling in NRSfM. We aim to extend the framework to address broader deformation cases, including topological changes and complex surface dynamics. Future work will focus on key challenges such as faster reconstruction, novel constraints, denser outputs, and self-occlusion handling. Furthermore, we plan to integrate advanced pose estimation techniques, including learning-based and keypoint-tracking methods, to develop a comprehensive online visual deformable SLAM system. In parallel, 3D Gaussian Splatting (3DGS)–based SLAM [71] has surged in popularity because, by enforcing photometric consistency across (nearly) all pixels rather than sparse keypoints, it leverages far more of the image signal and typically achieves higher accuracy. Future work will introduce the novel local physical constraints into the 4DGS SLAM system [72], combining image rendering with assumed physical information to enhance accuracy.

Acknowledgments

We are grateful to Dr. Jose Lamarca for his support with [2] and datasets in Section VI-D, and to the reviewers for their constructive comments.

VIII. Proof of Claim 1

Proof. Let's consider the composite function $\Phi_1 = \Psi_{21} \circ \Phi_2 \circ \eta_{12}$, for the locally diffeomorphic mapping Ψ_{21}, we have $(\bar{\boldsymbol{e}}_1^*, \bar{\boldsymbol{e}}_2^*) = \Upsilon\mathbf{R}(\boldsymbol{e}_1^*, \boldsymbol{e}_2^*)$, $\Upsilon = \text{diag}(\lambda_1, \lambda_2, \lambda_3)$. For the cross basis $\bar{\boldsymbol{e}}_3^* = \bar{\boldsymbol{e}}_1^* \times \bar{\boldsymbol{e}}_2^*$, we have: $\bar{\boldsymbol{e}}_3^* = \Upsilon^*\mathbf{R}\boldsymbol{e}_3^*$, $\Upsilon^* = \text{diag}(\lambda_2\lambda_3, \lambda_1\lambda_3, \lambda_1\lambda_2)$. In short, the moving frame of the composite mapping Φ_1, can be written as:

$$\begin{aligned}\overline{E}(\Phi_1) &= E(\Psi_{21} \circ \Phi_2 \circ \eta_{12}) = (\bar{\boldsymbol{e}}_1^*, \bar{\boldsymbol{e}}_2^*, \bar{\boldsymbol{e}}_3^*) \\ &= (\Upsilon\mathbf{R}(\boldsymbol{e}_1^*, \boldsymbol{e}_2^*), \Upsilon^*\mathbf{R}\boldsymbol{e}_3^*) = \Upsilon\mathbf{R}(\boldsymbol{e}_1^*, \boldsymbol{e}_2^*, \boldsymbol{C}\boldsymbol{e}_3^*) \\ &= \Upsilon\mathbf{R}(\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{C}\boldsymbol{e}_3)\mathbf{J}_{\eta 3\times 3},\end{aligned} \tag{19}$$

where $\boldsymbol{C} = (\Upsilon\mathbf{R})^{-1}\,\Upsilon^*\,\mathbf{R}$.

Based on the definition of the connections, we apply the derivative to the moving frame. The differential vectors are:

$$\begin{aligned}(d\bar{\boldsymbol{e}}_1^*, d\bar{\boldsymbol{e}}_2^*, d\bar{\boldsymbol{e}}_3^*) &= \Upsilon\mathbf{R}(d\boldsymbol{e}_1, d\boldsymbol{e}_2, \boldsymbol{C}d\boldsymbol{e}_3)\mathbf{J}_{\eta 3\times 3} \\ &+ \Upsilon\mathbf{R}(e_1, e_2, \boldsymbol{C}\boldsymbol{e}_3)\text{diag}(d\mathbf{J}_{\eta_{12}}, d\det(\mathbf{J}_{\eta_{12}})).\end{aligned} \tag{20}$$

For the left hand side of the differential vectors (20), we can represent the connection by the coordinate axises. Let's consider

$$\begin{aligned}d\bar{\boldsymbol{e}}_j^* &= \frac{\partial\,\bar{\boldsymbol{e}}_j^*}{\partial\,\bar{u}}d\bar{u} + \frac{\partial\,\bar{\boldsymbol{e}}_j^*}{\partial\,\bar{v}}d\bar{v} \\ \frac{\partial\,\bar{\boldsymbol{e}}_j^*}{\partial\,\bar{u}} &= \Gamma_{j1}^1(\Phi_1)\bar{\boldsymbol{e}}_1^* + \Gamma_{j1}^2(\Phi_1)\bar{\boldsymbol{e}}_2^* + \Gamma_{j1}^3(\Phi_1)\bar{\boldsymbol{e}}_3^* \\ \frac{\partial\,\bar{\boldsymbol{e}}_j^*}{\partial\,v} &= \Gamma_{j2}^1(\Phi_1)\bar{\boldsymbol{e}}_1^* + \Gamma_{j2}^2(\Phi_1)\bar{\boldsymbol{e}}_2^* + \Gamma_{j2}^3(\Phi_1)\bar{\boldsymbol{e}}_3^*,\end{aligned} \tag{21}$$

we have:

$$\begin{aligned}d\bar{\boldsymbol{e}}_j^* &= (\Gamma_{j1}^1(\Phi_1)\bar{\boldsymbol{e}}_1^* + \Gamma_{j1}^2(\Phi_1)\bar{\boldsymbol{e}}_2^* + \Gamma_{j1}^3(\Phi_1)\bar{\boldsymbol{e}}_3^*)d\bar{u} \\ &+ (\Gamma_{j2}^1(\Phi_1)\bar{\boldsymbol{e}}_1^* + \Gamma_{j2}^2(\Phi_1)\bar{\boldsymbol{e}}_2^* + \Gamma_{j2}^3(\Phi_1)\bar{\boldsymbol{e}}_3^*)d\bar{v}.\end{aligned} \tag{22}$$

Rewrite $\bar{\omega}_j^i = \Gamma_{j1}^i(\Phi_1)d\bar{u} + \Gamma_{j2}^i(\Phi_1)d\bar{v}$, we have: $d\bar{\boldsymbol{e}}_j^* = \bar{\omega}_j^1\bar{\boldsymbol{e}}_1^* + \bar{\omega}_j^2\bar{\boldsymbol{e}}_2^* + \bar{\omega}_j^3\bar{\boldsymbol{e}}_3^*$. Then, we can get:

$$\begin{aligned}(d\bar{\boldsymbol{e}}_1^*, d\bar{\boldsymbol{e}}_2^*, d\bar{\boldsymbol{e}}_3^*) &= (\bar{\boldsymbol{e}}_1^*, \bar{\boldsymbol{e}}_2^*, \bar{\boldsymbol{e}}_3^*)\left(\bar{\omega}_j^i\right)_{3\times 3} \\ &= \Upsilon\mathbf{R}(\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{C}\boldsymbol{e}_3)\mathbf{J}_{\eta 3\times 3}\left(\bar{\omega}_j^i\right)_{3\times 3}.\end{aligned} \tag{23}$$

Similarly, for the right hand side of the differential vectors (23), based on $(d\boldsymbol{e}_1, d\boldsymbol{e}_2, d\boldsymbol{e}_3) = (\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3)\left(\omega_j^i\right)_{3\times 3}$, we also have:

$$\begin{aligned}&(d\boldsymbol{e}_1, d\boldsymbol{e}_2, \boldsymbol{C}d\boldsymbol{e}_3) \\ &= \left((\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3)\left(\omega_j^i\right)_{3\times 2}, \boldsymbol{C}(\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3)\left(\omega_3^i\right)_{3\times 1}\right) \\ &= (E(\Phi_2)\left(\omega_j^i\right)_{3\times 2},\ \boldsymbol{C}E(\Phi_2)\left(\omega_j^i\right)_{3\times 1}),\end{aligned} \tag{24}$$

where $(\cdot)_{\bullet\times\circ}$, $\left(\omega_j^i\right)_{3\times 2}$ and $\left(\omega_3^i\right)_{3\times 1}$ denote the sub-matrices of ω_j^i corresponding to its first two columns and its last column, respectively.

Introducing (23) and (24) into (20), we have:

$$\begin{aligned}&\Upsilon\mathbf{R}(\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{C}\boldsymbol{e}_3)\mathbf{J}_{\eta 3\times 3}\left(\bar{\omega}_j^i\right)_{3\times 3} = \Upsilon\mathbf{R} \\ &(E(\Phi_2)\left(\omega_j^i\right)_{3\times 2}, \boldsymbol{C}E(\Phi_2)\left(\omega_3^i\right)_{3\times 1})\mathbf{J}_{\eta 3\times 3} \\ &+ \Upsilon\mathbf{R}(\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{C}\boldsymbol{e}_3)\text{diag}(d\mathbf{J}_{\eta_{12}}, d\det(\mathbf{J}_{\eta_{12}})).\end{aligned} \tag{25}$$

Let's delete $\Upsilon\mathbf{R}$, we have:

$$\begin{aligned}&\left(\bar{\omega}_j^i\right)_{3\times3} = f_1(\Upsilon,\mathbf{R},\left(\omega_j^i\right)_{3\times3})+\\&\mathbf{J}_{\eta^3\times3}^{-1}\text{diag}(d\mathbf{J}_{\eta_{12}}, d\det(\mathbf{J}_{\eta_{12}})),\end{aligned}\tag{26}$$

where,

$$\begin{aligned}&f_1(\Upsilon,\mathbf{R},\left(\omega_j^i\right)_{3\times3}) = \mathbf{J}_{\eta^3\times3}^{-1}(\boldsymbol{e}_1,\boldsymbol{e}_2,\boldsymbol{C}\boldsymbol{e}_3)^{-1}\\&(E(\Phi_2)\left(\omega_j^i\right)_{3\times2}, \boldsymbol{C}E(\Phi_2)\left(\omega_3^i\right)_{3\times1})\mathbf{J}_{\eta^3\times3}.\end{aligned}\tag{27}$$

Introduce $\omega_j^i = \Gamma_{j1}^i(\Phi_2)du + \Gamma_{j2}^i(\Phi_2)dv$, $\bar{\omega}_j^i = \Gamma_{j1}^i(\Phi_1)$ $d\bar{u}+\Gamma_{j2}^i(\Phi_1)d\bar{v}$ into the above equation (26). We can find that the function is greatly related to the conformal scale (matrix form) Υ and the rotation matrix $\mathbf{R}$, which means that the connections of the mappings Φ_2 and $\Phi_2\circ\eta_{12}$ do not have the invariance property. So Claim 1 is proved. □

IX. Proof of Theorem 1

Proof. Let's consider the composite function $\Phi_1 = \Psi_{21}\circ\Phi_2\circ\eta_{12}$, for the conformal case $\lambda_1=\lambda_2=\lambda_3=\lambda$, the moving frame of the composite mapping, Φ_1, can be written as: $\overline{E}(\Phi_1) = \mathbf{R}E(\Phi_2)\mathbf{J}_{\eta_{3\times3}}\Lambda$. Its differential vector is:

$$\begin{aligned}&(d\bar{\boldsymbol{e}}_1^*, d\bar{\boldsymbol{e}}_2^*, d\bar{\boldsymbol{e}}_3^*) = \mathbf{R}(d\boldsymbol{e}_1, d\boldsymbol{e}_2, d\boldsymbol{e}_3)\mathbf{J}_{\eta_{3\times3}}\Lambda\\&+\mathbf{R}E(\Phi_2)\text{diag}(d\mathbf{J}_{\eta_{12}}, d\det(\mathbf{J}_{\eta_{12}}))\Lambda.\end{aligned}\tag{28}$$

For the left-hand side of the differential vectors (28), we can represent the connection by the coordinate axes. Let's consider:

$$\begin{aligned}&d\bar{\boldsymbol{e}}_j^* = \frac{\partial\,\bar{\boldsymbol{e}}_j^*}{\partial\,\bar{u}}d\bar{u} + \frac{\partial\,\bar{\boldsymbol{e}}_j^*}{\partial\,\bar{v}}d\bar{v},\\&\frac{\partial\,\bar{\boldsymbol{e}}_j^*}{\partial\,\bar{u}} = \Gamma_{j1}^1(\Phi_1)\bar{\boldsymbol{e}}_1^* + \Gamma_{j1}^2(\Phi_1)\bar{\boldsymbol{e}}_2^* + \Gamma_{j1}^3(\Phi_1)\bar{\boldsymbol{e}}_3^*,\\&\frac{\partial\,\bar{\boldsymbol{e}}_j^*}{\partial\,v} = \Gamma_{j2}^1(\Phi_1)\bar{\boldsymbol{e}}_1^* + \Gamma_{j2}^2(\Phi_1)\bar{\boldsymbol{e}}_2^* + \Gamma_{j2}^3(\Phi_1)\bar{\boldsymbol{e}}_3^*,\end{aligned}\tag{29}$$

we have:

$$\begin{aligned}&d\bar{\boldsymbol{e}}_j^* = (\Gamma_{j1}^1(\Phi_1)\bar{\boldsymbol{e}}_1^* + \Gamma_{j1}^2(\Phi_1)\bar{\boldsymbol{e}}_2^* + \Gamma_{j1}^3(\Phi_1)\bar{\boldsymbol{e}}_3^*)d\bar{u}\\&+(\Gamma_{j2}^1(\Phi_1)\bar{\boldsymbol{e}}_1^* + \Gamma_{j2}^2(\Phi_1)\bar{\boldsymbol{e}}_2^* + \Gamma_{j2}^3(\Phi_1)\bar{\boldsymbol{e}}_3^*)d\bar{v}.\end{aligned}\tag{30}$$

Rewrite:

$$\bar{\omega}_j^i = \Gamma_{j1}^i(\Phi_1)d\bar{u} + \Gamma_{j2}^i(\Phi_1)d\bar{v},\tag{31}$$

we have:

$$d\bar{\boldsymbol{e}}_j^* = \bar{\omega}_j^1\bar{\boldsymbol{e}}_1^* + \bar{\omega}_j^2\bar{\boldsymbol{e}}_2^* + \bar{\omega}_j^3\bar{\boldsymbol{e}}_3^*.\tag{32}$$

Further, based on:

$$\begin{aligned}&(d\boldsymbol{e}_1, d\boldsymbol{e}_2, d\boldsymbol{e}_3) = (\boldsymbol{e}_1,\boldsymbol{e}_2,\boldsymbol{e}_3)(\omega_j^i)_{3\times3} = E(\Phi_2)(\omega_j^i)_{3\times3},\\&(d\bar{\boldsymbol{e}}_1^*, d\bar{\boldsymbol{e}}_2^*, d\bar{\boldsymbol{e}}_3^*) = E(\Psi_{21}\circ\Phi_2\circ\eta_{12})(\bar{\omega}_j^i)_{3\times3}\\&\qquad = (\bar{\boldsymbol{e}}_1^*,\bar{\boldsymbol{e}}_2^*,\bar{\boldsymbol{e}}_3^*)(\bar{\omega}_j^i)_{3\times3} = \overline{E}(\Phi_1)(\bar{\omega}_j^i)_{3\times3},\end{aligned}\tag{33}$$

and (10), the previous equation (28) can be written as:

$$\begin{aligned}&\mathbf{R}E(\Phi_2)\mathbf{J}_{\eta_{3\times3}}\Lambda\left(\bar{\omega}_j^i\right)_{3\times3} = \mathbf{R}E(\Phi_2)\left(\omega_j^i\right)_{3\times3}\mathbf{J}_{\eta_{3\times3}}\Lambda\\&+\mathbf{R}E(\Phi_2)\text{diag}(d\mathbf{J}_{\eta_{12}}, d\det(\mathbf{J}_{\eta_{12}}))\Lambda.\end{aligned}\tag{34}$$

Multiplying $E(\Phi_2)^{-1}\mathbf{R}^{-1}$ on both sides, because of:

$$\begin{aligned}&\bar{\omega}_j^i = \Gamma_{j1}^i(\Phi_2)du + \Gamma_{j2}^i(\Phi_2)dv,\\&\bar{\omega}_j^i = \Gamma_{j1}^i(\Phi_1)d\bar{u} + \Gamma_{j2}^i(\Phi_1)d\bar{v},\\&\text{diag}(d\mathbf{J}_{\eta_{12}}, d\det(\mathbf{J}_{\eta_{12}})) = \text{diag}(\frac{\partial\mathbf{J}_{\eta_{12}}}{\partial\bar{u}},\\&\frac{\partial\det(\mathbf{J}_{\eta_{12}})}{\partial\bar{u}})d\bar{u} + \text{diag}(\frac{\partial\mathbf{J}_{\eta_{12}}}{\partial\bar{v}}, \frac{\partial\det(\mathbf{J}_{\eta_{12}})}{\partial\bar{v}})d\bar{v},\end{aligned}\tag{35}$$

and the orthogonality of $d\bar{u}$ and $d\bar{v}$, we have:

$$\begin{aligned}&(1)\ \mathbf{J}_{\eta_{3\times3}}\Lambda\Gamma_{j1}^i(\Phi_1) = \Gamma_{j1}^i(\Phi_2)\frac{\partial u}{\partial\bar{u}}\mathbf{J}_{\eta_{3\times3}}\Lambda+\\&\Gamma_{j2}^i(\Phi_2)\frac{\partial v}{\partial\bar{u}}\mathbf{J}_{\eta_{3\times3}}\Lambda + \text{diag}(\frac{\partial\mathbf{J}_{\eta_{12}}}{\partial\bar{u}}, \frac{\partial\det(\mathbf{J}_{\eta_{12}})}{\partial\bar{u}})\Lambda,\\&(2)\ \mathbf{J}_{\eta_{3\times3}}\Lambda\Gamma_{j2}^i(\Phi_1) = \Gamma_{j1}^i(\Phi_2)\frac{\partial u}{\partial\bar{v}}\mathbf{J}_{\eta_{3\times3}}\Lambda+\\&\Gamma_{j2}^i(\Phi_2)\frac{\partial v}{\partial\bar{v}}\mathbf{J}_{\eta_{3\times3}}\Lambda + \text{diag}(\frac{\partial\mathbf{J}_{\eta_{12}}}{\partial\bar{v}}, \frac{\partial\det(\mathbf{J}_{\eta_{12}})}{\partial\bar{v}})\Lambda.\end{aligned}\tag{36}$$

So Theorem 1 is proved. □

X. Proof of Corollary 1

Proof. As an example, because the deduction process of the first equation of the connection (7) is very similar to the second one, we only show the deduction process of the first one. For the conformal deformation Ψ_{21}, writing $\Gamma_{j1}^i(\Phi_1) = (\bar{\boldsymbol{T}}_{kl}^1)$, $\Gamma_{j1}^i(\Phi_2) = (\boldsymbol{T}_{kl}^1)$, and $\Gamma_{j2}^i(\Phi_2) = (\boldsymbol{T}_{kl}^2)$, $k,l=1,2$ as 2×2 block matrices, we can re-write (12) in Theorem 1 as the block matrix formulation:

$$\begin{aligned}&\begin{pmatrix}\mathbf{J}_{\eta_{12}} & \mathbf{0}\\ \mathbf{0} & \det(\mathbf{J}_{\eta_{12}})\end{pmatrix}\begin{pmatrix}\lambda\boldsymbol{I}_{2\times2} & \mathbf{0}\\ \mathbf{0} & \lambda^2\end{pmatrix}\begin{pmatrix}\bar{\boldsymbol{T}}_{11}^1 & \bar{\boldsymbol{T}}_{12}^1\\ \bar{\boldsymbol{T}}_{21}^1 & \bar{\boldsymbol{T}}_{22}^1\end{pmatrix}\\&=\begin{pmatrix}\boldsymbol{T}_{11}^1 & \boldsymbol{T}_{12}^1\\ \boldsymbol{T}_{21}^1 & \boldsymbol{T}_{22}^1\end{pmatrix}\frac{\partial u}{\partial\bar{u}}\mathbf{J}_{\eta3\times3}\begin{pmatrix}\lambda\boldsymbol{I}_{2\times2} & \mathbf{0}\\ \mathbf{0} & \lambda^2\end{pmatrix}\\&+\begin{pmatrix}\boldsymbol{T}_{11}^2 & \boldsymbol{T}_{12}^2\\ \boldsymbol{T}_{21}^2 & \boldsymbol{T}_{22}^2\end{pmatrix}\frac{\partial v}{\partial\bar{u}}\mathbf{J}_{\eta3\times3}\begin{pmatrix}\lambda\boldsymbol{I}_{2\times2} & \mathbf{0}\\ \mathbf{0} & \lambda^2\end{pmatrix}\\&+\text{diag}(\frac{\partial\mathbf{J}_{\eta_{12}}}{\partial\bar{u}}, \frac{\partial\det(\mathbf{J}_{\eta_{12}})}{\partial\bar{u}})\text{diag}(\lambda,\lambda,\lambda^2)\\&\Rightarrow\begin{pmatrix}\lambda\mathbf{J}_{\eta_{12}}\bar{\boldsymbol{T}}_{11}^1 & \lambda\mathbf{J}_{\eta_{12}}\bar{\boldsymbol{T}}_{12}^1\\ \lambda^2\det(\mathbf{J}_{\eta_{12}})\bar{\boldsymbol{T}}_{21}^1 & \lambda^2\det(\mathbf{J}_{\eta_{12}})\bar{\boldsymbol{T}}_{22}^1\end{pmatrix}\\&=\frac{\partial u}{\partial\bar{u}}\begin{pmatrix}\lambda\boldsymbol{T}_{11}^1\mathbf{J}_{\eta_{12}} & \lambda^2\boldsymbol{T}_{12}^1\det(\mathbf{J}_{\eta_{12}})\\ \lambda\boldsymbol{T}_{21}^1\mathbf{J}_{\eta_{12}} & \lambda^2\boldsymbol{T}_{22}^1\det(\mathbf{J}_{\eta_{12}})\end{pmatrix}\\&+\frac{\partial v}{\partial\bar{u}}\begin{pmatrix}\lambda\boldsymbol{T}_{11}^2\mathbf{J}_{\eta_{12}} & \lambda^2\boldsymbol{T}_{12}^2\det(\mathbf{J}_{\eta_{12}})\\ \lambda\boldsymbol{T}_{21}^2\mathbf{J}_{\eta_{12}} & \lambda^2\boldsymbol{T}_{22}^2\det(\mathbf{J}_{\eta_{12}})\end{pmatrix}\\&+\begin{pmatrix}\lambda\frac{\partial\mathbf{J}_{\eta_{12}}}{\partial\bar{u}} & \mathbf{0}\\ \mathbf{0} & \lambda^2\frac{\partial\det(\mathbf{J}_{\eta_{12}})}{\partial\bar{u}}\end{pmatrix}\Rightarrow\\&\begin{cases}\lambda\mathbf{J}_{\eta_{12}}\bar{\boldsymbol{T}}_{11}^1 = \frac{\partial u}{\partial\bar{u}}\lambda\boldsymbol{T}_{11}^1\mathbf{J}_{\eta_{12}} + \frac{\partial v}{\partial\bar{u}}\lambda\boldsymbol{T}_{11}^2\mathbf{J}_{\eta_{12}} + \lambda\frac{\partial\mathbf{J}_{\eta_{12}}}{\partial\bar{u}}\\ \lambda^2\det(\mathbf{J}_{\eta_{12}})\bar{\boldsymbol{T}}_{22}^1 = \frac{\partial u}{\partial\bar{u}}\lambda^2\boldsymbol{T}_{22}^1\det(\mathbf{J}_{\eta_{12}})\\ +\frac{\partial v}{\partial\bar{u}}\lambda^2\boldsymbol{T}_{22}^2\det(\mathbf{J}_{\eta_{12}}) + \lambda^2\frac{\partial\det(\mathbf{J}_{\eta_{12}})}{\partial\bar{u}}\\ \lambda\mathbf{J}_{\eta_{12}}\bar{\boldsymbol{T}}_{12}^1 = \frac{\partial u}{\partial\bar{u}}\lambda^2\boldsymbol{T}_{12}^1\det(\mathbf{J}_{\eta_{12}}) + \frac{\partial v}{\partial\bar{u}}\lambda^2\boldsymbol{T}_{12}^2\det(\mathbf{J}_{\eta_{12}})\\ \lambda^2\det(\mathbf{J}_{\eta_{12}})\bar{\boldsymbol{T}}_{21}^1 = \frac{\partial u}{\partial\bar{u}}\lambda\boldsymbol{T}_{21}^1\mathbf{J}_{\eta_{12}} + \frac{\partial v}{\partial\bar{u}}\lambda\boldsymbol{T}_{21}^2\mathbf{J}_{\eta_{12}}\end{cases},\end{aligned}\tag{37}$$

where $\bar{\boldsymbol{T}}_{11}^1,\ \boldsymbol{T}_{11}^1,\ \boldsymbol{T}_{11}^2 \in \mathbb{R}_{2\times2}$, $\bar{\boldsymbol{T}}_{12}^1,\ \boldsymbol{T}_{12}^1,\ \boldsymbol{T}_{12}^2 \in \mathbb{R}_{2\times1}$, $\bar{\boldsymbol{T}}_{21}^1,\ \boldsymbol{T}_{21}^1,\ \boldsymbol{T}_{21}^2 \in \mathbb{R}_{1\times2}$, and $\bar{\boldsymbol{T}}_{22}^1,\ \boldsymbol{T}_{22}^1,\ \boldsymbol{T}_{22}^2 \in \mathbb{R}_{1\times1}$.

Deleting λ or λ^2, in the final 4 equations, we see that, the front two are not related to the conformal scale λ, which means that the second-order leading principal minors and the last

elements of the connections $\Gamma^i_{jk}(\Phi_1)$ and $\Gamma^i_{jk}(\Phi_2 \circ \eta_{12})$ are invariant. So Corollary 1 is proved. □

XI. Proof of Corollary 2

Proof. Based on the block matrix formulation in Corollary 2, viewing the conformal scale λ as the variable, in (37), we find that the relation between the (1,3)-th, (2,3)-th, (3,1)-th, and (3,2)-th elements of the connections $\Gamma^i_{jk}(\Phi_1)$ and $\Gamma^i_{jk}(\Phi_2 \circ \eta_{12})$ of the mapping Φ_1 and $\Phi_2 \circ \eta_{12}$ are linear functions, satisfying:

$$\begin{aligned}
&\lambda \det(\mathbf{J}_{\eta_{12}})\bar{\boldsymbol{T}}^1_{21} = \frac{\partial u}{\partial \bar{u}}\boldsymbol{T}^1_{21}\mathbf{J}_{\eta_{12}} + \frac{\partial v}{\partial \bar{u}}\boldsymbol{T}^2_{21}\mathbf{J}_{\eta_{12}},\\
&(\frac{\partial u}{\partial \bar{u}}\boldsymbol{T}^1_{12} + \frac{\partial v}{\partial \bar{u}}\boldsymbol{T}^2_{12})\det(\mathbf{J}_{\eta_{12}})\lambda = \mathbf{J}_{\eta_{12}}\bar{\boldsymbol{T}}^1_{12},\\
&\lambda \det(\mathbf{J}_{\eta_{12}})\bar{\boldsymbol{T}}^2_{21} = \frac{\partial u}{\partial \bar{v}}\boldsymbol{T}^1_{21}\mathbf{J}_{\eta_{12}} + \frac{\partial v}{\partial \bar{v}}\boldsymbol{T}^2_{21}\mathbf{J}_{\eta_{12}},\\
&(\frac{\partial u}{\partial \bar{v}}\boldsymbol{T}^1_{12} + \frac{\partial v}{\partial \bar{v}}\boldsymbol{T}^2_{12})\det(\mathbf{J}_{\eta_{12}})\lambda = \mathbf{J}_{\eta_{12}}\bar{\boldsymbol{T}}^2_{12},
\end{aligned} \tag{38}$$

For these 8 linear functions, moving other terms into the right-hand side, they can all be written as the following formulation $\lambda - \alpha_i = 0,\ i = 1, \cdots, 8$. Hence, the sum of the squares problem of the conformal scale can be formulated as:

$$\min_{\lambda} \sum_{i=1}^{8} (\lambda - \alpha_i)^2,\ s.t.\ \lambda \in \mathbb{R}. \tag{39}$$

Ignoring the constraint $\lambda \in \mathbb{R}$, its closed-form solution is:

$$\lambda = \frac{\sum_{i=1}^n \alpha_i \pm \sqrt{\left(\sum_{i=1}^n \alpha_i\right)^2 - n\sum_{i=1}^n \alpha_i^2}}{n},\ n = 8. \tag{40}$$

Based on the AM-QM inequality (arithmetic mean, and quadratic mean), we have $\left(\sum_{i=1}^n \alpha_i\right)^2 - n\sum_{i=1}^n \alpha_i^2 \leq 0$. Hence, the closest real value solution is $\lambda = \sum_{i=1}^n \alpha_i / n$. By introducing all relationships in (38), we can get the solution in (13). So Corollary 2 is proved. □

References

[1] S. Parashar and A. Bartoli, "3DVFX: 3D video editing using non-rigid structure-from-motion," *Eurographics* (Short Papers), pp. 29-32, 2019.

[2] J. Lamarca, S. Parashar, A. Bartoli, and J. M. M. Montiel, "DefSLAM: Tracking and mapping of deforming scenes from monocular sequences," *IEEE Trans. on Robot.*, vol. 37, no. 1, pp. 291-303, 2021.

[3] J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós. "NR-slam: Non-rigid monocular slam." *IEEE Trans. on Robot.*, vol. 40, pp. 4252-4264, 2024.

[4] T. Deng, G. Shen, C. Xun, S. Yuan, T. Jin, H. Shen, Y. Wang, J. Wang, H. Wang, D. Wang, and W. Chen, "Mne-slam: Multi-agent neural slam for mobile robots," in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2025, pp. 1485-1494.

[5] L. Torresani, A. Hertzmann, and C. Bregler, "Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 30, no. 5, pp. 878-892, 2008.

[6] I. Akhter, Y. Sheikh, S. Khan, and T. Kanade, "Trajectory space: A dual representation for nonrigid structure from motion," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 33, no. 7, pp. 1442-1456, 2010.

[7] Y. Chen, L. Zhao, Y. Zhang, and S. Huang, "Dense isometric non-rigid shape-from-motion based on graph optimization and edge selection," *IEEE Robot. and Autom. Lett.*, vol. 5, no. 4, pp. 5889-5896, 2020.

[8] A. Chhatkuli, D. Pizarro, T. Collins, and A. Bartoli, "Inextensible non-rigid structure-from-motion by second-order cone programming," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 40, no. 10, pp. 2428-2441, 2017.

[9] A. Sengupta, K. Makki and A. Bartoli, "Using specularities to boost non-rigid structure-from-motion," in *Proc. IEEE Int. Conf. on Robot. Auto. (ICRA)*, 2024, pp. 2652-2659.

[10] A. Chhatkuli, D. Pizarro, and A. Bartoli, "Nonrigid shape-from-motion for isometric surfaces using infinitesimal planarity," in *Proc. British Mach. Vis. Conf. (BMVC)*, 2014, pp. 1-12.

[11] S. Parashar, D. Pizarro, and A. Bartoli, "Isometric non-rigid shapefrom-motion with riemannian geometry solved in linear time," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 40, no. 10, pp. 2442-2454, 2018.

[12] S. Parashar, D. Pizarro, and A. Bartoli, "Local deformable 3d reconstruction with cartan's connections," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 42, no. 12, pp. 3011-3026, 2021.

[13] S. Parashar, M. Salzmann, and P. Fua, "Local non-rigid structure-from-motion from diffeomorphic mappings," in *Proc. IEEE Conf. Comput. Vis. Patt. Recog. (CVPR)*, 2020, pp. 2059-2067.

[14] V. Sidhu, E. Tretschk, V. Golyanik, A. Agudo, and C. Theobalt, "Neural dense non-rigid structure from motion with latent space constraints," in *Eur. Conf. Comput. Vis (ECCV)*, 2020, pp. 204-222.

[15] A. D. Bue, "A factorization approach to structure from motion with shape priors," in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2008, pp. 1-8.

[16] Y. Dai, H. Li, and M. He, "A simple prior-free method for non-rigid structure-from-motion factorization," *Int. J. Comput. Vis.*, vol. 107, no. 2, pp. 101-122, 2014.

[17] S. Graßhof and S. S. Brandt. "Tensor-based non-rigid structure from motion." in *Proc. of the IEEE/CVF Winter Conf. on Appl. of Comput. Vis. (WACV)*, 2022, pp. 3011-3020.

[18] I. Akhter, Y. Sheikh, S. Khan, and T. Kanade, "Nonrigid structure from motion in trajectory space," in *Proc. Neural Inf. Process. Syst. (NIPS)*, 2009, pp. 1-8.

[19] A. Agudo and F. M. Noguer, "Force-based representation for non-rigid shape and elastic model estimation," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 40, no. 9, pp. 2137-2150, 2018.

[20] A. Agudo, F. M. Noguer, B. Calvo, and J. Montiel, "Sequential nonrigid structure from motion using physical priors," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 38, no. 5, pp. 979-994, 2016.

[21] A. D. Bue, F. Smeraldi, and L. Agapito, "Non-rigid structure from motion using non- parametric tracking and non-linear optimization," in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2004, pp. 1-8.

[22] L. Torresani, A. Hertzmann, and C. Bregler, "Nonrigid structure-frommotion: Estimating shape and motion with hierarchical priors," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 30, no. 5, pp. 878-892, 2008.

[23] P. F. U. Gotardo and A. M. Martínez, "Kernel non-rigid structure from motion," in *IEEE Int. Conf. Mach. Lear (ICCV)*, 2011, pp. 802-809.

[24] Y. Wang, D. Xu, W. Huang, X. Ye, and M. Jiang, "Temporal-aware neural network for dense non-rigid structure from motion." *Electronics*, vol. 12, no. 18, 3942, 2023.

[25] J. Fayad, A. D. Bue, L. Agapito, and P. M. Aguiar, "Non-rigid structure from motion using quadratic deformation models," in *Proc. British Mach. Vis. Conf. (BMVC)*, 2009, pp. 1-11.

[26] D. Novotny, N. Ravi, B. Graham, N. Neverova, and A. Vedaldi. "C3dpo: Canonical 3d pose networks for non-rigid structure from motion," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 7688-7697.

[27] C. Wang and S. Lucey, "PAUL: Procrustean autoencoder for unsupervised lifting," in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2021, pp. 434-443.

[28] H. Deng, T. Zhang, Y. Dai, J. Shi, and H. Li, "Deep non-rigid structure-from-motion: A sequence-to-sequence translation perspective," in *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 46, no. 12, pp. 10814-10828, 2024.

[29] F. Bookstein, "Principal warps: thin-plate splines and the decomposition of deformations," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 11, no. 6, pp. 567-585, 1989.

[30] J. Taylor, A. D. Jepson, and K. N. Kutulakos, "Non-rigid structure from locally-rigid motion," in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2010, pp. 2761-2768.

[31] C. Russell, R. Yu, and L. Agapito, "Video pop-up: Monocular 3d reconstruction of dynamic scenes," in *Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 583-598.

[32] S. Parashar, D. Pizarro, and A. Bartoli. "Isometric non-rigid shape-from-motion in linear time," in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2016, pp. 4679-4687.

[33] J. J. G. Rodríguez, J. Lamarca, J. Morlana, J. D. Tardós, and J. M. Montiel, "Sd-defslam: Semi-direct monocular slam for deformable and intracorporeal scenes," in *Proc. IEEE Int. Conf. on Robot. Auto. (ICRA)*, 2021, pp. 5170-5177.

[34] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. on Robot.*, vol. 31, no. 5, pp. 1147-1163, 2015.

[35] J. Lamarca and J. M. M. Montiel, "Camera tracking for SLAM in deformable maps," in *Eur. Conf. on Comput. Vis. (ECCV) Workshops*, 2018.

[36] E. Wang, Y. Liu, J. Xu, and X. Chen, "Non-rigid scene reconstruction of deformable soft tissue with monocular endoscopy in minimally invasive surgery," *Int. J. Comput. Assist. Radiol. Surg.*, vol. 19, no. 12, pp. 2433-2443, 2024.

[37] S. Parashar, Y. Long, M. Salzmann, and P. Fua, "A closed-form, pairwise solution to local non-rigid structure-from-motion," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 46, no. 11, pp. 7027-7040, 2024.

[38] P. Ji, H. Li, Y. Dai, and I. Reid, ""Maximizing rigidity" revisited: A convex programming approach for generic 3D shape reconstruction from multiple perspective views," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 929-937.

[39] N. Sundaram, T. Brox, and K. Keutzer, "Dense point trajectories by gpu-accelerated large displacement optical flow," in *Eur. Conf. Comput. Vis. (ECCV)*, 2010, pp. 438-451.

[40] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 1999, pp. 1150-1157.

[41] H. Bay, "Surf: Speeded up robust features," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 404-417, 2006.

[42] Y. Chen, S. Huang, L. Zhao, and G. Dissanayake, "Cramér–rao bounds and optimal design metrics for pose-graph SLAM," *IEEE Trans. on Robot.*, vol. 37, no. 2, pp. 627-641, 2021.

[43] Y. Chen, K. M. B. Lee, C. Yoo, and R. Fitch, "Broadcast your weaknesses: cooperative active pose-graph SLAM for multiple robots," *IEEE Robot. and Autom. Lett.*, vol. 5, no. 2, pp. 2200-2007, 2020.

[44] J. M. Lee. Riemannian manifolds: an introduction to curvature. Springer, 1997.

[45] J. C. Bezdek and R. J. Hathaway, "Convergence of alternating optimization," *Neural Parallel Sci. Comput.*, vol. 11, no. 4, pp. 351-368, 2003.

[46] I. Alhashim and P. Wonka, "High quality monocular depth estimation via transfer learning," arXiv preprint arXiv:1812.11941, 2018.

[47] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2017, pp. 2261-2269.

[48] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and F. F. Li, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2009, pp. 248–255.

[49] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Int. Conf. Mach. Lear (ICML)*, 2015, pp. 448-456.

[50] H. Fu, M. Gong, C. Wang, N. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation,"in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2018, pp. 2002-2011.

[51] J. Matt, Absolute Orientation - Horn's method, MATLAB Central File Exchange, 2023.

[52] D. Henrion, J. B. Lasserre, and J. Löfberg, "GloptiPoly 3: moments, optimization and semidefinite programming," *Optim. Methods Softw.*, vol. 24, no. 4–5, pp. 761-779.

[53] S. Vicente and L. Agapito, "Soft inextensibility constraints for template-free non-rigid reconstruction," in *Eur. Conf. Comput. Vis. (ECCV)*, 2012, pp. 426-440.

[54] S. H. N. Jensen, M. E. B. Doest, H. Aanæs, and A. Del Bue, "A benchmark and evaluation of non-rigid structure from motion," *Int. J. Comput. Vis.*, vol. 129, no. 4, pp. 882-899, 2021.

[55] S. Parashar, A. Bartoli, and D. Pizarro, "Robust isometric non-rigid structure-from-motion," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 44, no. 10, pp. 6409-6423, 2022.

[56] M. D. Ansari, V. Golyanik, and D. Stricker, "Scalable dense monocular surface reconstruction," in *Int. Conf. on 3D Vis. (3DV)*, 2017, pp. 78-87.

[57] M. Lee, S. Cho, and J. Oh, "Consensus of non-rigid reconstructions," in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2016, pp. 4670-4678.

[58] P. Mountney, D. Stoyanov, and G. Z. Yang, "Three-dimensional tissue deformation recovery and tracking," *IEEE Signal Process. Mag.*, vol. 27, no. 4, pp. 14-24, 2010.

[59] D. Stoyanov, G. P. Mylonas, F. Deligianni, A. Darzi, and G. Z. Yang, "Soft-tissue motion tracking and structure estimation for robotic assisted mis procedures," in *Proc. Med. Image Comput. Comput.-Assisted Intervention (MICAI)*, 2005, pp. 139-146.

[60] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM," *IEEE Trans. on Robot.*, vol. 37, no. 6, pp. 1874-1890, 2021.

[61] G. Yang, D. Sun, V. Jampani, D. Vlasic, F. Cole, H. Chang, D. Ramanan, W.T. Freeman, and C. Liu, "Lasr: Learning articulated shape reconstruction from a monocular video," in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2021, pp. 15980-15989.

[62] G. Yang, D. Sun, V. Jampani, D. Vlasic, F. Cole, C. Liu, and D. Ramanan, "Viser: Video-specific surface embeddings for articulated 3d shape reconstruction," in *Proc. Adv. Neural Inf. Process. (NeurIPS)*, 34, pp. 19326-19338.

[63] I. Akhter, Y. Sheikh, S. Khan, and T. Kanade, "Trajectory space: A dual representation for nonrigid structure from motion," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 33, no. 7, pp. 1442-1456, 2011.

[64] M. Paladini, A. Del Bue, J. Xavier, L. Agapito, M. Stosić, and M. Dodig, "Optimal metric projections for deformable and articulated structure-from-motion," *Int. J. Comput. Vis.*, vol. 96, no. 2, pp. 252-276, 2012.

[65] Y. Dai, H. Deng, and M. He, "Dense non-rigid structure-from-motion made easy – a spatial-temporal smoothness based solution," in *Proc. Int. Conf. on Image Processing (ICIP)*, 2017, pp. 4532-4536.

[66] S. Kumar, A. Cherian, Y. Dai, and H. Li, "Scalable dense non-rigid structure-from- motion: A grassmannian perspective," in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2018, pp. 254-263.

[67] S. Kumar, "Jumping manifolds: Geometry aware dense non-rigid structure from motion," in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2019, pp. 5346-5355.

[68] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. on Comput. Vis. Patt. Recog. (CVPR)*, 2016, pp. 770-778.

[69] C. Ionescu, D. Papava, V. Olaru and C. Sminchisescu, "Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments," *IEEE Trans. on Patt. Anal. and Mach. Intell.*, vol. 36, No. 7, pp. 1325-1339, 2014.

[70] G. Moon, S. Yu, H. Wen, and K. M. Lee, "InterHand2.6M: A dataset and baseline for 3D interacting hand pose estimation from a single RGB image," in *Eur. Conf. on Comput. Vis. (ECCV)*, 2020, pp. 548-564.

[71] H. Matsuki, R. Murai, P.H. Kelly, and A.J. Davison, "Gaussian splatting slam," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 18039-18048).

[72] Y. Li, Y. Fang, Z. Zhu, K. Li, Y. Ding, F. Tombari, 4D Gaussian Splatting SLAM, arXiv preprint arXiv:2503.16710, 2025.

Yongbo Chen received the Ph.D. degree in Robotics from the Robotics Institute, UTS, Sydney, Australia, in August 2021. From 2021–2024, he was a Postdoctoral Research Fellow in the School of Computing, The Australian National University (ANU), Canberra, Australia. He is now an Associate Professor with the School of Automation and Intelligent Sensing, Shanghai Jiao Tong University, Shanghai, China. His research interests are simultaneous localization and mapping (SLAM), partially observable Markov decision process (POMDP), optimization and learning techniques in Robotics.

Yanghao Zhang received the Ph.D. degree in robotics from UTS, Sydney, Ultimo, NSW, Australia, in 2022. From 2022-2024, he was a Postdoctoral Research Fellow with the College of Engineering and Computer Science, Australian National University, Canberra, ACT, Australia. He is now a researcher on Embodied AI and Robotics with the Robotics Institute, UTS, Sydney, Australia. His research interests include visual simultaneous localization and mapping and deformation reconstruction.

Shaifali Parashar received Ph.D. degree in computer vision from the Université Auvergne, Clermont-Ferrand, France, in 2017. She was a Postdoctoral Researcher with Computer Vision Laboratory (CVLab), École Polytechnique Fédérale de Lausanne (EPFL) in Lausanne, Switzerland. She is currently a Centre national de la recherche scientifique (CNRS) Research scientist at Institut National des Sciences Appliquées de Lyon (LIRIS, INSA-Lyon). Her research interests include 3D computer vision including nonrigid 3D reconstruction, deformation modeling and deformable SLAM.

Liang Zhao received the Ph.D. degree in photogrammetry and remote sensing from Peking University, Beijing, China, in 2013. From 2014 to 2016, he worked as a Postdoctoral Research Associate with the Hamlyn Centre for Robotic Surgery, Department of Computing, Faculty of Engineering, Imperial College London, London, U.K. From 2016 to 2024, he was a Senior Lecturer at the UTS Robotics Institute, UTS, Sydney, Australia. He is now a Reader in Robot Systems in the School of Informatics, The University of Edinburgh, United Kingdom. His research interests include surgical robotics, autonomous robot SLAM, monocular SLAM, aerial photogrammetry, optimization techniques in mobile robot localization and mapping and image-guided robotic surgery.

Shoudong Huang received his Ph.D. in Automatic Control from Northeastern University, Shenyang, China, in 1998. He is currently a Professor and Deputy Director of the UTS Robotics Institute, as well as Deputy Head of School (Research) in the School of Mechanical and Mechatronic Engineering at the University of Technology Sydney. His research interests include nonlinear system state estimation and control, mobile robot SLAM, and surgical robotics.