

# DeepSDF x Sim(3): Extending DeepSDF for automatic 3D shape retrieval and similarity transform estimation.

Oladapo A. Afolabi\*

Allen Yang

Shankar S. Sastry

University of California, Berkeley



Figure 1: Qualitative result on Redwood dataset. Left to right in alternating fashion, input scan in red, our estimate in gray overlaid with input scan in red.

## ABSTRACT

Recent advances in computer graphics and computer vision have allowed for the development of neural network generative models for 3D shapes based on signed distance functions (SDFs). These models are useful for shape representation, retrieval and completion. However, this approach to shape retrieval and completion has been limited by the need to have query shapes in the same canonical scale and pose as those observed during training, restricting its effectiveness on real world scenes. In this work, we present a formulation that overcomes this issue by jointly estimating shape and similarity transformation parameters. We conduct experiments to demonstrate the effectiveness of this formulation on synthetic and real datasets and report favorable comparisons to strong baselines. Finally, we also emphasize the viability of this approach as a form of data compression useful in augmented reality scenarios.

## 1 INTRODUCTION

The growth of low cost commodity RGB-D sensors has made it relatively easy to build 3D models of real world scenes. Such models are useful for automatic content creation as well as other applications in Augmented and Virtual Reality. These scans are typically created with a pipeline that couples Simultaneous Localization and Mapping based pose estimation and depth image integration using Signed Distance Functions (SDFs). Subsequently, mesh or pointcloud models of the scene may be extracted using from the SDF using variants of the Marching Cubes algorithms [23].

While it is possible to produce very accurate models using this pipeline, models may suffer from missing parts and holes due to occlusion and the difficulty involved in painstakingly scanning every area of a scene. Furthermore, pointcloud or mesh representations of the scene may not be organized in a way that is intuitive to edit, and may take up an unnecessary amounts of memory to store. These characteristics are undesirable for applications such as Augmented Reality where it may be necessary to transmit the model realtime and allow everyday users interact with the model. To alleviate this problem, multiple lines of work such as [4, 19, 22, 42], propose to decompose the scene into objects and their pose and scale. In this paradigm, one favored way to represent an object is to replace it with a suitable Computer Aided Design (CAD) model from a database. The benefit of this is that the models are more likely to be complete

and can be stored using their index in the database rather than a full mesh model. However, since models from CAD databases such as [8] are stored in a canonical pose and scale, this approach is only useful when one can estimate the appropriate scale and pose as well as the most similar object in the database to that in the scene. This leads to a shape retrieval and transform estimation task.

While this approach proffers a solution to the problems posed by scanning it also introduce another problem in that, it is not guaranteed that we can find a suitable model in the database. Thus we may lose some representation power. Nevertheless, it has been discovered that neural network based 3D generative models such as [1, 14, 27] allow one to produce a larger set of shapes than are available during training (or in the database) by learning a latent space that allows for interpolation and generalization of shapes. What this implies is that we may now be able to obtain the benefit of using CAD model representation without sacrificing too much representation power. In this work we make use of a neural network 3D generative model based on SDFs to formulate the problem of shape estimation/retrieval and similarity transform (rigid body transform and scale) estimation as a simple optimization problem. We present a gradient descent based solution and compare our results to baselines for shape retrieval and transform estimation. Our approach is shown to produce excellent results on synthetic and real world data. Our contributions in this work are as follows:

1. Formulate shape retrieval and similarity transform estimation as an optimization problem.
2. Demonstrate a parameterization of the problem to allow for easy incorporation in popular deep learning frameworks.
3. Conduct experiments showing the effectiveness of our approach on synthetic and real datasets.
4. Conduct an experiment to show the viability of our approach as a form of 3D data compression.

The rest of the paper is organized as follows: In Section 2 we give an overview of work related to ours. We present the notation and problem statement in Sections 3 and 4, and detail our approach and experiments in Section 5. Finally, we report the results in Section 6 and end with the conclusion and proposals for future work in Section 7.

\*e-mail: oafolabi@berkeley.edu

## 2 RELATED WORKS

### 2.1 3D shape alignment using signed distance functions

Aligning 3D shapes is a well studied problem with varying solutions depending on the representations of the 3D shape. For example, when the geometry of a 3D object is represented using a point cloud, solutions for estimating rigid body and orientation preserving similarity transformations have been proposed [3, 16, 17, 43, 44]. In this subsection, we focus on related works pertaining to aligning 3D shapes represented using signed distance functions. Similar to the problem of aligning 3D point sets, the problem of aligning Signed Distance Functions has also been well studied; [38] provides a good overview of some of these methods. The works of [7, 37] present techniques to estimate rigid body transformations between two shapes represented using Signed Distance Functions. Similarly, we also present a method can estimate rigid body transform between shapes represented by SDFs. However, in addition our presentation also addresses the case of similarity transformations. We are not entirely unique in this regard. [10, 25] also, tackle the problem of estimating similarity transformations between shapes represented by SDFs. Their algorithm transforms the problem by making use of geometric moments and the Fourier transform. We instead make use of a simple gradient descent based algorithm to find a solution to our optimization problem. [18, 26] also make use of gradient descent algorithms on optimization problems that estimate a similarity transformation between SDFs by making use of sum of squared differences and mutual information as loss functions respectively. Yet, non of the works above explicitly allow for shape retrieval as well as transformation estimation in one optimization problem. By building off the work of [27], we are able to provide a method that both retrieves and estimates the transformation of a shape given measurements of its SDF.

### 2.2 3D Features for shape retrieval and alignment

While the methods above do not explicitly deal with shape retrieval, there exists a body of work that does. Classical shape retrieval/recognition algorithms such as in [15, 35, 40], usually follow a two step approach. A first step involves shape retrieval from a database by making use of shape descriptors such as [2, 32, 41]. This step is then followed by a correspondence search using 3D keypoints and descriptors and then an alignment based on these correspondences. It has been argued in other work [36], that dense alignment via SDFs can produce much better results since it does not make use of a correspondence search which may introduce errors. Our work, favors this approach. We compare our results to classical 3D feature based retrieval and alignment techniques and show superior performance.

More recent developments have seen a shift from hand-crafted descriptors to data driven ones such as [12, 45]. Alternative data driven approaches [29] make use of deep neural networks to replace the retrieval module entirely. However, it may be argued that the sequential approach of retrieval and then alignment does not allow for corrections of errors in the alignment phase caused by a mismatch during the retrieval stage. To address this problem [5] proposes an end-to-end shape retrieval and alignment module. Similar to this work, our formulation jointly optimizes over the space of shapes and transforms. However, unlike [5], our formulation is not a feed-forward neural network but an optimization problem that makes use of a neural network model. The main benefit of this is that the optimization process can be viewed as introducing feedback that allows iterative minimization of errors from predictions made by the network. A feed-forward module can only make a prediction once per input and has no mechanism to minimize errors after training. In addition, the space of shapes we optimize over allows us retrieve objects that were not seen during training.

## 3 NOTATION AND PRELIMINARIES:

We assume the reader is familiar with similarity transformations and signed distance functions, but to make this work self contained, we introduce some notation as definitions below.

### 3.1 3D Similarity Transformations:

A 3D similarity transformation  $g \in Sim(3)$  is a tuple  $(s, R, t)$ , with  $s \in \mathbb{R}^+$ ,  $R \in SO(3)$ ,  $t \in \mathbb{R}^3$ . The action of  $g$  on a point  $x \in \mathbb{R}^3$  is as follows:

$$g(x) = sRx + t \quad (1)$$

### 3.2 Signed Distance Functions

Let  $S$  be a solid 3D shape with a closed surface, such that we have a well defined notion of inside and outside. In addition, we define the following sets:

$$\partial\Omega_S \text{ as the set of points on the surface of } S \quad (2)$$

$$\Omega_S^- \text{ as the set of points in the interior of } S \quad (3)$$

$$\Omega_S = \partial\Omega_S \cup \Omega_S^- \quad (4)$$

$$\Omega_S^+ = \Omega_S^c \quad (5)$$

Then a signed distance function is an implicit representation of the shape defined as:

$$\Phi(x, \Omega_S) = \begin{cases} -\min_{y \in \partial\Omega_S} \|x - y\|_2 & \text{if } x \in \Omega_S \\ \min_{y \in \partial\Omega_S} \|x - y\|_2 & \text{if } x \in \Omega_S^c \end{cases} \quad (6)$$

Intuitively, for a fixed shape  $S$ , its SDF at a point  $x$  tells us the distance from  $x$  to the surface of  $S$ , with the sign determined by whether or not  $x$  lies in the interior of  $S$ .

### 3.3 Action of similarity transformations on SDFs

Let  $S, S'$  be two shapes related by a similarity transformation  $g$ , i.e.

$$\partial\Omega_{S'} = \{g(x) | x \in \partial\Omega_S\}$$

Under this condition, it is well known [26] that

$$\Phi(g(x), \Omega_{S'}) = s\Phi(x, \Omega_S) \quad (7)$$

with  $g(x)$  and  $s$  as defined in eq. 1.

### 3.4 Approximating SDFs with deep neural networks

Given a class of shapes or 3D objects  $\mathcal{S} = \{S_1, S_2, S_3, \dots, S_n\}$ , e.g. the set of chairs in a database, we may hypothesize that we can approximate the SDF representing these shapes using deep neural networks. Concretely, we first assume that we can assign to each shape in our set an element of  $\mathbb{R}^d$  i.e.  $\exists h : \mathbb{R}^d \rightarrow \mathcal{P}(\mathbb{R}^3)$  such that  $\forall S \in \mathcal{S} \exists z \in \mathbb{R}^d : h(z) = \Omega_S$ . The dimension of  $\mathbb{R}^d$ ,  $d$ , is a design choice in this scenario. We refer to  $\mathbb{R}^d$  as the latent space. We also assume,  $h$  is not a one-to-many mapping.

Then, we may define  $f(x, z) = \Phi(x, h(z))$ . It was shown in [27] that it is possible to estimate  $f$  with  $\hat{f}$  by using deep neural networks, if we restrict our set of shapes to those shape in a canonical pose and scale. For example, the work [27] was shown to work for shapes that have been normalized to lie in a unit sphere and generally facing the same direction. While it may be possible to learn an estimate for shapes in all scales and poses, we show that this may be unnecessary and inefficient. It was also shown in [27], that in learning  $\hat{f}$ , one also learn a latent space that may generate new shapes not in the training dataset.

The implication of these results is that, given a shape  $S$  and a set of  $m$  samples of its SDF,  $\chi_S = \{(x_i, \phi_i = \Phi(x_i, \Omega_S))\}_{i=1}^m$ , we may be able to estimate  $S$  by solving the following optimization problem:

$$\min_{z \in \mathbb{R}^d} \sum_{(x_i, \phi_i) \in \chi_S} |\hat{f}(x_i, z) - \phi_i| \quad (8)$$

Provided that  $S$  is also a shape in the canonical pose and scale, we may use this problem formulation for shape retrieval. In addition, since we also learn a latent space that may include shapes not in the training dataset, we may be able to retrieve shapes never seen during the training process used to learn  $\hat{f}$ . Moreover, in the event that we have incomplete knowledge of the SDF of  $S$ , this formulation gives some robustness and has been shown to be useful for shape completion as well.

To make the problem truly equivalent to shape retrieval, it is necessary to keep track of the latent vectors of the shapes in the database and then project the optimal result on this set. We find this projection step to be unnecessary since in most cases retrieval is only used to lookup the mesh model that is most similar to the shape. If we know the optimal latent shape, we can obtain a mesh for that shape through a forward pass through the network followed by marching cubes. Hence, it is sufficient for retrieval tasks to only obtain the optimal latent vector. We have only provided a summary of this work here. The interested reader may consult [27] for more details.

The restriction to shapes in the canonical pose and scale, limits the usefulness of this work in real world applications. Our main contribution in this work is to present a principled way to overcome this problem without making use of any more data.

#### 4 PROBLEM STATEMENT

We assume that we are given a set of shapes  $\mathcal{S} = \{S_1, S_2, S_3, \dots, S_n\}$  belonging to the same class, all in a canonical scale and pose, as well as a learned estimate of their signed distance function  $\hat{f}(\dots)$ .

Given a new shape  $S$  belonging to the same class, but not necessarily in  $\mathcal{S}$  or in the same canonical scale and pose, and  $m$  samples of its SDF  $\chi_S = \{(x_i, \phi_i = \Phi(x_i, \Omega_S))\}_{i=1}^m$ , can we recognize the object? To recognize the object, it is sufficient to find the shape of the object from the associated latent space of  $\hat{f}$  as well as the similarity transformation aligning this shape estimate to  $S$ .

Specifically, with our approach, the input to our algorithm is the set of SDF samples, and initial guesses for the transformation parameters and optimal latent vector. The output of our algorithm is the optimal latent vector and the optimal similarity transformation.

#### 5 APPROACH

Our main idea is to modify eq. 8 to also include an optimization over similarity transformation parameters and to choose a parameterization that is easy to implement. Making use of eq. 7 and eq. 8, we may formulate the problem as:

$$\min_{s, R, t, z} \sum_{(x_i, \phi_i) \in \chi_S} \left| s \hat{f} \left( \frac{R^{-1}(x_i - t)}{s}, z \right) - \phi_i \right| \quad (9)$$

with  $s \in \mathbb{R}^+$ ,  $R \in SO(3)$ ,  $t \in \mathbb{R}^3$ ,  $z \in \mathbb{R}^d$ . To solve the problem, one may make use of gradient descent algorithms. However care must be taken, since  $R$  does not live in a vector space. While it is possible to properly take gradient steps even in  $SO(3)$  [11], we take an alternative approach by re-parameterizing  $R$ .

Concretely, using the axis-angle representation for rotation matrices, for any rotation matrix  $R$ ,  $\exists \omega \in \mathbb{S}^2$ , and  $\theta \in [-\pi, \pi]$ :  $R = \exp(\hat{\omega}\theta)$ , where  $\omega = [\omega_1, \omega_2, \omega_3]^\top$

$$\hat{\omega} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}$$

Since,  $\omega$  lies on the unit sphere, we may parameterize it using spherical coordinates as

$$\omega_1 = \sin(\psi)\cos(\rho) \quad (10)$$

$$\omega_2 = \sin(\psi)\sin(\rho) \quad (11)$$

$$\omega_3 = \cos(\psi) \quad (12)$$

$$(13)$$

where, to avoid potential confusion due to overloading symbols, we have used  $\psi$  to represent the polar angle and  $\rho$  to represent the azimuthal angle. Consequently, we may write:

$$R(\psi, \rho, \theta) = \exp \left( \begin{bmatrix} 0 & -\cos(\psi) & \sin(\psi)\sin(\rho) \\ \cos(\psi) & 0 & -\sin(\psi)\cos(\rho) \\ -\sin(\psi)\sin(\rho) & \sin(\psi)\cos(\rho) & 0 \end{bmatrix} \theta \right) \quad (14)$$

and the initial problem eq. 9 as:

$$\min_{s, \psi, \rho, \theta, t, z} \sum_{(x_i, \phi_i) \in \chi_S} \left| s \hat{f} \left( \frac{[R(\psi, \rho, \theta)]^{-1}(x_i - t)}{s}, z \right) - \phi_i \right| \quad (15)$$

In this way, no special handling of gradients has to be taken. We find that this explicit formulation allows for easy implementation and incorporation into common deep learning frameworks with automatic differentiation such as [28].

#### 5.1 Implementation Details

To test the formulation presented, we have implemented it using PyTorch [28] and report some implementation details here for reproducibility. We have made use of the closed form expression of the matrix exponential  $\exp(\hat{\omega}\theta)$  provided by Rodrigues's formula [24, 31]

$$\exp(\hat{\omega}\theta) = I + \hat{\omega}\sin(\theta) + \hat{\omega}^2(1 - \cos(\theta)) \quad (16)$$

We have also made use of Adam [21] as implemented in [28] for solving the optimization problem eq. 15. To ensure that we remain in the feasible set of the problem, we restrict  $s \in [s, \bar{s}]$  with  $s = 0.01, \bar{s} = 10$ . To learn  $\hat{f}$  we have made use of the open source code provided by [27], with latent vectors in  $\mathbb{R}^{256}$ , and have trained on the same subset of Shapenet [8] chairs reported in their work. All our experiments are performed using the chair class but we expect similar performance on other classes of objects. To encourage convergence of the network [27] makes use of a penalty term on the norm of  $z$  weighed by  $10^{-4}$ . We have also included this term in the problem we solve eq. 15. We left it out in the above description for clarity of explanation.

#### 5.2 Experiments

We perform two experiments to validate our approach for shape and transformation estimation, one on synthetic data and the other on real scans of object. We also include an experiment to assess the viability of our approach as a means for 3D data compression.

##### 5.2.1 Synthetic data

Gradient descent based methods are susceptible to converging to locally optimal solutions. Consequently, it is important to find good initialization points for the variables we are optimizing over. We would like to observe the behavior of our approach as we change the range of initialization values.

Sometimes, we may know a subset of the variables confidently and can make use of this to obtain good performance. To empirically evaluate the performance of our method, we have assumed that

we know the axis of rotation  $\omega$ , while we vary other initialization parameters uniformly in a range about the true value. We then report performance metrics in this scenario. We have made this assumption to reduce the size of the parameter space. This is a reasonable assumption since many objects lie on a planar surface. We can estimate the normal to the planar surface and use that as the axis of rotation.

Specifically, we randomly sample five shapes from the set of shapes used to train  $\hat{f}$ , and five sets of transformation parameters, one for each shape. Note that during training the shapes are all normalized to lie in a unit sphere and facing a canonical direction (the second experiment includes the case for potentially unseen shapes). For each of these sampled shapes we perform two sets of sub-experiments. We draw each of the ground-truth transformation parameters randomly from the following sets:

$$\begin{aligned} s &\sim \text{Uniform}([\frac{1}{2}, 2]) \\ \theta &\sim \text{Uniform}([-\pi, \pi]) \\ \|t\|_2 &\sim \text{Uniform}([0, 4]) \end{aligned} \quad (17)$$

We choose the direction of  $t$  and  $\omega$  uniformly from the surface of a unit sphere. In the first sub-experiment, we then choose the initialization parameters as  $s^0 = s(1 + \Delta s)$ ,  $\theta^0 = \theta + \Delta\theta$ ,  $t^0 = t + \Delta t$ .  $\Delta s, \Delta\theta, \Delta t$  are chosen as follows:

$$\begin{aligned} \Delta s &\sim \text{Uniform}([-0.3, 0.3]) \\ \Delta\theta &\sim \text{Uniform}([\frac{-\pi}{9}, \frac{\pi}{9}]) \\ \|\Delta t\|_2 &\sim \text{Uniform}([0, 0.15]) \end{aligned} \quad (18)$$

the direction of  $\Delta t$  is chosen uniformly from the surface of the unit sphere. We choose  $z^0 \sim \text{Normal}(\mathbf{0}, \sigma \mathbf{I})$ ,  $\sigma = 0.01$ . Using these initialization parameters, we solve the optimization problem eq. 15.

We repeat this initialization procedure 100 times for each shape and set of groundtruth transformation parameters, and record the result.

The second sub-experiment is similar except that  $\Delta s, \Delta\theta, \Delta t$  are chosen as follows:

$$\begin{aligned} \Delta s &\sim \text{Uniform}([-0.5, 0.5]) \\ \Delta\theta &\sim \text{Uniform}([\frac{-2\pi}{9}, \frac{2\pi}{9}]) \\ \|\Delta t\|_2 &\sim \text{Uniform}([0, 0.2]) \end{aligned} \quad (19)$$

We provide results from this experiment to give us some intuition regarding the behavior of our algorithm if we initialize it within the parameter ranges described above.

### 5.2.2 Real data

The second experiment attempts to evaluate our algorithm on real data. We have made use of the Redwood dataset [9] and selected the subcategory of chairs. The Redwood dataset contains scans (represented as triangle meshes) of real objects. The dataset does not provide information about the groundtruth pose or scale of the shape, or the most similar shape in ShapeNet [8] corresponding to the shape. However, we can still validate our approach without this information since deviations from the groundtruth will result in misaligned shapes. We have manually segmented the dataset so that we remove other objects in the background, leaving behind only the floor and the main object in each scene. We can then segment out the floor by finding the largest plane in the scene (

usually corresponding to the floor). This leaves behind the shape of interest. We assume that all objects in the scene lie in the floor and are upright. This allows us obtain an estimate for  $\omega$  as the normal to the floor plane (using the rest of the points to determine the upward pointing normal, since all objects are above the floor plane). The segmented object typically has artifacts and noise that will negatively impact our optimization. To remove these artifacts we filter the segmented object by clustering triangles and removing triangles with small areas. We also make use of [20, 46] to remove spurious internal faces. To obtain SDF samples from the segmented mesh, we sample 250000 points on the surface and perturb them by  $\pm\epsilon$  in the direction of the normal at each point. We set  $\epsilon = 0.01$ . We also sample 25000 points from the surface of the mesh and perturb them in the direction of the normals by a random distance chosen uniformly in  $[0.07, 0.2]$  for each point. These points give us some information about the free, unoccupied space. We find the SDF of the segmented shape at each of these points by computing the distance of each point to the surface and using the outward pointing normal to the surface to determine its sign. Using this method, we collect 89 shapes and samples of their SDF. However, we use 31 randomly selected shapes from these 89 for validating the baselines described below, leaving only 58 for testing. We discarded all other scenes that could not be segmented properly. For each shape, we initialize  $s$  using the radius of the bounding sphere for that shape and  $t$  using the center of the bounding sphere. We estimate  $\omega$  using the normal vector to the floor plane and  $z$  is initialized by drawing from  $\text{Normal}(\mathbf{0}, \sigma \mathbf{I})$ ,  $\sigma = 0.01$ . The only parameter left is  $\theta$ . For  $\theta$  we grid up the space in increments of 30 degrees and run our algorithm for each value of  $\theta \in [0, \frac{\pi}{6}, \frac{\pi}{12}, \dots, 2\pi)$ . We then select the best result from each of these initialization of  $\theta$ . We run our algorithm on each of the segmented scenes and report our results.

### 5.2.3 Baselines

We compare our work to an approach that makes use of PointNet [30] for shape retrieved as described in [29] and an ICP [6] based technique for alignment. Specifically, we estimate a rough alignment for the similarity transformation using the same initialization procedure as that for our algorithm described above. We estimate the translation as the center of the bounding sphere, the scale as the radius of the bounding sphere, and the axis of rotation as the normal to the ground plane. We grid up the space over  $\theta$  in increments of 10 degrees and choose the overall transform that minimizes the F-score (see Section 5.2.5) between the retrieved model estimate, transformed by the similarity transform estimated by our initialization method, and the segmented object from the dataset. We then proceed to iteratively estimate correspondences using nearest neighbors and estimate the best transformation using a RANSAC based version of [43]. We do this for 50 iterations and return the result as an estimate for the similarity transformation. We refer to this baseline as Grid search + ICP.

As a second baseline, we make use of the same retrieval module but use Harris3D keypoints and the SHOT [34] descriptor for correspondence finding instead of our initialization module. We then make use of a RANSAC based approach for similarity transform estimation as implemented in [33]. We refer to this method as Harris3D + SHOT and also report these results. We are aware of other state of the art SDF based methods for jointly estimating shape and similarity transformation [5], however, we could not find a publicly available implementation to use for testing.

### 5.2.4 3D shape compression

Finally, we also conduct an experiment to explore the viability of our approach as a form of 3D data compression. We would like to understand the trade-off between representation power and storage size. Given the neural network parameters of  $\hat{f}$ , we can choose to store only the latent vector representing a shape as well as the

associated similarity transform. This can be useful in a scenario where one may need to transmit the shape data. It is assumed that the receiver has access to the network parameters to decode the latent vector and convert it back to a mesh. In our experiment the network size was approximately 7MB. This is a constant cost shared across many uses and is consequently amortized.

For this experiment, we save each mesh in our Redwood test set under two mesh simplification schemes, mesh simplification using quadric error metrics [13, 46] and vertex clustering [46]. We have varied the parameters of each method. For mesh simplification using quadric error metrics, we set the target number of faces to be reduced by a factor of 100 and 1000 from the number of faces in the original mesh. We refer to these experiments as QE-100, QE-1000 respectively. For vertex clustering, we vary the size of the grid cells used for clustering to be a factor of 0.1 and 0.2 of the radius of the bounding sphere for the shape. We refer to these as VC-0.1, VC-0.2 respectively. We save all meshes and compute the average size of the files as well as the mean F-score (compared to the original mesh) to give an indication of the representation power at each simplification parameter. We compare these scores against our mean F-score and the cost of saving the latent vector and transformation parameters as a 4x4 matrix.

### 5.2.5 Metrics

We have made use of the F-score as recommended in [39] to provide a quantitative evaluation of our work. Given a groundtruth shape  $S$  and an estimate shape  $\hat{S}$ , as well as samples of points on their surfaces  $\partial\Omega_S, \partial\Omega_{\hat{S}}$ <sup>1</sup> the F-score is the harmonic mean of precision and recall metrics. Precision and recall are defined as follows:

To compute the precision, for each point in the estimate shape, we first compute its distance to the groundtruth as:

$$e_y = \min_{x \in \partial\Omega_S} \|y - x\|_2$$

The precision is then computed as the percentage of points within distances less than  $\epsilon_p$ .

$$P(\epsilon_p) = \frac{100}{|\partial\Omega_{\hat{S}}|} \sum_{y \in \partial\Omega_{\hat{S}}} \mathbf{1}_{\{e_y < \epsilon_p\}} \quad (20)$$

Recall is computed in the opposite direction. Again, the error is first computed as:

$$e_x = \min_{y \in \partial\Omega_{\hat{S}}} \|x - y\|_2 \quad (21)$$

and recall as:

$$R(\epsilon_r) = \frac{100}{|\partial\Omega_S|} \sum_{x \in \partial\Omega_S} \mathbf{1}_{\{e_x < \epsilon_r\}} \quad (22)$$

Finally, the F-score is computed as:

$$F(\epsilon) = \frac{2P(\epsilon)R(\epsilon)}{P(\epsilon) + R(\epsilon)} \quad (23)$$

The F-score ranges from 0 to 1, with higher values indicating a better match between the estimate and the groundtruth. The F-score is very sensitive to deviations in shape and thus makes an excellent metric. To obtain pointclouds from the meshes, we have sampled 3000 points from the surface of each mesh. We have also set  $\epsilon = 0.05r$ , where  $r$  is the radius of the bounding sphere of the shape  $S$ .

An alternative metric may have been computing the distance between the estimated parameters and the groundtruth parameters,

<sup>1</sup>For simplicity, we have retained the same notation as that used for the set of all points on the surface. To compute the F-score, we can only use a sampling of the set of points on the surface.

	Harris3D + SHOT	Grid search + ICP	Ours
F-score	0.273	0.295	<b>0.902</b>

Table 1: Quantitative results on Redwood dataset

	Ours	QE-100	QE-1000	VC-0.1	VC-0.2
F-score	0.902	0.977	0.778	<b>0.978</b>	0.839
size (KB)	<b>1.088</b>	20.120	3.454	34.833	9.073

Table 2: Comparison of storage size and representation power for various mesh simplification algorithms

however, we find that solutions that are close in 3D space may not be close in the metric space. For example, we find that the learned shapes have slightly different centers from their groundtruth counterparts, as such a good result may have a different translation value than the groundtruth. The exception to this is the rotation since all the learned shapes face the same direction. In our experiments, we assume the axis of rotation is known and report the error in the angle of rotation about this axis  $|\hat{\theta} - \theta|$ .

## 6 RESULTS

Plotted below are the results from the experiments with synthetic data as described previously. Fig. 2 shows the histogram of F-score results with parameters from eq. 17 and eq. 18. We observe that in most cases, over 90% of the samples have an F-score greater than 0.8. We have also included a histogram of the angular error in Fig. 3 to give some intuition on the level of accuracy. We observe that most of the angular errors are within 2°.

The second set of parameters eq. 17 and eq. 19 increase the initialization error. Consequently, as we see in Fig. 4 and Fig. 5 we obtain a decrease in performance. The usefulness of this experiment is that it gives some intuition about how much error we can afford to have during initialization to obtain a desired level of performance.

In addition, we report the results from experiments on the Redwood dataset [9] in Table 1. We note that the shapes in the Redwood dataset are not necessarily in the training dataset or Shapenet [8] database. We report a significant improvement over the baselines. We believe this vast improvement is due to the expressiveness of the learned latent space over just a fixed database of CAD models as well as a reduction in errors that would otherwise be introduced by wrong correspondences. As reported in [4], data differences between noisy real scans and clean CAD models can make it difficult to find good correspondences using geometric descriptors. We also include sample images of test shapes and the predictions from our methods in Fig. 1 and Fig. 6.

Finally, we report the results from validating the ability to use our method as a form of 3D data compression in Table 2. We note that our method provides a good compromise between storage size and representation power. We provide much better representation power than two of the baselines with at least a third of the storage space and comparable representation power to the other baselines with over 20x less storage space.

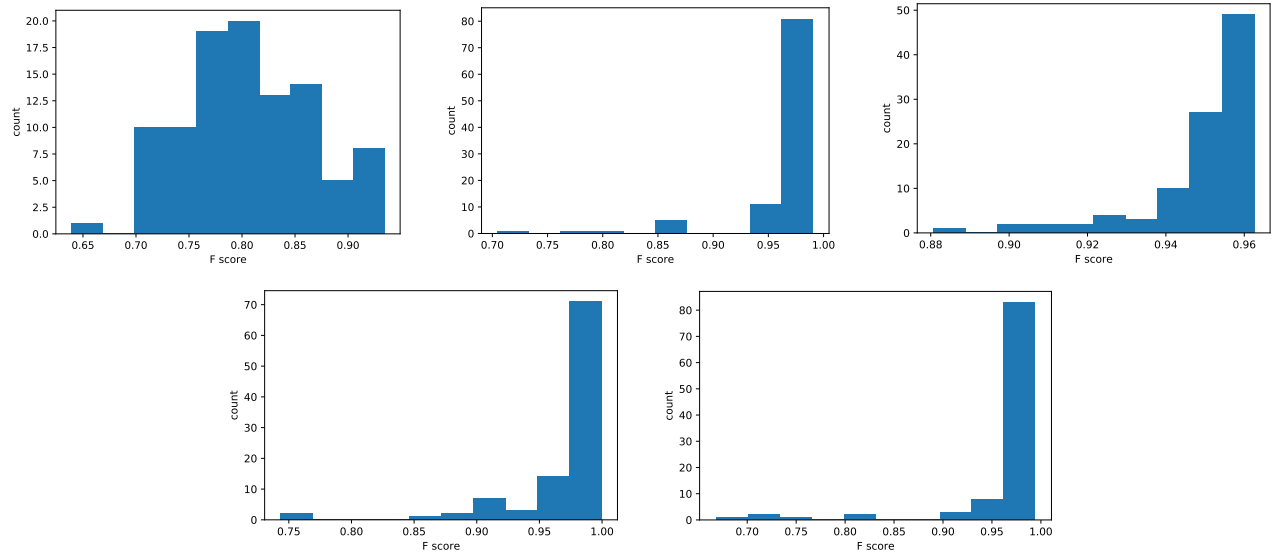


Figure 2: Histogram of F-scores on five randomly sampled sets of shape, scale and pose after one hundred runs with random initializations.

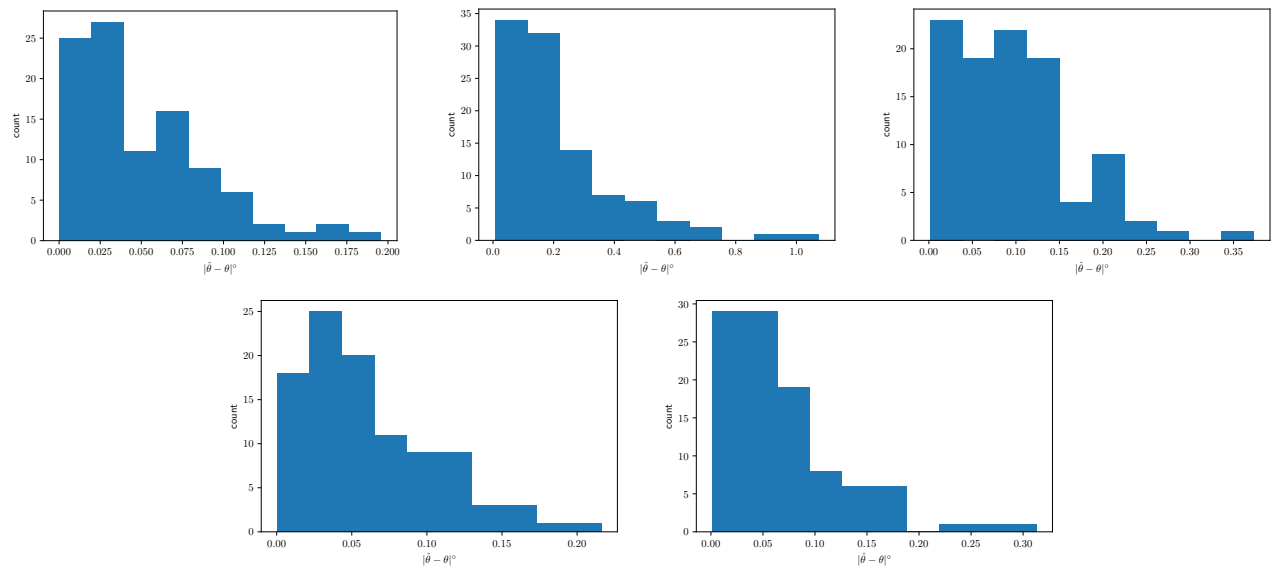


Figure 3: Histogram of angular errors on five randomly sampled sets of shape, scale and pose after one hundred runs with random initializations.

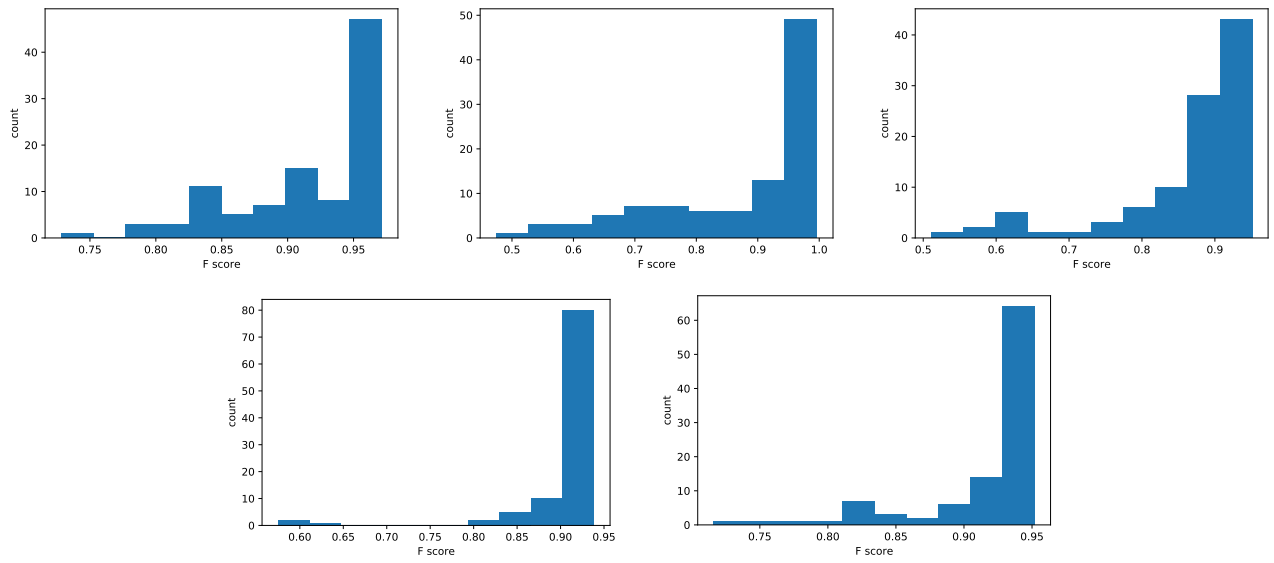


Figure 4: Histogram of F-scores on five randomly sampled sets of shape, scale and pose after one hundred runs with random initializations.

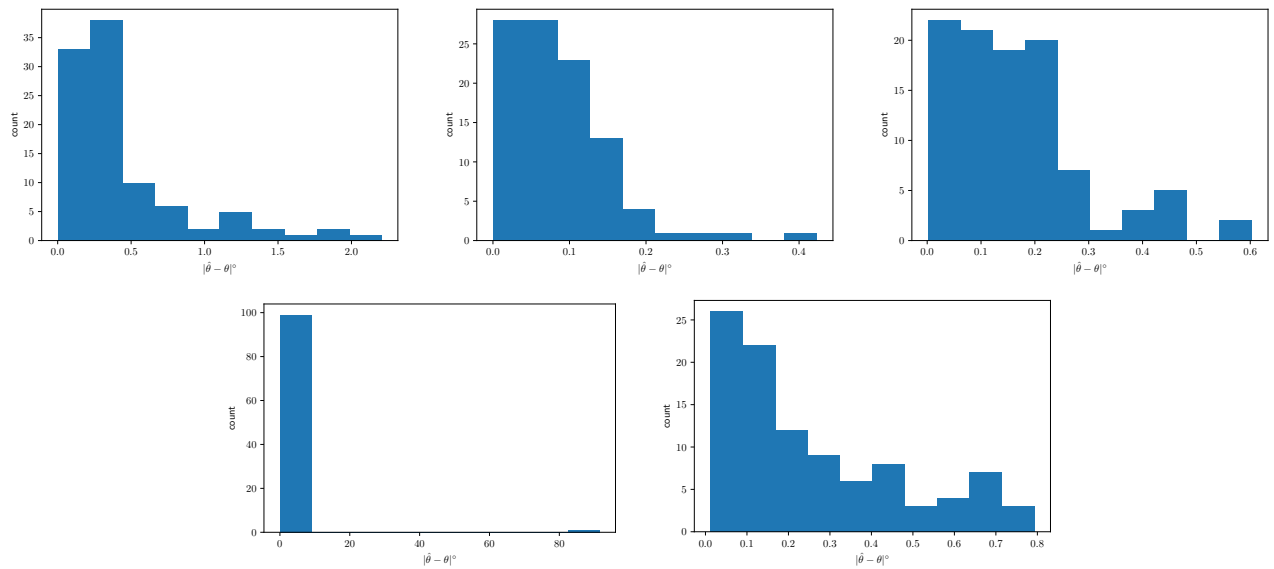


Figure 5: Histogram of angular errors on five randomly sampled sets of shape, scale and pose after one hundred runs with random initializations.

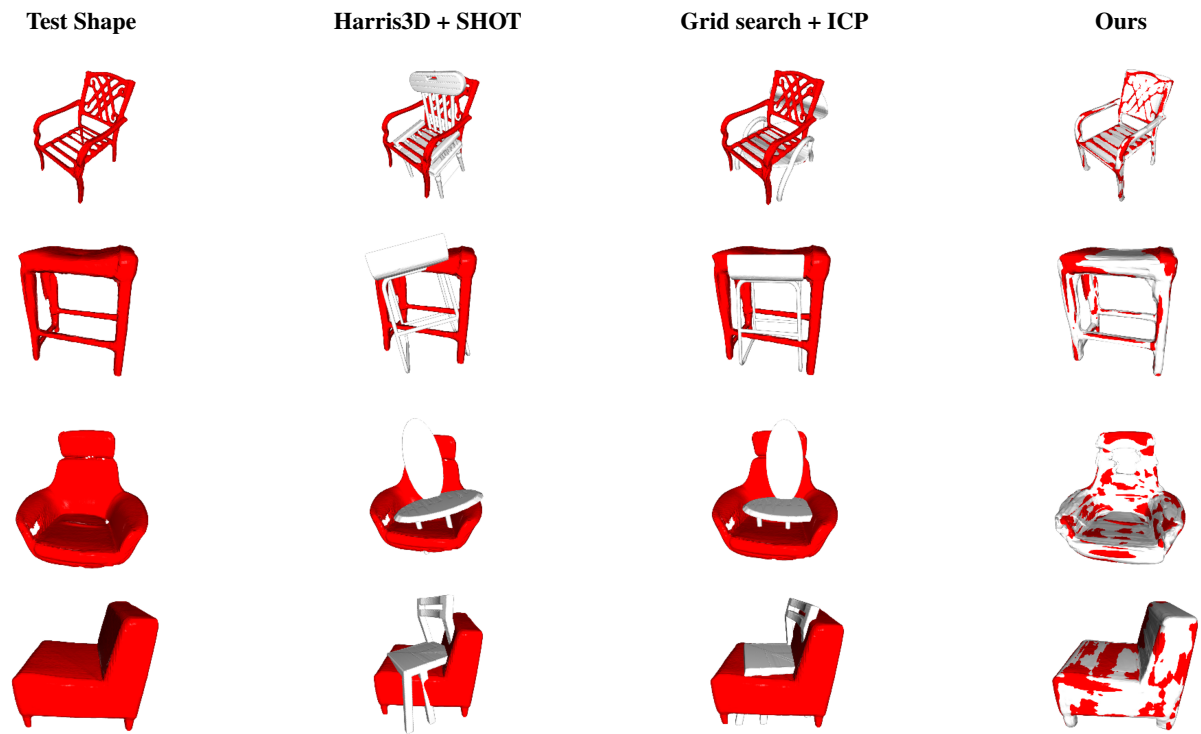


Figure 6: Qualitative result on Redwood dataset. Left to Right, test shape, results using Harris3D and SHOT for alignment, results using grid search and ICP for alignment, our proposed method. We have shown each method's result in gray overlaid with the input mesh in red.



## 7 CONCLUSION

In this paper we have presented a formulation for shape and similarity transform estimation from signed distance function data as an optimization problem. We have shown that it is possible to obtain good results with a gradient descent optimization scheme and good initialization of the parameters. We have also shown results on a dataset of real scans and obtained superior performance to baselines. While we have only tested our work on a fixed class of shapes we believe the method generalizes for any class of shapes. Our method currently relies on good segmentation of the scene and knowledge of the class of objects. As future work, we plan on tackling these problems as well as incorporating this method as a tool to 3D indoor scene understanding and representation.

## REFERENCES

- [1] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas. Learning representations and generative models for 3d point clouds. *arXiv preprint arXiv:1707.02392*, 2017.
- [2] A. Aldoma, F. Tombari, R. B. Rusu, and M. Vincze. Our-cvfh-oriented, unique and repeatable clustered viewpoint feature histogram for object recognition and 6dof pose estimation. In *Joint DAGM (German Association for Pattern Recognition) and OAGM Symposium*, pp. 113–122. Springer, 2012.
- [3] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on pattern analysis and machine intelligence*, (5):698–700, 1987.
- [4] A. Avetisyan, A. Dai, and M. Nießner. Scan2cad: Learning cad model alignment in rgb-d scans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2614–2623, 2019.
- [5] A. Avetisyan, A. Dai, and M. Nießner. End-to-end cad model retrieval and 9dof alignment in 3d scans. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2551–2560, 2019.
- [6] P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, vol. 1611, pp. 586–606. International Society for Optics and Photonics, 1992.
- [7] E. Bylow, J. Sturm, C. Kerl, F. Kahl, and D. Cremers. Real-time camera tracking and 3d reconstruction using signed distance functions. In *Robotics: Science and Systems*, vol. 2, p. 2, 2013.
- [8] 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 preprint arXiv:1512.03012*, 2015.
- [9] S. Choi, Q.-Y. Zhou, S. Miller, and V. Koltun. A large dataset of object scans. *arXiv preprint arXiv:1602.02481*, 2016.
- [10] Z. Cui, S. Mahmoodi, and M. Bennett. A robust and high-performance shape registration technique using characteristic functions. In *2018 IEEE International Conference on Image Processing, Applications and Systems (IPAS)*, pp. 1–6. IEEE, 2018.
- [11] F. Dellaert, M. Kaess, et al. Factor graphs for robot perception. *Foundations and Trends® in Robotics*, 6(1-2):82–95, 2017.
- [12] H. Deng, T. Birdal, and S. Ilic. 3d local features for direct pairwise registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3244–3253, 2019.
- [13] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pp. 209–216, 1997.
- [14] K. Genova, F. Cole, D. Vlasic, A. Sarna, W. T. Freeman, and T. Funkhouser. Learning shape templates with structured implicit functions. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 7154–7164, 2019.
- [15] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, and J. Wan. 3d object recognition in cluttered scenes with local surface features: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2270–2287, 2014.
- [16] B. K. Horn. Closed-form solution of absolute orientation using unit quaternions. *Josa a*, 4(4):629–642, 1987.
- [17] B. K. Horn, H. M. Hilden, and S. Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *JOSA A*, 5(7):1127–1135, 1988.
- [18] X. Huang, N. Paragios, and D. N. Metaxas. Shape registration in implicit spaces using information theory and free form deformations. *IEEE transactions on pattern analysis and machine intelligence*, 28(8):1303–1318, 2006.
- [19] H. Izadinia, Q. Shan, and S. M. Seitz. Im2cad. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5134–5143, 2017.
- [20] S. Katz, A. Tal, and R. Basri. Direct visibility of point sets. In *ACM SIGGRAPH 2007 papers*, pp. 24–es. 2007.
- [21] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [22] Y. Li, A. Dai, L. Guibas, and M. Nießner. Database-assisted object retrieval for real-time 3d reconstruction. In *Computer Graphics Forum*, vol. 34, pp. 435–446. Wiley Online Library, 2015.
- [23] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987.
- [24] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An invitation to 3-d vision: from images to geometric models*, vol. 26. Springer Science & Business Media, 2012.
- [25] S. Mahmoodi, M. S. Al-Huseiny, and M. S. Nixon. Similarity registration for shapes based on signed distance functions. In *International Symposium on Visual Computing*, pp. 599–609. Springer, 2012.
- [26] N. Paragios, M. Rousson, and V. Ramesh. Non-rigid registration using distance functions. *Computer Vision and Image Understanding*, 89(2-3):142–165, 2003.
- [27] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 165–174, 2019.
- [28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, eds., *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.
- [29] Q.-H. Pham, M.-K. Tran, W. Li, S. Xiang, H. Zhou, W. Nie, A. Liu, Y. Su, M.-T. Tran, N.-M. Bui, et al. Shrec18: Rgb-d object-to-cad retrieval. *Proc. 3DOR*, 2, 2018.
- [30] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.
- [31] O. Rodrigues. *De l’attraction des sphéroïdes, Correspondence sur l’É-cole Impériale Polytechnique*. PhD thesis, PhD thesis, Thesis for the Faculty of Science of the University of Paris, 1816.
- [32] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE international conference on robotics and automation*, pp. 3212–3217. IEEE, 2009.
- [33] R. B. Rusu and S. Cousins. 3d is here: Point cloud library (pcl). In *2011 IEEE international conference on robotics and automation*, pp. 1–4. IEEE, 2011.
- [34] S. Salti, F. Tombari, and L. Di Stefano. Shot: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding*, 125:251–264, 2014.
- [35] Y. Shan, B. Matei, H. S. Sawhney, R. Kumar, D. Huber, and M. Hebert. Linear model hashing and batch ransac for rapid and accurate object recognition. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2, pp. II–II. IEEE, 2004.
- [36] M. Slavcheva. *Signed Distance Fields for Rigid and Deformable 3D Reconstruction*. PhD thesis, Technische Universität München, 2018.
- [37] M. Slavcheva, W. Kehl, N. Navab, and S. Ilic. Sdf-2-sdf: Highly accurate 3d object reconstruction. In *European Conference on Computer Vision*, pp. 680–696. Springer, 2016.
- [38] M. Steiner. Statistical shape models with signed distance functions.
- [39] M. Tatarchenko, S. R. Richter, R. Ranftl, Z. Li, V. Koltun, and T. Brox. What do single-view 3d reconstruction networks learn? In *Proceedings*

- of the *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3405–3414, 2019.
- [40] F. Tombari and L. Di Stefano. Object recognition in 3d scenes with occlusions and clutter by hough voting. In *2010 Fourth Pacific-Rim Symposium on Image and Video Technology*, pp. 349–355. IEEE, 2010.
  - [41] F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. In *European conference on computer vision*, pp. 356–369. Springer, 2010.
  - [42] S. Tulsiani, S. Gupta, D. F. Fouhey, A. A. Efros, and J. Malik. Factoring shape, pose, and layout from the 2d image of a 3d scene. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 302–310, 2018.
  - [43] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (4):376–380, 1991.
  - [44] M. W. Walker, L. Shao, and R. A. Volz. Estimating 3-d location parameters using dual number quaternions. *CVGIP: image understanding*, 54(3):358–367, 1991.
  - [45] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1802–1811, 2017.
  - [46] Q.-Y. Zhou, J. Park, and V. Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.