2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

6D Pose Estimation for Flexible Production with Small Lot Sizes based on CAD Models using Gaussian Process Implicit Surfaces

Jianjie Lin[1], Markus Rickert[1] and Alois Knoll[2]

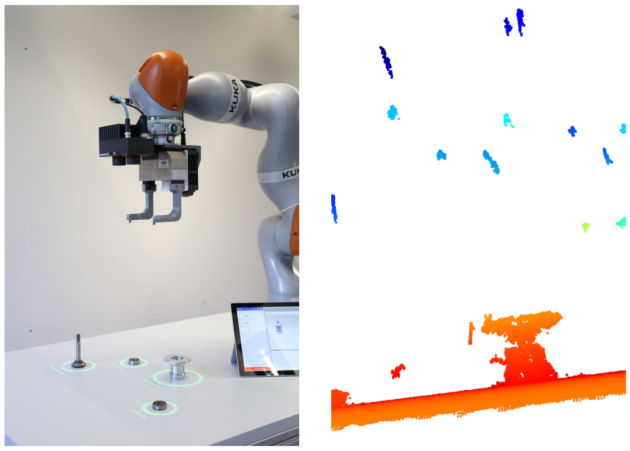
***Abstract*— We propose a surface-to-surface (S2S) point registration algorithm by exploiting the Gaussian Process Implicit Surfaces for partially overlapping 3D surfaces to estimate the 6D pose transformation. Unlike traditional approaches, that separate the corresponding search and update steps in the inner loop, we formulate the point registration as a nonlinear non-constraints optimization problem which does not explicitly use any corresponding points between two point sets. According to the implicit function theorem, we form one point set as a Gaussian Process Implicit Surfaces utilizing the signed distance function, which implicitly creates three manifolds. Points on the same manifold share the same function value, indicated as $\{1, 0, -1\}$. The problem is thus converted into finding a rigid transformation that minimizes the inherent function value. This can be solved by using a Gauss-Newton (GN) or Levenberg-Marquardt (LM) solver. In the case of a partially overlapping 3D surface, the Fast Point Feature Histogram (FPFH) algorithm is applied to both point sets and a Principal Component Analysis (PCA) is performed on the result. Based on this, the initial transformation can then be computed. We conduct experiments on multiple point sets to evaluate the effectiveness of our proposed approach against existing state-of-the-art methods.**

I. Introduction

While the majority of European companies consists of small and medium-sized enterprises, only a very limited number of them currently uses robot systems in their production. In contrast to larger companies, they mainly deal with small lot sizes and constantly changing production processes. Adapting a robot systems to new products and parts is however very time-consuming and requires expert knowledge in robotics that is not commonly found in shop floor workers [1]. While a number of modern robot systems currently on the market proposes an easier programming concept based on reusable skills, these approaches still require a manual adaptation to new processes. In contrast to this, approaches from service robotics are able to automatically solve declarative goal specifications by using semantic knowledge in combination with reasoning and inference [2].

Synthesizing robot programs based on semantic product, process, and resource descriptions enables an automatic adaptation to new processes and involves handling the recognition of objects and parts in the environment. These parts are typically designed in CAD systems and described via a boundary representation [3]. In order to grasp them with a robot, a full 6D pose estimation is required. Given the small lot size production of SMEs with constantly changing objects, it is not feasible to train object recognition models over a long period of time. Recognizing these parts efficiently by directly using their CAD models during execution is therefore essential in achieving short changeover times. Fig. 1 shows an example of such an assembly use case for a mechanical gearbox together with a point cloud scene captured by the 3D camera sensor attached to the robot.

[1]Jianjie Lin, Markus Rickert are with fortiss, An-Institut Technische Universität München, Munich, Germany {lin,rickert}@fortiss.org
[2]Alois Knoll are with Robotics and Embedded System, Department of Informatics, Technische Universität München, Munich, Germany knoll@in.tum.de



(a) (b)

Fig. 1: Robot setup for assembling a gear box with (a) a lightweight robot and a 3D camera sensor, (b) point cloud scene of mechanical gearbox parts on the table.

Point registration is one of the main approaches in computing the pose transformation by two given point sets and is widely used in MRI/CAT scan alignment [4] and robot manipulation [5]. The problem is especially challenging when two noisy point sets only partially overlap without initial alignment. A standard approach for point registration is based on the Iterative Closest Point (ICP) algorithm [6], [7]. It is interesting due to its intuitive and straightforward implementation. The identification of corresponding points follows a greedy search algorithm that is subjective to local minima and identifies incorrect points for some rotations. Furthermore, the success of ICP heavily relies on a good initial alignment. There are many variants which aim at optimizing the process for correspondence search, such as widening the convergence basin, heuristic global search, relaxed assignments, or distance fields, which however often fail to achieve a better performance and typically follow the principle of point to point (p2p) or point to surface (p2s) registration. Furthermore, with increasingly powerful neural networks, many researchers started applying deep learning to the problem of computing a pose transformation [8]. However, these approaches still follow the concept of finding the corresponding point and use a RANASAC-based Perspective-n-Point(PnP) algorithm to acquire the 6D pose. In addition, they require a large data set to encode a surrogate task and cannot be transferred to another task efficiently. For a flexible production application however, efficiency is a key factor required in any suitable algorithm.

This work, to the best of our knowledge, is the first one to consider the surface-to-surface (s2s) point registration and to describe one surface as an implicit function by employing Gaussian process regression, also referred to as Gaussian Process Implicit Surfaces (GPIS). We define three manifolds by borrowing the idea of signed distance functions (SDF) [9] with values $\{1, 0, -1\}$. All points on a surface should have the same value of 0. Instead of searching the corresponding points between two different point sets, the goal is now to find a rigid transformation that makes function value zero by transferring the point using this transformation. We evaluate the presented GPIS-based point registration on multiple point sets. In comparison to state-of-the-art algorithms, the presented approach can arrive at or exceed the same accuracy. We prioritize robustness over convergence speed and our approach can achieve more robust results than most of the existing algorithms. The primary advantage in contrast to other algorithms is that our approach does not require finding the corresponding point iteratively. The initialization for the transformation can be calculated based on a FPFH and PCA. The registration problem is efficiently solved with a Lie algebra-based Gaussian Newton solver.

II. Related Work

6D pose estimation is widely used and extensively studied and point registration technology is commonly used to find the spatial relationship between two point sets. Most of the advanced algorithms in this field are based on ICP and several variants exist [10], [7]. The typical work flow for geometric registration consists of two stages: initial (global) alignment and local refinement. Initial alignment is either based on simple Euclidean distance or on more complex sampling-based algorithms that identify matching points by utilizing local geometrical descriptors like Fast Point Feature Histogram (FPFH) [11], [12]. RANSAC [13] can be applied against outliers. In the following, the initial rigid transformation can be estimated by using a least squared method or by using the branch-and-bound framework (BnB) [14], which is a global optimization algorithm. In both cases, finding a good initial alignment can be computationally expensive. After the initial alignment, local refinement is executed by alternating the steps for finding the nearest neighborhood and the steps for updating the transformation based on the greedy search algorithm. This is susceptible to local minima and can only produce an accurate result with a good initialization. Several variants can be used to improve the performance: Fitzgibbon et al. [15] proposed a Levenberg-Marquardt algorithm that uses finite differences to optimize the objective function. Granger et al. [6] applied Expectation-Maximization (EM) principles to consider Gaussian noise, which can improve the robustness of local registration. Li et al. [16] utilize the Gaussian mixture model to model the surface uncertainty so that it increases the robustness of the registration. Myronenko et al.[17] introduced Coherent Point Drift (CPD), which is agnostic as to the used transformation model and similar to GMM takes a probabilistic approach to the alignment of point sets. Chavdar et al. [18] applied stochastic optimization to consider noise robustness, outlier resistance, and optimal global alignment. Yang et al. [14] proposed Globally Optimal ICP (Go-ICP) by using the Branch and Bound framework to derive the lower and upper bound for the error function and to integrate a local ICP in the same frame. Guo et al. [12] introduced the Fast Global Registration (FGR) algorithm that uses a scaled Geman-McClure estimator to describe the error function, optimizes the objective function by using Block Coordinate Descent, and applies FPFH to search the corresponding set before optimization.

All algorithms mentioned above share the requirement of finding the correct corresponding pair. This is followed by using either greedy search or a global optimizer to get the final rigid transformation. With continuous improvement in the field of deep learning, many researchers began to learn the 6D pose directly by using RGB images [19]. However, a large number of point sets is required to learn the surrogate task and the underlying relationship between learned loss function and the pose accuracy is still unclear. Such an approach cannot be easily deployed in a flexible production line. In this paper, we introduce a different approach for GPIS-based point registration (GPIS-S2SPR), that does not need to explicitly use corresponding points. By applying Gaussian Process Implicit Surfaces, we can achieve a more robust performance compared to state-of-the-art algorithms.

III. Problem Formulation

The standard ICP algorithm aims to estimate a rigid transformation $\boldsymbol{T} = \{\boldsymbol{R}, \boldsymbol{t}\}$ between given two point sets $\mathcal{X} = \{\boldsymbol{x}_i\}, i = 0, \ldots, n$ and $\mathcal{Y} = \{\boldsymbol{y}_i\}, i = 0, \ldots, m$, that minimizes the object function

$$E(\boldsymbol{T}) = \min \sum_{i=1}^{n} \Big(\boldsymbol{x}_i - \boldsymbol{T}\boldsymbol{y}_{\arg\min_{j=0,\ldots,m} \left(\|\boldsymbol{x}_i - \boldsymbol{T}\boldsymbol{y}_j\|^2 \right)} \Big) \quad (1)$$

by iteratively finding corresponding points. The objective function in (1) however is a non-convex function and therefore susceptible to local minima.

In this paper, we consider the problem from a different angle: aligning two point sets independent of corresponding points. We model one of the point set as an implicit surface, which can capture the local structure of the surface and then transfer another point set to this implicit surface. We assume that all points on the same surface share the same function value of 0. Then, the point registration problem is turned into finding a transformation that minimizes the objective function

$$E(\boldsymbol{T}) = \frac{1}{2} \sum_{j=0}^{m} \left\| f(\boldsymbol{T}\boldsymbol{y}_j, \mathcal{X}) \right\|^2 . \quad (2)$$

IV. Preliminary Knowledge

A. Gaussian Process Implicit Surfaces

According to the implict function theorem, the implicit function can be formally described as $f(x) = 0$, where f is a scalar function that takes an input $x \in \mathbb{R}^d$ and results in a $d-1$ dimensional manifold $\mathcal{S}$. For point registration, we will constrain d to dimension 3. Many existing methods can be used to describe the implicit surface, e.g., trimmed B-splines [20], transcendental functions, or thin plate splines [21]. In this work, Gaussian Process Implicit Surfaces [22] will be used to describe the 3D mesh surface. Unlike thin plate splines, the Radial Basis Function (RBF) kernel can only describe a local patch whose distribution will rapidly converge to zero when a point is not near the queried point and not on the surface, which however is in contrast to our assumption. As an alternative, thin plate splines will be selected via

$$k_{ij}(r) = 2r^3 - 3Cr^2 + C^3\,, \tag{3}$$

with a maximum radius of C inside the training point cloud sets $\mathcal{X}$. The parameter r is the distance between two points. Gaussian Process Implicit Surfaces can be considered as a standard regression problem, which is expressed as $\boldsymbol{f} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{K}(\mathcal{X},\mathcal{X}) + \boldsymbol{\sigma}^{\mathrm{T}}\boldsymbol{I}\boldsymbol{\sigma}\right)$, where $\mathcal{N}$ denotes a normal distribution with the mean $\mu(\mathcal{X}) = \mathbf{0}$ and variance $\boldsymbol{K}(\mathcal{X},\mathcal{X}) + \boldsymbol{\sigma}^{\mathrm{T}}\boldsymbol{I}\boldsymbol{\sigma}$. The covariance matrix $\boldsymbol{K}(.,.)$ consists of $k_{ij}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $\boldsymbol{\sigma}$ corresponds to the noise.

The goal therefore is to predict the value by evaluating the following formulas for the queried point $\mathbf{y}_j$:

$$f_*(\boldsymbol{y}_j|\mathcal{X}) = \boldsymbol{k}_*^{\mathrm{T}}\left[\boldsymbol{K} + \boldsymbol{\sigma}^{\mathrm{T}}\boldsymbol{I}\boldsymbol{\sigma}\right]^{-1}\boldsymbol{f} = \boldsymbol{k}_*^{\mathrm{T}}\boldsymbol{\alpha} \tag{4}$$

$$V(\boldsymbol{y}_j|\mathcal{X}) = k(\boldsymbol{y}_j, \boldsymbol{y}_j) - \boldsymbol{k}_*^{\mathrm{T}}\left[\boldsymbol{K} + \boldsymbol{\sigma}^{\mathrm{T}}\boldsymbol{I}\boldsymbol{\sigma}\right]^{-1}\boldsymbol{k}_*\,. \tag{5}$$

The kernel function $\boldsymbol{k}_*(\mathcal{X}, \boldsymbol{y}_j)$ is used to describe the correlation between source point set $\mathcal{X}$ and target point set $\boldsymbol{y}_j$. The function value $f_*(\boldsymbol{y}_j|\mathcal{X})$ is a prediction of $\boldsymbol{y}_j$ with the corresponding variance, which is expressed as $V(\boldsymbol{y}_j|\mathcal{X})$ and can be used to evaluate the reliability of the predicted value. For simplification, this value is not included inside the objective function. The 3D model points according to SDF are denoted as $\left\{\boldsymbol{X}_i \in \mathcal{X}^0 \mid \boldsymbol{X}_i = \{\boldsymbol{x}_i, \sigma_i, 0\}\right\}$. To aid the training of an implicit function, two additional point sets will be created: $\left\{\boldsymbol{X}_i \in \mathcal{X}^1 \mid \boldsymbol{X}_i = \{\boldsymbol{x}_i, \sigma_i, 1\}\right\}$ lies outside the surface and $\left\{\boldsymbol{X}_i \in \mathcal{X}^{-1} \mid \boldsymbol{X}_i = \{\boldsymbol{x}_i, \sigma_i, -1\}\right\}$ lies inside the surface. How to generate these two additional point sets is presented in [23]. The final training point sets consist of $\mathcal{X} = \mathcal{X}^0 \cup \mathcal{X}^1 \cup \mathcal{X}^{-1} \in \mathbb{R}^{n\times 3}$.

After modeling the training point sets as the GPIS in the sense of a manifold, the objective function E can be further expressed as

$$f_j(\boldsymbol{y}_j|\mathcal{X}) = \boldsymbol{k}_j(\mathcal{X}, \boldsymbol{y}_j)^{\mathrm{T}}\boldsymbol{\alpha} = \sum\nolimits_{i=0}^{n} k(\boldsymbol{x}_i, \boldsymbol{y}_j)\alpha_i \tag{6}$$

$$E = \frac{1}{2}\sum\nolimits_{j=0}^{m} f_j^2(\boldsymbol{y}_j|\mathcal{X}) = \sum\nolimits_{j=0}^{m} \boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{k}_j\boldsymbol{k}_j^{\mathrm{T}}\boldsymbol{\alpha}\,, \tag{7}$$

where m is the number of points in the target point cloud and n is the number of points in the source point cloud. f_j is the predictive value given by $\boldsymbol{y}_j$ that is equal to zero if the target point lies on the mesh surface. The main benefits of this formulation are that no corresponding points between two point sets are required and that it converts the problem into a standard nonlinear squares problem, which can be solved by standard convex solvers.

B. Lie Algebra for Optimization

The transformation matrix $\boldsymbol{T}$ consists of a rotation matrix $\boldsymbol{R}$ and a translation $\boldsymbol{t}$, which can be interpreted in terms of Lie groups SE(3) [24] with the corresponding Lie algebra $\mathfrak{se}(3)$. It can also be converted to a Lie group by utilizing the exponential map $\boldsymbol{T} = \exp(\boldsymbol{\xi}^\wedge)$, where $\boldsymbol{\xi}^{\mathrm{T}} = \begin{bmatrix}\boldsymbol{\rho}^{\mathrm{T}} & \boldsymbol{\phi}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{1\times 6}$ with $\boldsymbol{\rho} \in \mathbb{R}^3$ and $\boldsymbol{\phi} \in \mathfrak{so}(3)$. In this optimized formulation, the target point set is transferred by $\boldsymbol{T}$ to align the source point set, which is embedded into $f(\mathcal{X})$. The gradient-based optimization uses the Jacobian matrix to search for the minimum solution. We apply the perturbation method for calculating the gradient of $\boldsymbol{T}$ and consider the directional of the derivative of $\boldsymbol{T}$ with respect to the perturbation $\boldsymbol{\epsilon} \in \mathbb{R}^6$, which can be computed as

$$\boldsymbol{T} = \exp\left(\boldsymbol{\epsilon}^\wedge\right)\boldsymbol{T}_{\mathrm{op}} \approx (\boldsymbol{I} + \boldsymbol{\epsilon}^\wedge)\boldsymbol{T}_{\mathrm{op}}\,, \quad \frac{\partial(\boldsymbol{T}\hat{\boldsymbol{y}}_j)}{\partial\boldsymbol{\epsilon}} = (\boldsymbol{T}\hat{\boldsymbol{y}}_j)^{\odot}\,. \tag{8}$$

The operator $(.)^{\odot} : \mathbb{R}^4 \to \mathbb{R}^{4\times 6}$ is defined as $[\boldsymbol{\varepsilon}^{\mathrm{T}}, \eta]^{\mathrm{T}\odot} = \begin{bmatrix}\eta\boldsymbol{I} & -\boldsymbol{\varepsilon}^\wedge \\ \mathbf{0}^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}}\end{bmatrix}$, where $\boldsymbol{\varepsilon} \in \mathbb{R}^3$ and η is a scalar that maps the vector space to a higher manifold. For simplification, we omit $\boldsymbol{x}_i$ in $k_i(\boldsymbol{x}_i, \boldsymbol{T}\hat{\boldsymbol{y}}_j)$ and use $k_i(\boldsymbol{T}\hat{\boldsymbol{y}}_j)$ instead. By integration of the perturbation formula (8) with the first order Taylor series in kernel function, it can be approximated as

$$\begin{aligned} k_i(\boldsymbol{T}\hat{\boldsymbol{y}}_j) &= k_i\big(\exp(\boldsymbol{\epsilon}^\wedge)\boldsymbol{T}_{\mathrm{op}}\hat{\boldsymbol{y}}_j\big) \approx k_i\big((\boldsymbol{I} + \boldsymbol{\epsilon}^\wedge)\boldsymbol{T}_{\mathrm{op}}\hat{\boldsymbol{y}}_j\big) \\ &\approx k_i(\boldsymbol{T}_{\mathrm{op}}\hat{\boldsymbol{y}}_j) + \left(\frac{\partial k_i}{\partial\boldsymbol{y}_j}\right)^{\mathrm{T}}\Big|_{\boldsymbol{y}_j = \boldsymbol{T}_{\mathrm{op}}\hat{\boldsymbol{y}}_j} \frac{\partial\boldsymbol{T}\hat{\boldsymbol{y}}_j}{\partial\boldsymbol{\epsilon}}\boldsymbol{\epsilon} \\ &\approx \beta_i + \boldsymbol{\delta_i}^{\mathrm{T}}\boldsymbol{\epsilon}\,, \end{aligned} \tag{9}$$

where $\beta_i = k_i(\boldsymbol{T}_{\mathrm{op}}\hat{\boldsymbol{y}}_j) \in \mathbb{R}$, $\left(\frac{\partial k_i}{\partial\boldsymbol{y}_j}\right)^{\mathrm{T}} \in \mathbb{R}^{1\times 4}$ and $\boldsymbol{\delta_i}^{\mathrm{T}} = \left(\frac{\partial k_i}{\partial\boldsymbol{y}_j}\right)^{\mathrm{T}}\Big|_{\boldsymbol{y}_j = \boldsymbol{T}_{\mathrm{op}}\hat{\boldsymbol{y}}_j}\left((\boldsymbol{T}\hat{\boldsymbol{y}}_j)^{\odot}\right) \in \mathbb{R}^{1\times 6}$. The derivative of the kernel function $\frac{\partial k_i}{\partial\boldsymbol{y}_j}$ in (9) is expressed as

$$\frac{\partial k_i}{\partial\boldsymbol{y}_j} = \frac{\partial k_i}{\partial r_{ij}}\frac{\partial r_{ij}}{\partial\boldsymbol{y}_j} = 6(r_{ij} - C)(\boldsymbol{y}_j - \boldsymbol{x}_i)\,.$$

V. GPIS-Based S2S Registration Algorithm

For the partially overlapping situation, the transformation matrix $\boldsymbol{T}$ is considered in the kernel's corresponding distance function with $r = \|\boldsymbol{x}_i - \boldsymbol{R}\hat{y}_j - \boldsymbol{t}\| = \|\boldsymbol{x}_i - \boldsymbol{T}\hat{\boldsymbol{y}}_j\|$. Therefore, the equation (6) is updated as $f_j = \boldsymbol{k}(\mathcal{X}, \boldsymbol{T}\hat{y}_j)^{\mathrm{T}}\boldsymbol{\alpha}$. By combining this with the approximation of the kernel function (9), the objective function (7) can be further simplified as

$$f_j = \sum\nolimits_{i=0}^{n}(\beta_i + \boldsymbol{\delta}_i^{\mathrm{T}}\boldsymbol{\epsilon})\,\alpha_n = \boldsymbol{k}^{\mathrm{T}}(\boldsymbol{\beta}, \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{\epsilon})\,\boldsymbol{\alpha} \tag{10}$$

$$E(\boldsymbol{T}) = \frac{1}{2}\sum\nolimits_{j=0}^{m}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{k}\,(\boldsymbol{\beta}, \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{\epsilon})\,\boldsymbol{k}^{\mathrm{T}}(\boldsymbol{\beta}, \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{\epsilon})\,\boldsymbol{\alpha}\,, \tag{11}$$

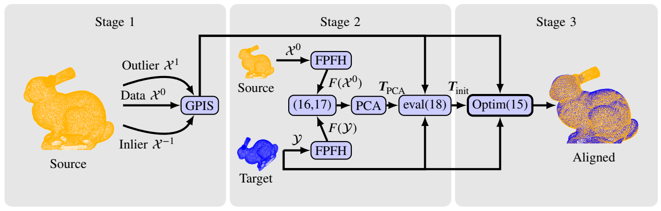


Fig. 2: The algorithm consists of three stages: In the first stage, two additional point sets $\mathcal{X}^1$ and $\mathcal{X}^2$ are created to augment the original point set as $\mathcal{X} = \mathcal{X}^0 \cup \mathcal{X}^1 \cup \mathcal{X}^{-1}$, which is used to form the implicit function GPIS. In stage two, we compute FPFH for each point in the source and target point set and a cross-checking is executed to identify a corresponding group. The PCA is utilized to compute the initial transformation by evaluating the objective function. In the last stage, the alignment is optimized by a convex solver.

with $\boldsymbol{k}^{\mathrm{T}}(\boldsymbol{\beta}, \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{\epsilon}) = \left[\beta_1 + \boldsymbol{\delta}_1^{\mathrm{T}}\boldsymbol{\epsilon}, \quad \cdots, \quad \beta_n + \boldsymbol{\delta}_n^{\mathrm{T}}\boldsymbol{\epsilon}\right] \in \mathbb{R}^{1\times n}$. As a result, (11) is converted to a nonlinear quadratic equation with the approximated nonlinear kernel function $\boldsymbol{k}(\boldsymbol{\beta}, \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{\epsilon})$ and the argument for this optimized problem is changed from the Lie group $\boldsymbol{T} \in \mathrm{SE}(3)$ to the perturbation variable of the Lie algebra $\boldsymbol{\epsilon} \in \mathfrak{se}(3)$.

A. Gradient of Objective Function

By taking the derivative of $\boldsymbol{J}$ with respect to $\boldsymbol{\epsilon}^{\mathrm{T}}$, we get

$$
\begin{aligned}
\frac{\partial E(\boldsymbol{T})}{\partial \boldsymbol{\epsilon}^{\mathrm{T}}} &= \sum_{j=0}^{m} \frac{\partial\left(\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{k}(\boldsymbol{\beta}, \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{\epsilon})\right)}{\partial \boldsymbol{\epsilon}^{\mathrm{T}}} \boldsymbol{k}^{\mathrm{T}}(\boldsymbol{\beta}, \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{\epsilon})\boldsymbol{\alpha} \\
&= \sum_{j=0}^{m}\left(\sum_{i=0}^{n} \frac{\partial(\alpha_i\beta_i + \alpha_i\boldsymbol{\epsilon}^{\mathrm{T}}\boldsymbol{\delta}_i)}{\partial \boldsymbol{\epsilon}^{\mathrm{T}}}\right)\boldsymbol{k}^{\mathrm{T}}(\boldsymbol{\beta}, \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{\epsilon})\boldsymbol{\alpha} \\
&= \sum_{j=0}^{m}\left(\sum_{i=0}^{n} \alpha_i\boldsymbol{\delta}_i\right)\boldsymbol{k}^{\mathrm{T}}(\boldsymbol{\beta}, \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{\epsilon})\boldsymbol{\alpha} \\
&= \sum_{j=0}^{m} \boldsymbol{\Delta}_j\left(\sum_{i=0}^{n} \beta_i\alpha_i + \boldsymbol{\delta}_i^{\mathrm{T}}\boldsymbol{\epsilon}\alpha_i\right), \qquad (12)
\end{aligned}
$$

where $\boldsymbol{\Delta}_j$ is defined as $\sum_{i=0}^{n} \alpha_i\boldsymbol{\delta}_i \in \mathbb{R}^{6\times 1}$ and can capture the surface's curvature by summing up all gradients in the kernel function. To get the optimum perturbation $\boldsymbol{\epsilon}^{\star}$ at the current position, the formula $\frac{\partial E(\boldsymbol{T})}{\partial \boldsymbol{\epsilon}^{\mathrm{T}}}$ is forced to be zero, leading to

$$
\begin{aligned}
\sum_{j=0}^{m} \boldsymbol{\Delta}_j \sum_{i=0}^{n} \boldsymbol{\delta}_i^{\mathrm{T}}\alpha_i\boldsymbol{\epsilon}^{\star} &= -\sum_{j=0}^{m} \boldsymbol{\Delta}_j \sum_{i=0}^{n} \beta_i\alpha_i \\
\boldsymbol{J}\boldsymbol{\epsilon}^{\star} &= -\sum_{j=0}^{m} \boldsymbol{\Delta}_j \sum_{i=0}^{n} \beta_i\alpha_i \qquad (13) \\
\boldsymbol{\epsilon}^{\star} &= -\boldsymbol{J}^{-1}\sum_{j=0}^{m} \boldsymbol{\Delta}_j \sum_{i=0}^{n} \beta_i\alpha_i\,, \qquad (14)
\end{aligned}
$$

with $\boldsymbol{J} = \sum_{j=0}^{m} \boldsymbol{\Delta}_j\boldsymbol{\Delta}_j^{\mathrm{T}} \in \mathbb{R}^{6\times 6}$. $\boldsymbol{T}$ is therefore updated as

$$
\boldsymbol{T}_{\mathrm{op},h} \leftarrow \exp\left((\boldsymbol{\epsilon}^{\star})^{\wedge}\right)\boldsymbol{T}_{\mathrm{op},h-1}, \qquad (15)
$$

which captures the local structural manifold by means of the Lie algebra. The optimization process follows the principle of the Gauss-Newton algorithm. We can further adapt $\boldsymbol{J}$ as $\sum_{j=0}^{m} \boldsymbol{\Delta}_j\boldsymbol{\Delta}_j^{\mathrm{T}} + \lambda\,\mathrm{diag}(\boldsymbol{S})$, which is the LM algorithm.

B. Initial Alignment Using PCA and FPFH

A good initial guess for the optimization is important in order to guarantee a good result and run-time. We present a new method for computing the initial alignment by employing the PCA and FPFH [11] algorithms. First, a FPFH is calculated for each point in the two point sets, which are referred to as $F(\mathcal{X}^0)$ and $F(\mathcal{Y})$. We then embed $F(\mathcal{X}^0)$ into a k-d tree $\mathrm{Kd}_{F(\mathcal{X}^0)}$ and a nearest neighbor search is performed for each feature $F(\boldsymbol{y}_j) \in F(\mathcal{Y})$, such that $\mathcal{G}_1$ is a group pair set of the results $\boldsymbol{x}_{i|j}$ for the queries $\boldsymbol{y}_j$:

$$
\mathcal{G}_1 = \left\{\{\boldsymbol{y}_j, \boldsymbol{x}_{i|j}\} \mid \mathrm{Kd}_{F(\mathcal{X}^0)}\left(F(\boldsymbol{y}_j)\right), \forall \boldsymbol{y}_j \in \mathcal{Y}\right\}. \qquad (16)
$$

Subsequently, we embed $F(\mathcal{Y})$ into another k-d tree $\mathrm{Kd}_{F(\mathcal{Y})}$ and perform a nearest neighbor search for each result stored in $\mathcal{G}_1$, such that $\mathcal{G}_2$ is a group pair set of the results $\boldsymbol{y}_{j|i}$ for the queries $\boldsymbol{x}_{i|j}$ that are stored in $\mathcal{G}_1$:

$$
\mathcal{G}_2 = \left\{\{\boldsymbol{x}_{i|j}, \boldsymbol{y}_{j|i}\} \mid \mathrm{Kd}_{F(\mathcal{Y})}\left(F(\boldsymbol{x}_{i|j})\right), \forall \boldsymbol{x}_{i|j} \in \mathcal{G}_1\right\} \qquad (17)
$$

We only keep the subset $\mathcal{G}_1 \cap \mathcal{G}_2$ that contains bidirectional nearest neighbors and refer to these as $\mathcal{X}'$ and $\mathcal{Y}'$. Statistical analysis techniques are then applied to remove any outliers in these groups [25]. The final selected points are grouped together and are denoted as $\mathcal{X}_{\text{group,FPFH}} \in \mathbb{R}^{n_{\text{FPFH}}\times 3}$ and $\mathcal{Y}_{\text{group,FPFH}} \in \mathbb{R}^{m_{\text{FPFH}}\times 3}$. After this, a PCA is used to compute the initial transformation $\boldsymbol{T}_{\text{PCA}}$ between $\mathcal{X}_{\text{group,FPFH}}$ and $\mathcal{Y}_{\text{group,FPFH}}$. Note, that $\boldsymbol{T}_{\text{PCA}}$ has four different possibilities according to the right-hand rule (Fig. 3). By evaluating the formula

$$
\boldsymbol{T}_{\text{init}} \leftarrow \min_{k\in\{0,\cdots 3\}} \sum_{j=0}^{m}\left\|f(\boldsymbol{T}_{\text{PCA},k}\hat{\boldsymbol{y}}_j, \mathcal{X})\right\|^2, \qquad (18)
$$

we select the transformation matrix $\boldsymbol{T}_{\text{init}}$ that has the smallest function value. The whole process is illustrated in Fig. 2 and the algorithm is summarized in Alg. 1.

VI. Evaluation

In order to compare our algorithm against the state of the art, we evaluated it against other registration algorithms

Algorithm 1 Optimization of transformation matrix by Gauss-Newton/Levenberg-Marquardt algorithm

```
Require: $\mathcal{X}$, $\mathcal{Y}$, $H$
 1: Modeling GPIS $f(\mathcal{X})$                                   ▷ Section IV-A
 2: Compute FPFH features $F(\mathcal{X}^0)$ and $F(\mathcal{Y})$      ▷ Section V-B
 3: Calculate $\mathcal{G}_1$ and $\mathcal{G}_2$                     ▷ (16), (17)
 4: $\{\mathcal{X}_{\text{group,FPFH}}, \mathcal{Y}_{\text{group,FPFH}}\} \leftarrow$ StaticalRemove($\mathcal{G}_1 \cap \mathcal{G}_2$)
 5: $\boldsymbol{T} \leftarrow$PCA($\mathcal{X}_{\text{group,FPFH}}, \mathcal{Y}_{\text{group,FPFH}}$)   ▷ (18)
 6: $\boldsymbol{T}_{\text{op},0} \leftarrow \arg\min \sum_{j=0}^{m} f(\mathcal{X}, \boldsymbol{T}\boldsymbol{y}_j)$
 7: for $h = 1 : H$ do
 8:     Approximate $k_i(\boldsymbol{y}_j)\ \forall \boldsymbol{y}_j \in \mathcal{Y}$    ▷ (9)
 9:     if Gauss-Newton then
10:         Set $\boldsymbol{J} = \sum_{j=0}^{m} \boldsymbol{\Delta}_j\boldsymbol{\Delta}_j^{\mathrm{T}}$
11:     end if
12:     if Levenberg-Marquardt then
13:         Set $\boldsymbol{J} = \sum_{j=0}^{m} \boldsymbol{\Delta}_j\boldsymbol{\Delta}_j^{\mathrm{T}} + \lambda\,\mathrm{diag}(\boldsymbol{S})$
14:     end if
15:     calculate $\boldsymbol{\epsilon}^{\star} = -\boldsymbol{J}^{-1}\sum_{j=0}^{m} \boldsymbol{\Delta}_j \sum_{i=0}^{n} \beta_i\alpha_i$   ▷ (14)
16:     Update $\boldsymbol{T}_{\text{op,h}}$                            ▷ (15)
17:     if $\|\boldsymbol{\epsilon}^{\star}\|_F \leq \epsilon$ then break
18:     end if
19: end for
20: return $\boldsymbol{T}_{\text{op}^{\star}}$
```

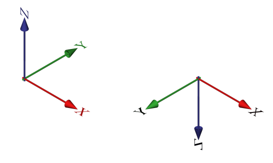
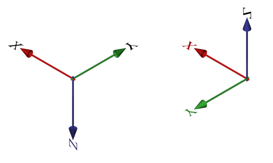



Fig. 3: The four possible coordinate systems for the PCA when computing the initial transformation matrix.

regarding accuracy RMSE and time on the Stanford 3D Scanning Repository's Happy Buddha, Stanford Bunny, and Chinese Dragon [26] as well as Blender's Suzanne model. PCL-ICP [25] is the standard implementation of the ICP algorithm in the Point Cloud Library, which is a local registration algorithm as it relies on a good initial alignment. SAC-IA-ICP [11] employs FPFH to get the initial alignment and then uses ICP to iteratively align the two point clouds. It is therefore considered a global point registration algorithm. GoICP and its variant GoICPT with trimming [14] are global registration algorithms that use the BnB algorithm in their implementations and also support partial overlapping point registration. Global registration RANSAC (Gl.RANSAC) [27] requires no initial alignment. Instead, it utilizes RANSAC for the initialization alignment by searching corresponding points in the FPFH feature space. Fast Global Registration (FGR) is another registration algorithm that utilizes FPFH for searching corresponding points. The algorithm presented in this paper is labeled GPIS-S2SPR. In this section, three different experiments were conducted. In order to reduce the computational burden, the tested point sets were downsampled for every algorithm into small scale numbers (1500–2500) by using voxel filtering. All evaluations were performed on a laptop with a 2.6 GHz Intel Core i7-6700HQ and 16 GB of RAM.

A. RMSE for Random Transformations

For exploring the capability of our algorithm, we evaluated algorithms using the Stanford Bunny point set with 50 290 points without Gaussian noise and only partial overlap (85%). We reduced the number points by applying a voxel grid filter with a size of 0.005 [25]. We ran each algorithm on a set of 40 random transformation matrices (Fig. 4).

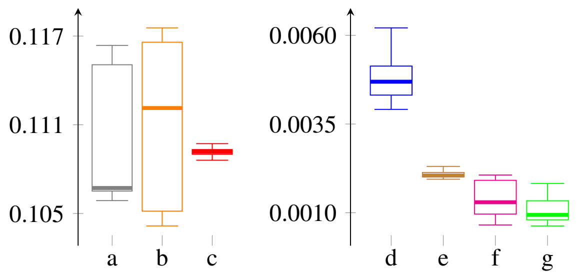


Fig. 4: RMSE for the Stanford Bunny with partial overlap (85%) without Gaussian noise: (a) GoICP, (b) GoICPT (10%), (c) PCL-ICP, (d) Gl.RANSAC, (e) SAC-IA-ICP, (f) FGR, (g) GPIS-S2SPR.

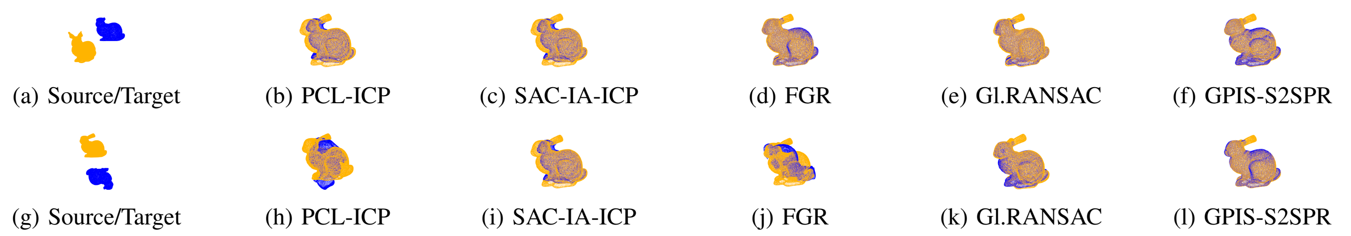
From the results, we can see that GoICP and its variant GoICP with trimming (GoICPT) (10%), as suggested in [14], performed worse in this experiment with median values of 0.110 and 0.108. PCL-ICP was evaluated using an identity matrix as initialization and behaved slightly better with a median value of 0.109 and a smaller variance in comparison to GoICP. Gl.RANSAC showed significant improvement with a median value of 0.005 and SAC-IA-ICP achieved a median value of 0.002. FGR achieved a median value of 0.001. In contrast to the previous algorithms however, its mean value of 0.017 deviated from the median and is much higher. This is due to its lack of robustness in handling large rotational changes, which is further evaluated in the next experiment (Section VI-B). Our algorithm showed the best overall performance, with a median value of 0.001 and a small variance.

B. Rotation and Translation Invariance

Rotation and translation invariance are essential factors for point registration. We conducted two experiments with the Stanford Bunny dataset to evaluate these two properties. For the first experiment, we rotated the source point set 50° around the y axis and translated it with a vector of $[0.1, 0.2, 0.3]$, as illustrated in the first row of Fig. 5. In the second one, we rotated the point set 180° around the z axis without translation, as shown in the second row of Fig. 5. The same initial alignment is used for each algorithm in both cases. We repeated both experiments 40 times. The best results are shown in Fig. 5. PCL-ICP and FGR show entirely different behaviors in these two cases, while they failed with a more than 0.1 RMSE value in the second case. For these two algorithms, we conducted further experiments with different rotations. FGR failed with a high probability for high rotation values and we therefore conclude, that FGR is not rotation invariant. SAC-IA-ICP showed no significant difference in both experiments, achieving roughly the same mean value of 0.009. Gl.RANSAC also showed similar performance in both cases with mean values of 0.003 and 0.006. In this experiment, rotation and translation showed no significant effect in our GPIS-S2SPR approach, with an approximate RMSE of 0.002.

C. Noise and Overlap Robustness

We evaluated the algorithms on all four point sets with the number of points varying from 30 000 to 50 000. Furthermore, we applied three different levels of noise based on a Gaussian distribution with variances set to 0, 0.00025, and 0.0005, respectively. We also evaluated the capability of point registration in a partially overlapping scenario, where only a subset of the points from the source point cloud is used for the target point cloud. Three different overlap factors were used in the experiments: 100%, 85%, and 65%. Furthermore, we used an identity matrix for the initial alignment in each test to maintain identical conditions. Each algorithm was executed 40 times for each configuration, leading to a total of 1440 times for all possible combinations.



(a) Source/Target (b) PCL-ICP (c) SAC-IA-ICP (d) FGR (e) Gl.RANSAC (f) GPIS-S2SPR

(g) Source/Target (h) PCL-ICP (i) SAC-IA-ICP (j) FGR (k) Gl.RANSAC (l) GPIS-S2SPR

Fig. 5: Stanford Bunny together with alignment results of selected algorithms for (a)–(f) 50° rotation around the y axis and small translation, (g)–(l) 180° rotation around the z axis with translation set to zero.

TABLE I: Benchmark results for all algorithms on four different point sets with three levels of Gaussian noise and three different overlap factors. The best RMSE value ϵ for each configuration is highlighted in *green*.

		noise = 0.00000						noise = 0.00025						noise = 0.00050					
		1.00		0.85		0.65		1.00		0.85		0.65		1.00		0.85		0.65	
		ϵ	t	ϵ	t	ϵ	t	ϵ	t	ϵ	t	ϵ	t	ϵ	t	ϵ	t	ϵ	t
Bunny [26]	PCL-ICP	0.081	0.5	0.090	0.4	0.067	0.3	0.046	0.4	0.082	0.6	0.046	0.3	0.069	0.4	0.107	0.6	0.073	0.3
	GoICP	0.115	20.3	0.110	20.1	0.097	20.0	0.046	20.1	0.024	20.1	0.063	20.2	0.089	20.1	0.101	20.1	0.046	20.1
	GoICPT	0.100	21.5	0.108	21.5	0.099	21.4	0.106	21.5	0.104	21.7	0.098	21.4	0.103	21.4	0.109	21.7	0.103	21.4
	SAC-IA-ICP	0.001	6.1	0.002	5.6	0.010	7.1	0.001	6.6	0.002	6.1	0.011	8.2	0.001	7.0	0.003	6.4	0.011	7.9
	Gl.RANSAC	0.001	1.6	0.005	1.7	0.005	2.3	0.001	1.7	0.005	1.8	0.005	1.9	0.001	1.7	0.005	1.9	0.005	2.3
	FGR	0.017	0.4	0.009	0.4	0.017	0.3	0.004	0.4	0.007	0.4	0.020	0.3	0.004	0.4	0.007	0.4	0.015	0.3
	GPIS-S2SPR	0.001	0.6	0.001	0.5	0.001	0.5	0.001	0.5	0.001	0.6	0.002	0.5	0.001	0.5	0.002	0.6	0.002	0.7
Suzanne [28]	PCL-ICP	0.134	0.8	0.108	0.8	0.116	0.7	0.049	0.6	0.127	1.3	0.095	0.6	0.131	0.9	0.115	0.6	0.106	0.9
	GoICP	0.089	23.5	0.055	21.6	0.086	22.0	0.089	21.8	0.081	21.8	0.089	21.8	0.064	21.8	0.072	21.7	0.117	21.5
	GoICPT	0.092	23.2	0.059	21.5	0.084	21.5	0.079	21.6	0.071	21.7	0.097	21.7	0.045	21.5	0.091	21.6	0.062	21.5
	SAC-IA-ICP	0.001	11.2	0.005	10.6	0.018	10.3	0.001	11.7	0.006	10.7	0.027	11.4	0.001	12.4	0.006	11.5	0.018	11.1
	Gl.RANSAC	0.014	1.7	0.018	1.7	0.040	2.0	0.016	1.8	0.011	1.8	0.039	2.3	0.013	1.9	0.025	1.9	0.022	2.7
	FGR	0.049	0.6	0.039	0.6	0.060	0.5	0.070	0.6	0.049	0.6	0.061	0.5	0.046	0.7	0.071	0.6	0.052	0.6
	GPIS-S2SPR	0.003	0.8	0.002	1.1	0.002	1.1	0.001	4.3	0.002	2.1	0.002	1.8	0.002	1.3	0.003	1.2	0.002	1.6
Dragon [26]	PCL-ICP	0.087	0.5	0.100	0.3	0.079	0.4	0.090	0.2	0.065	0.3	0.096	0.4	0.071	0.5	0.082	0.5	0.101	0.4
	GoICP	0.017	21.6	0.022	21.7	0.018	21.4	0.034	21.5	0.035	21.7	0.021	21.5	0.018	21.5	0.053	21.5	0.033	21.6
	GoICPT	0.022	21.4	0.014	21.4	0.021	20.0	0.011	19.9	0.009	20.0	0.084	20.1	0.013	20.1	0.044	20.2	0.017	20.2
	SAC-IA-ICP	0.001	5.9	0.003	5.3	0.009	4.8	0.001	6.0	0.003	5.4	0.009	4.9	0.001	6.5	0.004	5.8	0.009	5.2
	Gl.RANSAC	0.001	2.1	0.005	2.6	0.005	2.9	0.001	2.3	0.005	2.5	0.004	2.8	0.001	2.6	0.005	2.5	0.005	2.8
	FGR	0.012	0.5	0.022	0.5	0.024	0.4	0.012	0.5	0.016	0.4	0.017	0.4	0.021	0.5	0.015	0.5	0.013	0.4
	GPIS-S2SPR	0.002	0.7	0.002	0.9	0.002	1.0	0.002	0.8	0.002	0.9	0.003	1.0	0.002	0.7	0.002	1.0	0.003	0.9
Happy Buddha [26]	PCL-ICP	0.094	0.2	0.086	0.4	0.110	0.3	0.095	0.5	0.076	0.2	0.071	0.2	0.124	0.4	0.047	0.4	0.064	0.4
	GoICP	0.043	21.5	0.032	21.4	0.085	21.3	0.075	21.4	0.050	21.3	0.052	21.3	0.051	21.4	0.067	21.5	0.012	21.4
	GoICPT	0.013	20.2	0.025	20.1	0.028	20.2	0.016	20.2	0.031	20.2	0.022	20.1	0.023	20.0	0.031	19.9	0.020	20.0
	SAC-IA-ICP	0.001	5.5	0.007	5.2	0.009	6.3	0.001	6.2	0.014	7.3	0.017	7.4	0.001	6.7	0.011	7.5	0.015	6.2
	Gl.RANSAC	0.002	1.1	0.005	1.3	0.006	1.8	0.002	1.0	0.005	1.4	0.006	1.6	0.002	1.1	0.005	1.5	0.005	1.8
	FGR	0.022	0.4	0.022	0.4	0.019	0.3	0.016	0.4	0.019	0.4	0.015	0.3	0.024	0.4	0.022	0.4	0.021	0.3
	GPIS-S2SPR	0.003	1.0	0.002	0.8	0.002	0.8	0.001	0.6	0.003	0.7	0.002	0.8	0.002	0.7	0.002	0.7	0.003	1.2

TABLE II: Mean RMSE over all noise levels and overlap factors for all point sets and algorithms. The best result is highlighted in *green*.

Data	PCL-ICP	GoICP	GoICPT	SAC-IA-ICP	Gl.RANSAC	FGR	**GPIS-S2SPR**
Bunny	0.073	0.077	0.103	0.005	0.004	0.011	0.001
Suzanne	0.109	0.083	0.076	0.009	0.022	0.055	0.002
Dragon	0.086	0.028	0.013	0.004	0.004	0.017	0.002
Buddha	0.085	0.052	0.052	0.008	0.004	0.020	0.002
Mean	0.088	0.060	0.061	0.007	0.009	0.026	0.002

The results with individual RMSE values ϵ and runtime t for each configuration are listed in Table I. The algorithms GoICP and GoICPT (10% trimming) consistently showed the worst performance in all test cases with regard to the mean value of RMSE and total computation time. The BnB algorithm in these algorithms is very expensive to compute and the constant switch between ICP and BnB was not able to achieve a global optimum solution. PCL-ICP was able to converge very fast but is susceptible to local minima. SAC-IA-ICP achieved the best performance in case of an overlapping factor of 100% with an RMSE of 0.001. However, the RMSE value increased drastically to 0.01 in case of a partial overlap, Gl.RANSAC explores the corresponding points in terms of FPFH and showed similar performance. FGR converged very fast but showed a bad performance in all experiments for the reasons explained in Section VI-B. Our GPIS-S2SPR algorithm does not rely on identifying corresponding points and is therefore stable for different transformations. In this experiment, the RMSE for GPIS-S2SPR is stable in terms of the overlapping factor and noise for all point sets. As it is able to consider noise in its formulation, the RMSE is approximated equal to 0.002 for all noise levels. Table II shows the total RMSE computed over all possible combinations for each point set and algorithm. GPIS-S2SPR showed the best overall performance. In terms of computation time, it is not always the fastest approach, since GPIS is time-consuming due to the computation of an inverse of the covariance matrix with a complexity of $\mathcal{O}(m^3)$.

D. Gearbox Assembly Application

We explore the capability of our algorithm by using real point cloud data captured by the system shown in Fig. 1b,

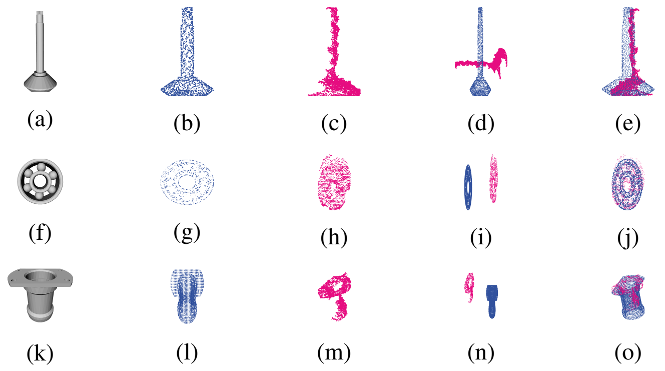


Fig. 6: The first column shows the CAD models of the gearbox components tree, bearing, and pipe. The second one shows the mesh sampled point cloud from the CAD model for each component. The cluster extracted from the point cloud scene (Fig. 1b) is shown in column three. Column four shows the initial source and target point cloud. The alignment results of GPIS-S2SPR are shown in the last column.

where the task is to assemble a mechanical gearbox. In order to grasp the objects on the table with the attached parallel gripper, we require a 6D pose transformation.

The gear box assembly task consists of four parts, a mechanical tree (Fig. 6a), two ball bearings (Fig. 6f), and a mechanical pipe (Fig. 6k). All parts are available in the form of CAD models. For the 6D pose estimation, we apply mesh sampling to acquire a detailed point cloud (PC) for each CAD model, shown in Fig. 6b, 6g, and 6l. The noisy point cloud data shown in Fig. 6c, 6h, and 6m is extracted from the actual data captured by the camera sensor as demonstrated in Fig. 1b. We combine the Euclidean Cluster Extraction and Region growing segmentation from PCL [25] to extract each component from the point cloud scene. The initial position relationship between the source and target point cloud is shown in the fourth column of Fig. 6, where the tree is lying on the table and the pipe is rotated by 90°. As only one 3D camera is used, we can only get a partial view of the objects. The noise added by the camera sensor is not a Gaussian distribution. We evaluate our algorithm with these three components, the results are shown in the last column of Fig. 6. Although the tree and ball bearing are two highly symmetrical components, the algorithm can match the bottom and the upper part with an error of 0.004 In the case of the pipe alignment, the pipe cluster has two disconnected parts and only an approximated 25% of object information is available, which further increases the complexity. Our algorithm can match the objects with an error of 0.009

E. Evaluation with Scanned Datasets

To further verify our algorithm, we evaluate the point sets from semantic-8 [29] and Urban Scenes Velodyne Point Cloud Dataset [30]. The corresponding results are shown in Fig. 7. We compare our algorithm with PCL-ICP in Fig. 7a, where the RMSE of PCL-ICP is 48 times that of our algorithm. In Fig. 7b–7h, each sub-figure consists of two images, where the left one is the initial state, and the right one is the result of point registration. It can be seen that our algorithm can work in different scenarios, such as urban scenes [30] and different kinds of buildings. Furthermore, we evaluate our algorithm with two additional point sets from [27] and Shapenet [31], which are shown in Fig. 8. The source point sets in Fig. 8 are indicated as blue points and the target point sets as orange points. The initial setting for source and target point sets are demonstrated in Fig. 8a, 8b and 8c. From Fig. 8d, 8e, and 8f, we can see that the alignment accuracy is very high in both point sets with an RMSE value of 0.002, 0.0001, and 0.0001, respectively.

VII. Conclusion

We propose a new algorithm for a partially overlapping 3D surface registration algorithm. In this algorithm, we abandon the traditional idea of point to point or point to plane correspondence search to register the points. Instead, we view the 3D surface as a Gaussian Process Implicit Surfaces, which utilizes the signed distance function to describe three manifolds. Furthermore, we convert the point registration as a nonlinear least-squares problem to find a rigid transformation between two point sets. For accelerating the optimization process, we use a Principal Component Analysis (PCA) together with Fast Point Feature Histograms descriptors to compute the initial transformation. Moreover, we derive the Jacobian matrix by applying the Lie algebra perturbation method, which approximated the kernel function with the first-order Taylor series. The whole optimization follows the principle of Gauss-Newton algorithm. By slightly adapting the Jacobian matrix with a damping value, we can convert the algorithm to Levenberg-Marquardt solver. Our approach demonstrated a higher accuracy performance and more robust rotation invariant properties compared to state-of-the-art methods by evaluating diverse experiments.

References

[1] A. Perzylo, M. Rickert, B. Kahl, N. Somani, C. Lehmann, A. Kuss, S. Profanter, A. B. Beck, M. Haage, M. R. Hansen, M. T. Nibe, M. A. Roa, O. Sornmo, S. G. Robertz, U. Thomas, G. Veiga, E. A. Topp, I. Kesslar, and M. Danzer, "SMErobotics: Smart robots for flexible manufacturing," *IEEE Robotics and Automation Magazine*, vol. 26, no. 1, pp. 78–90, 2019.

[2] M. Tenorth, A. Perzylo, R. Lafrenz, and M. Beetz, "The RoboEarth language: Representing and exchanging knowledge about actions, objects and environments," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation*, May 2012, pp. 1284–1289.

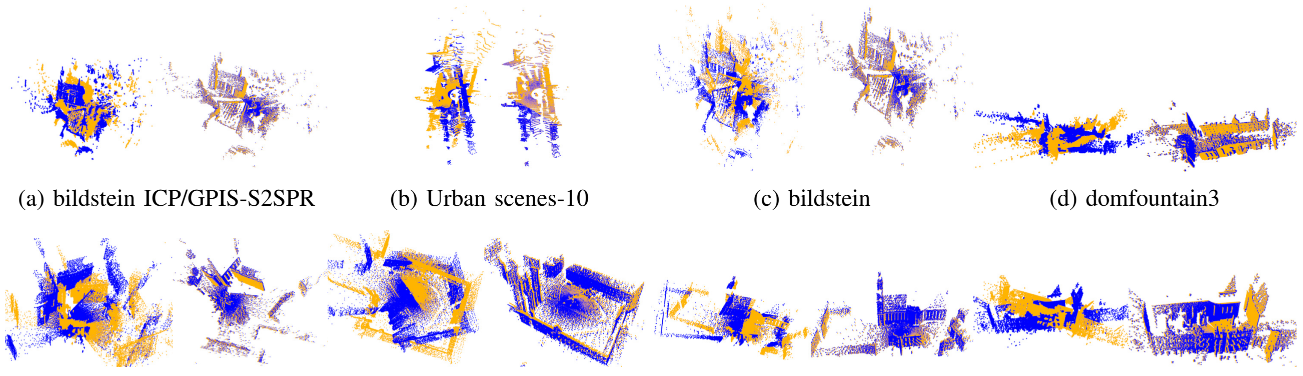
[3] A. Perzylo, N. Somani, M. Rickert, and A. Knoll, "An ontology for cad data and geometric constraints as a link between product models and semantic robot task descriptions," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, Hamburg, Germany, Sept. 2015, pp. 4197–4203.

[4] J. Feldmar and N. Ayache, "Rigid and affine registration of smooth surfaces using differential properties," in *Proc. of the European Conf. on Computer Vision*, 2005, pp. 397–406.

[5] F. Pomerleau, F. Colas, and R. Siegwart, "A review of point cloud registration algorithms for mobile robotics," *Foundations and Trends in Robotics*, vol. 4, no. 1, pp. 1–104, 2015.

[6] S. Granger and X. Pennec, "Multi-scale EM-ICP: A fast and robust approach for surface registration," in *Proc. of the European Conf. on Computer Vision*, 2002, pp. 418–432.

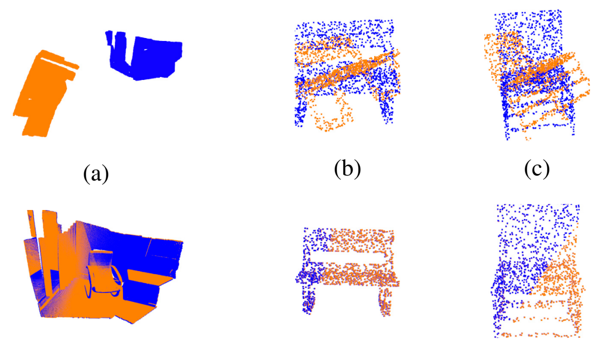
[7] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Proc. of Robotics: Science and Systems*, Seattle, USA, 2009.



(a) bildstein ICP/GPIS-S2SPR (b) Urban scenes-10 (c) bildstein (d) domfountain3

(e) stgallencathedral6 (f) stgallencathedral1 (g) stgallencathedral3 (h) neugasse

Fig. 7: Comparison between PCL-ICP and GPIS-S2SPR with examples from the large scale point cloud classification dataset Semantic-8 [29] and Urban Scenes Velodyne Point Cloud Dataset [30]. In (a), the alignment result of PCL-ICP is on the left and the result of GPIS-S2SPR on the right. From (b)–(h), left shows the initial pose of source and target point set, while right shows the result of applying GPIS-S2SPR.



(a) (b) (c)

(d) (e) (f)

Fig. 8: Evaluation of GPIS-S2SPR with additional scanned point sets from [27] and Shapenet [31]. The first row shows the initial setting for source and target pointsets. The second row shows the alignment results.

[8] A. Krull, E. Brachmann, F. Michel, M. Ying Yang, S. Gumhold, and C. Rother, "Learning analysis-by-synthesis for 6D pose estimation in RGB-D images," in *Proc. of the IEEE Intl. Conf. on Computer Vision*, 2015, pp. 954–962.

[9] H. Zhao, "A fast sweeping method for Eikonal equations," *Mathematics of Computation*, vol. 74, no. 250, pp. 603–627, 2005.

[10] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP variants on real-world data sets," *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, 2013.

[11] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation*, 2009, pp. 3212–3217.

[12] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan, and N. M. Kwok, "A comprehensive performance evaluation of 3D local feature descriptors," *International Journal of Computer Vision*, vol. 116, no. 1, pp. 66–89, 2016.

[13] R. Raguram, J.-M. Frahm, and M. Pollefeys, "A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus," in *Proc. of the European Conf. on Computer Vision*, 2008, pp. 500–513.

[14] J. Yang, H. Li, D. Campbell, and Y. Jia, "Go-ICP: A globally optimal solution to 3D ICP point-set registration," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 38, no. 11, pp. 2241–2254, 2016.

[15] A. W. Fitzgibbon, "Robust registration of 2D and 3D point sets," *Image and Vision Computing*, vol. 21, no. 13-14, pp. 1145–1153, 2003.

[16] Q. Li, R. Xiong, and T. Vidal-Calleja, "A GMM based uncertainty model for point clouds registration," *Robotics and Autonomous Systems*, vol. 91, pp. 349–362, 2017.

[17] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 32, no. 12, pp. 2262–2275, 2010.

[18] C. Papazov and D. Burschka, "Stochastic optimization for rigid point set registration," in *Proc. of the Intl. Symposium on Visual Computing*, 2009, pp. 1043–1054.

[19] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," in *Proc. of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, USA, 2018.

[20] Y.-P. Hu and T.-C. Sun, "Moving a B-spline surface to a curve—A trimmed surface matching algorithm," *Computer-Aided Design*, vol. 29, no. 6, pp. 449–455, June 1997.

[21] J. Duchon, "Splines minimizing rotation-invariant semi-norms in Sobolev spaces," in *Constructive Theory of Functions of Several Variables*, ser. Lecture Notes in Mathematics. Springer, 2006, vol. 571, pp. 85–100.

[22] O. Williams and A. Fitzgibbon, "Gaussian process implicit surfaces," in *Proc. of the Workshop on Gaussian Processes in Practice*, Bletchley Park, UK, Apr. 2007.

[23] M. Li, K. Hang, D. Kragic, and A. Billard, "Dexterous grasping under shape uncertainty," *Robotics and Autonomous Systems*, vol. 75, pp. 352–364, Jan. 2016.

[24] A. Kirillov, Jr, *An introduction to Lie groups and Lie algebras*, ser. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2008.

[25] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation*, 2011.

[26] G. Turk and B. Mullins, "Large geometric models archive," https://www.cc.gatech.edu/projects/large_models/, Mar. 2019.

[27] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847 [cs.CV]*, 2018.

[28] *Blender – A 3D Modelling and Rendering Package*, Blender Foundation, 2019. [Online]. Available: http://www.blender.org

[29] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys, "SEMANTIC3D.NET: A new large-scale point cloud classification benchmark," in *Proc. of the ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. IV-1-W1, 2017, pp. 91–98.

[30] K. Lai and D. Fox, "Object recognition in 3D point clouds using web data and domain adaptation," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 1019–1037, 2010.

[31] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, *et al.*, "ShapeNet: An information-rich 3D model repository," *arXiv:1512.03012 [cs.GR]*, 2015.