KernelGPA: A Deformable SLAM Back-end

Fang Bai* and Adrien Bartoli†

*ENCOV, TGI, Institut Pascal, UMR6602 CNRS. Université Clermont Auvergne, France
Email: Fang.Bai@yahoo.com
†ENCOV, TGI, Institut Pascal, UMR6602 CNRS. Université Clermont Auvergne, France
Department of Clinical Research and Innovation. CHU de Clermont-Ferrand, France
Email: Adrien.Bartoli@gmail.com

Abstract—Simultaneous localization and mapping (SLAM) in the deformable environment has encountered several barricades. One of them is the lack of a global registration technique. Thus current SLAM systems heavily rely on template based methods. We propose KernelGPA, a novel global registration technique to bridge the gap. We define nonrigid transformations using a kernel method, and show that the principal axes of the map can be solved globally in closed-form, up to a global scale ambiguity along each axis. We propose to solve both the global scale ambiguity and rigid poses in a unified optimization framework, yielding a cost that can be readily incorporated in sensor fusion frameworks. We demonstrate the registration performance of KernelGPA using various datasets, with a special focus on computerized tomography (CT) registration. We release our code[1] and data to foster future research in this direction.

I. INTRODUCTION

The simultaneous localization and mapping (SLAM) in the rigid (or static) environment has witnessed a huge success [13]. However in the nonrigid environment, SLAM is largely an open problem [25]. One of the major barricades of extending SLAM from the rigid to nonrigid environment is the shortage of principled global registration techniques that handle deformations. At this stage, there are only several options available in this regards, *i.e.*, the low-rank shape basis decomposition [14], the isometric nonrigid structure-from-motion [40], and more recently the generalized Procrustes analysis (GPA) with the linear basis warps (LBWs) [4].

On the other hand, matching a deformed shape to a template, termed shape-from-template (SfT) in [8, 37], is a relatively mature problem. In computer graphics, the deformation between a template and a deformed shape can be computed within many deformation frameworks, *e.g.*, the as-rigid-as-possible deformation model [49], the embedded deformation graph [1, 50], Lie-bodies [19], and isometric deformations [8]; see the review papers [53, 34]. In robotics, the SfT methods have been implemented in several deformable tracking systems. To name a few, we refer to DynamicFusion [38], Surfelwarp [20], KillingFusion [46], SobolevFusion [47], MIS-SLAM [48], and DefSLAM [36]. None of these systems has a principled global deformable registration, except for the recent work DefSLAM [36] which implements the isometric nonrigid structure-from-motion method from [40].

In this work, we propose a novel global registration technique for deformable SLAM. To make the context clear, we consider the following constraints:

1) **No temporal information.** We assume observations are made without sequential information, thus technologies based on tracking do not apply here.
2) **No template.** We assume a template of the environment is not available. We also disallow alternation methods by incrementally constructing and matching to a template.
3) **No aids on pose estimation.** We assume additional information on the sensor's pose is not available.

In general, we assume that the only available information is the geometric correspondences between sensor observations. Our target is to obtain: a) the estimate of the environment, b) the nonrigid transformations that can explain the sensor data, and c) the estimate of the sensor's rigid poses.

We consider the case where sensor readings are interpreted as point-clouds. The closest work to this purpose is the DefGPA developed in [4], where the globally optimal map and the deformable transformations are estimated in closed-form using the LBW models. In the language of SLAM, the results in [4] show that *the principal axes of the environment map can be estimated globally, up to a global scale ambiguity along each axis.* The authors proposed a method to estimate the global scale ambiguity using pairwise rigid Procrustes analysis, by assuming sufficient co-visibility across sensor measurements at different poses. This is a strong assumption for SLAM, as observations in SLAM are typically local.

In this work, we give a non-trivial extension of DefGPA [4] to deformable SLAM:

- First, we use a kernel method to model deformations, and term the resulting GPA as KernelGPA. We show that KernelGPA can be solved globally in closed-form, up to an unknown global scale ambiguity. KernelGPA provides extra modeling power compared to the DefGPA framework [4] based on the LBWs.
- Second, we give insights that the global scale ambiguity and the rigid transformations are correlated, by assuming the transformation to be *as-rigid-as-possible (ARAP)*. Therefore we can solve both in a unified optimization framework. We do not require good co-visibility as in [4], and the new cost term provides a natural extension to the sensor fusion framework.

---

[1]Code and data: https://bitbucket.org/clermontferrand/deformableprocrustes

The rest of this paper is organized as follows. We briefly review related work on deformation models and deformable registration techniques in Section II. We present our deformable SLAM formulation in Section III and draw its connection to the GPA problem with deformations. We present a deformable transformation model using kernel methods in Section IV, and in Section V we derive that the principal axes of the environment map can be solved globally in closed-form up to a global scale ambiguity $\mathbf{\Lambda}$. Then in Section VI, we propose to solve both $\mathbf{\Lambda}$ and the rigid poses in a unified optimization framework to accommodate local observations in SLAM. In Section VII, we provide implementation details to reproduce the work. We present our experimental results in Section VIII, and conclude the paper in Section IX.

II. RELATED WORK

A. Deformation Models

We shall use landmarks as the environment representation and define deformations accordingly. This representation has a long history in shape analysis [28, 29]. There has been a rich class of smooth deformation models (also termed smooth warps) developed based on landmark representations, *e.g.,* the Free-Form Deformations (FFD) [45, 51], the Radial Basis Functions (RBF) [10, 17] and the Thin-Plate Spline (TPS) [15, 10]. Beyond smooth models, there exist a class of models defined piece-wisely by implementing local transformations associated to a set of down-sampled points and modeling the deformations on other parts by interpolation. Representatives of such models include the ARAP deformation model [49], the embedded deformation graph [1, 50], and Lie-bodies [19].

Beyond landmark based models, other models based on curves [27, 54] or surfaces have been proposed. Some well-known models include level sets [39], medial surfaces [11], Q-maps [32, 33], and Square Root Normal Fields (SRNF) [26, 35]. Some models implement an articulated skeleton structure. Representative works include the medial axis representations (M-rep) [16], and SCAPE [2]. We refer interested readers to the review papers [53, 34] for more details.

B. Global Registration Techniques

1) Generalized Procrustes analysis: The generalized Procrustes analysis (GPA) framework was used as a fundamental technique in shape analysis to obtain an initial alignment. Both the rigid and affine transformations were recovered in the classical literature [28, 22, 43]. Recently, a novel GPA technique with deformation models was proposed in [4]. The deformation model in [4] is termed LBWs, which includes the affine transformation and a rich class of nonlinear deformation models [45, 51, 10, 17, 7] using radial-basis functions, *e.g,* the well-known TPS [10].

The work [4] is the closest to ours. However, we use a kernel method to model deformations, which is a novel deformation model compared to the LBWs used in [4]. In addition, our estimation method on the global scale does not require the global co-visibility of correspondences, thus is more suitable for SLAM applications. The work in [4] shows that GPA with

LBWs can be solved globally by an eigenvalue decomposition. We will show that the proposed KernelGPA can be solved in a similar manner, while being more powerful in modeling deformations.

2) Nonrigid structure-from-motion: Structure-from-motion (SfM) is a well-known global registration method that handles camera projections [23]. We do not consider projective cameras in this work, thus will only mention several nonrigid SfM (NRSfM) methods for references. One line of NRSfM methods use low-rank shape bases [12, 52, 14]. These methods model deformations as a linear combination of the basis shapes, which are jointly factorized by the singular value decomposition (SVD). Another line of NRSfM methods use differential geometry, where the deformations are constrained to be isometric or conformal, *e.g.,* the isometric NRSfM [40] which has been successfully implemented in DefSLAM [36]. We refer interested readers to a recent work using Cartan's connections [41] and references therein.

III. FORMULATION OF DEFORMABLE SLAM AND ITS CONNECTION TO GPA

A. Deformable SLAM

Environment Modeling. We define the surrounding environment of the sensor, *i.e.,* the map in SLAM, by a collection of 3D points $\mathbf{M} \in \mathbb{R}^{3 \times m}$. The onboard sensor observes the environment $\mathbf{M}$ at discrete time points $t = 1, 2 \ldots, n$. We denote the sensor's pose at time t by $(\mathbf{R}_t \in \mathrm{SO}(3), \mathbf{t}_t \in \mathbb{R}^3)$ and its reading at time t by $\mathbf{P}_t \in \mathbb{R}^{3 \times m_t}$ given as the relative-measurement of a portion of environment points $\mathbf{M}_t \in \mathbb{R}^{3 \times m_t}$. Concretely, we can write $\mathbf{M}_t = \mathbf{M}\mathbf{\Gamma}_t$ with the help of a *visibility matrix* $\mathbf{\Gamma}_t$ whose columns are constructed from the standard basis vectors in $\mathbb{R}^m$:

$$\mathbf{\Gamma}_t = [\mathbf{e}_{k_t}] \in \mathbb{R}^{m \times m_t}.$$

The basis vector $\mathbf{e}_{k_t}$ in $\mathbf{\Gamma}_t$ means that the k-th point in $\mathbf{M}$ occurs in $\mathbf{M}_t$. It can be easily verified that $\mathbf{\Gamma}_t^\top \mathbf{1}_m = \mathbf{1}_{m_t}$.

Deformable Transformation. In deformable SLAM, the environment deforms over time. We model the environment deformation as a time varying function $\mathbf{\Phi}_t(\cdot)$. In particular the deformed environment $\tilde{\mathbf{M}}_t$ at time t is:

$$\tilde{\mathbf{M}}_t = \mathbf{\Phi}_t(\mathbf{M}_t) = \mathbf{\Phi}_t(\mathbf{M}\mathbf{\Gamma}_t).$$

In the noise-free case, the sensor reading $\mathbf{P}_t$ at t corresponds to the measurement of the deformed 3D environment $\tilde{\mathbf{M}}_t$:

$$\mathbf{P}_t = \mathbf{R}_t^\top(\tilde{\mathbf{M}}_t - \mathbf{t}_t\mathbf{1}^\top) \iff \mathbf{R}_t\mathbf{P}_t + \mathbf{t}_t\mathbf{1}^\top = \tilde{\mathbf{M}}_t.$$

We fairly assume the deformation function $\mathbf{\Phi}_t(\cdot)$ is invertible, and thus define an inverse mapping $\mathbf{\Phi}_t^{-1}(\cdot)$ such that:

$$\mathbf{M}\mathbf{\Gamma}_t = \mathbf{\Phi}_t^{-1}(\tilde{\mathbf{M}}_t) = \mathbf{\Phi}_t^{-1}(\mathbf{R}_t\mathbf{P}_t + \mathbf{t}_t\mathbf{1}^\top) = \mathbf{y}_t(\mathbf{P}_t).$$

Here we use $\mathbf{y}_t(\cdot)$ as a composition of both the rigid transformation $(\mathbf{R}_t, \mathbf{t}_t)$ and the deformation $\mathbf{\Phi}_t^{-1}(\cdot)$.

Deformable SLAM. We formally define deformable SLAM as the problem that estimates 1) the nonrigid transformations $\mathbf{y}_t(\cdot)$, and 2) the surrounding environment map $\mathbf{M}$ using a
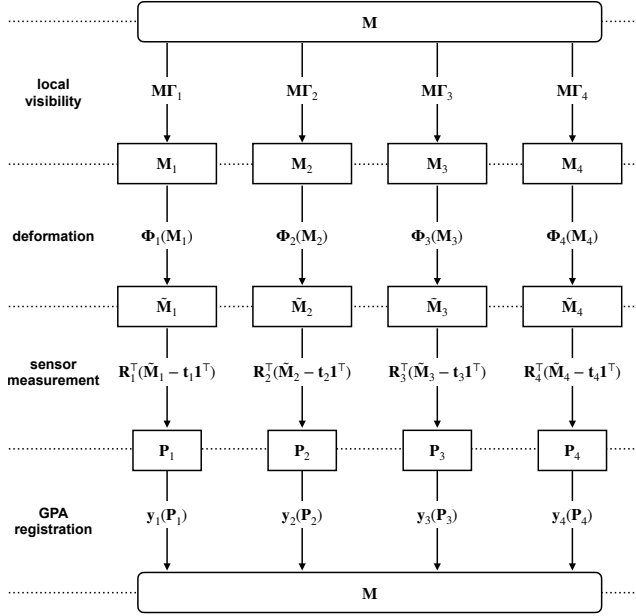
Fig. 1: Deformable SLAM as the generalized Procrustes analysis (GPA) problem with deformable transformations.

collection of sensor readings $(\mathbf{P}_t, \mathbf{\Gamma}_t)$ at all time points $t = 1, 2 \ldots, n$. Formally, we formulate deformable SLAM as:

$$\min \sum_{t=1}^{n} \varphi_t \quad \text{with } \varphi_t = \left\| \mathbf{y}_t(\mathbf{P}_t) - \mathbf{M}\mathbf{\Gamma}_t \right\|_{\mathcal{F}}^2. \quad (1)$$

The rigid poses $(\mathbf{R}_t, \mathbf{t}_t)$ are composed in $\mathbf{y}_t(\cdot)$ in this formulation and will be decomposed afterwards.

B. Connection to Generalized Procrustes Analysis

The deformable SLAM formulation (1) is essentially the problem of *generalized Procrustes analysis* (GPA) with deformable transformation models, see Fig. 1. If we model the deformable transformations $\mathbf{y}_t(\cdot)$ as the LBWs, this problem can be solved by the deformable GPA method in [4], which states that the principal axes of $\mathbf{M}$ can be globally solved in closed-form based on the eigenvalue decomposition.

In this work, we propose to use a kernel method to model the transformation $\mathbf{y}_t(\cdot)$, and show that the resulting problem can be recast as an eigenvalue problem in (20) which can be solved by a result in [4]. We build upon this result to estimate both $\mathbf{M}$ and $\mathbf{y}_t(\cdot)$ up to an unknown global scale $\mathbf{\Lambda}$. Then we propose to solve $\mathbf{\Lambda}$ and rigid poses $(\mathbf{R}_t, \mathbf{t}_t)$ by the cost (25).

IV. Transformation Model

A. Nonlinear Transformation Model

We start our discussion by modeling transformation $\mathbf{y}_t(\mathbf{p})$ as the combination of an affine transformation $[\mathbf{A}_t, \mathbf{a}_t]$ and a nonlinear deformation, which is termed LBWs in [4]:

$$\mathbf{y}_t(\mathbf{p}) = \mathbf{A}_t\mathbf{p} + \mathbf{a}_t + \mathbf{W}_t^\top \boldsymbol{\theta}_t(\mathbf{p}), \quad (2)$$

where $\mathbf{W}_t \in \mathbb{R}^{l \times 3}$ is a parameter matrix to be estimated and $\boldsymbol{\theta}_t(\cdot) : \mathbb{R}^3 \rightarrowtail \mathbb{R}^l$ is a mapping to the l-dimensional feature

space. The components of $\boldsymbol{\theta}_t(\cdot)$ are typically chosen as radial basis functions [17], *e.g.,* the TPS [45, 10].

Applying transformation model (2) to the point-cloud $\mathbf{P}_t$ (whose columns represent points), we have:

$$\mathbf{y}_t(\mathbf{P}_t) = \mathbf{A}_t\mathbf{P}_t + \mathbf{a}_t\mathbf{1}^\top + \mathbf{W}_t^\top \boldsymbol{\theta}_t(\mathbf{P}_t), \quad (3)$$

where we have abused the notation to write:

$$\boldsymbol{\theta}_t(\mathbf{P}_t) = [\boldsymbol{\theta}_t(\mathbf{P}_t[1]), \boldsymbol{\theta}_t(\mathbf{P}_t[2]), \ldots \boldsymbol{\theta}_t(\mathbf{P}_t[m_t])],$$

and we denote $\mathbf{P}_t[k]$ the k-th point in $\mathbf{P}_t$.

B. Dual Formulation

The nonlinear deformation component $\mathbf{W}_t^\top \boldsymbol{\theta}_t(\mathbf{P}_t)$ is a linear regression model which admits a dual formulation [9]. To show the point, we examine the following regularized regression problem with cost:

$$\eta_t(\mathbf{W}_t) = \left\| \mathbf{W}_t^\top \boldsymbol{\theta}_t(\mathbf{P}_t) - \mathbf{Z}_t \right\|_{\mathcal{F}}^2 + \mu_t \left\| \mathbf{W}_t \right\|_{\mathcal{F}}^2, \quad (4)$$

where $\mu_t \left\| \mathbf{W}_t \right\|_{\mathcal{F}}^2$ is a regularization term to avoid overfitting, and $\mathbf{Z}_t$ the regression output which can be considered as $\mathbf{Z}_t = \mathbf{M}\mathbf{\Gamma}_t - \mathbf{A}_t\mathbf{P}_t + \mathbf{a}_t\mathbf{1}^\top$ in the context of this paper. The first order optimality condition of the cost (4) satisfies:

$$\mathbf{W}_t = \boldsymbol{\theta}_t(\mathbf{P}_t) \underbrace{\left(-\frac{1}{\mu_t} \mathbf{W}_t^\top \boldsymbol{\theta}_t(\mathbf{P}_t) + \frac{1}{\mu_t} \mathbf{Z}_t \right)^\top}_{\mathbf{\Omega}_t} = \boldsymbol{\theta}_t(\mathbf{P}_t)\mathbf{\Omega}_t,$$

$$(5)$$

where we introduce the *dual variable* $\mathbf{\Omega}_t$. By introducing a *kernel matrix* $\mathbf{K}_t = \boldsymbol{\theta}_t(\mathbf{P}_t)^\top \boldsymbol{\theta}_t(\mathbf{P}_t)$, we rewrite the nonlinear deformation component $\mathbf{W}_t^\top \boldsymbol{\theta}_t(\mathbf{P}_t)$, *i.e.,* the transpose of (5) post-multiplied by $\boldsymbol{\theta}_t(\mathbf{P}_t)$, into a *dual formulation*:

$$\mathbf{W}_t^\top \boldsymbol{\theta}_t(\mathbf{P}_t) = \mathbf{\Omega}_t^\top \underbrace{\boldsymbol{\theta}_t(\mathbf{P}_t)^\top \boldsymbol{\theta}_t(\mathbf{P}_t)}_{\mathbf{K}_t} = \mathbf{\Omega}_t^\top \mathbf{K}_t. \quad (6)$$

The elements of $\mathbf{K}_t = [k_{i,j}]$ are vector inner-products in the features space $\boldsymbol{\theta}_t(\cdot)$, as $k_{i,j} = \boldsymbol{\theta}_t(\mathbf{P}_t)[i]^\top \boldsymbol{\theta}_t(\mathbf{P}_t)[j]$. The spirit of the kernel method is to design the inner products in the feature space directly from the data space as $k(\mathbf{P}_t[i], \mathbf{P}_t[j])$ without explicitly assigning the basis functions in $\boldsymbol{\theta}_t(\cdot)$.

The kernel design allows more flexibility in terms of modeling power than the LBWs in [4]. From equation (6), we can find a kernel representation for each LBWs. In contrast, not every kernel model can be described by LBWs *e.g.,* a Gaussian kernel induces infinite dimension in the feature space.

C. The Proposed Transformation Model

By using the dual formulation in (6), we rewrite the transformation model (3) as:

$$\mathbf{y}_t(\mathbf{P}_t) = \mathbf{A}_t\mathbf{P}_t + \mathbf{a}_t\mathbf{1}^\top + \mathbf{\Omega}_t^\top \mathbf{K}_t, \quad (7)$$

with $[\mathbf{A}_t, \mathbf{t}_t]$ and $\mathbf{\Omega}_t$ being the transformation parameters to be estimated. The regularization term $\mu_t \left\| \mathbf{W}_t \right\|_{\mathcal{F}}^2$ can be rewritten as a function of the dual variable $\mathbf{\Omega}_t$ as:

$$\mu_t \left\| \mathbf{W}_t \right\|_{\mathcal{F}}^2 = \mu_t \left\| \boldsymbol{\theta}_t(\mathbf{P}_t)\mathbf{\Omega}_t \right\|_{\mathcal{F}}^2 = \mu_t \text{tr}\left(\mathbf{\Omega}_t^\top \mathbf{K}_t \mathbf{\Omega}_t \right). \quad (8)$$

Lastly, the transformation model $\mathbf{y}_t(\mathbf{p})$ in (2) acting on an arbitrary point $\mathbf{p}$ becomes:

$$\mathbf{y}_t(\mathbf{p}) = \mathbf{A}_t\mathbf{p} + \mathbf{a}_t + \mathbf{\Omega}_t^\top\mathbf{k}_t(\mathbf{p}), \qquad (9)$$

where $\mathbf{k}_t(\mathbf{p}) = \boldsymbol{\theta}_t(\mathbf{P}_t)^\top\boldsymbol{\theta}_t(\mathbf{p})$ denotes a vector whose i-th element is given by $k(\mathbf{P}_t[i], \mathbf{p})$ using the kernel function.

V. GLOBALLY OPTIMAL SOLUTION TO PRINCIPAL AXES OF THE ENVIRONMENT MAP

A. Constraints to Resolve Environment Ambiguities

It is well-known that SLAM in the rigid case is defined up to a global transformation, with 6 DOFs ambiguities. We first remove these 6 DOFs by the following 6 constraints as in [4]:

$$\begin{cases} \mathbf{M}\mathbf{1} = \mathbf{0} & (10) \\ \mathbf{M}\mathbf{M}^\top = \mathbf{\Lambda} = \mathrm{diag}(\lambda_1^2, \lambda_2^2, \lambda_3^2), & (11) \end{cases}$$

where $\lambda_1, \lambda_2, \lambda_3$ are unknown parameters.

In the classical rigid SLAM literature [13, 44, 5], the transformation ambiguity (*i.e.*, the 6 gauge freedoms caused by a global rigid transformation) is removed by fixing one of the robot poses, *e.g.*, the first robot pose to the identity of SE(3). However, the way of removing 6 gauge freedoms as above is not yet realized in the robotics community. We briefly recapitulate the ideas in [4] in the appendix for completeness.

Importantly, the estimate of both the transformation model and the map can be established up to an unknown $\mathbf{\Lambda}$ (see [4] for affine models and TPS warps). We shall derive similar results using kernel models. In other words, we can determine the diagonal elements of $\mathbf{\Lambda}$ afterwards with additional properties of underlying deformations, *e.g.*, by assuming the deformation is as-rigid-as-possible.

B. Formulation of Deformable SLAM

Using transformation model (7) and regularization (8), we write the cost function at time t as:

$$\varphi_t(\mathbf{A}_t, \mathbf{a}_t, \mathbf{\Omega}_t, \mathbf{M}) = \left\|\mathbf{A}_t\mathbf{P}_t + \mathbf{a}_t\mathbf{1}^\top + \mathbf{\Omega}_t^\top\mathbf{K}_t - \mathbf{M}\mathbf{\Gamma}_t\right\|_{\mathcal{F}}^2 \\ + \mu_t\mathrm{tr}\left(\mathbf{\Omega}_t^\top\mathbf{K}_t\mathbf{\Omega}_t\right). \tag{12}$$

Then we use constraints $\mathbf{M}\mathbf{1} = \mathbf{0}$ and $\mathbf{M}\mathbf{M}^\top = \mathbf{\Lambda}$ to fix 6 gauge freedoms of the problem, and propose the following formulation for deformable SLAM:

$$\min \sum_{t=1}^n \varphi_t(\mathbf{A}_t, \mathbf{a}_t, \mathbf{\Omega}_t, \mathbf{M}) \quad \text{s.t. } \mathbf{M}\mathbf{1} = \mathbf{0}, \ \mathbf{M}\mathbf{M}^\top = \mathbf{\Lambda}. \tag{13}$$

In formulation (13), we only use relative observations to the environment in the local coordinate frame of each robot pose. This is the minimal set of information required to establish the estimate for both the transformation model and the map.

For many SLAM applications, *e.g.*, outdoor mapping, additional sensors, *e.g.*, Inertial Measurement Units (IMU) are used to assist pose estimation. As we shall see shortly, it is straightforward to extend the solution of formulation (13) to standard sensor fusion frameworks.

In the remainder of this section, we derive the globally optimal solution to formulation (13) in closed-form.

C. Eliminating Transformation Parameters

We notice that in problem (13), the transformation parameters $\mathbf{A}_t$, $\mathbf{a}_t$ and $\mathbf{\Omega}_t$ are linearly dependent on $\mathbf{M}$. This presents a separable structure and allows us to reduce the optimization to $\mathbf{M}$ only using the variable projection method [21]. We derive the linear dependence of $\mathbf{A}_t$, $\mathbf{a}_t$ and $\mathbf{\Omega}_t$ on $\mathbf{M}$ as follows.

We denote $\tilde{\mathbf{P}}_t = [\mathbf{P}_t^\top, \mathbf{1}]^\top$. Given $\mathbf{M}$ and $\mathbf{\Omega}_t$, the affine part $[\mathbf{A}_t, \mathbf{a}_t]$ for time t is:

$$\begin{bmatrix} \mathbf{A}_t, & \mathbf{a}_t \end{bmatrix} = -(\mathbf{\Omega}_t^\top\mathbf{K}_t - \mathbf{M}\mathbf{\Gamma}_t)\tilde{\mathbf{P}}_t^\dagger \tag{14}$$

where $\tilde{\mathbf{P}}_t^\dagger = \tilde{\mathbf{P}}_t^\top(\tilde{\mathbf{P}}_t\tilde{\mathbf{P}}_t^\top)^{-1}$ is the Moore–Penrose pseudo-inverse. Here we assume that the 3D point-cloud $\tilde{\mathbf{P}}_t$ is non-degenerate, thus the inverse of $\tilde{\mathbf{P}}_t\tilde{\mathbf{P}}_t^\top$ exists. If this is not the case, we can model the problem in the 2D space and use λ_1, λ_2 only in (11). It is also recommended to use $(\tilde{\mathbf{P}}_t\tilde{\mathbf{P}}_t^\top)^\dagger$ to replace $(\tilde{\mathbf{P}}_t\tilde{\mathbf{P}}_t^\top)^{-1}$ as a numerical workaround. Back substituting (14) to $\varphi_t(\mathbf{A}_t, \mathbf{a}_t, \mathbf{\Omega}_t, \mathbf{M})$ and denoting $\boldsymbol{\mathcal{P}}_t = \tilde{\mathbf{P}}_t^\top(\tilde{\mathbf{P}}_t\tilde{\mathbf{P}}_t^\top)^{-1}\tilde{\mathbf{P}}_t$, we obtain for time t:

$$\varphi_t(\mathbf{\Omega}_t, \mathbf{M}) = \left\|(\mathbf{\Omega}_t^\top\mathbf{K}_t - \mathbf{M}\mathbf{\Gamma}_t)(\mathbf{I} - \boldsymbol{\mathcal{P}}_t)\right\|_{\mathcal{F}}^2 \\ + \mu_t\mathrm{tr}\left(\mathbf{\Omega}_t^\top\mathbf{K}_t\mathbf{\Omega}_t\right). \tag{15}$$

It remains to derive the dependence of $\mathbf{\Omega}_t$ on $\mathbf{M}$. Given $\mathbf{M}$, the optimal $\mathbf{\Omega}_t$ from $\varphi_t(\mathbf{\Omega}_t, \mathbf{M})$ satisfies the optimality equation:

$$\mathbf{K}_t(\mathbf{I} - \boldsymbol{\mathcal{P}}_t)(\mathbf{K}_t\mathbf{\Omega}_t - \mathbf{\Gamma}_t^\top\mathbf{M}^\top) + \mu_t\mathbf{K}_t\mathbf{\Omega}_t = \mathbf{O}, \tag{16}$$

where we have used the fact that $\mathbf{I} - \boldsymbol{\mathcal{P}}_t$ is idempotent. By assuming that the kernel matrix $\mathbf{K}_t$ is invertible, we write the estimate of $\mathbf{\Omega}_t$ w.r.t. the estimate of $\mathbf{M}$ as:

$$\mathbf{\Omega}_t = \mathbf{H}_t^\top\mathbf{\Gamma}_t^\top\mathbf{M}^\top, \tag{17}$$

where we define the data dependent matrix constant:

$$\mathbf{H}_t = (\mathbf{I} - \boldsymbol{\mathcal{P}}_t)\left(\mathbf{K}_t(\mathbf{I} - \boldsymbol{\mathcal{P}}_t) + \mu_t\mathbf{I}\right)^{-1}. \tag{18}$$

Substituting (17) into (14), we obtain the estimate of $[\mathbf{A}_t, \mathbf{a}_t]$ w.r.t. the estimate of $\mathbf{M}$ as:

$$\begin{bmatrix} \mathbf{A}_t, & \mathbf{a}_t \end{bmatrix} = -\mathbf{M}\mathbf{\Gamma}_t(\mathbf{H}_t\mathbf{K}_t - \mathbf{I})\tilde{\mathbf{P}}_t^\dagger. \tag{19}$$

To fully determine the affine parameters $[\mathbf{A}_t, \mathbf{a}_t]$ via (19) and the deformable parameters $\mathbf{\Omega}_t$ via (17), it remains to determine the estimate of the environment map $\mathbf{M}$.

D. Optimal Estimate of the Environment Map

Substituting (17) into (15) (*i.e.*, the cost $\varphi_t(\mathbf{\Omega}_t, \mathbf{M})$), we obtain the final cost function in $\mathbf{M}$:

$$\varphi_t(\mathbf{M}) = \mathrm{tr}\left(\mathbf{M}\mathbf{\Gamma}_t\boldsymbol{\mathcal{Q}}_t\mathbf{\Gamma}_t^\top\mathbf{M}^\top\right),$$

where

$$\boldsymbol{\mathcal{Q}}_t = (\mathbf{H}_t\mathbf{K}_t - \mathbf{I})(\mathbf{I} - \boldsymbol{\mathcal{P}}_t)(\mathbf{K}_t^\top\mathbf{H}_t^\top - \mathbf{I}) + \mu_t\mathbf{H}_t\mathbf{K}_t\mathbf{H}_t^\top.$$

Lastly, we reduce the cost $\sum_{t=1}^n \varphi_t(\mathbf{A}_t, \mathbf{a}_t, \mathbf{\Omega}_t, \mathbf{M})$ in problem (13) to $\sum_{t=1}^n \varphi_t(\mathbf{M}) = \mathrm{tr}\left(\mathbf{M}\left(\sum_{t=1}^n \mathbf{\Gamma}_t\boldsymbol{\mathcal{Q}}_t\mathbf{\Gamma}_t^\top\right)\mathbf{M}^\top\right)$ and reach the following optimization problem in $\mathbf{M}$ only:

$$\min_{\mathbf{M}} \ \mathrm{tr}\left(\mathbf{M}\boldsymbol{\mathcal{Q}}\mathbf{M}^\top\right) \quad \text{s.t. } \mathbf{M}\mathbf{1} = \mathbf{0}, \ \mathbf{M}\mathbf{M}^\top = \mathbf{\Lambda}, \tag{20}$$

where we define $\mathcal{Q} = \sum_{t=1}^{n} \mathbf{\Gamma}_t \mathcal{Q}_t \mathbf{\Gamma}_t^\top$.

Formulation (20) can be solved globally in closed-form if the all-one vector $\mathbf{1}$ is an eigenvector of $\mathcal{Q}$ [4].

Proposition 1. In problem (20), $\mathcal{Q}\mathbf{1} = \mathbf{0}$ which means $\mathbf{1}$ is an eigenvector of $\mathcal{Q}$ corresponding to eigenvalue 0.

Proof: It can be verified that $\mathbf{\Gamma}_t^\top \mathbf{1}_m = \mathbf{1}_{m_t}$ as $\mathbf{e}_k^\top \mathbf{1}_m = 1$. The matrix $\mathcal{P}_t$ is the orthogonal projector to the range space of $\tilde{\mathbf{P}}_t^\top$. Given that $\mathbf{1}_{m_t}$ is a column of $\tilde{\mathbf{P}}_t^\top$ thus in the range of $\tilde{\mathbf{P}}_t^\top$, we have $\mathcal{P}_t \mathbf{1}_{m_t} = \mathbf{1}_{m_t}$ which means $(\mathbf{I} - \mathcal{P}_t)\mathbf{1}_{m_t} = \mathbf{0}$. From above, we conclude $(\mathbf{I} - \mathcal{P}_t)\mathbf{\Gamma}_t^\top \mathbf{1}_m = \mathbf{0}$ and $\mathbf{H}_t^\top \mathbf{\Gamma}_t^\top \mathbf{1}_m = \mathbf{0}$. Therefore $\mathcal{Q}_t \mathbf{1} = \mathbf{0}$ thus $\mathcal{Q}\mathbf{1} = \mathbf{0}$. ∎

We recapitulate necessary results to describe the solution to problem (20), and refer readers to [4] for detailed proofs.

Lemma 1. (Theorem 1 in [4].) If $\mathbf{u}$ is an eigenvector of $\mathbf{\Pi}$, then the solution to the optimization problem

$$\max_{\mathbf{X}} \ \mathrm{tr}\left(\mathbf{X}^\top \mathbf{\Pi}\mathbf{X}\mathbf{\Lambda}\right) \quad \text{s.t.} \quad \mathbf{X}^\top \mathbf{X} = \mathbf{I}, \ \mathbf{X}^\top \mathbf{u} = \mathbf{0}, \quad (21)$$

is the d top eigenvectors of $\mathbf{\Pi}$ excluding the vector $\mathbf{u}$.

By letting $\mathbf{X} = \mathbf{M}^\top \sqrt{\mathbf{\Lambda}}^{-1}$ and $\mathbf{u} = \mathbf{1}$, we have the following Lemma 2 which is in the same spirit of Proposition 1 in [4]. We have rewritten the maximization problem as a minimization problem to accommodate the form in (20).

Lemma 2. (Proposition 1 in [4].) If $\mathbf{1}$ is an eigenvector of $\mathbf{\Pi}$, then the globally optimal solution to problem

$$\min_{\mathbf{M}} \ -\mathrm{tr}\left(\mathbf{M}\mathbf{\Pi}\mathbf{M}^\top\right) \quad \text{s.t.} \quad \mathbf{M}\mathbf{M}^\top = \mathbf{\Lambda}, \ \mathbf{M}\mathbf{1} = \mathbf{0}, \quad (22)$$

is $\mathbf{M} = \sqrt{\mathbf{\Lambda}}\mathbf{X}^\top$ where $\mathbf{X}$ comprises the d top eigenvectors of $\mathbf{\Pi}$ excluding the vector $\mathbf{1}$.

We consider $\mathbf{\Pi} = -\mathcal{Q}$ in (22) which is problem (20). From Lemma 2, we obtain the following result immediately.

Proposition 2. The solution to problem (20) is $\mathbf{M} = \sqrt{\mathbf{\Lambda}}\mathbf{X}^\top$ where $\mathbf{X}$ comprises the d top eigenvectors of $-\mathcal{Q}$ excluding the vector $\mathbf{1}$. As $\mathcal{Q}$ is positive definite thus $-\mathcal{Q}$ is negative definite, the d top eigenvectors of $-\mathcal{Q}$ excluding the vector $\mathbf{1}$ are the d bottom eigenvectors of $\mathcal{Q}$ excluding the vector $\mathbf{1}$.

E. Optimal Estimate of the Transformation Models

Upon obtaining the map estimate $\mathbf{M} = \sqrt{\mathbf{\Lambda}}\mathbf{X}^\top$, we decide the transformation parameters $[\mathbf{A}_t, \mathbf{a}_t]$ by (19) and $\mathbf{\Omega}_t$ by (17). The optimal transformation in (9) can be written as:

$$\mathbf{y}_t(\mathbf{p}) = -\mathbf{M}\mathbf{\Gamma}_t(\mathbf{H}_t\mathbf{K}_t - \mathbf{I})\tilde{\mathbf{P}}_t^\dagger \tilde{\mathbf{p}} + \mathbf{M}\mathbf{\Gamma}_t\mathbf{H}_t\mathbf{k}_t(\mathbf{p}),$$

where $\tilde{\mathbf{p}} = [\mathbf{p}^\top, 1]^\top$.

Similarly the optimal transformation $\mathbf{y}_t(\mathbf{P}_t)$ in (7) for the point-cloud $\mathbf{P}_t$ is:

$$\mathbf{y}_t(\mathbf{P}_t) = -\mathbf{M}\mathbf{\Gamma}_t(\mathbf{H}_t\mathbf{K}_t - \mathbf{I})\mathcal{P}_t + \mathbf{M}\mathbf{\Gamma}_t\mathbf{H}_t\mathbf{K}_t = \sqrt{\mathbf{\Lambda}}\mathbf{S}_t, \quad (23)$$

where we have used $\mathbf{M} = \sqrt{\mathbf{\Lambda}}\mathbf{X}^\top$ and have defined

$$\mathbf{S}_t = -\mathbf{X}^\top\mathbf{\Gamma}_t(\mathbf{H}_t\mathbf{K}_t - \mathbf{I})\mathcal{P}_t + \mathbf{X}^\top\mathbf{\Gamma}_t\mathbf{H}_t\mathbf{K}_t. \quad (24)$$

It is worth noting that $\mathbf{S}_t$ is estimated without knowing $\mathbf{\Lambda}$.

Therefore, both the transformation and map estimates are decided up to an unknown $\mathbf{\Lambda}$ which can be estimated afterwards as a posterior by using additional information on the allowed deformations.

VI. ESTIMATION OF RIGID TRANSFORMATIONS AND THE GLOBAL SCALE AMBIGUITY

A. Jointly Solving for Rigid Transformations and $\mathbf{\Lambda}$

We want the solution to be as-rigid-as-possible, that means we want $\mathbf{\Phi}_t(\cdot)$ to be close to an identity mapping. This results in the following constraint:

$$\mathbf{R}_t\mathbf{P}_t + \mathbf{t}_t\mathbf{1}^\top \approx \mathbf{y}_t(\mathbf{P}_t) = \sqrt{\mathbf{\Lambda}}\mathbf{S}_t,$$

where $\mathbf{S}_t$ has been calculated as equation (24).

We thus propose to jointly estimate the rigid transformation $[\mathbf{R}_t, \mathbf{t}_t]$ and the unknown $\mathbf{\Lambda}$ together by solving the following optimization problem:

$$\min_{\mathbf{R}_t, \, \mathbf{t}_t, \, \mathbf{\Lambda}} \ \sum_{t=1}^{n} \left\| \mathbf{R}_t\mathbf{P}_t + \mathbf{t}_t\mathbf{1}^\top - \sqrt{\mathbf{\Lambda}}\mathbf{S}_t \right\|_{\mathcal{F}}^2. \quad (25)$$

In what follows, we show how to solve formulation (25) robustly, and then discuss the extension to sensor fusion.

B. Solving Formulation (25) Robustly

1) Exploiting Separability: Formulation (25) is a nonlinear least squares problem which needs to be solved iteratively. To proceed, we exploit the separability to eliminate the translation $\mathbf{t}_t$ first. Given $\mathbf{R}_t$ and $\mathbf{\Lambda}$, the translation estimate $\mathbf{t}_t$ is:

$$\mathbf{t}_t = -\frac{1}{m_t}\left(\mathbf{R}_t\mathbf{P}_t - \sqrt{\mathbf{\Lambda}}\mathbf{S}_t\right)\mathbf{1}, \quad (26)$$

where $m_t = \mathbf{nnz}(\mathbf{\Gamma}_t)$ is the number of columns in $\mathbf{P}_t$.

We denote the zero-centered matrices:

$$\bar{\mathbf{P}}_t = \mathbf{P}_t - \frac{1}{m_t}\mathbf{P}_t\mathbf{1}\mathbf{1}^\top, \quad \bar{\mathbf{S}}_t = \mathbf{S}_t - \frac{1}{m_t}\mathbf{S}_t\mathbf{1}\mathbf{1}^\top,$$

and rewrite formulation (25) as follow:

$$\min_{\mathbf{R}_t, \, \mathbf{\Lambda}} \ \sum_{t=1}^{n} \left\| \mathbf{R}_t\bar{\mathbf{P}}_t - \sqrt{\mathbf{\Lambda}}\bar{\mathbf{S}}_t \right\|_{\mathcal{F}}^2 \quad (27)$$

We solve formulation (27) iteratively using the standard Lie group optimization technique. A good initialization is required.

2) Initialization: In formulation (27), given $\mathbf{\Lambda}$, the affine estimate of $\mathbf{R}_t$ without considering the SO(3) constraints are:

$$\mathbf{R}_t = \sqrt{\mathbf{\Lambda}}\bar{\mathbf{S}}_t\bar{\mathbf{P}}_t^\dagger = \sqrt{\mathbf{\Lambda}}\mathbf{L}_t, \quad (28)$$

where we define $\mathbf{L}_t = \bar{\mathbf{S}}_t\bar{\mathbf{P}}_t^\dagger$. We enforce the orthonormal constraints $\mathbf{R}_t^\top\mathbf{R}_t = \mathbf{I}$ afterwards by minimizing:

$$\mathbf{\Lambda}_0 = \underset{\mathbf{\Lambda}}{\arg\min} \ \sum_{t=1}^{n} \left\| \mathbf{L}_t^\top\mathbf{\Lambda}\mathbf{L}_t - \mathbf{I} \right\|_{\mathcal{F}}^2.$$

This is a linear least-squares problem in $\mathbf{\Lambda}$ that can give the initialization $\mathbf{\Lambda}_0$ in closed-form.

Given $\mathbf{\Lambda}_0$, the initial rotation $\mathbf{R}_t$ for each time t can be solved from formulation (27) in closed-form by pairwise special orthogonal Procrustes analysis [3, 24] between $\bar{\mathbf{P}}_t$ and $\sqrt{\mathbf{\Lambda}}\bar{\mathbf{S}}_t$, or we can round $\sqrt{\mathbf{\Lambda}}\mathbf{L}_t$ to a proper rotation $\mathbf{R}_t$.
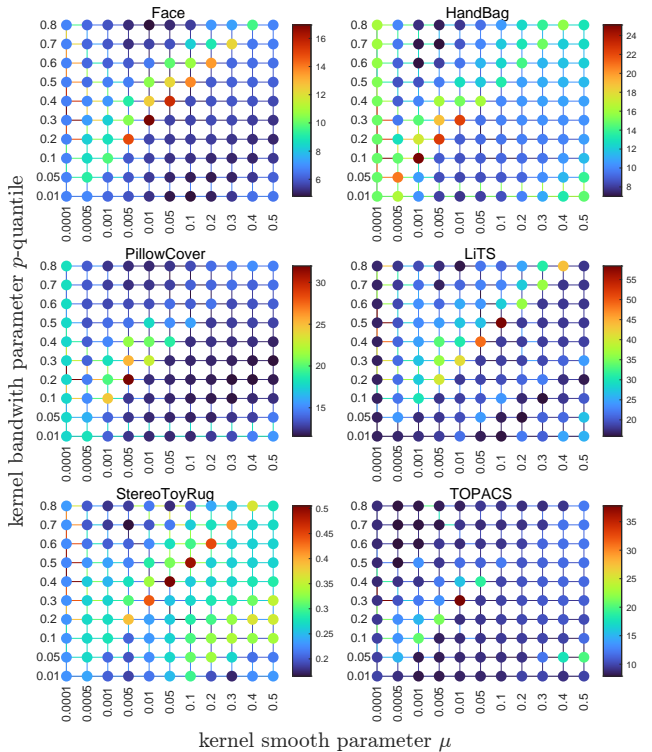
Fig. 2: The tuning parameters of KernelGPA. We use the 20-fold cross-validation error (CVE) as the metric encoded by the colors. We use the Gaussian kernel and decide the bandwidth σ by the p-quantile of the pairwise Euclidean distances. We set $\mu_t = \mu$ for $t = 1, 2 \ldots, n$ and search for μ.

VII. Implementation Details

A. Model Selection

We implement $k(\cdot, \cdot)$ using the Gaussian kernel:

$$k_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right), \qquad (29)$$

and set the bandwidth parameter σ to the p-quantile of the Euclidean distances between training points:

$$\sigma = p\text{-quantile}\left(\|\mathbf{x}_i - \mathbf{x}_j\|\right), \quad \text{for all } (i \neq j). \qquad (30)$$

We use the same regularization strength $\mu_t = \mu$ for all $t = 1, 2 \ldots, n$. We suggest searching for tuning parameters at $p = 0.2$ and $\mu = 0.05$. Some examples are provided in Fig. 2, using the G-fold cross-validation defined in what follows.

B. Cross-Validation

We use cross-validations to perform model selection and evaluation. We define the Cross-Validation Error (CVE) as:

$$\text{CVE} = \sqrt{\frac{1}{\kappa} \sum_{t=1}^{n} \|\mathcal{Y}_t(\mathbf{P}_t) - \mathbf{M}^\star \mathbf{\Gamma}_t\|_F^2},$$

where $\mathbf{M}^\star$ is the estimate of the environment, and $\kappa = \sum_{t=1}^{n} \mathbf{nnz}(\mathbf{\Gamma}_t) = \sum_{t=1}^{n} m_t$ is a normalization constant by

summing up the number of nonzeros in $\mathbf{\Gamma}_t$. The point-cloud $\mathcal{Y}_t(\mathbf{P}_t)$ is computed by the G-fold cross-validation as follows.

We group all the points in the environment to G mutually exclusive subsets $g_1, g_2, \ldots, g_G$. We index the points in g_k ($k = [1 : G]$) occurring in the point-cloud $\mathbf{P}_t$ as $\mathbf{P}_t[g_k]$. Now for each subset g_k we solve the GPA with all the points except those in g_k. This solution is denoted as $^{g_k}\mathbf{M}^\star$ for the optimal map estimate and $^{g_k}\mathbf{y}_t^\star()$ ($t \in [1 : n]$) for the t-th optimal transformation. We predict the transformed positions of the points in $\mathbf{P}_t[g_k]$ as $^{g_k}\mathbf{y}_t^\star(\mathbf{P}_t[g_k])$.

Then we correct the gauge of $^{g_k}\mathbf{y}_t^\star(\mathbf{P}_t[g_k])$ to the reference frame of $\mathbf{M}^\star$ using the similarity Procrustes between $^{g_k}\mathbf{M}^\star$ and $\mathbf{M}^\star$. Denote the corrected points as $^{g_k}_C\mathbf{y}_t^\star(\mathbf{P}_t[g_k])$:

$$^{g_k}_C\mathbf{y}_t^\star(\mathbf{P}_t[g_k]) = s_{g_k}\mathbf{R}_{g_k}{}^{g_k}\mathbf{y}_t^\star(\mathbf{P}_t[g_k]) + \boldsymbol{t}_{g_k}\mathbf{1}^\top,$$

where the scale s_{g_k}, rotation $\mathbf{R}_{g_k}$ and translation $\boldsymbol{t}_{g_k}$ are in closed form [4]. Lastly, the point-cloud $\mathcal{Y}_t(\mathbf{P}_t)$ is defined as:

$$\mathcal{Y}_t(\mathbf{P}_t) = [^{g_1}_C\mathbf{y}_t^\star(\mathbf{P}_t[g_1]), {}^{g_2}_C\mathbf{y}_t^\star(\mathbf{P}_t[g_2]), \ldots, {}^{g_G}_C\mathbf{y}_t^\star(\mathbf{P}_t[g_G])].$$

If we set G as the number of points in the environment $\mathbf{M}$, we obtain the *leave-one-out cross-validation* used in [4]. Practically, we use 20-fold cross-validation to choose tuning parameters, and the leave-one-out cross-validation to evaluate the fitness of the registration.

VIII. Experiments

A. Comparison to DefGPA [4]

We first rerun the experiment in [4] (*i.e.,* Table 2 in [4]) and report the new statistics in Table I. We use three evaluation metrics: RMSE_r defined as the root-mean-squared-error (RMSE) of the cost (1), CVE-Leave1 for the leave-one cross-validation and CVE-Gfold for the G-fold cross-validation defined in Section VII-B. We set $G = 20$.

We use GPA with Euclidean (EUC) transformation, affine (AFF) transformation and the TPS(c) warps as benchmark methods. We denote TPS(c) the TPS warp by assigning c control points evenly along each principal axis (see [4] for details), and use the same tuning parameters as in [4]. For KernelGPA, we use hyper-parameters $p = 0.2$, $\mu = 0.05$ for all the datasets. The effect of different kernel tuning parameters is reported in Fig. 2 by the 20-fold cross-validation.

1) Estimation of $\mathbf{\Lambda}$: We estimate $\mathbf{\Lambda}$ by [4] (*i.e.,* Algorithm 1 in [4]) and by the proposed method (*i.e.,* by minimizing cost (25) in Section VI) respectively. We report the statistics for each method in Table I, and find that for most of the cases, there is no significance difference between the proposed $\mathbf{\Lambda}$ estimation method and the one in [4]. As we will show shortly in Section VIII-B and Fig. 3, this is because there are little (in the Face dataset) or no missing correspondences (in others) in these data, thus the two methods are not distinguishable.

2) Performance of KernelGPA: We report the statistics of KernelGPA in the last column of Table I. We see that the proposed KernelGPA outperforms other methods in terms of the CVE-Leave1 in all cases, and the CVE-Gfold for most of cases. This clearly shows that KernelGPA can generate more coherent registrations, as it provides more modeling power.

		EUC	AFF		TPS(3)		TPS(5)		TPS(7)		KernelGPA	
Λ Estimation			by [4]	ours	by [4]	ours	by [4]	ours	by [4]	ours	by [4]	ours
Face	RMSE_r	6.84	6.54	6.64	4.64	4.68	3.88	3.90	3.65	3.66	1.52	1.48
	CVE-Leave1	7.01	6.91	7.01	5.17	5.20	4.52	4.54	4.33	4.34	2.74	**2.68**
	CVE-20fold	7.30	7.79	7.90	6.40	6.44	5.89	5.91	5.74	5.76	5.42	**5.29**
	Time [s]	0.3579	0.0449	0.0794	0.0435	0.0286	0.0194	0.0358	0.0290	0.0404	0.1058	0.1351
HandBag	RMSE_r	30.23	18.90	19.08	9.16	9.19	7.12	7.13	6.75	6.76	3.27	3.24
	CVE-Leave1	30.48	19.26	19.45	9.59	9.62	7.62	7.63	7.30	7.30	3.95	**3.92**
	CVE-20fold	31.84	21.72	21.88	12.36	12.28	10.69	10.58	10.41	10.30	7.12	**7.00**
	Time [s]	0.9535	0.0101	0.0387	0.0189	0.0531	0.0261	0.0556	0.0474	0.0789	0.0660	0.2733
Pillow	RMSE_r	23.44	16.84	16.94	7.82	7.84	5.47	5.49	5.02	5.04	2.96	2.99
	CVE-Leave1	24.01	18.11	18.21	10.09	10.11	8.25	8.30	7.99	8.06	7.79	**7.76**
	CVE-20fold	25.18	20.83	20.80	13.54	13.32	11.89	11.99	**11.72**	11.86	13.02	12.44
	Time [s]	0.2532	0.0058	0.0263	0.0143	0.0272	0.0164	0.0361	0.0270	0.0439	0.0178	0.0624
LiTS	RMSE_r	19.85	15.74	15.93	13.50	13.64	11.57	11.67	10.77	10.86	6.29	5.83
	CVE-Leave1	20.50	17.44	17.62	16.63	16.78	16.09	16.21	16.02	16.13	15.58	**14.37**
	CVE-20fold	20.83	18.01	18.19	17.49	17.64	17.43	17.56	17.45	17.57	18.77	**16.80**
	Time [s]	0.2279	0.0040	0.0391	0.0160	0.0263	0.0806	0.0859	0.4159	0.4278	0.0111	0.1268
ToyRug	RMSE_r	0.88	0.47	0.20	0.29	0.15	0.23	0.12	0.21	0.11	0.06	0.06
	CVE-Leave1	0.98	0.59	0.30	0.43	0.27	0.39	0.25	0.38	0.23	0.28	**0.16**
	CVE-20fold	1.00	0.62	0.33	0.49	0.32	0.44	0.29	0.42	0.27	0.30	**0.17**
	Time [s]	14.2786	0.0615	0.3916	0.2072	0.4876	0.4898	0.7582	1.1369	1.4432	0.1731	0.7257

TABLE I: The statistics of different registration methods by rerunning the experiments in [4]. We compare with the Euclidean GPA (EUC), and DefGPA with the affine transformation (AFF) and three TPS warps with different number of control points along the principal axes *i.e.,* TPS(3), TPS(5), TPS(7) as in [4]. For KernelGPA, we use parameters $p = 0.2$, $\mu = 0.05$ for all the cases. We use Matlab implementations for all the methods.
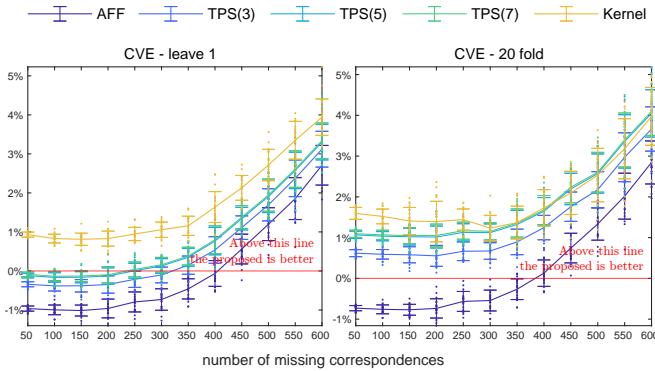

Fig. 3: Performance improvements of the proposed Λ estimation method over the one in [4]. We gradually remove the correspondences in the HandBag dataset [6] (with 155×8 correspondences in total), and compute the relative decrease of the CVE metrics by 20-trial Monte-Carlo runs.

B. Estimation of Λ

We give a detailed evaluation of the proposed Λ estimation method and the one in [4]. In specific, we gradually reduce the number of correspondences of the data, and compute the CVE-Leave1 and CVE-Gfold statistics for each case. We perform this experiment using the HandBag dataset which contains 8 point-clouds with 155 points for each. We use the relative improvement CVE_{Gain} to benchmark the performance:

$$\text{CVE}_{\text{Gain}} = \frac{\text{CVE by [4]} - \text{CVE by proposed}}{\text{CVE by [4]}}.$$

If $\text{CVE}_{\text{Gain}} > 0$ then the proposed Λ estimation method is better than the one in [4], and otherwise if $\text{CVE}_{\text{Gain}} < 0$.

We report the result in Fig. 3 with 20-trial Monte-Carlo runs for each case. It is clear that the proposed Λ estimation method outperforms the one in [4], if the data contains more missing correspondences which is the case for SLAM. Moreover, the worst case scenario shows a comparable accuracy with a merely 1% difference. In addition, the proposed method Λ estimation does not rely on pairwise co-visibility, thus can handle even more missing correspondences where the one in [4] will run into algorithm failures.

C. Pose Estimation

Owing to the lack of deformable datasets with ground-truth poses, we use simulation data to evaluate the quality of the estimated rigid poses obtained from Section VI.

We use a circular pose trajectory and a ground-truth point-cloud on a hemisphere as shown Fig. 4. At each pose, we firstly transform the ground-truth point-cloud with a smooth deformation field generated by the TPS warp and then compute the relative measurement (with Gaussian additive noise 0.01) in the local coordinate frame of the robot pose. We use a single variable σ_{deform} to control the extent of the used deformation.

We use the as-rigid-as-possible approach in Section VI (by solving cost (25)) to estimate both the rigid poses and Λ for all the deformable GPA methods, and report the relative pose error (RPE) [30] with respect to different σ_{deform} in Fig. 4.

We find that for most of the cases, the pose estimates of different methods are very close. This can be explained as follows. In the cost (25), given enough modeling power, the
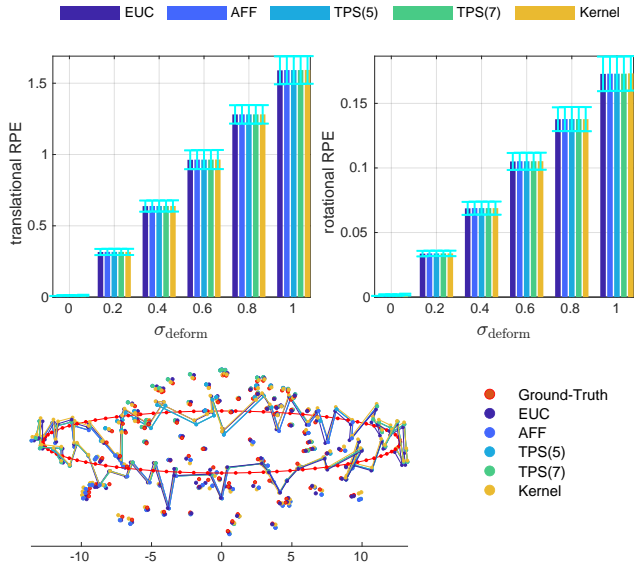
Fig. 4: The RPE of estimated rigid poses w.r.t. σ_{deform} by a 20 run Monte-Carlo simulation for each case. The rigid poses for all deformable GPA methods are solved from cost (25). We visualize an example of the estimated pose trajectory and the reference point-cloud $\mathbf{M}$ for each methods at $\sigma_{\text{deform}} = 1$.

term $\mathbf{y}_t(\mathbf{P}_t) = \sqrt{\boldsymbol{\Lambda}}\mathbf{S}_t$ approximately equals to $\mathbf{M}\boldsymbol{\Gamma}_t$, thus the cost (25) is essentially close to the cost of the rigid GPA.

This result further confirms the point that the proposed $\boldsymbol{\Lambda}$ estimation is close to the one in [4] based on the pairwise rigid Procrustes. However the proposed $\boldsymbol{\Lambda}$ estimation can better handle missing correspondences (in Fig. 3) as the solution is global thus eliminating the bias by using pairwise Procrustes.

D. TOPACS: A Real Computerized Tomography (CT) Dataset

Deformable SLAM has found its application in medical applications, *e.g.,* computerized tomography (CT) registration. CT data lack texture information compared to conventional images, rendering the registration a challenging problem. We now compare different GPA methods using real CT data, which we term the TOPACS dataset.

1) The TOPACS dataset: The CT data we use, given in Fig. 5, contain 6 scans, which are processed by the SURF3D features [42] resulting in 6 point-clouds (with 20000 points for each point-cloud). Initial correspondences are found by matching feature descriptors and then refined by an ICP algorithm. The global correspondences are found by a graph matching algorithm, and the ambiguous ones are removed based on distances. We categorize the correspondences into four sets $\mathcal{C}_3$, $\mathcal{C}_4$, $\mathcal{C}_5$, $\mathcal{C}_6$ by their occurrences across the six point-clouds. For example, $\mathcal{C}_3$ collects the correspondences occurring in exactly three point-clouds, and others are defined analogously.

2) Hyper-parameters of DefGPA and KernelGPA: We use cross-validation to find the suitable tuning-parameters for both the TPS warps and the kernel model. The result for KernelGPA is shown in Fig. 2, and that for DefGPA (by the TPS warp) in
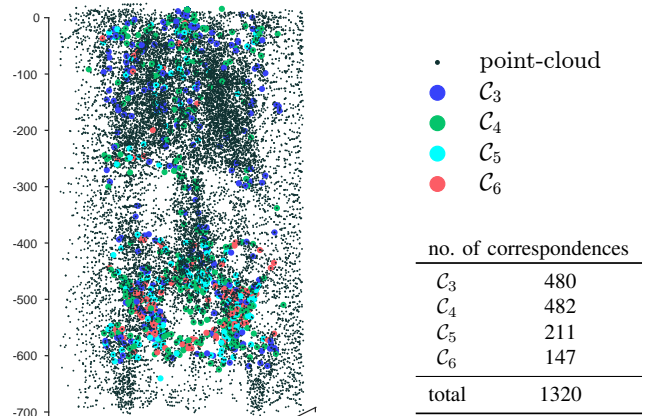

Fig. 5: The TOPACS point-cloud dataset. The point-clouds are processed from real CT scans using SURF3D features. This dataset contains 6 point-clouds, with 20000 points for each point-cloud. There are in total 1320 correspondences classified into four categories according to their occurrences.
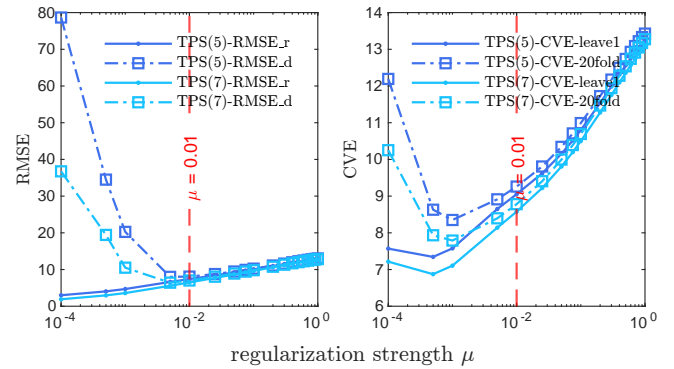

Fig. 6: Parameters of the TPS model for the TOPACS dataset.

Fig. 6. For DefGPA, we use $\mu = 0.01$ for both the TPS(5) and TPS(7) models. For KernelGPA, we use $p = 0.2$, $\mu = 0.05$.

3) Performance: We use one of the correspondence categories as the training set and the rest as the test set. In specific, we use the training set to estimate the GPA transformations, and then use the estimated transformations to align the points in the test set to the same coordinate frame. We compute the point dispersion of the transformed points in the test set, and

training	test	EUC	AFF	TPS(3)	TPS(5)	TPS(7)	Kernel
$\mathcal{C}_3$	$\mathcal{C}_4\mathcal{C}_5\mathcal{C}_6$	18.17	14.66	10.65	9.19	8.70	**8.60**
$\mathcal{C}_4$	$\mathcal{C}_3\mathcal{C}_5\mathcal{C}_6$	18.54	14.48	10.56	9.40	9.02	**7.86**
$\mathcal{C}_5$	$\mathcal{C}_3\mathcal{C}_4\mathcal{C}_6$	19.13	14.88	11.29	10.21	9.92	**9.26**
$\mathcal{C}_6$	$\mathcal{C}_3\mathcal{C}_4\mathcal{C}_5$	20.48	16.74	13.39	12.61	12.48	**12.14**

TABLE II: The performance of each GPA method on the TOPACS dataset. We report the standard-deviation of the transformed points in the test set. The smaller the statistics, the better consistency the registration has.
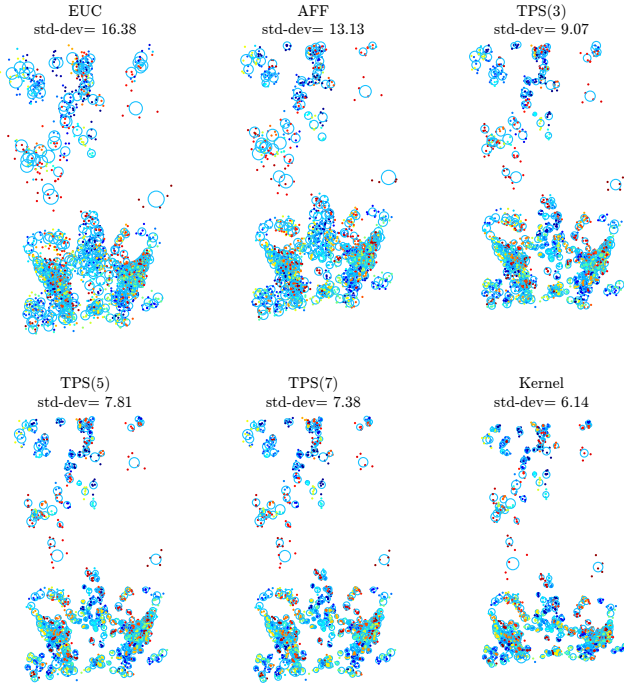
EUC
std-dev= 16.38

AFF
std-dev= 13.13

TPS(3)
std-dev= 9.07

TPS(5)
std-dev= 7.81

TPS(7)
std-dev= 7.38

Kernel
std-dev= 6.14

Fig. 7: The GPA registration by different transformation models on the TOPACS dataset. In this example, we use $\mathcal{C}_4$ to estimate the transformation and $\mathcal{C}_5\mathcal{C}_6$ to validate the fitness. The transformed correspondences in $\mathcal{C}_5\mathcal{C}_6$ are color-coded dots. The mean of each transformed correspondence is plotted as a hollow circle with its size encoding the standard-deviation.

report the average standard-deviation in Table II. We provide a qualitative result Fig. 7 where we use $\mathcal{C}_4$ to estimate the transformation and $\mathcal{C}_5$, $\mathcal{C}_6$ to validate the fitness.

It is clear that the result improves from the rigid model (EUC) to DefGPA (AFF, TPS(3), TPS(5) and TPS(7)), and the proposed KernelGPA (using the kernel model) achieves the best result both qualitatively and quantitatively.

IX. CONCLUSION

We propose KernelGPA, a novel generalized Procrustes analysis method based on a kernel method. At its core, KernelGPA uses a kernel method to define the nonrigid transformation. Analogous to the previous studied DefGPA using the LBWs, the proposed KernelGPA can be solved globally in closed-form to obtain the optimal estimate of the principal axes of the map. We propose a new method to estimate the global scale ambiguity $\mathbf{\Lambda}$ and the rigid poses altogether in a unified optimization framework, which handles missing correspondences and forms a proper back-end for deformable SLAM. KernelGPA outperforms DefGPA [4] in terms of registration performance on various datasets.

Sensor Fusion. It is interesting to explore the possibility of using the as-rigid-as-possible cost (25) in the sensor fusion framework. We sketch the general idea as follows for future

researchers. To begin with, we denote cost (25) explicitly as:

$$\phi_{\mathrm{ARAP}}(\mathbf{R}_t, \mathbf{t}_t, \mathbf{\Lambda}) = \sum_{t=1}^{n} \left\| \mathbf{R}_t\mathbf{P}_t + \mathbf{t}_t\mathbf{1}^\top - \sqrt{\mathbf{\Lambda}}\mathbf{S}_t \right\|_{\mathcal{F}}^2,$$

which we have shown to be sufficient to estimate $\mathbf{R}_t$, $\mathbf{t}_t$, $\mathbf{\Lambda}$ from Section VI. In SLAM applications, often additional sensor information, *e.g.,* IMUs, are available to assist pose estimation. The IMU measurements can be preintegrated to form a cost term ϕ_{IMU} [18]. Other sensor measurements can be used in an analogous manner. In general, we propose the following cost for sensor fusion:

$$\phi_{\mathrm{Fusion}}(\mathbf{R}_t, \mathbf{t}_t, \mathbf{\Lambda}, \dots) = \phi_{\mathrm{ARAP}} + \phi_{\mathrm{IMU}} + \cdots,$$

and pose sensor fusion for deformable SLAM as:

$$\min \ \phi_{\mathrm{Fusion}}(\mathbf{R}_t, \mathbf{t}_t, \mathbf{\Lambda}, \dots).$$

This problem is well-posed as the cost ϕ_{ARAP} constrains $\mathbf{R}_t$, $\mathbf{t}_t$, $\mathbf{\Lambda}$ entirely. The compound cost ϕ_{Fusion} can be minimized by standard graph optimization software packages [31]. Other parameters can be initialized based on the initialization of $\mathbf{R}_t$, $\mathbf{t}_t$, $\mathbf{\Lambda}$ proposed in Section VI-B2.

APPENDIX

Let us define $\mathbb{C}\mathrm{ov}(\mathbf{M}) = \bar{\mathbf{M}}\bar{\mathbf{M}}^\top$ with $\bar{\mathbf{M}} = \mathbf{M} - \frac{1}{m}\mathbf{M}\mathbf{1}\mathbf{1}^\top$ being the zero-centered point-cloud of $\mathbf{M}$. Then it can be verified (by Lemma 2 in [4]) that $\mathbb{C}\mathrm{ov}(\mathbf{M})$ satisfies:

$$\mathbb{C}\mathrm{ov}(\mathbf{R}\mathbf{M} + \mathbf{t}\mathbf{1}^\top) = \mathbf{R}\mathbb{C}\mathrm{ov}(\mathbf{M})\mathbf{R}^\top, \tag{31}$$

for any arbitrary rotation $\mathbf{R}$ and translation $\mathbf{t}$. This result basically states that $\mathbb{C}\mathrm{ov}(\mathbf{M})$ is purely decided by the rotation $\mathbf{R}$, and is independent of the translation $\mathbf{t}$. Thus it allows the possibility to use $\mathbb{C}\mathrm{ov}(\mathbf{M})$ to fix the rotation gauge. If $\mathbf{M}\mathbf{1} = \mathbf{0}$, *i.e.,* constraint (10), we have $\mathbb{C}\mathrm{ov}(\mathbf{M}) = \mathbf{M}\mathbf{M}^\top$.

Let us denote the eigenvalue decomposition of $\mathbb{C}\mathrm{ov}(\mathbf{M})$ as $\mathbb{C}\mathrm{ov}(\mathbf{M}) = \mathbf{M}\mathbf{M}^\top = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top$. Then the following is a valid eigenvalue decomposition of $\mathbf{R}\mathbb{C}\mathrm{ov}(\mathbf{M})\mathbf{R}^\top$:

$$\mathbf{R}\mathbb{C}\mathrm{ov}(\mathbf{M})\mathbf{R}^\top = \mathbf{R}\mathbf{M}\mathbf{M}^\top\mathbf{R}^\top = \mathbf{R}\mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top\mathbf{R}^\top, \tag{32}$$

which has the same eigenvalues as $\mathbb{C}\mathrm{ov}(\mathbf{M})$ which are the diagonal elements of $\mathbf{\Lambda}$. In other words, the eigenvalues of $\mathbb{C}\mathrm{ov}(\mathbf{M})$ are preserved under rotations, thus can be used to fix the rotation gauge. By letting $\mathbf{M}\mathbf{M}^\top = \mathbf{\Lambda}$, *i.e.,* constraint (11), we fix the rotation to $\mathbf{R} = \mathbf{Q}^\top$.

The constraint $\mathbf{M}\mathbf{1} = \mathbf{0}$ does not cause any bias to the estimation if the deformation model contains a free translation [4], *i.e.,* if matrix $\mathcal{Q}$ in formulation (20) has an all-ones eigenvector with eigenvalue 0. This is verified as $\mathcal{Q}\mathbf{1} = \mathbf{0}$ for the kernel model by Proposition 1. The constraint $\mathbf{M}\mathbf{M}^\top = \mathbf{\Lambda}$ implies an as-rigid-as-possible solution. This is verified by the implicit eigenvalue decomposition in (32) and the explicit estimation of $\mathbf{\Lambda}$ by the cost (25).

REFERENCES

[1] Brett Allen, Brian Curless, and Zoran Popović. The space of human body shapes: reconstruction and parameterization from range scans. *ACM transactions on graphics (TOG)*, 22(3):587–594, 2003.

[2] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. SCAPE: shape completion and animation of people. In *ACM SIGGRAPH 2005 Papers*, pages 408–416. 2005.

[3] K Somani Arun, Thomas S Huang, and Steven D Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on pattern analysis and machine intelligence*, (5):698–700, 1987.

[4] Fang Bai and Adrien Bartoli. Procrustes analysis with deformations: A closed-form solution by eigenvalue decomposition. *International Journal of Computer Vision*, 130(2):567–593, 2022.

[5] Fang Bai, Teresa Vidal-Calleja, and Giorgio Grisetti. Sparse pose graph optimization in cycle space. *IEEE Transactions on Robotics*, 37(5):1381–1400, 2021.

[6] Adrien Bartoli. Towards 3d motion estimation from deformable surfaces. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 3083–3088. IEEE, 2006.

[7] Adrien Bartoli, Mathieu Perriollat, and Sylvie Chambon. Generalized thin-plate spline warps. *International Journal of Computer Vision*, 88(1):85–110, 2010.

[8] Adrien Bartoli, Yan Gérard, Francois Chadebecq, Toby Collins, and Daniel Pizarro. Shape-from-template. *IEEE transactions on pattern analysis and machine intelligence*, 37(10):2099–2118, 2015.

[9] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[10] Fred L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on pattern analysis and machine intelligence*, 11 (6):567–585, 1989.

[11] Sylvain Bouix, Jens C Pruessner, D Louis Collins, and Kaleem Siddiqi. Hippocampal shape analysis using medial surfaces. *Neuroimage*, 25(4):1077–1089, 2005.

[12] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3d shape from image streams. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 2, pages 690–696. IEEE, 2000.

[13] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332, 2016.

[14] Yuchao Dai, Hongdong Li, and Mingyi He. A simple prior-free method for non-rigid structure-from-motion factorization. *International Journal of Computer Vision*, 107(2):101–122, 2014.

[15] Jean Duchon. Interpolation des fonctions de deux variables suivant le principe de la flexion des plaques minces. *Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique*, 10(R3):5–12, 1976.

[16] P Thomas Fletcher, Conglin Lu, Stephen M Pizer, and Sarang Joshi. Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE transactions on medical imaging*, 23(8):995–1005, 2004.

[17] Mike Fornefett, Karl Rohr, and H Siegfried Stiehl. Radial basis functions with compact support for elastic registration of medical images. *Image and vision computing*, 19 (1-2):87–96, 2001.

[18] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-manifold preintegration for real-time visual–inertial odometry. *IEEE Transactions on Robotics*, 33(1):1–21, 2016.

[19] Oren Freifeld and Michael J Black. Lie bodies: A manifold representation of 3d human shape. In *European Conference on Computer Vision*, pages 1–14. Springer, 2012.

[20] Wei Gao and Russ Tedrake. Surfelwarp: Efficient non-volumetric single view dynamic reconstruction. In *Robotics: Science and System (RSS)*, 2018.

[21] Gene Golub and Victor Pereyra. Separable nonlinear least squares: the variable projection method and its applications. *Inverse problems*, 19(2):R1, 2003.

[22] Colin Goodall. Procrustes methods in the statistical analysis of shape. *Journal of the Royal Statistical Society: Series B (Methodological)*, 53(2):285–321, 1991.

[23] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[24] Berthold KP Horn, Hugh M Hilden, and Shahriar Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *JOSA A*, 5(7):1127–1135, 1988.

[25] Shoudong Huang, Yongbo Chen, Liang Zhao, Yanhao Zhang, and Mengya Xu. Some research questions for slam in deformable environments. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7653–7660, 2021.

[26] Ian H Jermyn, Sebastian Kurtek, Eric Klassen, and Anuj Srivastava. Elastic shape matching of parameterized surfaces using square root normal fields. In *European conference on computer vision*, pages 804–817. Springer, 2012.

[27] Shantanu H Joshi, Eric Klassen, Anuj Srivastava, and Ian Jermyn. A novel representation for riemannian analysis of elastic curves in rn. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7. IEEE, 2007.

[28] David G Kendall. Shape manifolds, procrustean metrics, and complex projective spaces. *Bulletin of the London mathematical society*, 16(2):81–121, 1984.

[29] Martin Kilian, Niloy J Mitra, and Helmut Pottmann. Geometric modeling in shape space. In *ACM SIGGRAPH*

2007 papers, pages 64–es. 2007.

[30] Rainer Kümmerle, Bastian Steder, Christian Dornhege, Michael Ruhnke, Giorgio Grisetti, Cyrill Stachniss, and Alexander Kleiner. On measuring the accuracy of slam algorithms. *Autonomous Robots*, 27(4):387–407, 2009.

[31] Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. g2o: A general framework for graph optimization. In *2011 IEEE International Conference on Robotics and Automation*, pages 3607–3613. IEEE, 2011.

[32] Sebastian Kurtek, Eric Klassen, Zhaohua Ding, and Anuj Srivastava. A novel riemannian framework for shape analysis of 3d objects. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 1625–1632. IEEE, 2010.

[33] Sebastian Kurtek, Eric Klassen, John C Gore, Zhaohua Ding, and Anuj Srivastava. Elastic geodesic paths in shape space of parameterized surfaces. *IEEE transactions on pattern analysis and machine intelligence*, 34(9): 1717–1730, 2011.

[34] Hamid Laga. A survey on nonrigid 3d shape analysis. *Academic Press Library in Signal Processing, Volume 6*, pages 261–304, 2018.

[35] Hamid Laga, Qian Xie, Ian H Jermyn, and Anuj Srivastava. Numerical inversion of srnf maps for elastic shape analysis of genus-zero surfaces. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2451–2464, 2017.

[36] Jose Lamarca, Shaifali Parashar, Adrien Bartoli, and JMM Montiel. Defslam: Tracking and mapping of deforming scenes from monocular sequences. *IEEE Transactions on robotics*, 37(1):291–303, 2020.

[37] Abed Malti and Cédric Herzet. Elastic shape-from-template with spatially sparse deforming forces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3337–3345, 2017.

[38] Richard A Newcombe, Dieter Fox, and Steven M Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 343–352, 2015.

[39] Stanley Osher and Ronald P Fedkiw. *Level set methods and dynamic implicit surfaces*, volume 153. Springer, 2003.

[40] Shaifali Parashar, Daniel Pizarro, and Adrien Bartoli. Isometric non-rigid shape-from-motion with riemannian geometry solved in linear time. *IEEE transactions on pattern analysis and machine intelligence*, 40(10):2442–2454, 2017.

[41] Shaifali Parashar, Daniel Pizarro, and Adrien Bartoli. Local deformable 3d reconstruction with cartan's connections. *IEEE transactions on pattern analysis and machine intelligence*, 42(12):3011–3026, 2019.

[42] Ivatury S. Raju and James C. Newman. surf3d: A 3-d finite-element program for the analysis of surface and corner cracks in solids subjected to mode-1 loadings. 1993.

[43] F James Rohlf and Dennis Slice. Extensions of the Procrustes method for the optimal superimposition of landmarks. *Systematic biology*, 39(1):40–59, 1990.

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

[45] Daniel Rueckert, Luke I Sonoda, Carmel Hayes, Derek LG Hill, Martin O Leach, and David J Hawkes. Nonrigid registration using free-form deformations: application to breast mr images. *IEEE transactions on medical imaging*, 18(8):712–721, 1999.

[46] Miroslava Slavcheva, Maximilian Baust, Daniel Cremers, and Slobodan Ilic. Killingfusion: Non-rigid 3d reconstruction without correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1386–1395, 2017.

[47] Miroslava Slavcheva, Maximilian Baust, and Slobodan Ilic. Sobolevfusion: 3d reconstruction of scenes undergoing free non-rigid motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2646–2655, 2018.

[48] Jingwei Song, Jun Wang, Liang Zhao, Shoudong Huang, and Gamini Dissanayake. Mis-slam: Real-time large-scale dense deformable slam system in minimal invasive surgery based on heterogeneous computing. *IEEE Robotics and Automation Letters*, 3(4):4068–4075, 2018.

[49] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, volume 4, pages 109–116, 2007.

[50] Robert W Sumner, Johannes Schmid, and Mark Pauly. Embedded deformation for shape manipulation. In *ACM SIGGRAPH 2007 papers*, pages 80–es. 2007.

[51] Richard Szeliski and James Coughlan. Spline-based image registration. *International Journal of Computer Vision*, 22(3):199–218, 1997.

[52] Jing Xiao, Jinxiang Chai, and Takeo Kanade. A closed-form solution to non-rigid shape and motion recovery. *International Journal of Computer Vision*, 67(2):233–246, 2006.

[53] Laurent Younes. Spaces and manifolds of shapes in computer vision: An overview. *Image and Vision Computing*, 30(6-7):389–397, 2012.

[54] Laurent Younes, Peter W Michor, Jayant M Shah, and David B Mumford. A metric on shape space with explicit geodesics. *Rendiconti Lincei-Matematica e Applicazioni*, 19(1):25–57, 2008.