STAG: Smart Tools and Applications in Graphics (2023)
K. Lupinetti, F. Banterle, N. Capece, and U. Erra (Editors)

GPU-Accelerating Hierarchical Descriptors for Point Set Registration

S. Dutta[1,2], B. Russig[1], and S. Gumhold[1]

[1]Technische Universität Dresden, Germany
[2]CNR-ISTI, Pisa, Italy

Abstract

We present a GPU-accelerated global registration method for registering partial shapes, a common and often performance-critical task in many robotics, vision, and graphics applications. Global registration based on descriptor matching is highly dependent on the quality at which a shape is sampled, and computing expressive descriptors typically incurs high computation time. In this paper, we augment a global pair-wise registration algorithm based on hierarchical shape descriptors with a GPU-accelerated descriptor construction process, reducing the time spent on building descriptors by an order of magnitude. This allows for building more expressive descriptors, achieving a dual gain in both performance and accuracy. We conducted extensive evaluations on a large set of pair-wise registration problems, demonstrating very competitive registration accuracy, often rendering subsequent refinement with a local method unnecessary.

CCS Concepts

• ***Computing methodologies*** → *Computer Graphics;Shape Modeling; Point-based models;*

1. Introduction

3D sensors are becoming ubiquitous and affordable, including various types of 3D laser scanners, and RGB-D cameras (Microsoft Kinect, Intel RealSense, Apple Truth Depth Cameras) [YLX*19]. These sensors acquire 3D shape information to varying levels of accuracy along with additional attributes, for instance, color, etc. The raw point clouds acquired by the sensors essentially require an alignment step that brings together individual scans to a common frame of reference and therefore synthesize a complete model or a large-scale scene. Precisely, it implies estimating transformation matrices between a pair of scans and simultaneously applying the estimated transformation resulting in a complete model. Traditionally the task of point cloud registration has been of utmost importance in numerous computer vision applications, and with recent developments in sensor technologies, has drastically diversified. The accuracy of this registration step is often impaired due to low overlap between scans, inherent noise of the environment and sensors, spatial extent, and sparsity of the scans resulting in registration artifacts(oscillation or even holes).

Among various forms of registration, the same source pairwise global registration, i.e.coarse registration is fundamental to the task of multi-view registration and importantly local refinements (e.g. the iterative Closest Point, (ICP) [BM92; RL01a] relies on a coarse alignment for faster convergence and higher accuracy.

Global registration algorithms generally follow a two-stage workflow, determining correspondences followed by transformation estimation. The correspondence-establishing stage consists of keypoint detection [SOG09; SB11; WZWL20], feature descriptors [JH99; PD15; QYW*22] descriptors matching, and finally some outlier filtering [FB81; SAH19]. Correspondences between descriptors are established based on similarity, resulting in a rigid transformation that best aligns the set of feature points and using some robust estimator. The widely used approach to attain pose invariance deploys a Local Reference Frame (LRF) centered on the feature point and attached to the surface regardless of its orientation. Thereby, descriptors can encode the local or global geometric information from the data with respect to a canonical reference associated with the feature point.

Although effective and fast algorithms pertaining to their computation have been devised in the field of registration, these descriptors are majorly impacted by sampling discrepancy of individual scans and measurement noise. Additionally, methods relying on localized feature descriptors often suffer from a lack of generality(works well with objects of a given topology), a certain percentage of false matches, an outcome out-rightly caused by insufficient discriminating capabilities of the descriptor and/or an underperforming similarity measure. An amalgamation of such factors creates a significant impact on the overall registration accuracy and convergence of the local iterative algorithm.

This paper is motivated by the goal of designing an approach that can solve the registration problem of partial, low-overlap shapes globally based on efficient descriptors. We rely on the continuous surface representation-based hierarchical descriptor proposed by Dutta et al. [DRG22], itself based on the registration

DOI: 10.2312/stag.20231295

pipeline and low-overlap similarity measure introduced by Ferrero et al. [TGA*12]. However, constructing these descriptors at higher resolutions quickly becomes prohibitively expensive, limiting their expressiveness and thus the achievable alignment accuracy. We overcome this performance bottleneck by mapping the most expensive part of descriptor computation to the programmable rasterization pipeline of modern GPUs. Our contribution can be summarized as follows:

- a conceptually effortless method of leveraging GPU hardware for the computation of hierarchical shape descriptors
- based on that, a modified pair-wise global registration pipeline (`GPU-HER`) that features
  - Fast processing times
  - High precision that compares favorably to state-of-the-art global methods, making local refinement unnecessary in most cases or at least giving close-to optimal initial guesses.

Additionally, the paper contributes a broad evaluation of the proposed approach and compares it to previous hierarchical descriptor-based approaches [TGA*12; DRG22] as well as the state-of-the-art global registration method by [ZPK16], demonstrating measurable improvements in terms of speed, accuracy and robustness.

2. Related Work

In this section, we present an overview of the existing methodologies concerning the point cloud registration algorithm. We refer the reader to the respective surveys [RL01b], Salvi et al. [SMFF07], Tam et al. [TCL*13], Bellekens et al. [BSM14], Huang et al.[HMZA21], Li et al. [LWZ21] for registration algorithms over the years. In accordance with the aforementioned surveys, registration methods can be classified based on the following criteria.

Pairwise/Multi-Way. Pairwise registration methods [BSM14] are focused on aligning two partially overlapping scans, a category into which our proposed method also fits. To achieve comprehensive dataset registration, scans can be progressively incorporated and aligned with either their predecessors or all previous scans. In contrast, multi-way registration [ZPK16] involves optimizing all transformations simultaneously. This is often done by iteratively performing pairwise registrations or by formulating the problem with a single registration objective.

Rigid/Non-rigid. Rigid registration techniques [BSM14] exclusively permit rigid body transformations for individual scans, while non-rigid methods provide flexibility for arbitrary transformations. Nonetheless, non-rigid approaches [DYDZ22] often incorporate regularization into the objective function to prevent degenerate solutions (e.g., by requiring the transform to maintain as much rigid as possible). However, a detailed discourse on non-rigid approaches is outside the purview of this paper.

Local/Global. Local methods [BM92; BTP13; POD*18] use an initial transform estimation (e.g. provided by the user) and refine this transformation to optimize an objective. To date, the classic iterative nearest point (ICP) [BM92] remains the most widely-used point cloud matching algorithm. Many researchers [BTP13; KYOB21] have made improvements to the ICP algorithm, which relies on a good initial alignment position. Without this, it quickly falls into the local optimum and cannot achieve a good registration accuracy. Contrarily, global methods [AGV*14; ZPK16; PD15] necessitate only the geometry as input and determine registration transforms regardless of initial alignment. While many global methods, including those mentioned, rely on descriptor matching for rigid motion parameter estimation, they typically generate coarse registrations that can subsequently be refined using local methods. Importantly, global methods are distinct in that they do not rely solely on spatial proximity for registration, often incorporating geometric feature descriptors or similar approaches.

Learning-based Registration. Data-driven approaches have become increasingly popular for point cloud registration in recent years, thanks to their remarkable performance in both 2D and 3D applications. In [ZF23], a detailed survey of deep-learning (DL) methods for global registration algorithms is meticulously presented. This survey tabulates the performance of these methods across various benchmark datasets. Furthermore, feature learning methods [ZSN*17; DBI18; HGU*21] utilize deep neural networks (NNs) to learn robust feature correspondence search. Following this step, the transformation matrix is obtained using one-step estimation techniques, such as SVD or RANSAC, eliminating the need for iterative processes. While these NNs can indeed deliver sturdy and precise correspondence searches, their performance is often hindered by the availability of sizeable training data and limited generalization capabilities. As a result, they encounter challenges when dealing with unknown scenes that exhibit distributions differing from the training data. Conversely, end-to-end learning-based approaches [CCG*22] tackle registration challenges with an all-encompassing network, incorporating both correspondence search and transformation estimation within a unified framework. This stands in contrast to feature learning methods, which are primarily centered around point feature learning.

Descriptors. Registration algorithms predominantly, global methods, rely on 3D descriptors, and as a result, substantial research is dedicated to devising efficient and robust geometric descriptors. An in-depth evaluation of point cloud descriptors, covering aspects of computational efficiency and accuracy, is provided in references [Ale12; GBS*16; HJX*18; RX20]. Point cloud descriptors can be broadly classified based on their descriptor extraction process into local, global, and hybrid categories. Local descriptors such as those proposed in [RBB09; TSD10] capture spatial distribution or geometric attributes in the vicinity of each point. These descriptors are computed by generating histograms that encapsulate information about the descriptors for individual points. While local descriptors are effective for cluttered scenes, they may lack the highest degree of discriminatory capabilities. An example of a widely used local descriptor is the Spin Image [JH99]. It constructs a cylindrical region around a designated keypoint, dividing it into radial and vertical volumes. The descriptor then tallies the number of points residing within each of these volumes. Conversely, global descriptors are computed for the entire point cloud, yielding a single context vector by aggregating a collection of features or incorporating the spatial distribution of the entire dataset. Global-based features, as observed in [DRG22], find applications in 3D object recognition and categorization. Some global descriptors can also be employed for object pose estimation, such as the Viewpoint Feature Histogram (VFH) [RBTH10]. Hybrid descriptors, as discussed in

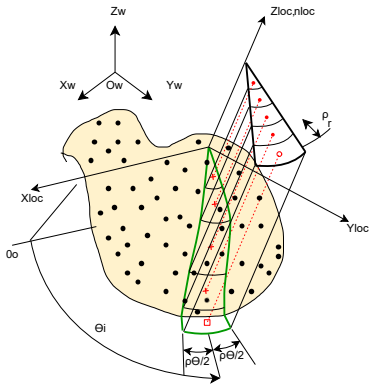


Figure 1: *Circon descriptor [TGA*12] of a continuous surface fitted to a point cloud [DRG22], spawned by a keypoint defining a local reference frame with cell division into sectors represented in green. The red crosses mark the surface points from which the cell values are measured. The continuous surface makes descriptors largely independent from the point cloud-imposed sampling.*

[OOFB08], merge local and global information to create a more complete surface description. In summary, the selection of a suitable descriptor depends on the specific application context, computational efficiency, and the capacity to accurately distinguish between point cloud features.

3. Methodology

Background. The proposed method relies on the core concept of descriptor matching in order to estimate the rigid transformation. Our approach uses the *Circon* descriptor proposed by Ferrero et al. [TGA*12] which represents an ordered set of radial contours around some point-of-interest within a point cloud. The descriptor associated with a particular point-of-interest is used to express the point cloud in a local reference frame.

To build a descriptor, the entire point cloud is divided into sectors, each representing an angle ρ_θ, with the point-of-interest in the center, and each sector is further divided radially into cells with length ρ_r. By convention, the descriptor cells are indexed by a pair of 0-based numbers i (row-index) and j (column-index) for the angular and radial directions, respectively.

Surface points are mapped into cells of the descriptor in a way that the cell-value represents the height (z-value) with respect to the local reference frame. Figure 1 depicts a descriptor associated with a point-of-interest (cf. section 3.1) encoding the structural information from a shape.

Ferrero et al. also tailored a global registration scheme for this descriptor, which finds the best alignment by tracing a hierarchy of successively finer descriptors. The way they use the point cloud directly to build the descriptors, however, ties the achievable accuracy to the point cloud resolution. Dutta et al. [DRG22] addressed this shortcoming by working with a continuous surface representation instead, enabling higher-resolution descriptors that can capture more surface features while compensating for under-sampled regions. The major drawback of this method is that sampling descriptor cell values from the continuous surface is expensive, as the inverse problem of finding a particular surface point corresponding to a given descriptor cell needs to be solved, which is performed using an elaborate ray-casting scheme.

GPU acceleration. We propose a new method for descriptor computation that naturally leverages GPU hardware. This enables us to build descriptors (section 3.2) of much higher resolution with little performance impact, combining a dual benefit of higher final alignment accuracy and an immense speed-up of overall descriptor computation time. The key idea is to tessellate the surface once at a sufficiently high fidelity to capture all important features and rasterize the resulting mesh from the local frame of the descriptor we are about to compute using an orthogonal projection. The resulting depth map can then easily be sampled in a forward fashion to obtain descriptor cell values.

We retained *non-uniform rational B-splines* (NURBS) for the continuous surface representation as suggested by [DRG22] on the account that there are now no performance considerations offsetting their advantages (ability to exactly represent relevant algebraic shapes, their use in CAD to design the real objects we are likely to scan), including the parameter-space trimming curve fitted to the silhouette of the point cloud to represent the shape border. Our proposed modified algorithm can be summarized as follows:

AL.1 (pre-processing) Fit NURBS surfaces [MV13] and trimming curves (implemented in PCL [RC11], default parameters) to the source and target point clouds. Euclidean clustering [Rus10] (see **Appendix A**: section A.2.1) on both clouds accounts for highly fragmented scans, and each cluster is fitted with its own surface. We refer to the union of all surfaces fitted to a point cloud as *the* surface of the point cloud.

AL.2 (pre-processing) Tessellate both surfaces and evaluate and store per-vertex *trimmed* status (section 3.2)

AL.3 (pre-processing) Point-of-interest (*poi*) selection for both surfaces (section 3.1).

AL.4 (pre-processing) Initial sorting of *poi* (section 3.4) pairs at a coarse resolution according to their similarity score (section 3.3), so step AL.5 can start with the most promising pairs.

AL.5 Traverse through the descriptor pyramid (section 3.4) for a pair of *poi* from source and target surface. Descriptors for the next level are computed on-the-fly using our GPU-accelerated method (section 3.2).

AL.6 Estimate transformation from point-pairs with the highest similarity score at maximum resolution (user-defined) and evaluate stopping criterion.

AL.7 Repeat from AL.5 until the stopping is criterion satisfied.

The computation of descriptors is performed potentially thousands of times as the hierarchies are traced in AL.5, representing the major bottleneck of the method. Our contribution is the speedup of this step via GPU acceleration (section 3.2), but in the interest of providing a comprehensive understanding of the whole method, we reiterate all key aspects of the algorithm in the following sections, in the order that they are required during execution.

3.1. Selecting Points-of-Interest

Points-of-interest for the source and target shapes define where descriptors are initially constructed and should be chosen in a way such that the resulting descriptors cover every location on the shape at least once. Since we are not restricted to points from the original

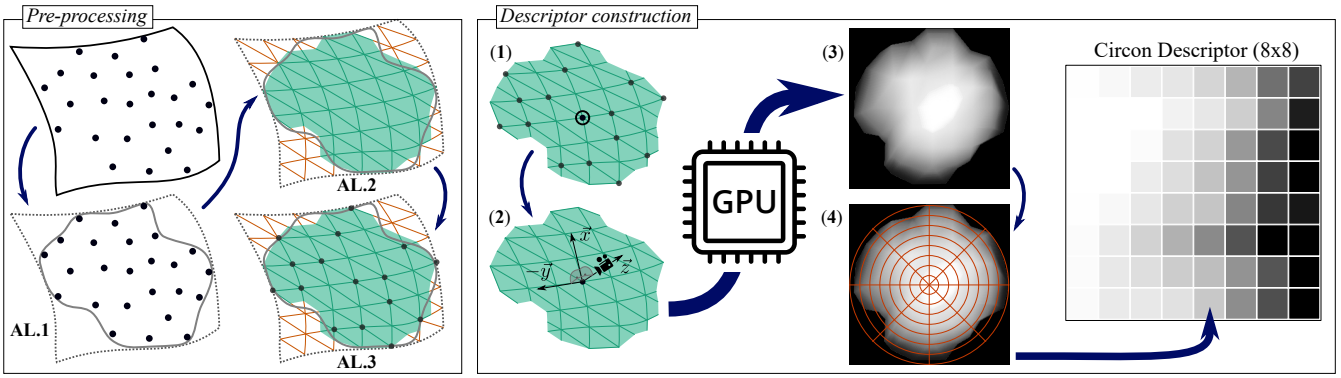


Figure 2: *The GPU-augmented descriptor construction pipeline.* ***Pre-processing****: (**AL.1**) The NURBS surface is fitted to the input point cloud (clustering is omitted for brevity), with the trimming curve fitted in NURBS parameter space. (**AL.2**) The surface is tessellated with per-vertex discard flags stored based on the trimming curve. Here, the intra-triangle trim boundary results from a threshold of $d = 0.5$. (**AL.3**) poi are selected from untrimmed vertices.* ***Descriptor construction*** *(required inside steps **AL.4-5**): (**1**) A poi becomes keyppoint for a descriptor. (**2**) A reference frame tangential to the surface at the keypoint is chosen. (**3**) The surface is rasterized by the GPU from the reference frame into a depth map. (**4**) The depth map is sampled at the descriptor cell centers to compute the cell values.*

point cloud, we will henceforth use the term *keypoint* to refer to any location on a surface around which we build a descriptor. We will refer to those surface locations that we select explicitly to be keypoints for the initial, most coarse descriptors, as *the* points-of-interest, or *poi* for short.

The original registration algorithm by Ferrero et al. [TGA*12] adopts a strategy to select non-edge points by performing edge detection and thresholding based on the Laplacian of the normal vectors, essentially restricting their selection to relatively flat areas. Dutta et al. [DRG22] did not find this to provide measurable benefits, and suggest choosing *poi* based on a random sampling of locations on the respective surfaces instead.

Our GPU-based descriptor construction requires a one-time tessellation of each surface (see section 3.2) – meaning we can just randomly choose a subset of m, n points from the position samples we use for tessellating the source and target surfaces to obtain the sets of points-of-interest $s_{poi} = \{s^i_{poi} \in P_S \mid i = 1..m\}$ and $t_{poi} = \{t^i_{poi} \in P_T \mid i = 1..n\}$, $m < |P_S|$, $n < |P_T|$, where P_S and P_T are the sets of position samples from the tessellation of the source (S) and target (T) surface respectively. In practice, we found $m = n = 100$ *poi* per shape to be sufficient.

3.2. Descriptor Construction

The local reference frame of a Circon descriptor is defined by the normal z_l at the keypoint (queried from the shape NURBS representation), some perpendicular vectors y_l and $x_l = z_l \times y_l$, as well as the keypoint itself as the origin. We refer to the plane with normal z_l that the keypoint lies on as the *reference plane* of the descriptor.

In this regard, the descriptors are identical to [TGA*12; DRG22]. However, for actually constructing descriptors, we propose to render the surface from the descriptor reference frame using the programmable GPU rasterization pipeline, resulting in a depth map that can be efficiently sampled to gather cell values. This gets rid of the major performance bottleneck in [DRG22], where cell values are obtained by ray-casting the transformed NURBS surface using a costly iterative procedure.

For efficient rendering, we first tessellate each NURBS surface into a triangle mesh. The tessellation depends on the surfaces only and remains constant, thus it is done just once during preprocessing. The fidelity of the meshes should be sufficient to reproduce all features of the NURBS surfaces, which in turn we assume to have been fitted with an appropriate amount of smoothing for the input data. We found 1024×1024 vertices to work quite well for most real-life data including high-quality scans while rendering virtually instantaneously (between $500\mu s \sim 1ms$ depending on the GPU). In performance-critical applications, an adaptive tessellation scheme could speed up descriptor construction even more, but would have to be weighed against increased pre-processing times or require hardware support like tessellation [BG21] or mesh shaders [Eri22]. In addition to the vertex positions in their respective *source* or *target* spaces, we also store a discard flag d indicating whether the vertex is outside the trimming curve ($d = 1$) or inside ($d = 0$).

We use an orthographic projection for rasterization to enable efficient lookup of the resulting depth map. Note that the resolution of the depth map (and corresponding frame buffer for rendering) should be high enough for the chosen maximum descriptor resolution. Determining this resolution is not straightforward, since Circon descriptor cells increase in size radially outwards. We use twice the resolution of the most detailed descriptor level, which works well in practice. As a possible enhancement, foveated rendering could be used to better match the radially decreasing spatial resolution of Circon descriptors.

Either way, since the orientation of the descriptor can be arbitrary, the frame buffer can be square irrespective of the actual shapes of *source* and *target*, which is reflected in the projection matrix. The matrix also has a scaling component that ensures the larger one of either *source* or *target* shape can fit into the orthographic frustum. In the case of OpenGL, this matrix is

$$O_{proj} = \begin{bmatrix} \frac{1}{f} & 0 & 0 & 0 \\ 0 & \frac{1}{f} & 0 & 0 \\ 0 & 0 & -\frac{1}{f} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

$$f = \frac{s}{2}, \quad \text{where } s \text{ is the largest shape extent} \tag{2}$$

The vertex shader then transforms each mesh vertex with the matrix

$$M = O_{proj} \cdot {}^{W}\mathrm{T}_{l} \tag{3}$$

where ${}^{W}\mathrm{T}_{l}$ is the sytem transformation into the local frame l of a descriptor from world space W (see section 3.4). The trimming flag d is barycentrically interpolated by the rasterizer and passed on to the fragment shader, where we can discard fragments we deem outside the trimming curve (we use $d = 0.5$ as the threshold).

This concludes the rendering process – for the remaining steps of descriptor construction, the depth map is made visible to host memory for further processing on the CPU, where calculating a descriptor cell value simply requires sampling the depth map at the corresponding pixel coordinates $(i_{\mathrm{D}}, j_{\mathrm{D}})$:

$$\begin{aligned} i_{\mathrm{D}} &= \left\lceil \frac{w}{2} + f\, j\, \rho_r \cos(-(i-1)\,\rho_\theta) \right\rceil \\ j_{\mathrm{D}} &= \left\lceil \frac{w}{2} + f\, j\, \rho_r \sin(-(i-1)\,\rho_\theta) \right\rceil \end{aligned} \tag{4}$$

where (i, j) form the 2D-index of the descriptor cell, ρ_r, ρ_θ are the radial and angular resolutions, and w is the pixel resolution of the $w \times w$ depth map. When the pixel depth $d_z = [0, 1]$ is smaller than one, the cell is valid and we un-project the corresponding point into the descriptor reference frame using the inverse projection O_{proj}^{-1}. Since we are only interested in the z-coordinate of the un-projected point, this reduces to

$$z = -(2d_z - 1)f \tag{5}$$

The actual cell value $c_{i,j}$ can now be obtained by quantizing the z-coordinate with respect to the height resolution ρ_z:

$$c_{i,j} = \lceil \frac{z}{\rho_z} \rceil \tag{6}$$

As the descriptor embeds the entire region around the keypoint, it represents a closed sequence such that the first and last row (each row represents a radial sector as pointed out in Figure 1) are considered adjacent and also the elements with the same column index correspond to an adjacent cell. Hence, the descriptor exhibits a cyclical property that is crucial for matching and determining the rotation parameter of the alignment transformation it represents (see section 3.4).

An overview of the modified, GPU-enabled descriptor construction pipeline is shown in Figure 2.

3.3. Similarity Measure

To compare the source and target descriptors for each resolution, the algorithm employs the tailor-made similarity measure by Ferrero et al. [TLRS09]. This measure allows for finding those keypoints on source and target (and associated orientations) for which the descriptors appear maximally similar.

In point clouds with low overlap regions, it is likely that the geometry around the corresponding keypoints of source and target scans could randomly appear similar, resulting in the selection of false correspondences. Likewise, differences originating from non-overlapping regions should not result in severely lowered similarity. To this end, the measure contains a heuristic for downweighting the contribution of cells deemed non-overlapping. As a result, the approach remains effective in cases of low overlap, making it particularly well-suited for challenging 3D registration tasks in complex environments. The detailed derivation and additional concepts are provided in [TGA*12].

3.4. Multi-resolution Descriptor and Matching

The descriptor computation in Section 3.2 and the similarity measure in Section 3.3 constitute the fundamental building blocks of the hierarchical registration pipeline. Circon descriptors can form a hierarchy, as each cell, in turn, implies another keypoint that spawns a descriptor. By doubling the resolution at each level, a tree of coarse-to-fine representations is formed, with the similarity to the target surface deciding which sub-tree to follow. The points-of-interest determined as per section 3.1 form the roots of such a hierarchy.

The algorithm initially exhaustively matches descriptors at a coarse resolution of 16×16 between the two sets s_{poi} and t_{poi}. The pairs (s^i_{poi}, t^j_{poi}) are sorted in decreasing order of their coarse similarity to obtain starting points for the hierarchical matching of descriptors. The highest-ranked pairs are tried first, avoiding initial match-ups that are unlikely to yield good results. The actual process of descriptor comparison then works on each (s^i_{poi}, t^j_{poi}) pair in isolation until the stopping criterion (section 3.5) indicates a valid alignment. The initial pair of descriptors at some resolution is called the *primary* descriptors for source and target. A search over each cell of the primary source descriptor and their associated keypoints results in a set of *secondary* source descriptors that have their greatest similarity to the target at some row shift k. This row shift represents rotation around the normal of the keypoint that spawned the secondary descriptor, determining the remaining free parameter needed to form a rigid transformation. When the rows of the secondary source descriptor had to be shifted k times for the highest similarity, then the rotation angle is $k\rho_\theta$, where ρ_θ is the angular step at the given resolution.

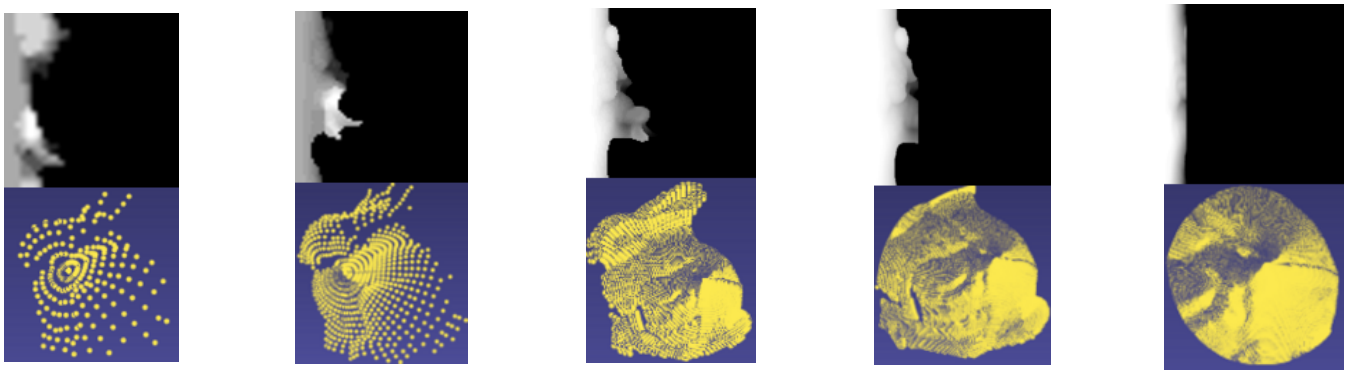
The 3D point and rotation associated with the secondary descriptor that gave the best match at the current resolution become the starting point for the primary descriptor at the next higher resolution. The search space further down in the hierarchy is reduced by halving the percentage of columns (i.e. radial steps away from the contour center) considered at each level. Figure 3 depicts a series of primary descriptors from coarse to fine resolution. The process terminates for a specific (s^i_{poi}, t^j_{poi}) pair once the specified maximum resolution is reached.

The alignment transformation for the current *poi* pair then results from the model transformation ${}^{W}\mathrm{T}_{s}^{-1}$ associated with reference frame of the best-matching secondary source descriptor at the highest resolution, the optimal row shift k and the system transformation ${}^{W}\mathrm{T}_{t}$ associated with the reference frame of the target descriptor:

$$\mathrm{T}_{align} = {}^{W}\mathrm{T}_{s}^{-1} \cdot \mathrm{R}(k) \cdot {}^{W}\mathrm{T}_{t} \tag{7}$$

The transformation ${}^{W}\mathrm{T}_{l}$ for some descriptor reference frame l results from the keypoint p that spawned the descriptor and its associated normal n (see section 3.2). The rotational part of the transformation from W to l is obtained as follows:

$${}^{W}\mathrm{R}_{l} = [\vec{x}_l \quad \vec{n} \times \vec{x}_l \quad \vec{n}]^T \tag{8}$$



(a) 32X32 **(b)** 64X64 **(c)** 128X128 **(d)** 256X256 **(e)** 512X512

Figure 3: *Descriptor cell values mapped to an image at the top and corresponding point cloud represented by the descriptor at the bottom of Bunny model. The restriction of higher resolution descriptors to 64 columns is visible in image (d).*

The full transformation ${}^{W}\mathrm{T}_{l}$ is given by

$$ {}^{W}\mathrm{T}_{l} = \begin{bmatrix} {}^{W}\mathrm{R}_{l} & -{}^{W}\mathrm{R}_{l} \cdot p \\ 0_{1\times 3} & 1 \end{bmatrix} \tag{9} $$

Finally, a rotation matrix parameterized by some row shift k is obtained as follows:

$$ \mathrm{R}(k) = \begin{bmatrix} \cos(\rho_\theta \cdot k) & \sin(\rho_\theta \cdot k) & 0 & 0 \\ -\sin(\rho_\theta \cdot k) & \cos(\rho_\theta \cdot k) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10} $$

Our contribution to GPU-accelerated descriptor construction enables us to extend the hierarchy with much higher-resolution descriptors.

3.5. Stopping Criterion

Ferrero et al. [TGA*12] suggest a sanity check to decide whether the current best match corresponds to a good alignment. To this end, three non-collinear points from the final source and target descriptors are chosen that each form a triangle around the respective Circon descriptor center. Such points can be easily obtained by selecting the same three equidistantly spaced cells from the first column of each descriptor and retrieving their respective surface points. These triangles form a fictitious correspondence pair and the centroid of each triangle defines a local reference frame with fictitious transformation T_{fict} between them. T_{fict} is compared with T_{align} in terms of a delta transformation:

$$ \Delta\mathrm{T} = \mathrm{T}_{align} \cdot \mathrm{T}_{fict}^{-1} \tag{11} $$

The rotational and translation component are separated from (ΔT) to obtain rotational angle (Δr) and translation distance (Δt_s):

$$ \Delta r = \sqrt{\frac{1}{3}(\omega^2 + \theta^2 + \varphi^2)} < \varepsilon_r \tag{12} $$

$$ \Delta t_s = \|\Delta T_s\|_2 < \varepsilon_t \tag{13} $$

ω, θ, φ are the ZYX Euler angles extracted from the rotational component and ΔT_s refers to translation part of ΔT respectively. The threshold ε_r, ε_t in our evaluations are $0.045rad$ and $0.055\bar{r}$, where $\bar{r}$ is the extent of the model. If the rotational difference Δr and translation distance Δt_s extracted from ΔT are smaller than their respective thresholds, the algorithm terminates and T_{align} is returned as the final alignment transformation.

4. Evaluation

4.1. Dataset

The evaluation is conducted using the extensive benchmark dataset provided by Petrelli et al. [PD15], denoted as the LRF dataset. This dataset comprises a variety of objects captured by sensors with differing quality levels, spanning from consumer-grade depth cameras to high-end laser scanners, leading to point clouds of varying point densities. Furthermore, this benchmark includes challenging scenarios with registration pairs exhibiting low overlap.

4.2. Methods

To demonstrate the efficacy of our proposed method, we perform a comparative analysis with ORI-HER [TGA*12] and HER [DRG22], the original precursor methods that employ hierarchical Circon descriptors with a similarity measure for coarse point cloud registration. Note that for our method, we quadrupled the resolution of the finest hierarchy level from 256^2 cells (ORI-HER and HER) to 512^2, expecting that the saved computation time can be spent to achieve higher accuracy.

Additionally, we compared to the state-of-the-art global registration method FGR [ZPK16]. This method has been shown to rival the accuracy of local methods and therefore ideal for comparison with our proposed approach. We were unable to reproduce the performance reported for state-of-the-art deep learning methods like the one by Huang et al. [HGU*21] on the LRF dataset, so we did not include such methods in the comparison.

To ensure fairness, values for each method's parameters were

	No.Pairs	Fine	Coarse	Fail
ORI−HER	4169	139	821	3209
FGR	4169	653	1237	2279
HER	4169	742	1319	2108
GPU−HER	4169	**918**	**1463**	**1788**

Table 1: *Categorization of registration view pairs from LRF datasets based on threshold as discussed in section 5.1*

	Min	Max	Avg	Std.dev
ORI−HER	1.11	29.49	5.43	3.97
FGR	0.33	24.51	3.25	2.53
HER	0.21	22.01	2.38	2.15
GPU−HER	**0.11**	**18.76**	**1.95**	**1.67**

Table 2: *Statistical comparison of RMSE based on Fine threshold as discussed in section 5.1*

chosen according to the defaults suggested by the respective authors wherever applicable (see **Appendix A**).

5. Results

The section presents the results of our experiments, which aim to validate that the proposed approach yields more accurate registration results compared to the state-of-the-art method. Additionally, we will delve into various characteristics of our algorithm. We ran all experiments on an AMD Ryzen 7 5800H clocked at 3.20 GHz with an Nvidia RTX 3060 6GB GPU. The rasterization part of the descriptor construction process is implemented against the OpenGL 3.3 API (Core profile) directly, without using any rendering middleware.

5.1. Accuracy

The accuracy assessment is based on datasets introduced in section 4.1 with multiple registration problems and root mean square error (RMSE) computation on the registration output after convergence. The RMSE measures the mean error over all source cloud point locations for some computed transformation T_{op} with respect to ground truth transformation T_{gt}:

$$\mathrm{RMSE}(T_{op}) = \frac{1}{\overline{d}} \sqrt{\frac{1}{n} \sum_{i=1}^{n} \|T_{op} p_i - T_{gt} p_i\|^2} \tag{14}$$

The normalization factor $\overline{d}$ signifies the sampling distance, determined as the average mesh edge length for the LRF dataset which employs polygon mesh representations for scans. We categorize the root mean square error (RMSE) outcomes into *fine* ($< 20\overline{d}$), *coarse* ($[20\overline{d}, 50\overline{d}]$), and *failed* segments, and evaluate the number of view pairs under each category in comparison to the total registration problems (Table 1). Focusing on the *fine* category, we analyze accuracy results in Figure 6 for the LRF datasets. The findings affirm our method's superior accuracy compared to FGR, ORI−HER, and HER. This conclusion is further supported by the statistical measures in Table 2 that depict our method's consistently lower values considering *fine* RMSE category. The capability to construct high-resolution descriptors from continuous surface representation, enabled by GPU acceleration, contributes to our method's accuracy enhancements. Visual comparisons of registration results in Figure 4 and Figure 5 validate our approach's ability to achieve finer alignment, as opposed to failure or inaccurate alignment exhibited by other methods.

	32×32	64×64	128×128	256×256	512×512
HER	0.023s	0.028s	0.09s	0.15s	0.29s
GPU−HER	**0.002s**	**0.004s**	**0.005s**	**0.008s**	**0.017s**

Table 3: *Average Descriptor Computation at each resolution for HER and GPU−HER*

5.2. Runtime

The initial portion of our runtime analysis centers around the average time taken for a single descriptor computation across various resolutions for the LRF dataset. Presented in Table 3, this comparative data, when juxtaposed with HER, underscores the significant speed-up in descriptor computation offered by our method. This enhancement is notably augmented by GPU acceleration, all the while maintaining accuracy without compromise.

Moving to the second part of the analysis, we delve into the computational time of ORI−HER, FGR, HER, and our approach, taking into account preprocessing steps and the core algorithm applied to models from the LRF dataset. This examination dissects the key components and provides insights into their individual runtime contributions. For instance, in HER and our method, this analysis encompasses surface fitting, coarse resolution computation with sorting, and the iterative core algorithm. As depicted in Figure 7, the initial computation and sorting of coarse resolution pairs incur negligible runtime in contrast to other pivotal stages. The keypoint computation time in ORI−HER is explicitly included in the comparison as it relies on a customized set of steps for their selection as earlier described in section 3.1. The FGR algorithm relies on feature descriptors from a pre-processing step, which tends to be expensive to compute. The time spent computing the alignment in the case of FGR is primarily dedicated to feature matching, which can vary significantly depending on the specific point clouds. Despite this variability, FGR stands out as the fastest method on average. Conversely, ORI−HER exhibits the highest runtime, attributed to its extensive processing of keypoint pairs and descriptor hierarchies in the quest for optimal transformations. The iterative core algorithm of HER has higher computational time as descriptors are comparatively expensive to compute, further restricting their resolution to 256×256 in the analysis. In the case of our proposed method, we observe a significant acceleration in descriptor hierarchy construction and a concurrent reduction in the time required for the iterative core algorithm. This effect stems from our method's capacity to construct high-resolution descriptors across the hierarchy swiftly, enabling convergence within the first few keypoint pairs. This reduces the overall time as illustrated in Figure 7, showcasing the time efficiency of our approach in comparison to its contemporaries.

(a) ORI−HER **(b)** FGR **(c)** HER **(d)** GPU−HER

Figure 4: *Registration of a view-pair From* LRF *Armadillo dataset*

(a) ORI−HER **(b)** FGR **(c)** HER **(d)** GPU−HER

Figure 5: *Registration of a view-pair From* LRF *OilPump dataset*

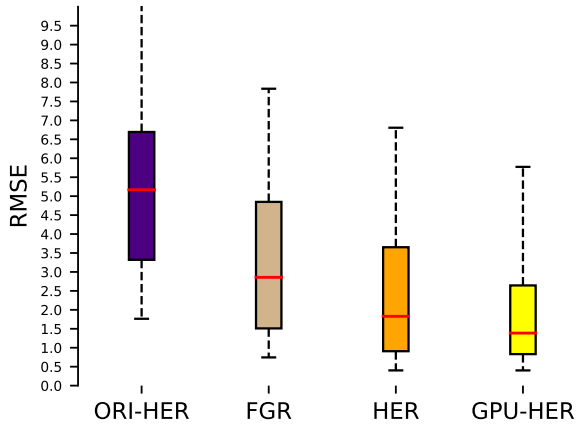


Figure 6: *Accuracy comparison of pairwise registration methods for the* LRF *datasets. The box height represents the 25th and 75th percentile. The whiskers represent the 5th and 95th percentiles. The middle line represents the median. Better registration results are characterized by lower RMSE.*

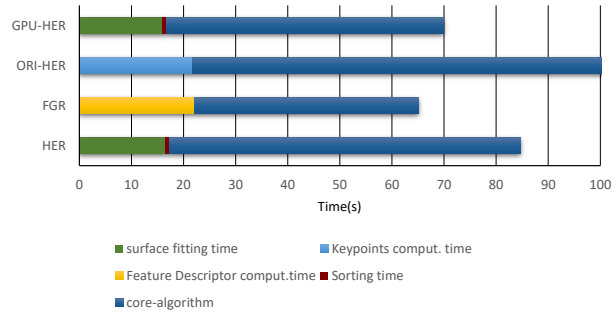


Figure 7: *Runtime comparison of the proposed approach* GPU−HER *with* HER, ORI−HER, FGR *including the preprocessing step for each methods.*

6. Conclusion and Outlook

In this paper, we have presented a hierarchical (coarse-to-fine) global registration method based on GPU-accelerated descriptors. By leveraging GPU hardware, we surmounted the bottleneck of computing high-resolution 2D descriptors from the continuous surface representation. This resulted in a dual efficacy i.e., offering higher accuracy and speed-up. Our evaluation, conducted on the diverse LRF dataset, demonstrates significant improvements in accuracy achieved by employing higher-resolution descriptors, along with a reduction in their computation time by an order of magnitude. Consequently, the overall time of the registration process is greatly reduced.

However, there are several areas that could be improved, still focusing primarily on speed: While we do advocate for NURBS surfaces due to their versatility and availability of implementations, the fitting process is computationally expensive. In the case of range scans, their 2.5-dimensional nature would allow for more optimized fitting algorithms than the generic, PCL-provided one we use. Additionally, more parts of the core algorithm could be transferred to the GPU, particularly similarity measure calculation and the secondary descriptor comparison loop. Finally, foveated rendering and adaptive tessellation, potentially also in a *poi*/keypoint-dependent way (a viable option on non-embedded hardware with tessellation units), could further reduce the computational effort of building very high-resolution Circon descriptors.

7. Acknowledgment

This work received funding from the Deutsche Forschungsgemeinschaft through DFG grant 389792660 as part of TRR 248, the Clusters of Excellence CeTI (EXC2050/1 grant 390696704), and from the German Federal Ministry of Education and Research via the Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI) Dresden/Leipzig (BMBF, 01/S18026A-F).

References

[AGV*14] AGUS, MARCO, GOBBETTI, ENRICO, VILLANUEVA, ALBERTO JASPE, et al. "SOAR: Stochastic Optimization for Affine global point set Registration". *Vision, Modeling Visualization*. The Eurographics Association, 2014. ISBN: 978-3-905674-74-3. DOI: 10.2312/vmv.20141282 2.

[Ale12] ALEXANDRE, LUÍS A. "3D Descriptors for Object and Category Recognition: a Comparative Evaluation". 2012 2.

[BG21] BUCHENAU, CHRISTOPH and GUTHE, MICHAEL. "Real-Time Curvature-aware Re-Parametrization and Tessellation of Bézier Surfaces". *Vision, Modeling, and Visualization*. Ed. by ANDRES, BJOERN, CAMPEN, MARCEL, and SEDLMAIR, MICHAEL. 2021 4.

[BM92] BESL, P.J. and MCKAY, NEIL D. "A method for registration of 3-D shapes". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.2 (1992), 239–256 1, 2.

[BSM14] BELLEKENS, BEN, SPRUYT, VINCENT, and MAARTEN WEYN, RAFAEL BERKVENS. "A survey of rigid 3D pointcloud registration algorithms". *Fourth International Conference on Ambient Computing, Applications, Services and Technologies, Proceedings*. 2014, 8–13 2.

[BTP13] BOUAZIZ, SOFIEN, TAGLIASACCHI, ANDREA, and PAULY, MARK. "Sparse Iterative Closest Point". *Computer Graphics Forum* 32.5 (2013), 113–123 2.

[CCG*22] CHEN, ZHILEI, CHEN, HONGHUA, GONG, LINA, et al. "UTOPIC: Uncertainty-aware Overlap Prediction Network for Partial Point Cloud Registration". *Computer Graphics Forum* (2022), 87–98 2.

[DBI18] DENG, H., BIRDAL, T., and ILIC, S. "PPFNet: Global Context Aware Local Features for Robust 3D Point Matching". *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, June 2018, 195–205 2.

[DRG22] DUTTA, SOMNATH, RUSSIG, BENJAMIN, and GUMHOLD, STEFAN. "3D Point Set Registration based on Hierarchical Descriptors". *J. WSCG* 30 (2022), 44–53 1–4, 6.

[DYDZ22] DENG, BAILIN, YAO, YUXIN, DYKE, ROBERTO M., and ZHANG, JUYONG. "A Survey of Non-Rigid 3D Registration". *Computer Graphics Forum* 41.2 (2022), 559–589 2.

[Eri22] ERIKSSON, OLIVER. "Smooth Particle Ribbons Through Hardware Accelerated Tessellation". MA thesis. KTH, School of Electrical Engineering and Computer Science (EECS), 2022, 20 4.

[FB81] FISCHLER, MARTIN A. and BOLLES, ROBERT C. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". (1981), 381–395 1.

[GBS*16] GUO, YULAN, BENNAMOUN, MOHAMMED, SOHEL, FERDOUS, et al. "A Comprehensive Performance Evaluation of 3D Local Feature Descriptors". *International Journal of Computer Vision* (2016), 66–89. ISSN: 0920-5691 2.

[HGU*21] HUANG, S., GOJCIC, Z., USVYATSOV, M., et al. "PREDATOR: Registration of 3D Point Clouds with Low Overlap". *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, 4265–4274 2, 6.

[HJX*18] HAN, XIAN-FENG, JIN, JESSE S., XIE, JUAN, et al. "A comprehensive review of 3D point cloud descriptors". *ArXiv* abs/1802.02297 (2018) 2.

[HMZA21] HUANG, XIAOSHUI, MEI, GUOFENG, ZHANG, JIAN, and ABBAS, RANA. "A comprehensive survey on point cloud registration". *ArXiv* abs/2103.02690 (2021) 2.

[JH99] JOHNSON, A.E. and HEBERT, M. "Using spin images for efficient object recognition in cluttered 3D scenes". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21.5 (1999), 433–449 1, 2.

[KYOB21] KOIDE, KENJI, YOKOZUKA, MASASHI, OISHI, SHUJI, and BANNO, ATSUHIKO. "Voxelized GICP for Fast and Accurate 3D Point Cloud Registration". *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021, 11054–11059 2.

[LWZ21] LI, LEIHUI, WANG, RIWEI, and ZHANG, XUPING. "A Tutorial Review on Point Cloud Registrations: Principle, Classification, Comparison, and Technology Challenges". *Mathematical Problems in Engineering* (2021) 2.

[MV13] MÖRWALD, T. and VINCZE, M. *Object Modelling for Cognitive Robotics*. Vienna University of Technology, 2013 3.

[OOFB08] OHBUCHI, RYUTAROU, OSADA, KUNIO, FURUYA, TAKAHIKO, and BANNO, TOMOHISA. "Salient local visual features for shape-based 3D model retrieval". *2008 IEEE International Conference on Shape Modeling and Applications* (2008), 93–102 3.

[PD15] PETRELLI, ALIOSCIA and DI STEFANO, LUIGI. "Pairwise Registration by Local Orientation Cues". *Computer Graphics Forum* (2015), 59–72 1, 2, 6.

[POD*18] PAVLOV, ARTEM L., OVCHINNIKOV, GRIGORY WV., DERBYSHEV, DMITRY YU., et al. "AA-ICP: Iterative Closest Point with Anderson Acceleration". *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, 3407–3412 2.

[QYW*22] QIN, ZHENG, YU, HAO, WANG, CHANGJIAN, et al. "Geometric Transformer for Fast and Robust Point Cloud Registration". *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), 11133–11142 1.

[RBB09] RUSU, RADU BOGDAN, BLODOW, NICO, and BEETZ, MICHAEL. "Fast Point Feature Histograms (FPFH) for 3D registration". *2009 IEEE International Conference on Robotics and Automation*. 2009, 3212–3217 2.

[RBTH10] RUSU, RADU BOGDAN, BRADSKI, GARY, THIBAUX, ROMAIN, and HSU, JOHN. "Fast 3D Recognition and Pose Using the Viewpoint Feature Histogram". *Proceedings of the 23rd IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Oct. 2010 2.

[RC11] RUSU, RADU BOGDAN and COUSINS, STEVE. "3D is here: Point Cloud Library (PCL)". *IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China, May 2011 3.

[RL01a] RUSINKIEWICZ, S. and LEVOY, M. "Efficient variants of the ICP algorithm". *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*. 2001, 145–152 1.

[RL01b] RUSINKIEWICZ, S. and LEVOY, M. "Efficient variants of the ICP algorithm". *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*. 2001, 145–152 2.

[Rus10] RUSU, RADU BOGDAN. "Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments". *KI - Künstliche Intelligenz* 24 (2010), 345–348 3.

[RX20] RAN, YINGYING and XU, XIAOBIN. "Point cloud registration method based on SIFT and geometry feature". *Optik* 203 (2020), 163902. ISSN: 0030-4026 2.

[SAH19] STECHSCHULTE, JOHN, AHMED, NISAR RAZZI, and HECKMAN, C. "Robust low-overlap 3-D point cloud registration for outlier rejection". *2019 International Conference on Robotics and Automation (ICRA)* (2019), 7143–7149 1.

[SB11] SIPIRAN, IVAN and BUSTOS, BENJAMIN. "Harris 3D: A Robust Extension of the Harris Operator for Interest Point Detection on 3D Meshes". (2011), 963–976 1.

[SMFF07] SALVI, JOAQUIM, MATABOSCH, CARLES, FOFI, DAVID, and FOREST, JOSEP. "A review of recent range image registration methods with accuracy evaluation". *Image and Vision Computing* 25.5 (2007), 578–596. ISSN: 0262-8856 2.

[SOG09] SUN, JIAN, OVSJANIKOV, MAKS, and GUIBAS, LEONIDAS. "A Concise and Provably Informative Multi-Scale Signature Based on Heat Diffusion". *Proceedings of the Symposium on Geometry Processing*. Eurographics Association, 2009, 1383–1392 1.

[TCL*13] TAM, GARY K.L., CHENG, ZHI-QUAN, LAI, YU-KUN, et al. "Registration of 3D Point Clouds and Meshes: A Survey from Rigid to Nonrigid". *IEEE Transactions on Visualization and Computer Graphics* (2013) 2.

[TGA*12] TORRE-FERRERO, CARLOS, GARCÍA, JOSÉ R. LLATA, ALONSO, LUCIANO, et al. "3D point cloud registration based on a purpose-designed similarity measure". *EURASIP Journal on Advances in Signal Processing* 2012 (2012), 1–15 2–6.

[TLRS09] TORRE-FERRERO, C., LLATA, J.R., ROBLA, S., and SARABIA, E.G. "A similarity measure for 3D rigid registration of point clouds using image-based descriptors with low overlap". *2009 IEEE 12th International Conference on Computer Vision Workshops*. 2009, 71–78 5.

[TSD10] TOMBARI, FEDERICO, SALTI, SAMUELE, and DI STEFANO, LUIGI. "Unique Signatures of Histograms for Local Surface Description". 2010, 356–369. ISBN: 978-3-642-15557-4 2.

[WZWL20] WU, WEIKUN, ZHANG, YAN, WANG, DAVID, and LEI, YUNQI. "SK-Net: Deep Learning on Point Cloud via End-to-end Discovery of Spatial Keypoints". *AAAI Conference on Artificial Intelligence*. 2020 1.

[YLX*19] YI, CHENG, LU, DENING, XIE, QIAN, et al. "Hierarchical tunnel modeling from 3D raw LiDAR point cloud". *Computer-Aided Design* 114 (2019), 143–154 1.

[ZF23] ZHAO, YANG and FAN, LEI. "Review on Deep Learning Algorithms and Benchmark Datasets for Pairwise Global Point Cloud Registration". *Remote Sensing* 15 (2023). ISSN: 2072-4292 2.

[ZPK16] ZHOU, QIAN-YI, PARK, JAESIK, and KOLTUN, VLADLEN. "Fast Global Registration". *Computer Vision – ECCV 2016*. Ed. by LEIBE, BASTIAN, MATAS, JIRI, SEBE, NICU, and WELLING, MAX. Springer International Publishing, 2016, 766–782 2, 6.

[ZSN*17] ZENG, ANDY, SONG, SHURAN, NIESSNER, MATTHIAS, et al. "3DMatch: Learning Local Geometric Descriptors from RGB-D Reconstructions". *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, 199–208 2.

Appendix

A. Choice of Parameters

This supplementary material provides a broader context to the parameters for our proposed approach (GPU−HER) along with HER, ORI−HER and FGR. In order to evaluate their performance, we adapted to the default setup proposed by the respective authors. However, It is worth highlighting certain parameters integral to methods and key to their optimal registration accuracy. In the following subsections, we focus individually on the parameters for each method.

A.1. ORI-HER

The essential aspect of the method is the selection of a set of points-of-interest (*poi*). The author relies on a mixture of algorithms focused primarily on range images to extract *poi*s considering descriptor characteristics. It comprises an edge detection on a range image followed by Laplace operation on the normal vector components. Point-of-interest is selected based on random sampling from a set of all non-edge pixels with a value lower than the median of the Laplacian. The primary logic behind the process is to select a candidate set that does not lie on the edges or boundaries of the object but rather on areas with a minimum variation of curvatures.

We replicated similar steps with minor modifications for evaluation on the LRF dataset, which consists of only generic point clouds with no range images. For each source and target point cloud of a registration view-pair, we evaluate all points to classify them as edge or non-edge points. It is followed by Laplacian (L_p) computation for all points based on their normals. Finally, pois are represented by a set of points that adheres to the criteria of being a non-edge point and within a specific bound defined by eqn 15 based on their Laplacian value. The lower bound (lb) is a fraction of the median of Laplacians ($med(L_p(P))$) for all points (P) in a scan. For our experiments, we used a multiplication factor (μ) within a range [0.20,0.5].

$$lb < Ł_p < med(L_p(P)) \tag{15}$$

where

$$lb = \mu \cdot med(L_p(P)) \tag{16}$$

A.2. HER & GPU-HER

The pre-processing steps of HER and GPU−HER for the respective source and target point clouds comprise of mainly two steps *(a)* Euclidean clustering to account for highly fragmented scans *(b)* Fitting surface to clusters.

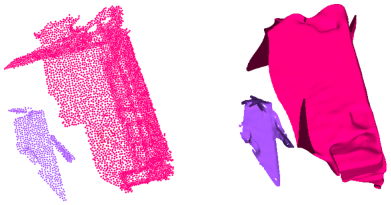


Figure 8: *Source point cloud with 2 clusters on the left with fitted NURBS surface to the right.*

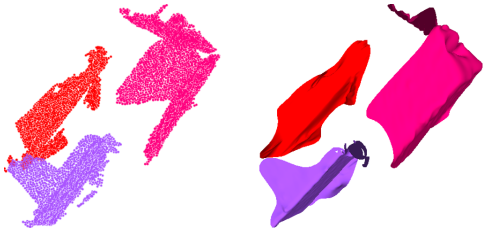


Figure 9: *Target point cloud with 3 clusters on the left with fitted NURBS surface to the right.*

A.2.1. Euclidean Clustering

We accomplish (a) by performing Euclidean clustering as implemented in the open-source Point Cloud Library (PCL). The Euclidean in the name signifies the use of Euclidean distance (L^2) metrics. Let $C_i = \{p_i \in P\}$ be a distinct cluster from $C_j = \{p_j \in P\}$ if:

$$min \|p_i - p_j\|_2 > d_{thres} \tag{17}$$

Equation 17 implies that if the minimum distance is larger than the threshold d_{thres}, then the set of points p_i and p_j belongs to two distinct clusters C_i and C_j respectively. In order to employ the notion of the minimal distance between two

sets of points, the clustering approach adapts Kd-tree-based approximate nearest neighbor search. The distance threshold d_{thres} defines a radius parameter within which the approximate set of neighbors is determined. The parameter d_{thres} is provided as an input to the method and largely depends on the dataset. Lower values result in a large number of clusters whereas too high values result in a single large cluster. Furthermore, to avoid small clusters, a parameter defining a minimum number of points representing a cluster could be provided that completely eliminates tiny clusters.

It is worth mentioning that in order to be useful for our purpose, there is no need for the clustering result to make semantic sense (although it often does), the clusters just need to be compact enough to enable sensible fitting with a single surface and boundary curve.

Figure 8 and 9 represent a pair of scans along with their respective clusters and fitted surfaces. These fitted clusters (as a union of surfaces) are input shapes for the core registration process. Our algorithm is specifically designed to tackle these complex shapes and can efficiently align pairs of scans even under challenging conditions.

	Values
interior_smoothness	0.2
interior_weight	1.0
boundary_smoothness	0.2
boundary_weight	0.0
degree	2
refinement	6
iterations	6

Table 4: *NURBS surface fitting parameters*

A.2.2. NURBS

The surface fitting step (b) is based on NURBS surface representation and their implementation as part of PCL. It comprises surface fitting to respective scans followed by trimming the boundaries of the object based on the B-spline curve. The robustness of the entire process depends on the underlying data and parameters are adapted to model the characteristics of the data. The surface fitting relies on a set of parameters illustrated in Table 4. The fitting parameters (`interior_smoothness`, `interior_weight`, `boundary_smoothness`) are used in their default settings as suggested by the PCL implementation. Importantly, the parameters largely preserved the overall surface smoothness without comprising intricate details. `interior_smoothness` defines the smoothness of the surface interior, `interior_weight` is the optimization weight awarded to points classified as interior. `boundary_smoothness` is the smoothness of the surface boundary and `boundary_weight` is the optimization weight awarded to points classified as being part of the boundary. Note that "boundary" in this case does not yet imply any trimming of the surface, although more or less the same set of points will determine the parameter-space trimming curve (see below).

The polynomial degree of the NURBS surface is defined by `degree`. The number of refinement iterations (`refinement`) defines the maximum iterations for which control-points are inserted, in a way that is approximately doubled for every iteration in each parametric direction of the surface. In most of our experiments, we obtained 64^2 of control points per surface. `iterations` refers to the number of iterations that are performed after refinement is completed. The `refinement` and `iterations` have the largest influence on the smoothness of the surface and are the only parameters we tuned for our approach, but we kept them constant across the dataset.

In order to trim the fitted surface, a NURBS curve is fitted to the boundary points in the parameter space of the surface. Again, we retained most of the default settings proposed by PCL (Table 5). The major influencing parameters are `ControlPoints_iterations` and `curve_accuracy`. `ControlPoints_iterations` defines the inner iterations of the curve fitting without insertion of control points. `curve_accuracy` refers to the average fitting accuracy.

	Values
ControlPoints_accuracy	5e-2
ControlPoints_iterations	6
ControlPoints_maximum	200
curve_accuracy	3.5e-3
curve_iterations	100
degree	2

Table 5: *Curve fitting parameter*

A.3. FGR

The Fast Global registration (FGR) adapted is based on the authors' implementation and has been inducted into our evaluation pipeline without major modifications. The parameters prescribed by the authors are maintained throughout the experiments and only the input point cloud-dependent scale multiplier required for FGR's feature descriptors was tuned to $30\overline{d}$ for the `LRF` dataset, where $\overline{d}$ is the average distance between neighboring points as defined in our paper.