ETH zürich

PRS Photogrammetry Remote Sensing

SO(3)-Equivariant Neural Network for Multi-object Relocalization and Reconstruction

Master's Thesis

Liyuan Zhu

liyzhu@student.ethz.ch

Photogrammetry and Remote Sensing
Department of Civil, Environmental and Geomatic Engineering
ETH Zurich

Supervisors:
Shengyu Huang
Prof. Dr. Iro Armeni
Prof. Dr. Konrad Schindler

July 2023

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

SO(3)-Equivariant Neural Network for Multi-object Relocalization and Reconstruction

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Zhu	Liyuan

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
2023/07/24	Liyuan Zhu

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.

Acknowledgements

Hereby, I express my gratitude to the following people, without whom I could not have completed my thesis.

- **Shengyu Huang** for his responsible guidance throughout the thesis process and for his fruitful output on 3D vision during our discussions. My working attitude and interest in 3D vision are positively affected by him.
- **Prof. Dr. Iro Armeni** for providing the initial idea of this project. I benefit from our weekly discussion and her insightful suggestions as a 3D scene understanding specialist.
- **Prof. Dr. Konrad Schindler** for his generosity in providing the computing resources and his high-level support from my semester project to this thesis.

I would also like to thank **Bingxin Ke, Yue Pan, Junyang Gou, and Dr. Torben Peters** for the discussions and proofreading. I spent an incredible two-year journey at ETH Zurich, and I thank everyone I have met. I have learned to find exciting research questions, be open-minded, and do rigorous research. Most importantly, I would like to thank **my parents** for their invaluable love, supporting my study in Switzerland, and encouraging me to explore more possibilities.

Abstract

In the built environment, humans interact with surrounding objects on a daily basis. It is essential that autonomous robots and systems understand the motion of moving objects to perform subsequent tasks. To the best of our knowledge, there are no datasets that capture such long-term changing environments with multiple objects. In this thesis, we synthesize a dataset to simulate the dynamic environment. We further divide dynamic scene understanding into three tasks: *(i)* shape matching *(ii)* relative pose estimation and *(iii)* shape reconstruction. We solve the trinity problem by using a single neural network in an iterative fashion. Experiments justify the effectiveness of our design choice, and we achieve the best or on-par results in all three tasks.

The **contributions** of this thesis are:

- A new **dataset** that simulates the long-term changing indoor environments.
- Novel **multi-object-centric** scene understanding using point clouds.
- A learning-based **pipeline** to recover the motions and shapes of objects.
- A novel **joint optimization** to aggregate multiple temporal scans.

Keywords: Equivariant Neural Networks, Implicit Neural Representation, Shape Matching, Pose Estimation, Shape Reconstruction, Dynamic Scene Understanding, and Joint Optimization.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

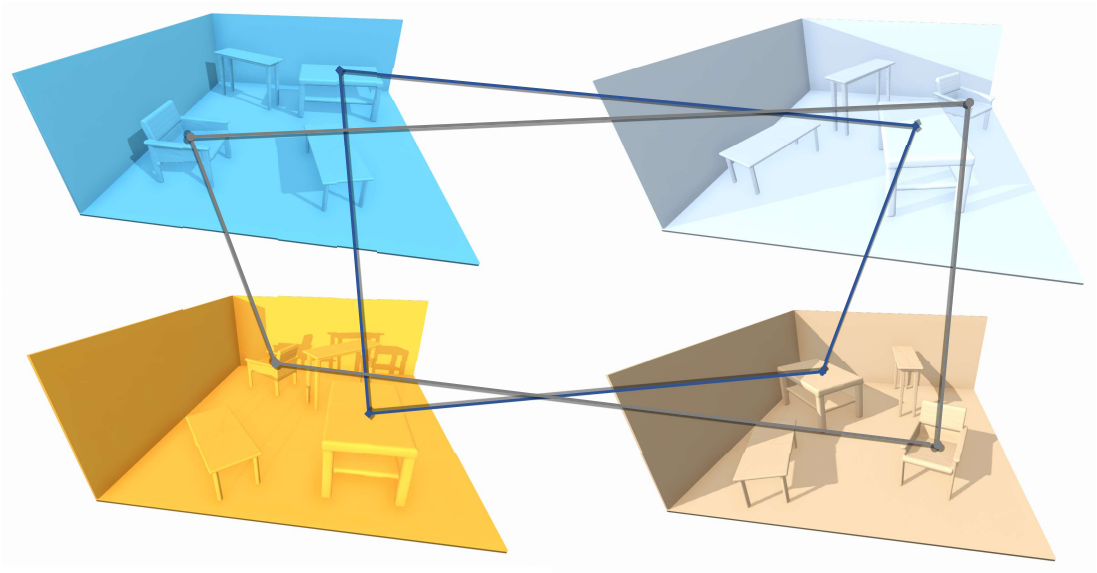


Figure 1.1: **A Dynamic Room with Moving Objects.** Connected lines refer to the associations of objects. Colors refer to different temporal stages.

1.1 Motivation

Understanding dynamic scenes is a long-standing and essential problem in the vision and photogrammetry community. Given the ability to analyze dynamic environments, intelligent systems *e.g.*, autonomous driving [8, 9], augmented reality [10], and mobile robots can perform complex tasks to facilitate further automation. However, previous work [11, 12, 13] focus on real-time dynamic scenes and have neglected the long-term dynamics of built environments. The **long-term changing environment** refers to a built environment in which sensor data cannot be captured constantly but irregularly, as shown in Figure 1.1. Therefore, the movements of moving objects cannot be modelled using trackers [14, 15, 12]. The second issue is the **data modality**. Due to the ease of data capture and annotation, most of the object pose estimation and reconstruction work [16, 12, 13, 17] use RGB/RGB-D images as input. **Point cloud** is a direct and convenient 3D representation that can also be used to perform the same tasks but is harder to annotate. Understanding long-term changes in the environment using point clouds is still an unexplored direction with sporadic related work [18]. More recently, **implicit neural representation** [6, 19, 20] has gained increasing popularity due to its continuity and differentiability. It is a powerful tool for surface reconstruction [21, 22] and pose estimation [23]. But in scene-level point clouds, individual objects *(i)* have arbitrary poses at different times and *(ii)* are often partially observed. These problems have hindered the integration of implicit neural representation and spatio-temporal scene understanding.

In this thesis, we study the **relocalization** and **reconstruction** of multiple objects in environments with long-term changes.

1.2 Research Questions

We aim to develop an algorithm to understand the motions and shapes of objects in the built environment:

- **Input:** point clouds of a dynamic scene.
- **Objectives:** motions and shapes of the foreground objects.
- **Assumptions:** rigid motions of moving objects and instance-level masks.

We come up with following **research questions**:

1. Are there existing datasets that captured the long-term changing environment?
2. Can we develop a concise learning-based solution for this dynamic problem?
3. Can we aggregate the temporal scans to improve pose estimation and reconstruction?

1.3 Thesis Structure

The structure of this thesis unfolds as follows:

1. In Chapter 1, we introduce the motivation and objectives of the thesis.
2. In Chapter 2, we review the related work from four aspects: *(i)* representations for spatiotemporal data, *(ii)* implicit neural representation, *(iii)* Object reconstruction and pose estimation, *(iv)* spatiotemporal scene representations and *(v)* realted datasets.
3. In Chapter 3, we present our learning-based method for solving the trinity problem. We tackle shape matching, registration, and aggregation using the learned model sequentially.
4. In Chapter 4, we demonstrate our dataset generation, data processing, and experimental settings. We compare our method with existing baselines, conduct ablation studies, and analyze the pros and cons of our method.
5. In Chapter 5, we summarize the thesis with findings, discussions, and future work.

Chapter 2

Related Work

In this chapter, we discuss previous work on implicit neural representations, object pose estimation and reconstruction, equivariant neural networks, 4D scene representations, and related datasets.

2.1 Implicit Neural Representations

Implicit Neural Representations, also known as **Neural Fields**, parameterize a field quantity for all spatial or temporal coordinates using neural networks [1]. The main types of neural fields are **Signed Distance Field (SDF)** [6] and **Radiance Field** [24]. In this thesis, we focus only on the neural implicits of SDF and its variants for surface reconstruction. [6, 20, 19] are the three concurrent and seminal works for neural SDFs. The input of neural SDFs can be images or point clouds, and the goal is to reconstruct the complete surface from single-view or partial observations. Neural networks learn the shape priors from the training data and encode them in latent code $\mathbf{z}$. The field quantity is conditioned on $\mathbf{z}$ during decoding.

As shown in Figure 3.1, there are two types of conditioning for neural fields. In **global conditioning**, a single latent code $\mathbf{z}$ [25, 19, 20] represents the global structure of a field and is the condition for all query coordinates. In **local conditioning** [21, 26, 22], the field is discretized by a coordinate-based data structure *e.g.* feature grid, feature volume, and the latent code for is coordinate dependent. Global conditioning is widely used by object-centric reconstruction for its consice representation and easy-to-encode category priors [27]. Local conditioning is preferred for precise surface reconstructions [21, 26, 22] as it captures local geometry information in better detail. [28, 29] integrate neural representations

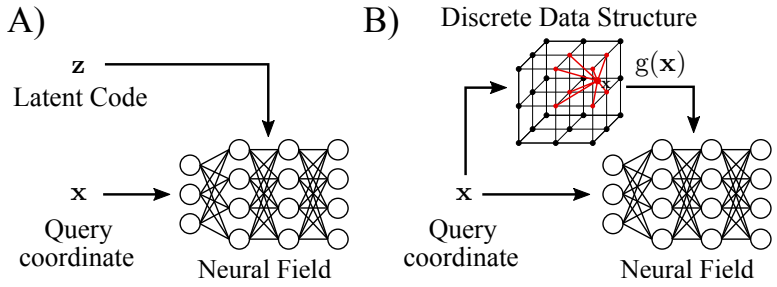


Figure 2.1: **Global Conditioning and Local Conditioning.** Image from [1].

with explicit representations for high-fidelity reconstruction and compression. To mitigate the poor generalizability of global conditioning, Duggal *et al*. [30] use an encoder as a robust initializer and test-time optimization to reconstruct shapes in the wild. [31, 32] leverage gradient-based meta-learning to generalize across different shapes. Truong *et al*. [33] include additional geometric constraints during run-time optimization. Huang *et al*. [34] further transfer neural radiance fields to LiDAR-based novel view synthesis.

2.2 Object Reconstruction and Pose Estimation

Object-centric mapping is a fundamental problem for computer vision and robotics. It is the task of estimating the 6 DoF transformation from the world reference frame to the object-centric frame (NOCS [35]), and to reconstruct the shapes of the object. It usually consists of the instance detection and reconstruction of every individual object from multi-view data. Recent advances in implicit neural representation [6, 19, 20] have stimulated the development of the multi-object version of this task.

In Table 2.1 we summarize the evolution of object-centric pose estimation and reconstruction. FroDo [16] infers the pose and shape of a single object from coarse to fine using DeepSDF [6] decoding. MOLTR [12] is a follow-up of FroDo and utilizes Bayesian tracking to solve the multi-object association problem. ELLIPSDF [36] and ODAM [38] further introduce geometric representations *i.e*. superquadrics to represent shape primitives and constrain multi-view optimization. Huang *et al*. [37] achieve object-part segmentation from the motion field with synchronization maps and iterative refinement. Irshad *et al*. [25] do not rely on existing detectors but instead develop a single-shot pipeline to regress object

Method	Venue	Input	MV	MO	Add-ons	Output	Key Features
FroDo [16]	CVPR 2020	RGB	✓	✗	Object detector Camera poses	Object Pose Shape	2 Decoders (sparse & dense) Rendering-based optimization
MOLTR [12]	RA-L 2020	RGB	✓	✓	Object detector Camera poses	Bounding Boxes Shapes	Bayesian Tracker Deep Shape Prior
ELLIPSDF [36]	CVPR 2021	RGB-D	✓	✓	Semgmentation Camera poses	Object Poses Shapes	Bi-level Representation Implicit Object Shape
MultiBodySync [37]	CVPR 2021	Point Clouds	✗	✗	-	Part Association Rigid Motion	Multi-Body Segmentation Permutation Synchronization
ODAM [38]	ICCV 2021	RGB	✓	✓	Object detector Camera poses	Bounding Volumes	Multi-view Association Super-quadrics Representation
Shapo [25]	ECCV 2022	RGB-D	✗	✓	-	Object Shapes Appearances	Disentangle shape and appearance Octree-based optimization
BundleSDF [13]	CVPR 2023	RGB-D	✓-	✗	Segmentation	Pose Tracking Shape	Dynamic Object Pose Graph Neural Object Field

Table 2.1: **Summary of object-centric mapping. MV** represents multi-view and **MO** represents multi-object. ✓- in BundleSDF denotes that it can be either single-view for a dynamic scene or multi-view for a static scene.

pose, shape, and appearance. BundleSDF [13] shifts the focus to dynamic objects and generalizes to unknown objects using Neural Object Fields and graph optimization.

2.3 Equivariant Neural Networks

The proliferation of 3D data capture [8, 39, 9, 40, 41] provides the ground for deep learning on point clouds [42, 43]. A point cloud is an **unordered** list of 3D coordinates (**subject to permutations**). Point clouds may have different sampling patterns, orientations, and partiality due to changes of view or 3D sensor (**subject to rotations**). In this thesis, we focus on the equivariant neural networks for point clouds.

PointNet [42] and ACNe [44] resolve permutations of point clouds using order-invariant/equivariant layers, such as pooling layers across point features. Rotations for 3D deep learning can be partly resolved by heavy data augmentations during training [42]. Tomas *et al.* [45] apply spherical harmonics to constrain the network and achieve SE(3) equivariance. SE(3) transformers [46] introduce equivariance to the self-attention module and significantly improve the efficiency of [45]. Deng *et al.* develop Vector Neurons [47], a general framework that can be applied to arbitrary point-cloud backbones and different tasks by vectorizing

scaler neurons in neural networks. EPN [48] breaks down Naïve 6D convolutions on SE(3) into separable convolutions in 3D Euclidean and SO(3) spaces and reduces the computational cost. GraphOnet [49] extends Vector Neurons to SE(3) equivariance. TF-ONet [50] facilitates equivariance for local shape modeling and improves generalizability. EFEM [51] is an example of Vector Neurons working in the wild for point cloud segmentation.

2.4 4D Scene Representations

Modeling dynamic 3D scenes is a challenging task. Mustafa *et al*. propose a segmentation and reconstruction pipeline that decomposes the scene into spatio-temporally coherent static and dynamic objects [52]. Huang *et al*. [53] apply DBSCAN [54] to cluster the spatio-temporal point cloud to obtain instance masks and associations. Caspr [55] extends NOCS [35] to a 4D Temporal-NOCS using Neural Ordinary Differential Equations [56]. Neural Scene Graph [57] combines 3D scene graph with NeRF [24]. It assigns a latent code to each object and models the dynamic scene by composition of NeRFs. Li *et al*. [58] extend NeRF [24] to 4D (Neural Scene Flow Fields) by explicitly modeling forward and backward scene flows as a vector field. DynIBaR [59] addresses the problems of complex motions and uncontrolled camera trajectories by respecting motion and aggregating nearby views. K-Plane and Hex-Plane [60, 61] are two concurrent works that factorize 4D volume into feature planes and efficiently synthesize novel views in space and time. Singer *et al*. [62] combine Hex-Plane representation and diffusion model [63] to generate 4D dynamic scenes from text and video.

2.5 3D Indoor Datasets

SceneParser [41] and ScanNet [40] are two large-scale indoor scene datasets for 3D object classification, detection, segmentation, and *etc*. Wald *et al*. [17, 64] collect a dataset that benchmarks object relocalization and scene graph generation in dynamic environments. NOCS [35] is a real-world dataset for 6D pose estimation and reconstruction in various scenes. [21, 65, 50] synthesize 3D room datasets by combining random shapes from ShapeNet [66]. NAVI [67] is a category-agnostic dataset with high-quality 3D shape and pose annotations.

Chapter 3
Methodology

In this chapter, we first introduce our problem setting and objectives. We then elaborate on our network architecture and the step-by-step pathway of solving shape matching, relative pose estimation, and reconstruction using a single network. Finally, we introduce our joint optimization of shapes and pose graphs to aggregate multi-temporal point clouds.

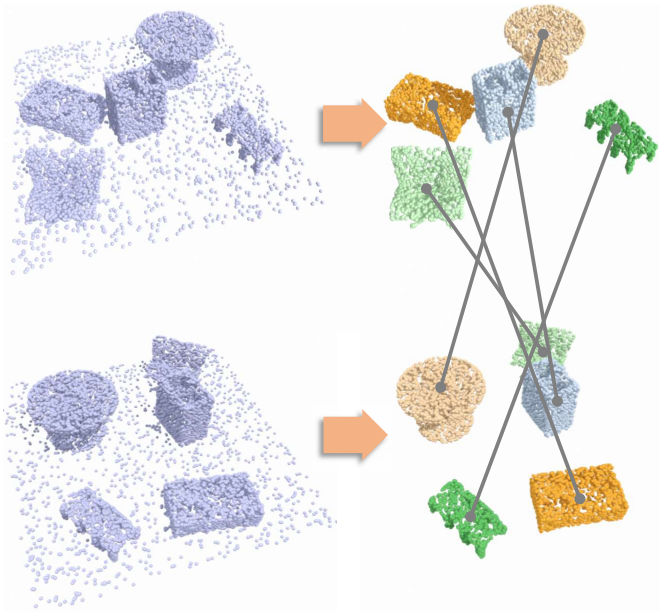


Figure 3.1: **An Example of Dynamic Rooms.** Instance masks can be obtained by off-the-shelf point cloud segmentation methods, *e.g.* [2, 3].

3.1 Problem Formulation

Problem Setting. Consider a collection of point clouds $\mathcal{X} = \{\mathbf{X}^t\}_{t=1}^{M}$ of a room captured by a mobile mapping system with irregular intervals. All point clouds in $\mathcal{X}$ are pre-aligned, sharing a global reference frame. Point cloud $\mathbf{X}^t$ represents the room at the time stage t and contains a list of moving objects $\mathcal{S}^t = \{\mathbf{O}_i^t\}_{i=1}^{N}$. Each object $\mathbf{O}_i^t$ within the room is associated with a pose $\mathbf{T}_i^t \in \mathrm{SE}(3)$ and has different poses at different t.

Assumptions. Given a point cloud $\mathbf{X}^t$, it is assumed that we can obtain the instance mask $\mathbf{M}_i^t$ for each object from existing methods [3, 2, 68]. We assume the rigid motion of objects and none existence of identical shapes to avoid ambiguities.

Objectives. ① Shape matching: the correspondence of the same objects $\mathbf{O}_i^t \in \mathcal{S}^t$ at different ts. ② Relative pose estimation: 6DoF transformation $\mathbf{T}_i^{t,t+k} \in \mathrm{SE}(3)$ of every object $\{\mathbf{O}_i\}_{i=1}^{N}$ from stage t to $t+k, k \in [0\,..\,N-1]$. ③ Surface reconstruction of every object $\{\mathbf{O}_i\}_{i=1}^{N}$ from sparse point clouds.

3.2 Network Architecture

We combine a Vector-Neuron (VN) encoder with a decoder (see Figure 3.2). The encoder Ω takes as input the point cloud $\mathbf{X} \in \mathbb{R}^{3\times N}$ (N is the number of input points) of an object. The encoder outputs invariant features $\mathbf{F}_{inv} \in \mathbb{R}^{1\times 256}$, equivariant features $\mathbf{F}_{eqv} \in \mathbb{R}^{3\times 256}$, scale estimate $\mathbf{F}_{scale} \in \mathbb{R}$, and centroid corrections $\mathbf{F}_{center} \in \mathbb{R}^{3\times 1}$,

$$[\mathbf{F}_{inv}, \mathbf{F}_{eqv}, \mathbf{F}_{scale}, \mathbf{F}_{center}] = \Omega(\mathbf{X}). \tag{3.1}$$

During decoding, we first normalize the query point $\mathbf{q} = (\mathrm{x}, \mathrm{y}, \mathrm{z})^{\mathrm{T}}$ into canonical space.

$$\mathbf{\Theta}_{NOCS} = \left\langle \mathbf{F}_{eqv}, \frac{\mathbf{q} - \mathbf{F}_{center}}{\mathbf{F}_{scale}} \right\rangle_{channel}, \tag{3.2}$$

$\mathbf{\Theta}_{NOCS}$ from Eq. 3.2, is concatenated with $\mathbf{F}_{inv}$, as input to the decoder (Eq. 3.3):

$$\mathrm{SDF} = \mathrm{MLP}\left(\mathrm{cat}[\mathbf{F}_{inv}, \mathbf{\Theta}_{NOCS}]\right), \tag{3.3}$$

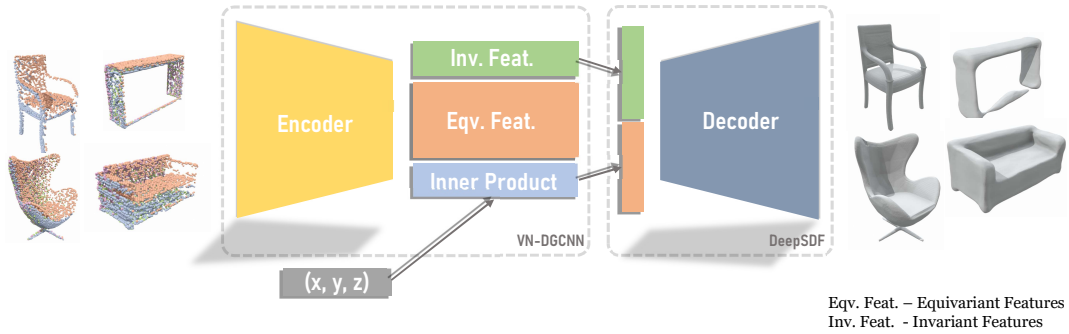


Figure 3.2: **Network Architecture for Surface Reconstruction.** Our network learns to predict the SDF values at the query coordinates $\mathbf{q} = (\mathrm{x}, \mathrm{y}, \mathrm{z})^{\mathrm{T}}$ The encoder takes point clouds as input and extract rotation equivariant and invariant features. The decoder takes the query coordinates and the feature embeddings as input and computes the corresponding SDF values.

where $\mathbf{\Theta}_{NOCS} \in \mathbb{R}^{1\times 256}$ denotes the canonicalized features, $\langle\cdot,\cdot\rangle_{channel}$ the channel-wise inner product between two feature vectors, MLP$(\cdot)$ a multi-layer perceptron, cat$[\cdot]$ concatenation. The canonicalization step in Eq. 3.2 maps the query point from an arbitrary SE(3) space to the normalized object coordinate system. $\mathbf{\Theta}_{NOCS}$ is invariant to SIM(3) (see Appendix A.1).

$$\mathrm{SIM}(3) = \left\{ \mathbf{T} \in \mathbb{R}^{4\times 4} : \mathbf{T} = \begin{bmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0}^{\top} & 1 \end{bmatrix} \right\}, \tag{3.4}$$

where $\mathbf{T}$ denotes a similarity transformation in 3D Euclidean Space, including scale $s \in \mathbb{R}$, rotation $\mathbf{R} \in \mathrm{SO}(3)$ and translation $\mathbf{t} \in \mathbb{R}^{3\times 1}$.

3.3 Shape Matching

Assume that we can obtain instances of foreground objects $\{\mathbf{O}\}_{i=1}^{N}$ in the point cloud of a scene, as shown in Figure 1.1. Given several point clouds of a scene at several temporal stages, it is unclear about the correspondences of the same objects at different times t_i and t_j, *i.e.*, $\{\mathbf{O}_i^{t_1}\}_{i=1}^{N}$ and $\{\mathbf{O}_i^{t_2}\}_{i=1}^{N}$ where i denotes the index of the point cloud in the scene. The goal of shape matching is to sort

out the pairs of point clouds of the same objects:

$$\{\mathbf{O}_i^{t_1}\}_{i=1}^N,\ \{\mathbf{O}_i^{t_2}\}_{i=1}^N \rightarrow \{(\mathbf{O}_i^{t_1}, \mathbf{O}_j^{t_2})\}_{i,j=1}^N, \tag{3.5}$$

where N denotes the number of objects in the scene and $(\cdot, \cdot)$ the matched pair.

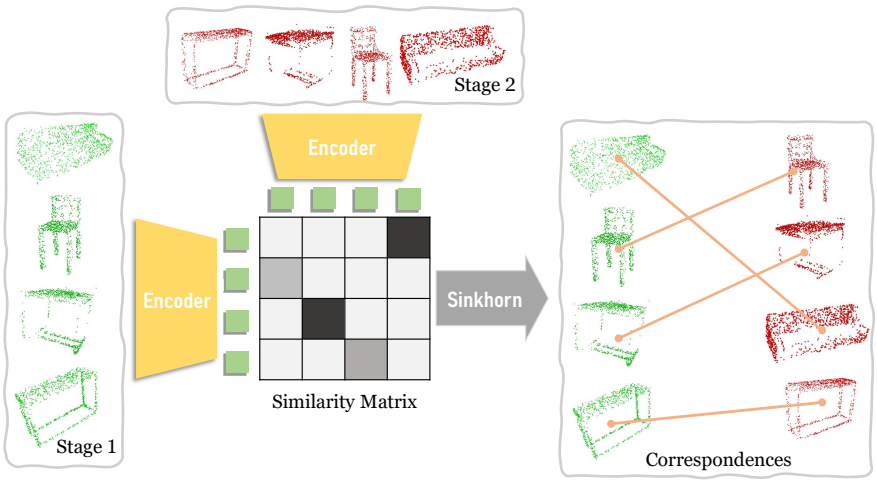


Figure 3.3: **Workflow of Object Matching**. It consists of encoding, similarity matrix, and Sinkhorn algorithm [4, 5].

Shape matching pipeline. As shown in Figure 3.3, we first extract embeddings of point-cloud instances using the encoder. In shape matching, we only use invariant embeddings $\{\mathbf{F}_i^t\}$, with i being the object id and t the temporal stage. The rotation invariant networks have the following property

$$\Phi(\mathbf{X}, \mathbf{W}) = \Phi(\mathbf{R}\mathbf{X}, \mathbf{W}) \tag{3.6}$$

with $\mathbf{W}$ being weights of the networks. This is indispensable for extracting matchable features from point clouds with large pose variations. We compute the inner product of all pairs of features $\{\langle \mathbf{F}_i^{t_1}, \mathbf{F}_j^{t_2} \rangle\}_{i,j=1}^N$ and obtain a similarity matrix $\mathbf{\Lambda} \in \mathbb{R}_+^{N \times N}$. The problem now becomes to find the best assignment $\mathbf{P}$ that maximizes total similarity $\sum_{i,j} \mathbf{\Lambda}_{i,j} \mathbf{P}_{i,j}$. To solve this linear assignment problem, we resort to the (differentiable) Sinkhorn algorithm [4, 5]. It is an efficient solution to discrete optimal transport [69]. With this, the best assignment of embeddings can be obtained, and objects at different time stages can be

matched. No additional networks are used to match objects, as feature embeddings contain structural information of each shape. We consider shape matching an essential step to the following multi-object registration (Section 3.4) and reconstruction (Section 3.5). It enables us to parse the changing environment from a multi-object-centric perspective.

3.4 Relative Pose Estimation

From Section 3.3, point clouds of the corresponding objects are matched but not aligned. The problem of solving unaligned point clouds is equivalent to pairwise point-cloud registration or relative pose estimation. Consider two point clouds $\mathbf{X}^{t_1} \in \mathbb{R}^{3\times N^1}$ and $\mathbf{X}^{t_2} \in \mathbb{R}^{3\times N^2}$ at two temporal stages. Our goal is to estimate the relative transformation $\mathbf{T} = [\mathbf{R} \in \mathrm{SO}(3), \mathbf{t} \in \mathbb{R}^3]$ that aligns $\mathbf{X}^{t_2}$ to $\mathbf{X}^{t_1}$.

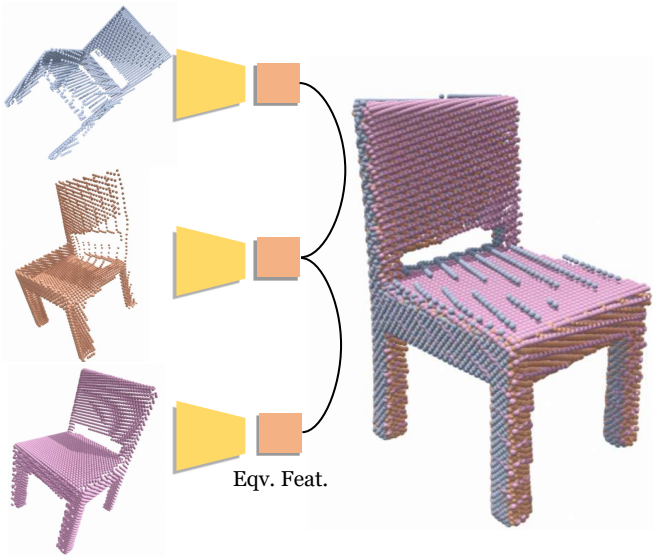


Figure 3.4: **Relative Pose Estimation of Matched Point Clouds**. Left is the inputs, middle is the embeddings, and right is the registration result.

Traditional methods [70, 71, 72, 73] extract superpoints and the corresponding feature descriptors, followed by feature matching and transformation estimation using inlier matches. We directly solve the registration problem using equivariant features.

$$\mathbf{X}^{t_1} = \mathbf{R}_{\mathrm{gt}}\mathbf{X}^{t_2} + \mathbf{t}_{\mathrm{gt}}. \tag{3.7}$$

Here $\mathbf{R}_{\text{gt}}$ and $\mathbf{t}_{\text{gt}}$ are ground truth. Following the rotation-equivariant property of the encoder Φ:

$$\mathbf{R}\Phi(\mathbf{X}, \mathbf{W}) = \Phi(\mathbf{R}\mathbf{X}, \mathbf{W}), \tag{3.8}$$

we can have

$$\Phi(\mathbf{X}^{t_1}, \mathbf{W}) = \Phi(\mathbf{R}\mathbf{X}^{t_2}, \mathbf{W}) = \mathbf{R}\Phi(\mathbf{X}^{t_2}, \mathbf{W}). \tag{3.9}$$

Therefore, rotations are preserved during encoding. Eq. 3.9 strictly holds only if the rotation centroids of the point clouds align with the origin of canonical space. We compensate for the small misalignment using $\mathbf{F}_{center} \in \mathbb{R}^3$ (see Appendix A.1). Then we have

$$\mathbf{F}_{eqv,1} = \mathbf{R}\mathbf{F}_{eqv,2}, \tag{3.10}$$

where $\mathbf{F}_{eqv,1} \in \mathbb{R}^{3\times256}$ and $\mathbf{F}_{eqv,2} \in \mathbb{R}^{3\times256}$ can be regarded as two pseudo point clouds with one-to-one correspondences in the latent feature space. The rotation can be solved using Singular Value Decomposition (SVD):

$$\mathbf{U},\ \mathbf{S},\ \mathbf{V}^{\mathrm{T}} = \mathrm{SVD}(\mathbf{F}_{eqv,1}\mathbf{F}_{eqv,2}^{\mathrm{T}}), \tag{3.11}$$

s.t.

$$\mathbf{R} = \mathbf{V}\mathbf{\Lambda}\mathbf{U}^{\mathrm{T}}, \tag{3.12}$$

where $\mathbf{\Lambda} = \mathrm{diag}(1, 1, \det(\mathbf{V}\mathbf{U}^{\mathrm{T}}))$. Zhu *et al.* name this method correspondence-free registration [74]. Intuitively, the network first infers the overall shape of the object, determines which part of an object each point cloud belongs to, and then aligns two point clouds based on their coverage w.r.t. the canonical shape.

3.5 Reconstruction

After associating and aligning the point clouds, we reconstruct their continuous surfaces using the output of the decoder. The space around the object is divided into 64^3 voxels, and the corresponding SDF values are queried through the decoder. Following the previous literature [6, 51, 16], we use Marching Cubes to extract the *zero level-set* as the surface of each object.

3.6 Joint Optimization

For reconstruction, we observe that Vector Neurons [47] suffer from *(i)* misalignment between the point cloud and reconstruction *(ii)* inaccurate reconstruction in occluded parts and *(iii)* inability to aggregate multiple scans. In our problem setting, we can accumulate point clouds of objects with varying observing angles and partialities that cover different parts of the objects. We propose a joint optimization procedure to refine the poses and shapes, enabling aggregation of multi-view multi-temporal point clouds. As shown in Figure 3.5, we initialize a pose graph $\mathbf{G} = \{\mathbf{R}_i\}_{i=0}^{T}$ that includes the relative poses among all pairs. The weights of the decoder are frozen during optimization. The pose graph and shape embeddings are iteratively refined through gradient descent optimization. Our assumption for the loss function is simple: all the points we observe should lie on the surface, which means that their SDF values should be *zero*. The optimizer minimizes the mean absolute value of the SDF values. We choose the parameter set with the smallest error as our final output. The pose graph* $\mathbf{G}$ connects

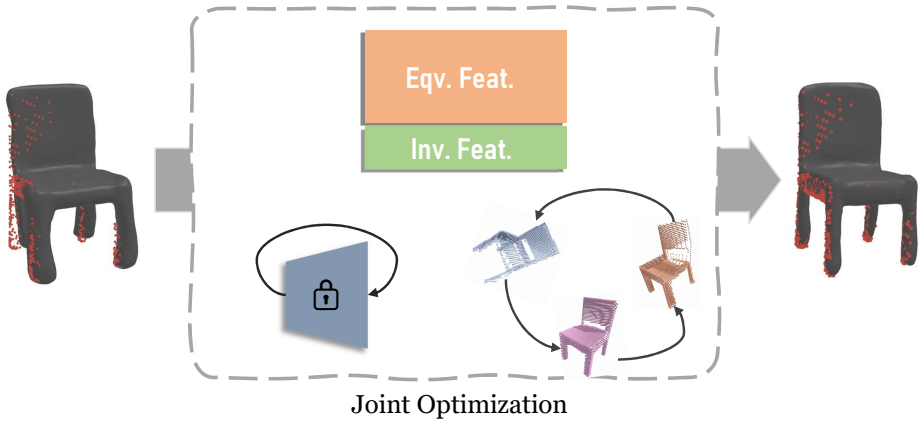


Figure 3.5: **Overview of Joint Optimization**. Left is the reconstruction before optimization and right is after optimization. The decoder indicates whether a point is inside or outside the surface. The optimization refines the pose graph and the reconstruction to make the point clouds lie on the surface.

the observations at multiple temporal stages. All point clouds that construct the

*To ensure that the gradient descent does not break the orthogonality of rotation matrices, we use unit quaternions to represent rotations, directly optimize on them, and normalize the quaternions after every update.

pose graph share the same set of feature embeddings, as they correspond to the same object. The pose graphs and feature embeddings contain pose and shape information, respectively. Through optimization, both pieces of information can be mutually and positively propagated. Moreover, this procedure implicitly fuses point clouds from multi-view multi-temporal observations. Details are illustrated in Algorithm 1.

Algorithm 1: Joint Optimization on Shape and Pose

`/* Aggregated Point Clouds of Object as Input */`
Data: $\{\mathbf{O}^t \in \mathbb{R}^{3\times N} | t \in \{0, 1, ..\mathrm{T}-1\}, N \in \mathbb{N}^+\}$
Input: $[\mathbf{F}_{inv}, \mathbf{F}_{eqv}, \mathbf{F}_{scale}, \mathbf{F}_{center}, \mathbf{R}_0 ... \mathbf{R}_{n-1}]$
`/* Initialization */`
$\mathbf{F}_{inv}, \mathbf{F}_{eqv}, \mathbf{F}_t, \mathbf{F}_{center} \leftarrow \Omega(\mathbf{O}^0)$;
for $t = 0;\ t < \mathrm{T};\ t = t + 1$ **do**
  $\mathbf{R}_t \leftarrow \mathcal{SVD}(\mathbf{F}_{eqv}^t, \mathbf{F}_{eqv}^{t+1})$;
end
$\epsilon \leftarrow 10^{-4}$: step size, $I \leftarrow 400$: number of iterations;
`/* Iterative Update */`
while $i < I$ **do**
  $\mathcal{L} \leftarrow 0$;
  for $t = 0;\ t < T;\ t = t + 1$ **do**
    for $\mathbf{q} \in \mathbb{R}^3$ in $\mathbf{O}^t \in \mathbb{R}^{3\times N}$ **do**
      $\mathbf{\Theta}_{NOCS} = \left\langle \mathbf{F}_{eqv}, \frac{\mathbf{O^t} - \mathbf{F}_{center}}{\mathbf{F}_{scale}} \right\rangle_{channel}$;
      $\mathcal{L} \leftarrow \mathcal{L} + \left| \mathrm{MLP}\left(\mathrm{cat}[\mathbf{F}_{inv}, \mathbf{\Theta}_{NOCS}]\right) \right| / N$;
    end
  end
  `/* Update Parameters using Adam Optimizer */`
  $[\mathbf{F}_{inv}, \mathbf{F}_{eqv}, \mathbf{F}_{center}, \mathbf{R}_0 ... \mathbf{R}_{n-1}] \leftarrow \mathrm{AdamUpdate}(\mathcal{L}, \epsilon)$;
  $i \leftarrow i + 1$;
end
`/* Terminate Iteration */`
Output: $[\mathbf{F}_{inv}, \mathbf{F}_{eqv}, \mathbf{F}_{center}, \mathbf{R}_0 ... \mathbf{R}_{n-1}] = \underset{[\mathbf{F}_{inv}, \mathbf{F}_{eqv}, \mathbf{F}_{center}, \mathbf{R}_0 ... \mathbf{R}_{n-1}]}{\arg\min} \mathcal{L}$

Chapter 4

Experiments and Results

In this chapter, we first demonstrate how we synthesize a dynamic room dataset (Section 4.1) and the processing of the training data (Section 4.2). In Section 4.4, we explain our experimental setting and evaluation metrics and compare our method with baselines. The ablation study in Section 4.5 justifies our design choices.

4.1 Dataset Generation

We synthesize the *dynamic room dataset* due to the lack of real-world data. We take four categories of objects (sofa, cabinet, chair, and table) from ShapeNet [66] as our *shape collection*. Firstly, objects are arbitrarily drawn from the *shape collection* and combined with a background of the room *i.e.*, walls and ground [21] to form a room. To imitate a long-term changing environment, object shapes remain unchanged but are randomly placed with different poses at each temporal stage. During generation, we ensure that objects in the room do not collide, are within the area of the room, and contact the ground. We store the transformations of objects as ground truth. We render point clouds and instance masks from the scene-level mesh through ray-casting with Pyrender [75]. We use 3D scene graphs [76] to store the layout and semantics of the scene. We drew 1000 shapes from ShapeNet and synthesized 500 dynamic rooms with moving objects.

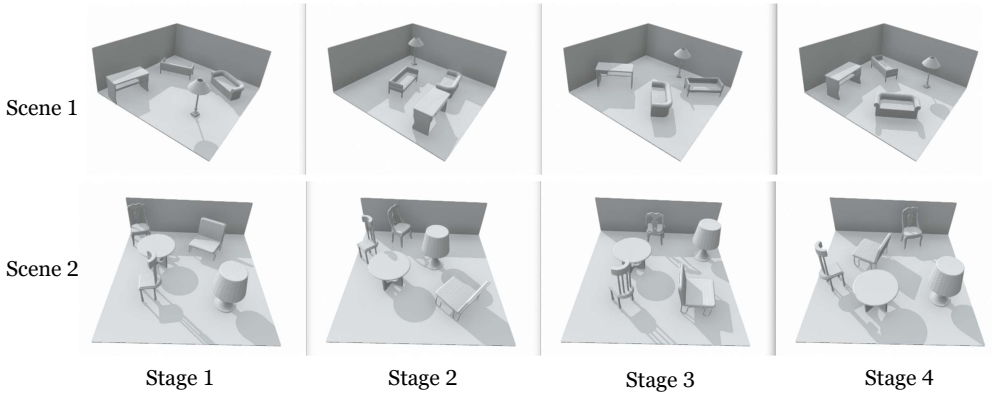


Figure 4.1: **Dynamic Room Dataset**. Each row represents the same room, and each column a temporal stage of the room.

4.2 Data Processing

The network is trained under the supervision of Signed Distance Fields (SDFs). We need to compute the SDF field for every shape to generate the training samples. We partly follow the processing pipelines of [19, 51], which include three steps: *(i)* making the mesh watertight, *(ii)* generating partial point cloud from depth rendering and *(iii)* sample SDF samples.

Watertight mesh for SDF. In computer graphics, watertight meshes usually describe meshes consisting of one closed surface. In this sense, watertight meshes do not contain holes and have a clearly defined interior [77]. CAD models of ShapeNet [66] are non-watertight and cannot be voxelized to signed distances, hindering further data processing. We use MeshFusion* [78] and Manifold-2† [79] to process raw meshes to make them watertight.

Point Cloud from Rendering. To mimic partial observations in real-world datasets, we render depth maps for meshes from multiple views. First, we construct a sphere around the mesh. The sphere completely covers the mesh and has the exact origin of the mesh. We uniformly sampled 24 points on the sphere as focal points of the depth camera. To ensure that the major part of the shape is

*https://github.com/davidstutz/mesh-fusion
†https://github.com/hjwdzh/Manifold

within the field of view (FOV), we put the principal point on the line connecting the focal point and the center of the sphere by solving the camera orientation $\mathbf{R}$:

$$z \begin{bmatrix} w/2 \\ h/2 \\ 1 \end{bmatrix} = \mathbf{K}[\mathbf{R}|\mathbf{T}] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{4.1}$$

where $\mathbf{K}$ and $[\mathbf{R}|\mathbf{T}]$ denote camera intrinsics and transformation from world to camera. w and h are the dimensions of the image and $[X, Y, Z, 1]^T$ are the coordinates of the focal point in the world frame. $\mathbf{R}$ is the unknown to solve. We render depth maps at 24 sampled positions for every mesh using OpenGL* [80] and backproject them to point clouds as network input.

Sampling Signed Distances. We use the GAPS† library to sample SDF values around meshes. We adopt two sampling strategies: uniform sampling and near-surface sampling. Uniform sampling captures the global structure of the shape and near-surface sampling captures high-frequency (detailed) signals. For each mesh, we sample 10^5 SDF samples, 50% uniform, and 50% near the surface. We also sample 10^5 points with normals on the surface for evaluation.

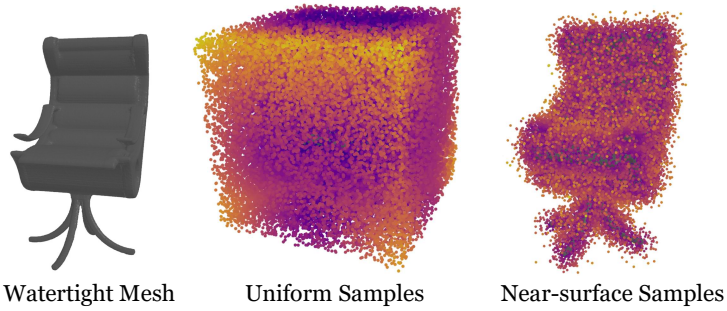


Figure 4.2: **SDF Sampling**. Color intensities denote SDF values.

*Additional conversion is needed since CG and CV have different definitions of camera axes.
†https://github.com/tomfunkhouser/gaps

4.3 Network Implementation

Backbone. We adopt Vector Neuron Dynamic Graph Convolutional Neural Network (VN-DGCNN [43, 51]) as encoder. The encoder outputs intermediate global embeddings $\mathbf{F}$, followed by four prediction heads that predict $(\mathbf{F}_{inv}, \mathbf{F}_{eqv}, \mathbf{F}_{scale}, \mathbf{F}_{center})$. Each prediction head is an 8-layer VN-MLP.

Training Loss. We train the VN encoder-decoder on our *shape collection* end-to-end. The loss $\mathcal{L}$ is calculated as:

$$\mathcal{L} = \omega_{\text{SDF}}\mathcal{L}_{\text{SDF}} + \omega_{\text{center}}\mathcal{L}_{\text{center}} + \omega_{\text{scale}}\mathcal{L}_{\text{scale}}. \tag{4.2}$$

Similar to [81, 51], we assign larger weight to near-surface samples to improve the reconstruction accuracy:

$$\mathcal{L}_{\text{SDF}} = \frac{\lambda_{\text{near}} \sum_{x \in \mathcal{Q}_{\text{near}}} \mathcal{L}_1(x) + \lambda_{\text{far}} \sum_{x \in \mathcal{Q}_{\text{far}}} \mathcal{L}_1(x)}{|\mathcal{Q}_{\text{near}}| + |\mathcal{Q}_{\text{far}}|}. \tag{4.3}$$

Here $\mathcal{L}_1 = |\text{SDF}(\mathbf{x})_{\text{pred}} - \text{SDF}_{\text{gt}}(\mathbf{x})|$. $\mathcal{Q}_{\text{far}}$ denotes the set of samples with SDF values larger than 0.1 and $\mathcal{Q}_{\text{near}}$ smaller than 0.1. λ_{far} and λ_{near} are set to 0.5 and 1.0, respectively. $\mathcal{L}_{\text{scale}} = |1 - \mathbf{F}_{scale}|$ and $\mathcal{L}_{\text{center}} = ||\mathbf{F}_{center}||_2$ are two regularization terms with ground truth fixed at 1 and $[0, 0, 0]$. Following EFEM [51], we set $\omega_{\text{SDF}} = 1.0$, $\omega_{\text{center}} = 0.2$ and $\omega_{\text{scale}} = 0.001$.

Training Details. We train the network for 160000 steps using Adam optimizer [82] with batch size 32 and initial learning rate 10^{-4} that is linearly decayed by 0.3 after 80000, 120000 and 140000 steps. The network and training framework are implemented in PyTorch [83]. During training, four types of data augmentations are applied to input point cloud:

- Random rescaling
- Random centroid translation
- Random points addition & removal
- Random plane addition & removal

We do not apply rotation augmentations as the network is equivariant to SO(3) and can generalize rotations at test time.

4.4 Experimental Evaluation

In this section, we describe our evaluation settings and metrics (Section 4.4.1) for shape matching, pose estimation, and reconstruction. Then we introduce the state-of-the-art baselines (Section 4.4.2) for the three tasks and compare our method with the baselines. We then proceed to evaluate our full pipeline quantitatively and showcase the qualitative results. Finally, we justify our design choice in the ablation study (Section 4.5).

4.4.1 Evaluation Metrics

We evaluate three tasks separately with independent metrics.

Shape Matching. We take inspiration from 2D image feature matching [84, 85] to evaluate 3D shape matching [86]. We create two evaluation modes: easy and hard. In easy mode, we retrieve only one shape from each category from the *shape collection* and we always match two lists of four shapes in current. In hard mode, we draw a random number of shapes from the *shape collection*. Shapes have different orientations under both modes. Matching Recall (MR) is computed from ground truth, and the confusion matrix is computed with the diagonal correspondences being the correct matches.

Pose Estimation. We follow the standardized point cloud registration [70, 72] and evaluate relative pose estimation on our dataset. We calculate the relative rotation error (RRE) - geodesic distance of the predicted rotation w.r.t. ground truth [87] and registration recall (RR) - the fraction of estimates whose RRE $< 5°$ and RRE $< 30°$.

$$\text{RRE} = \arccos\left(\frac{\text{trace}\left(\mathbf{R}^{\text{T}}\overline{\mathbf{R}}\right) - 1}{2}\right) \tag{4.4}$$

Reconstruction. To assess the reconstruction fidelity, we compute the Chamfer Distance (CD) and Volumetric intersection over Union (IoU) [21, 19]. Consider $\mathcal{M}_{\text{pred}}$ and $\mathcal{M}_{\text{GT}}$ the set of points inside or outside the predicted mesh and the ground truth mesh. We compute the two-way chamfer distance *i.e.* the accuracy

and completeness score of $\mathcal{M}_{\text{pred}}$ w.r.t. $\mathcal{M}_{\text{GT}}$:

$$\begin{aligned} \text{Accuracy}\left(\mathcal{M}_{\text{pred}} \mid \mathcal{M}_{\text{GT}}\right) &\equiv \frac{1}{|\partial\mathcal{M}_{\text{pred}}|}\int_{\partial\mathcal{M}_{\text{pred}}} \min_{\mathbf{q}\in\partial\mathcal{M}_{\text{GT}}} \|\mathbf{p}-\mathbf{q}\|\mathrm{d}\mathbf{p} \\ \text{Completeness}\left(\mathcal{M}_{\text{pred}} \mid \mathcal{M}_{\text{GT}}\right) &\equiv \frac{1}{|\partial\mathcal{M}_{\text{GT}}|}\int_{\partial\mathcal{M}_{\text{GT}}} \min_{\mathbf{p}\in\partial\mathcal{M}_{\text{pred}}} \|\mathbf{p}-\mathbf{q}\|\mathrm{d}\mathbf{q} \end{aligned} \tag{4.5}$$

Here $\partial\mathcal{M}_{\text{GT}}$ and $\partial\mathcal{M}_{\text{GT}}$ are the surfaces of $\mathcal{M}_{\text{pred}}$ and $\mathcal{M}_{\text{GT}}$, respectively. The Chamfer-L_1 can be defined as below:

$$\begin{aligned} &\text{Chamfer-}L_1\left(\mathcal{M}_{\text{pred}}, \mathcal{M}_{\text{GT}}\right) = \\ &\quad\frac{1}{2}\left(\text{Accuracy}\left(\mathcal{M}_{\text{pred}} \mid \mathcal{M}_{\text{GT}}\right) + \text{Completeness}\left(\mathcal{M}_{\text{pred}} \mid \mathcal{M}_{\text{GT}}\right)\right) \end{aligned} \tag{4.6}$$

The Volumetric IoU is equal to the intersection devided by the union of two sets:

$$\text{IoU}\left(\mathcal{M}_{\text{pred}}, \mathcal{M}_{\text{GT}}\right) \equiv \frac{|\mathcal{M}_{\text{pred}} \cap \mathcal{M}_{\text{GT}}|}{|\mathcal{M}_{\text{pred}} \cup \mathcal{M}_{\text{GT}}|}. \tag{4.7}$$

4.4.2 Baselines

We compare the performance of our method to state-of-the-art baselines.

Shape Matching. We use MendNet [30] as a baseline for shape matching. MendNet consists of a pointnet encoder and a deepsdf decoder. We reimplement[‡] the model and train it on our *shape collection* from scratch with the same training hyperparameters.

Pose Estimation. We adopt three baseline methods. FPFH+RANSAC [73, 88] is one hand-crafted baseline. FPFH is used to extract and match point features, followed by RANSAC filtering. RPMNet [72] and Predator [70] are two learning-based baselines. We use the weights from the official implementations.

Reconstruction. MendNet [30], OccNet [19] and ConvONet [21] are the three baseline methods for reconstruction. We train MendNet and OccNet on our *shape collection* under the same training settings. We directly use the weights of ConvONet pre-trained on ShapeNet, as its training set is a superset of ours.

[‡]The source code of this work is unavailable.

4.4.3 Quantitative Results

Method	I/I		I/SO(3)		SO(3)/SO(3)	
	Easy	Hard	Easy	Hard	Easy	Hard
MendNet [30] w/ NN	96.88	93.31	30.53	22.22	43.05	33.52
MendNet [30] w/ Sinkhorn	**99.80**	**98.26**	31.30	22.46	46.67	36.47
Ours w/ NN	88.23	80.49	88.23	80.49	88.23	80.49
Ours w/ Sinkhorn	95.80	89.15	**95.80**	**89.15**	**95.80**	**89.15**

Table 4.1: **Results of Shape Matching Recall [%]** (↓). I/SO(3) denotes that the model is trained in canonical poses and evaluated in arbitrary poses. NN denotes the nearest neighbour matching.

Shape Matching. We evaluate model performances on our dataset with six settings: {I/I, I/SO(3), SO(3)/SO(3)} × {Easy, Hard}. Table 4.1 shows that MendNet better fits the data under the ideal setting I/I but cannot generalize SO(3). MendNet's performance is improved with rotation augmentation. Remarkably, our model that was trained only with canonical poses (I) can seamlessly generalize SO(3). The (hard) matching recall of MendNet drops to 36.47%, which basically fails to output meaningful shape pairs, while our model remains high with 89.15% recall. The advantage of our method can be attributed to the rotation invariance design. Lastly, the Sinkhorn algorithm [4, 5] can provide a performance gain around 3%–6% over nearest-neighbour matching. Figure A.2 (Appendix) shows three similarity matrices computed from the embeddings.

Pose Estimation. Table 4.2 shows the performance of FPFH, RPMNet, Predator and ours. Figure 4.3 depicts the error cumulative distribution functions (ECDF). Our encoder-only method can achieve on-par RRE with three baselines. With joint optimization (full model), the MeanRRE and MedRRE are reduced from 4.98° and 4.32° to 3.98° and 2.56°, respectively. Ours with joint optimization exceeds all three baselines in terms of MeanRRE, MedRRE and Recall@5°. Predator has the highest recall of less than 30°. We consider Recall@30° as a minor metric, since the largest rotation angle in our dataset is 60° and 30° is already half of largest the ground truth value. The EFDF curves also show that our model outperforms under different thresholds. Additionally, performance can be further refined using ICP, with improvements on all metrics. Conceptually,

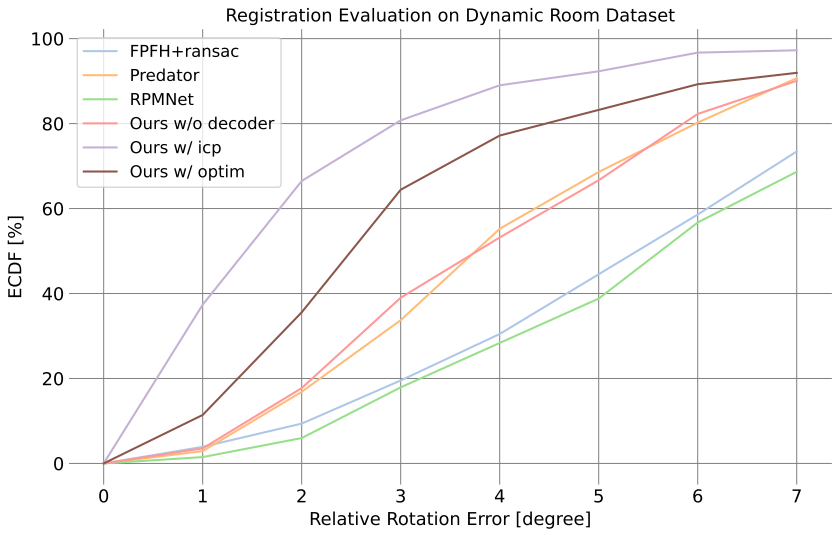


Figure 4.3: **ECDF Curve of Relative Rotation Errors**. ECDF - empirical cumulative distribution functions, compute the quantiles at different thresholds.

our method solves the point cloud registration problem differently. The baseline methods extract distinct geometry feature vectors of both point clouds and find the correspondences to solve poses. Our method implicitly aligns the posed point cloud to a canonical part of the object and infers the complete shape of the object. Through joint optimization, our method adjusts the misalignment and refines the poses from coarse to fine.

Reconstruction. We compare the reconstruction results of our method with three baselines in Table 4.3. We achieve the highest IoU 0.57 and the second best $L1$ -Chamfer 2.41. Our method outperforms MendNet and OccNet on both metrics, and it can be accredited to the VN-DGCCN backbone, enabling better expressity. ConvONet has the best $L1$-Chamfer due to its local conditioning using tri-plane feature grids.

Method	MeanRRE [°] ↓	MedRRE [°] ↓	Recall@5° ↑	Recall@30° ↑
FPFH [73]	6.38	6.07	0.28	0.88
RPMNet [72]	6.90	6.45	0.13	0.72
Predator [70]	<u>4.38</u>	<u>3.87</u>	<u>0.59</u>	**0.94**
Ours w/ encoder	4.98	4.32	0.47	0.88
Ours w/ joint opt	**3.98**	**2.56**	**0.62**	<u>0.91</u>
Ours w/ ICP	2.36	1.38	0.84	0.95

Table 4.2: **Results of Pose Estimation**. We compute the mean and median of RRE, and the recall of 5°and 30°. We report our performances with the encoder only, with joint optimization. We further evaluate the performance of our full model with iterative closest point (ICP) to show the best possible performance.

Method	Train/Eval	$L1$-Chamfer ↓	IoU ↑
MendNet [30]	I/I	3.58	<u>0.52</u>
MendNet [30]	SO(3)/SO(3)	10.33	0.25
OccNet [19]	I/I	4.22	0.46
ConvONet [21]	I/I	**1.50**	0.42
Ours	I/SO(3)	<u>2.41</u>	**0.57**

Table 4.3: **Evaluation of Surface Reconstruction**. We report $L1$-Chamfer [$\times 10^{-3}$] and volumetric IoU. The train and evaluation settings are highlighted.

4.4.4 Qualitative Results

In this section, we show the quantitative results of *1)* point cloud registration *2)* scene-level reconstruction and *3)* temporal scene reconstruction.

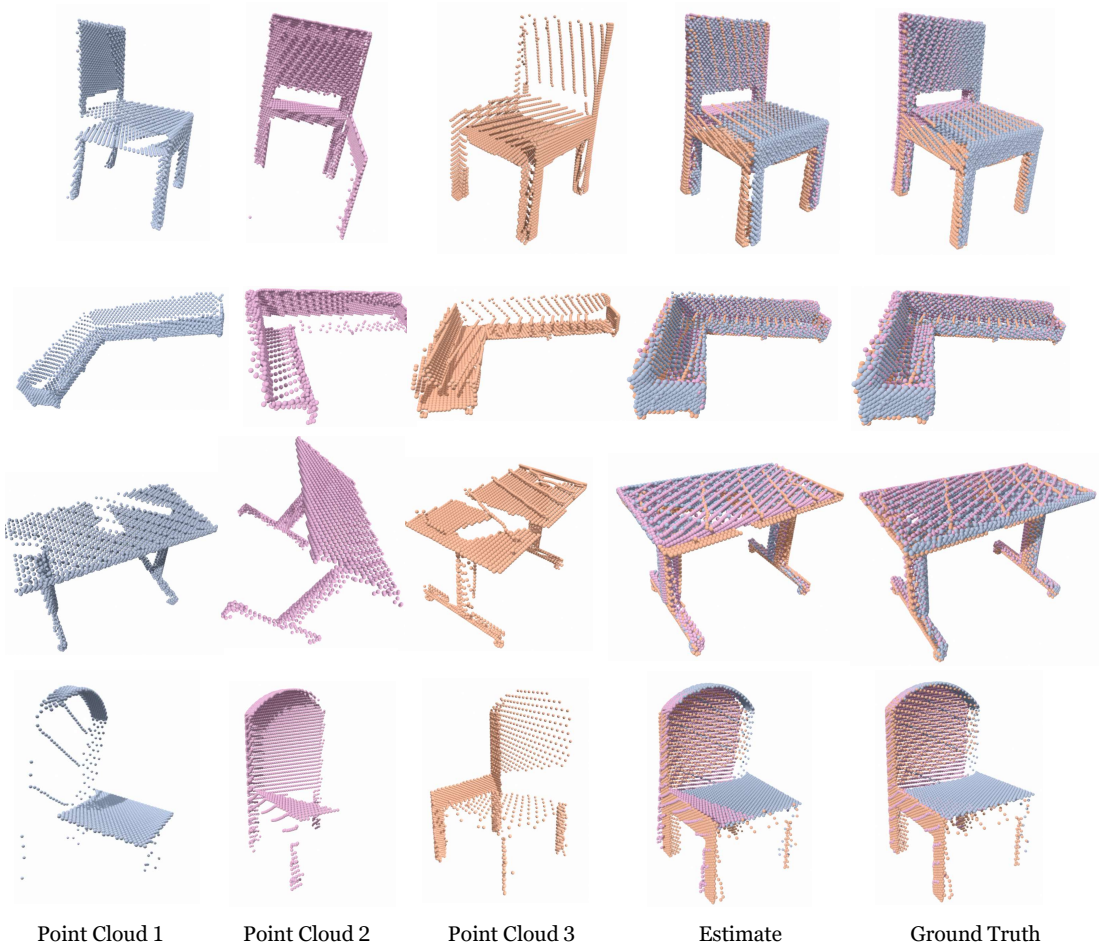


Figure 4.4: **Qualitative Results of Multi-temporal Point Clouds**. From left to right: input point clouds ×3, registration estimtate and ground truth.

Point Cloud Registration. Figure 4.4 shows the multi-stage (multi-view) point cloud registration of objects from different categories using our full model. The second last row is an example of the registration of symmetrical objects. Our

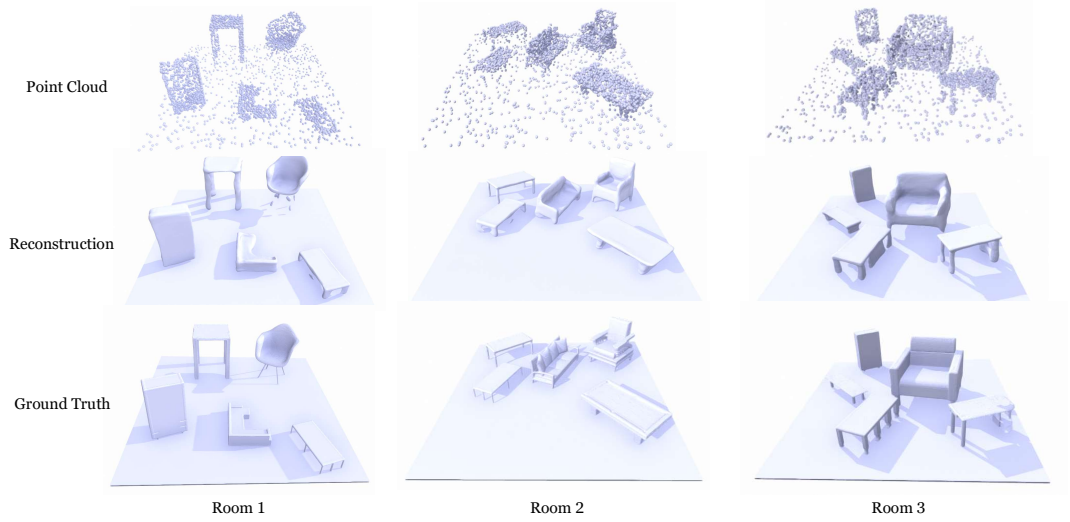


Figure 4.5: **Qualitative Results of Scene-level Reconstruction**. From top to bottom: point cloud, reconstruction, and ground truth.

estimate and the ground truth rotation of the orange point cloud are complementary to 180°. The bottom row of Figure 4.4 is a remarkable example of our successful registrations: input point clouds have extremely low overlap, making it difficult to find correspondences in the overlap region. Our method does not rely on finding the overlap but rather on completing the surface.

Scene-level Reconstruction. Figure 4.5 shows three examples of scene-level reconstruction. Here, we use ground-truth object masks. The objects are fed through the network sequentially, reconstructed under corresponding poses, and merged to generate the room-level reconstruction. We also predict a neural field for the background[††]. We found that the background can be easily learned and accurately reconstructed as it has simple and sharp geometry. Some high-frequency details *e.g.* the pillows on the couch in room 2 are smeared due to global conditioning. Global conditioning methods can perform better at learning shape priors but worse at reconstructing local details than local conditioning.

Dynamic Room Reconstruction. We show the reconstruction of one room with multiple time stages in Figure 4.6. We disentangle the dynamic room

[††]We train a network with the same architecture to reconstruct the background.

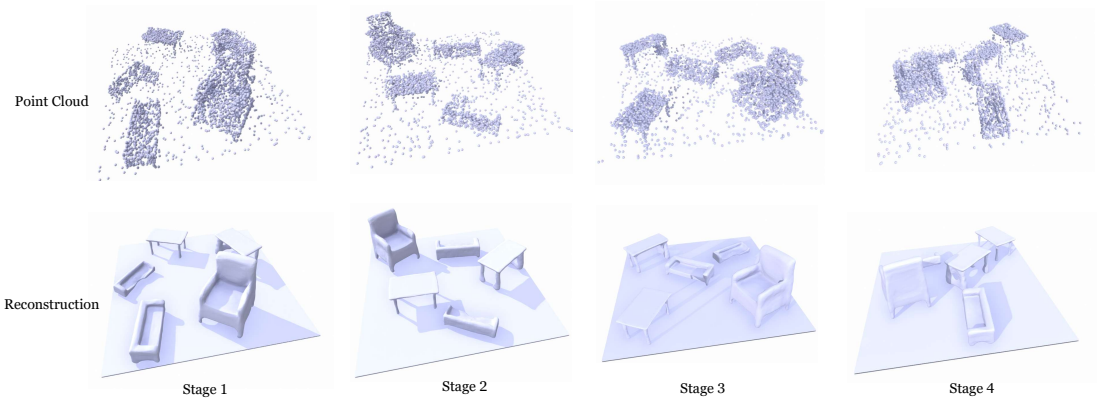


Figure 4.6: **Qualitative Results of Dynamic Room Reconstruction**. Point clouds (top) and reconstruction (bottom) of the same room at different stages.

through our proposed match→align→reconstruct pipeline with optimization. We use stage 1 as a reference and then propagate poses to other stages. Point clouds of objects at different stages are aggregated. Our method is able to accurately recover the motion of objects and reconstruct the surfaces accordingly.

4.5 Ablation Study

PE & Decoder	$L1$-Chamfer ↓	$L1$-USS ↓	$L1$-NSS ↓	IoU ↑
Inner. +DeepSDF	**3.58**	**0.027**	**0.015**	**0.454**
Inner. +MLP	3.65	0.029	0.016	0.436
Inv. +DeepSDF	7.34	0.041	0.019	0.343
Inv. +MLP	7.04	0.037	0.021	0.325

Table 4.4: **Ablation Study on Decoding Strategies**. Inner product (Inner.) and invariant feaures (Inv.). $L1$-Chamfer [$\times 10^{-3}$], USS - Uniform SDF Sampling, NSS - Near Surface Sampling.

Decoding Strategy. DeepSDF decoder, as shown in 4.7, is different from regular multi-layer perceptrons (MLP), where DeepSDF includes re-concatenation. Positional encoding can be achieved in two ways: by directly using 3D coordinates and by computing the inner product between query coordinates and equivariant embeddings. We ablate the decoding strategy in two aspects: decoder architecture and positional encoding. As shown in Table 4.4, using the inner product as positional encoding and DeepSDF as decoder has the best performance. The inner product can unproject the 3D coordinates into a high-dimensional space, providing more high-frequency information. The re-concatenation in DeepSDF can reconstruct more regularized and complete surfaces, making the network easier to train.

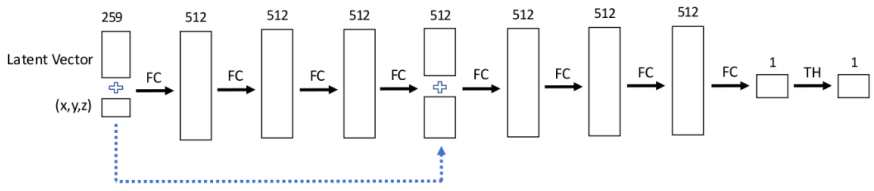


Figure 4.7: **DeepSDF Architecture** [6]. The latent code and query coordinates are concatenated as the input. At the middle layer of the decoder, the input feature is concatenated with the intermediate feature as the new input.

Category Agnosticity. [6, 89] learn category-level priors for shape reconstruc-

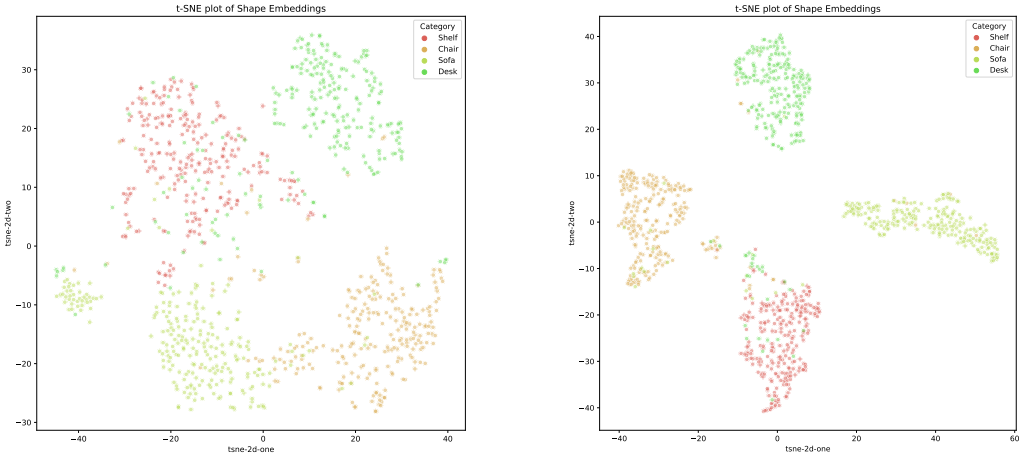
tion. We seek to avoid a multi-network solution because such pipelines will introduce extra classification networks. We compare the performance of training a single network to that of training multiple networks (see Table 4.5). We find that single-category and multi-category are comparable in terms of Mean $L1$-Chamfer, but single-category is better than multi-category on Median $L1$-Chamfer. This experiment shows that shape priors can be better learned with category-specific networks with the cost of more parameters and external reliance on classification.

Category	Shelf	Chair	Sofa	Desk	All
Trained on	Mean $L1$-Chamfer [$\times 10^{-3}$] $\downarrow$				
Single Category	**2.218**	4.736	2.033	**2.841**	**2.626**
Multiple Categories	2.587	**4.016**	**1.350**	4.348	3.075
	Median $L1$-Chamfer [$\times 10^{-3}$] $\downarrow$				
Single Category	**1.434**	**2.245**	**0.782**	**1.728**	**1.504**
Multiple Categories	2.072	2.252	1.049	2.108	1.741

Table 4.5: **Ablation Study on Category Agnosticity**. Single category refers to training a network for each shape category and multiple category refers to training one network for multiple categories.

Impact of Classification Head. Following the previous ablation, we explore the possibility of adding an additional classification head to the network. The head is a VN-linear layer, with the sigmoid function being the activation layer. To train the head, we introduce cross-entropy loss to the loss proxy. From experiments, our classification head can achieve 91% accuracy. Figure 4.8 shows the distributions of latent embeddings. The classification head can make the embeddings better regularized and more compact within each category. One potential disadvantage of the classification head is that the in-category distribution becomes narrower, which might harm the performance on shape matching of point clouds from the same class.

Parameters to Optimize. To find the best set of parameters for joint optimization, we experimented with different optimization settings. Table 4.7 shows the optimization setting on the left and the corresponding performance on the right. The best performance is achieved when we optimize all four parameters.



(a) w/o classification head (b) w/ classification head

Figure 4.8: **t-SNE Plots of Shape Embeddings**. Left is the latent distribution of the model trained without a classification head, and right is the distribution with a classification head.

The step sizes of parameters during optimization are different. We mainly optimize $\mathbf{F_{eqv}}$ and $\mathbf{R}_0 ... \mathbf{R}_{n-1}$ and only apply small adjustments to $\mathbf{F_{inv}}$ and $\mathbf{F_t}$. The step sizes are set as follows:

Parameter	$\mathbf{F_{inv}}$	$\mathbf{F_t}$	$\mathbf{F_{eqv}}$	$\mathbf{R}_0 ... \mathbf{R}_{n-1}$
Step Size $[\times 10^{-5}]$	1.0	1.0	10.0	5.0

Table 4.6: **Step Sizes of Parameters during Joint Optimization**.

Optimizing Parameters				Evaluation Metrics			
				RRE [°]		$L1$-Chamfer [$\times 10^{-3}$]	
$\mathbf{R}_0 ... \mathbf{R}_{n-1}$	$\mathbf{F}_{\mathbf{eqv.}}$	$\mathbf{F}_{\mathbf{inv.}}$	$\mathbf{F}_{\mathbf{t}}$	Mean ↓	Median ↓	Mean ↓	Median ↓
✗	✗	✗	✗	11.85	5.28	5.37	2.31
✓	✗	✗	✗	<u>7.88</u>	3.45	5.38	2.30
✓	✗	✗	✓	**7.81**	3.41	5.67	2.30
✓	✓	✗	✓	8.38	<u>1.81</u>	<u>3.95</u>	<u>2.06</u>
✓	✓	✓	✓	8.70	**1.71**	**3.81**	**1.86**

Table 4.7: **Ablation Study on Optimization Settings**. ✗ denotes fixing the parameter during optimization and vice versa (✓).

4.5.1 Temporal Aggregation

Our method is capable of aggregating point clouds and integrating multi-temporal information. In this section, we showcase the performance gain of our aggregation in both qualitative and quantitative results. Figure 4.9 shows the evolution of reconstruction as the number of stages increases. With the point cloud from one stage only, the observation is partial and sparse, causing errors and ambiguities in the reconstruction. As we accumulate more points, the quality of the reconstruction improves accordingly (from left to right in Figure 4.9).

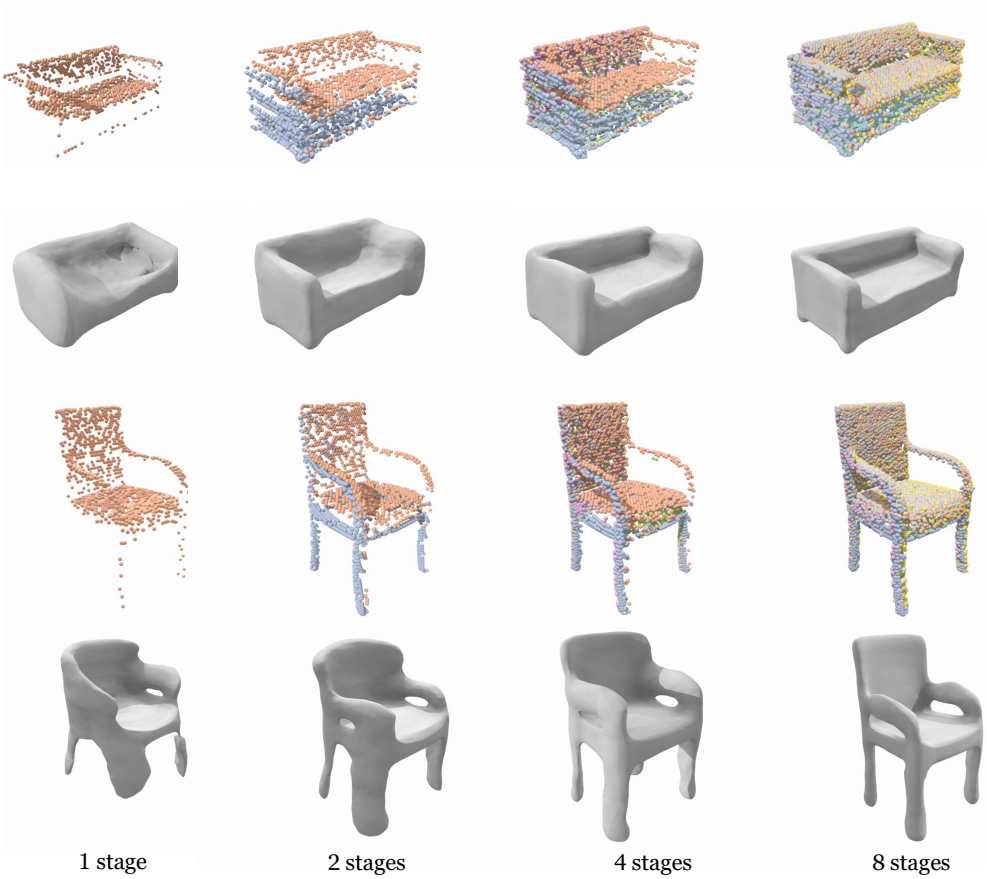


Figure 4.9: **Qualitative Results of Object-Level Temporal Aggregation**. From left to right: point clouds accumulated from 1, 2, 4 and 8 temporal stages. Points of the same color refer to the same temporal stage.

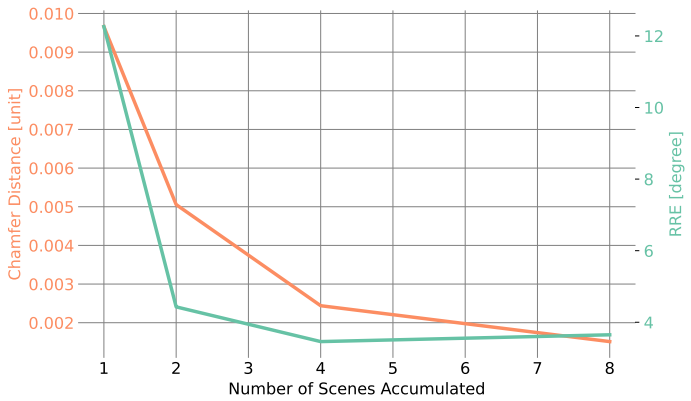


Figure 4.10: **Quantitative Results of Temporal Aggregation.** The horizontal axis is the number of scenes accumulated. The vertical axes denote $L1$-Chamfer $\downarrow$ (left) and relative rotation error $\downarrow$ (right).

We quantify the performance of pose estimation and reconstruction during the temporal aggregation in Figure 4.10. We can see apparent improvements when the aggregation starts at 1 stage and the performance of both tasks begins to saturate after four stages.

Chapter 5

Conclusion and Outlook

This thesis investigated a novel way of parsing the changing environment and benefitting from equivariant deep neural networks. We first made a clear definition of the problem of a changing environment and conducted a comprehensive literature review. Specifically, shape matching, point cloud registration, and shape reconstruction are the three tasks that we formulated as our framework. We first synthesized a dynamic room dataset as the basis of our experimentation. We utilize an equivariant network to store object shape priors, serving as the backbone to realize our framework. We make delicate manipulation of the intermediate features (invariant and equivariant) of the network based on the information they store (*e.g.* structure, rotations, and local geometry). We solve the problem from coarse to fine with our joint optimization procedure. Our experiments demonstrate the feasibility of solving the trinity problem using a single network and the superior performance of our method compared to existing baselines. There are several limitations with our method and many more to explore in the future.

5.1 Limitations

Symmetrical and Identical Shapes. Our method faces challenges on objects with more than one symmetric axis or point symmetry because such symmetries cause ambiguities in the rotations. Another challenging scenario is a scene with more than one identical shapes. Identical shapes can cause ambiguities during shape matching and lead to multiple solutions. We avoid such problems during our dataset generation by assuming non-repetitive shapes in the scene. This assumption can be problematic with scenes like offices and classrooms where there exist a large number of chairs and desks with identical shapes.

Real-World Datasets. We tested and evaluated our method on the synthetic dataset. There are several challenges before we can make our method work on real datasets: *1)* Selection of instance segmentation networks and integration with our pipeline *2)* Training the network with more categories and looking into the impact of the number of increasing classes *3)* Dealing with noisy point clouds as input, since the off-the-shelf point cloud segmentation networks might predict incomplete or wrong segmentations.

5.2 Future Work

The first thing in future work is to find a real-world dataset and test our framework on it. Our framework can be further integrated into a long-term 4D scene representation, namely a spatiotemporal scene graph (ST-Scene Graph) to conduct downstream tasks like scene reasoning. The global conditioning of reconstruction ignores local geometric details and needs improvements like locally equivariant networks. It is intricate to strike a balance between learned shape priors and flexibility for run-time optimization. It is intriguing to see whether our method, originally designed for indoor data, can generalize to outdoor scenarios, *e.g.* autonomous driving. With the development of pre-trained text -image embedding models (*e.g.* CLIP [90]), our method can take advantage of open-world scene understanding [91] and extend to open-scene object relocalization and reconstruction. Another exciting line of research is the capture or creation of large 3D datasets with temporal changes and standardize the dynamic scene understanding paradigm.

Bibliography

[1] Y. Xie, T. Takikawa, S. Saito, O. Litany, S. Yan, N. Khan, F. Tombari, J. Tompkin, V. Sitzmann, and S. Sridhar, "Neural fields in visual computing and beyond," in *Computer Graphics Forum*, vol. 41, no. 2, 2022, pp. 641–676.

[2] J. Schult, F. Engelmann, A. Hermans, O. Litany, S. Tang, and B. Leibe, "Mask3d: Mask transformer for 3d semantic instance segmentation," in *ICRA*, 2023.

[3] T. Vu, K. Kim, T. M. Luu, T. Nguyen, and C. D. Yoo, "Softgroup for 3d instance segmentation on point clouds," in *CVPR*, 2022.

[4] R. Sinkhorn and P. Knopp, "Concerning nonnegative matrices and doubly stochastic matrices," *Pacific Journal of Mathematics*, vol. 21, no. 2, pp. 343–348, 1967.

[5] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," 2013.

[6] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed distance functions for shape representation," in *CVPR*, 2019.

[7] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[8] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *CVPR*, 2020.

[9] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *CVPR*, 2020.

[10] E. Iannucci, Z. Chen, I. Armeni, M. Pollefeys, H. Pfister, and J. Beyer, "Arrow: A real-time ar rowing coach," 2023.

[11] S. Huang, Z. Gojcic, J. Huang, A. Wieser, and K. Schindler, "Dynamic 3d scene analysis by point cloud accumulation," in *ECCV*, 2022.

[12] K. Li, H. Rezatofighi, and I. Reid, "Moltr: Multiple object localization, tracking and reconstruction from monocular rgb videos," *Robotics and Automation Letters*, vol. 6, no. 2, pp. 3341–3348, 2021.

[13] B. Wen, J. Tremblay, V. Blukis, S. Tyree, T. Müller, A. Evans, D. Fox, J. Kautz, and S. Birchfield, "Bundlesdf: Neural 6-dof tracking and 3d reconstruction of unknown objects," in *CVPR*, 2023.

[14] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *ICCV*, 2009.

[15] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "Mot16: A benchmark for multi-object tracking," *arXiv:1603.00831*, 2016.

[16] M. Runz, K. Li, M. Tang, L. Ma, C. Kong, T. Schmidt, I. Reid, L. Agapito, J. Straub, S. Lovegrove *et al.*, "Frodo: From detections to 3d objects," in *CVPR*, 2020.

[17] J. Wald, A. Avetisyan, N. Navab, F. Tombari, and M. Nießner, "Rio: 3d object instance re-localization in changing indoor environments," in *CVPR*, 2019.

[18] A. Adam, T. Sattler, K. Karantzalos, and T. Pajdla, "Objects can move: 3d change detection by geometric transformation consistency," in *ECCV*, 2022.

[19] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3d reconstruction in function space," in *CVPR*, 2019.

[20] Z. Chen and H. Zhang, "Learning implicit fields for generative shape modeling," in *CVPR*, 2019.

[21] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *ECCV*, 2020.

[22] J. Huang, Z. Gojcic, M. Atzmon, O. Litany, S. Fidler, and F. Williams, "Neural kernel surface reconstruction," in *CVPR*, 2023.

[23] L. Yen-Chen, P. Florence, J. T. Barron, A. Rodriguez, P. Isola, and T.-Y. Lin, "inerf: Inverting neural radiance fields for pose estimation," in *IROS*, 2021.

[24] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[25] M. Z. Irshad, S. Zakharov, R. Ambrus, T. Kollar, Z. Kira, and A. Gaidon, "Shapo: Implicit representations for multi-object shape, appearance, and pose optimization," in *ECCV*, 2022.

[26] F. Williams, Z. Gojcic, S. Khamis, D. Zorin, J. Bruna, S. Fidler, and O. Litany, "Neural fields as learnable kernels for 3d reconstruction," in *CVPR*, 2022.

[27] Y. Pan, F. Magistri, T. Läbe, E. Marks, C. Smitt, C. McCool, J. Behley, and C. Stachniss, "Panoptic mapping with fruit completion and pose estimation for horticultural robots," *arXiv:2303.08923*, 2023.

[28] W. Yifan, S. Wu, C. Oztireli, and O. Sorkine-Hornung, "Iso-points: Optimizing neural implicit surfaces with hybrid representations," in *CVPR*, 2021.

[29] H. Chen, M. Gwilliam, S.-N. Lim, and A. Shrivastava, "Hnerv: A hybrid neural representation for videos," in *CVPR*, 2023.

[30] S. Duggal, Z. Wang, W.-C. Ma, S. Manivasagam, J. Liang, S. Wang, and R. Urtasun, "Mending neural implicit modeling for 3d vehicle reconstruction in the wild," in *WACV*, 2022.

[31] V. Sitzmann, E. Chan, R. Tucker, N. Snavely, and G. Wetzstein, "Metasdf: Meta-learning signed distance functions," 2020.

[32] G. Chou, I. Chugunov, and F. Heide, "Gensdf: Two-stage learning of generalizable signed distance functions," vol. 35, pp. 24 905–24 919, 2022.

[33] P. Truong, M.-J. Rakotosaona, F. Manhardt, and F. Tombari, "Sparf: Neural radiance fields from sparse and noisy poses," in *CVPR*, 2023.

[34] S. Huang, Z. Gojcic, Z. Wang, F. Williams, Y. Kasten, S. Fidler, K. Schindler, and O. Litany, "Neural lidar fields for novel view synthesis," 2023.

[35] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas, "Normalized object coordinate space for category-level 6d object pose and size estimation," in *CVPR*, 2019.

[36] M. Shan, Q. Feng, Y.-Y. Jau, and N. Atanasov, "Ellipsdf: joint object pose and shape optimization with a bi-level ellipsoid and signed distance function description," in *ICCV*, 2021.

[37] J. Huang, H. Wang, T. Birdal, M. Sung, F. Arrigoni, S.-M. Hu, and L. J. Guibas, "Multibodysync: Multi-body segmentation and motion estimation via 3d scan synchronization," in *CVPR*, 2021.

[38] K. Li, D. DeTone, Y. F. S. Chen, M. Vo, I. Reid, H. Rezatofighi, C. Sweeney, J. Straub, and R. Newcombe, "Odam: Object detection, association, and mapping using posed rgb video," in *ICCV*, 2021.

[39] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *IJRR*, 2013.

[40] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *CVPR*, 2017.

[41] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3d semantic parsing of large-scale indoor spaces," in *CVPR*, 2016.

[42] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *CVPR*, 2017.

[43] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics (ToG)*, vol. 38, no. 5, pp. 1–12, 2019.

[44] W. Sun, W. Jiang, E. Trulls, A. Tagliasacchi, and K. M. Yi, "Acne: Attentive context normalization for robust permutation-equivariant learning," in *CVPR*, 2020.

[45] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley, "Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds," *arXiv:1802.08219*, 2018.

[46] F. Fuchs, D. Worrall, V. Fischer, and M. Welling, "Se (3)-transformers: 3d roto-translation equivariant attention networks," 2020.

[47] C. Deng, O. Litany, Y. Duan, A. Poulenard, A. Tagliasacchi, and L. J. Guibas, "Vector neurons: A general framework for so (3)-equivariant networks," in *CVPR*, 2021.

[48] M. Zhu, M. Ghaffari, W. A. Clark, and H. Peng, "E2pn: Efficient se(3)-equivariant point network," in *CVPR*, 2023.

[49] Y. Chen, B. Fernando, H. Bilen, M. Nießner, and E. Gavves, "3d equivariant graph implicit functions," in *ECCV*, 2022.

[50] E. Chatzipantazis, S. Pertigkiozoglou, E. Dobriban, and K. Daniilidis, "Se(3)-equivariant attention networks for shape reconstruction in function space," in *ICLR*, 2023.

[51] J. Lei, C. Deng, K. Schmeckpeper, L. Guibas, and K. Daniilidis, "Efem: Equivariant neural field expectation maximization for 3d object segmentation without scene supervision," in *CVPR*, 2023.

[52] A. Mustafa, H. Kim, J.-Y. Guillemaut, and A. Hilton, "Temporally coherent 4d reconstruction of complex dynamic scenes," in *CVPR*, 2016.

[53] S. Huang, Z. Gojcic, J. Huang, A. Wieser, and K. Schindler, "Dynamic 3d scene analysis by point cloud accumulation," in *ECCV*, 2022.

[54] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD*, vol. 96, no. 34, 1996, pp. 226–231.

[55] D. Rempe, T. Birdal, Y. Zhao, Z. Gojcic, S. Sridhar, and L. J. Guibas, "Caspr: Learning canonical spatiotemporal point cloud representations," 2020.

[56] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," 2018.

[57] J. Ost, F. Mannan, N. Thuerey, J. Knodt, and F. Heide, "Neural scene graphs for dynamic scenes," in *CVPR*, 2021.

[58] Z. Li, S. Niklaus, N. Snavely, and O. Wang, "Neural scene flow fields for space-time view synthesis of dynamic scenes," in *CVPR*, 2021.

[59] Z. Li, Q. Wang, F. Cole, R. Tucker, and N. Snavely, "Dynibar: Neural dynamic image-based rendering," in *CVPR*, 2023.

[60] S. Fridovich-Keil, G. Meanti, F. R. Warburg, B. Recht, and A. Kanazawa, "K-planes: Explicit radiance fields in space, time, and appearance," in *CVPR*, 2023.

[61] A. Cao and J. Johnson, "Hexplane: A fast representation for dynamic scenes," in *CVPR*, 2023.

[62] U. Singer, S. Sheynin, A. Polyak, O. Ashual, I. Makarov, F. Kokkinos, N. Goyal, A. Vedaldi, D. Parikh, J. Johnson *et al.*, "Text-to-4d dynamic scene generation," *arXiv:2301.11280*, 2023.

[63] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," 2021.

[64] J. Wald, N. Navab, and F. Tombari, "Learning 3d semantic scene graphs with instance embeddings," *IJCV*, 2022.

[65] Z. Song and B. Yang, "Ogc: Unsupervised 3d object segmentation from rigid dynamics of point clouds," 2022.

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

[67] V. Jampani, K.-K. Maninis, A. Engelhardt, A. Karpur, K. Truong, K. Sargent, S. Popov, A. Araujo, R. Martin-Brualla, K. Patel, D. Vlasic, V. Ferrari, A. Makadia, C. Liu, Y. Li, and H. Zhou, "Navi: Category-agnostic image collections with high-quality 3d shape and pose annotations," 2023. [Online]. Available: https://navidataset.github.io/

[68] Y. Yue, S. Mahadevan, J. Schult, F. Engelmann, B. Leibe, K. Schindler, and T. Kontogianni, "Agile3d: Attention guided interactive multi-object 3d segmentation," *arXiv:2306.00977*, 2023.

[69] G. Peyré, M. Cuturi *et al.*, "Computational optimal transport: With applications to data science," *Foundations and Trends® in Machine Learning*, vol. 11, no. 5-6, pp. 355–607, 2019.

[70] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, and K. Schindler, "Predator: Registration of 3d point clouds with low overlap," in *CVPR*, 2021.

[71] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3d point cloud matching with smoothed densities," in *CVPR*, 2019, pp. 5545–5554.

[72] Z. J. Yew and G. H. Lee, "Rpm-net: Robust point matching using learned features," in *CVPR*, 2020.

[73] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *ICRA*, 2009.

[74] M. Zhu, M. Ghaffari, and H. Peng, "Correspondence-free point cloud registration with so (3)-equivariant implicit shape representations," in *CoRL*, 2022.

[75] M. Matl, "Pyrender," https://github.com/mmatl/pyrender, 2019.

[76] I. Armeni, Z.-Y. He, J. Gwak, A. R. Zamir, M. Fischer, J. Malik, and S. Savarese, "3d scene graph: A structure for unified semantics, 3d space, and camera," in *CVPR*, 2019.

[77] D. Stutz, "A formal definition of watertight meshes," https://davidstutz.de/a-formal-definition-of-watertight-meshes/, 2018.

[78] D. Stutz and A. Geiger, "Learning 3d shape completion under weak supervision," *CoRR*, vol. abs/1805.07290, 2018. [Online]. Available: http://arxiv.org/abs/1805.07290

[79] J. Huang, H. Su, and L. Guibas, "Robust watertight manifold surface generation method for shapenet models," *arXiv:1802.01698*, 2018.

[80] M. Woo, J. Neider, T. Davis, and D. Shreiner, *OpenGL programming guide: the official guide to learning OpenGL, version 1.2*. Addison-Wesley Longman Publishing Co., Inc., 1999.

[81] Q. Xu, W. Wang, D. Ceylan, R. Mech, and U. Neumann, "Disn: Deep implicit surface network for high-quality single-view 3d reconstruction," 2019.

[82] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014.

[83] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," 2019.

[84] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, 2004.

[85] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superglue: Learning feature matching with graph neural networks," in *CVPR*, 2020.

[86] J. Xie, Y. Fang, F. Zhu, and E. Wong, "Deepshape: Deep learned shape descriptor for 3d shape matching and retrieval," in *CVPR*, 2015.

[87] R. Brégier, "Deep regression on manifolds: a 3D rotation case study," 2021.

[88] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[89] S. Sridhar, D. Rempe, J. Valentin, B. Sofien, and L. J. Guibas, "Multiview aggregation for learning category-specific shape reconstruction," 2019.

[90] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *ICML*, 2021.

[91] S. Peng, K. Genova, C. Jiang, A. Tagliasacchi, M. Pollefeys, T. Funkhouser *et al.*, "Openscene: 3d scene understanding with open vocabularies," in *CVPR*, 2023.

Appendix A

Supplemental Materials

A.1 SIM(3) Equivariance

In EFEM [51], Lei et al. show the equivariance to SIM(3) by extending vector neurons to invariant, equivariant, scale and centroid features $\Theta = [\Theta_{\text{inv}}, \Theta_{\text{c}}, \Theta_{\text{s}}, \Theta_{\text{R}}]$.

$$
\begin{aligned}
&\widehat{SDF}(s\mathbf{x}\mathbf{R}+\mathbf{t}; \Phi(s\mathbf{X}\mathbf{R}+\mathbf{t})) = \widehat{SDF}(s\mathbf{X}\mathbf{R}+\mathbf{t}; g \circ \Theta) \\
&= \text{MLP}\left(\left\langle \Theta_{\text{R}}\mathbf{R}, \frac{s\mathbf{x}\mathbf{R}+t-(s\Theta_{\text{c}}\mathbf{R}+t)}{s\Theta_{\text{s}}} \right\rangle_{\text{channel}} \oplus \Theta_{\text{inv}}\right) \\
&= \text{MLP}\left(\left\langle \Theta_{\text{R}}\mathbf{R}, \frac{(\mathbf{x}-\Theta_{\text{c}})\,\mathbf{R}}{\Theta_{\text{s}}} \right\rangle_{\text{channel}} \oplus \Theta_{\text{inv}}\right) \\
&= \text{MLP}\left(\left\langle \Theta_{\text{R}}, \frac{\mathbf{x}-\Theta_{\text{c}}}{\Theta_{\text{s}}} \right\rangle_{\text{channel}} \oplus \Theta_{\text{inv}}\right) = \widehat{SDF}(\mathbf{x}; \Phi(\mathbf{X})),
\end{aligned}
\tag{A.1}
$$

where $s, \mathbf{R}$ and $\mathbf{t}$ represent the scale, rotation and translation of the point cloud to the canonical space. $\oplus$ refers to channel-wise concatenation and Φ the vector neuron encoder. $g = (\mathbf{R}, \mathbf{t}, s)$ denotes the SIM(3) transformation and $\circ$ applies the transformation to embeddings. Since the centroid correction and scale terms are estimated, the network is not strictly equivariant to SIM(3).

A.2 Training Log

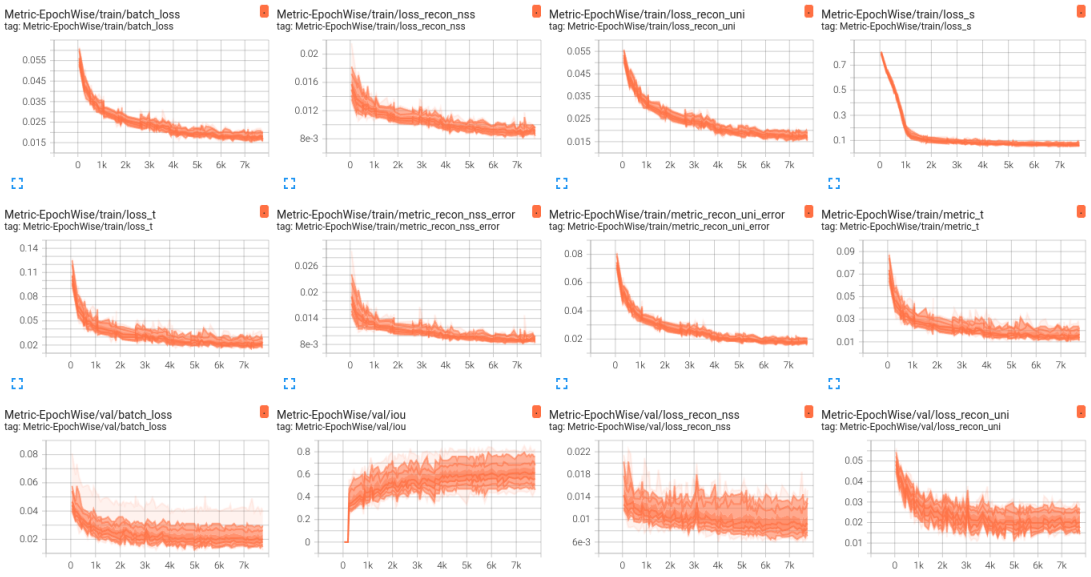


Figure A.1: **Training Log from Tensorboard [7].** From left to right: total batch loss, near-surface loss, uniform SDF loss, scale loss, centroid loss and iou. Curves during training and evaluation.

A.3 Similarity Matrices of Shape Matching

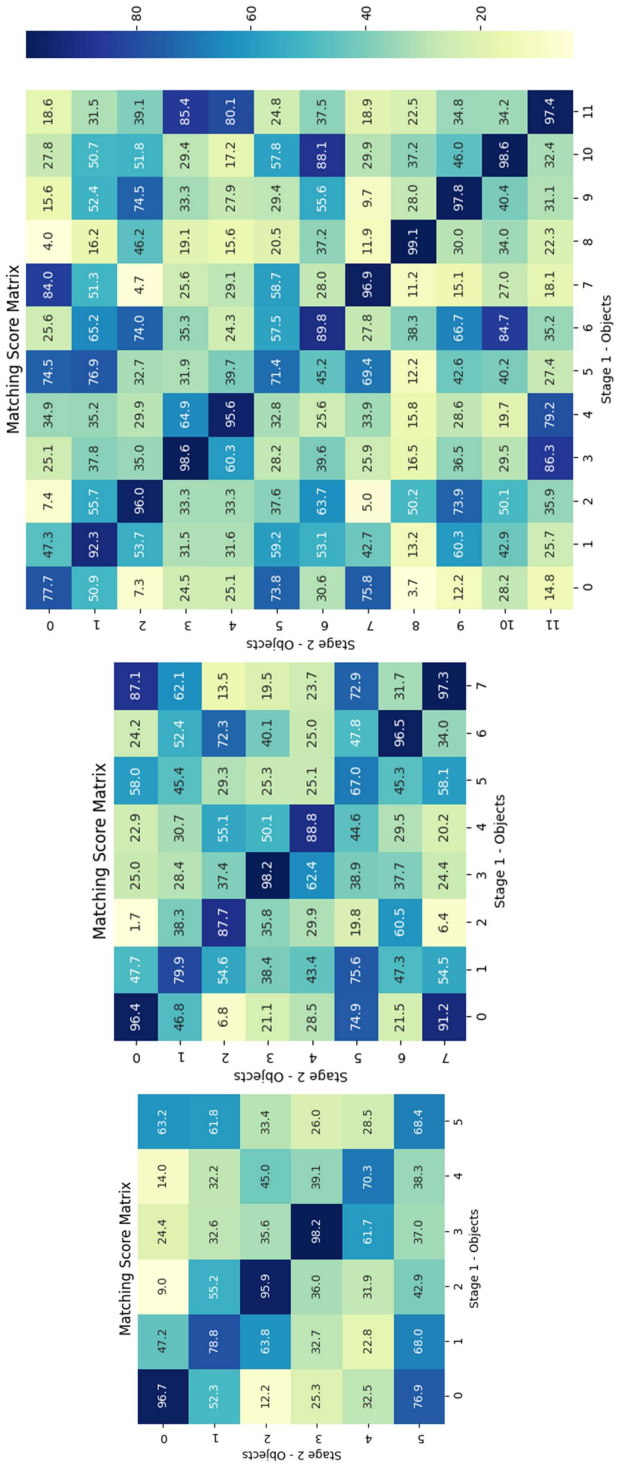


Figure A.2: **Score Matrics of Shape Matching.** From left to right: 6-object matching, 8-object matching and 12-object matching. The correct matches correspond to the diagonal of each matrix.

A.4 Additional Registration Results

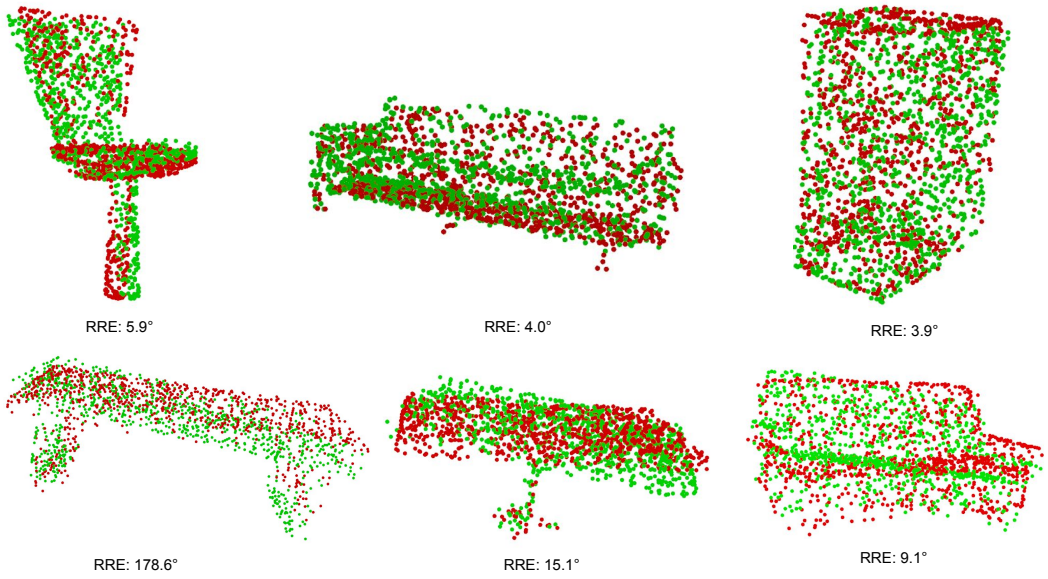
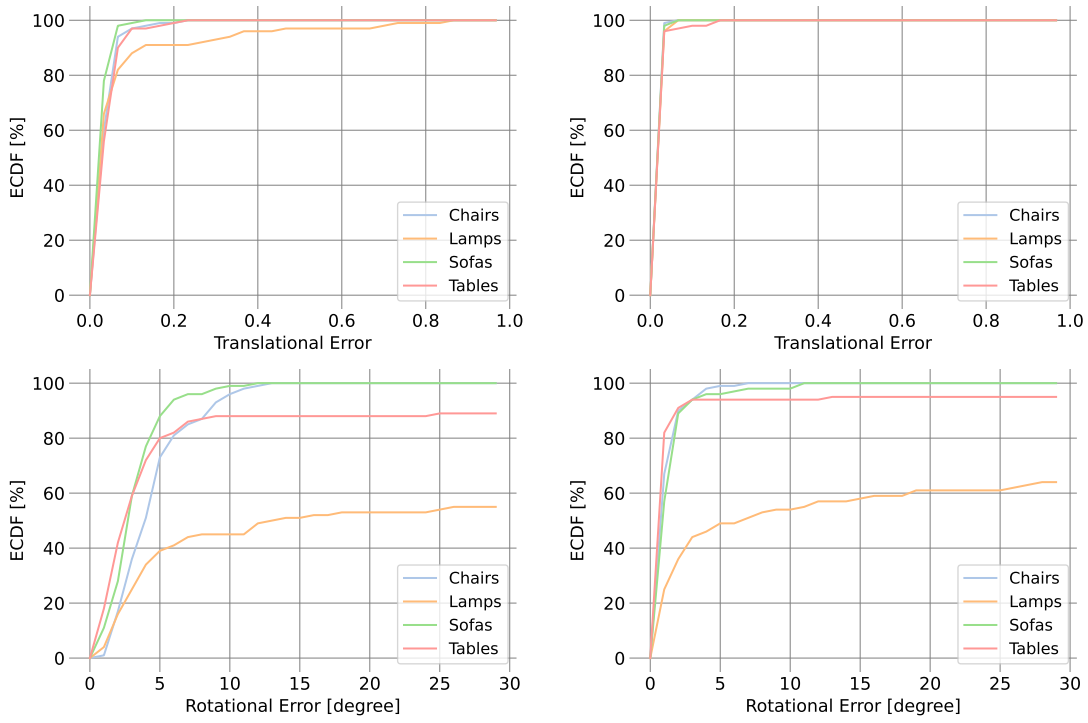


Figure A.3: **Registration Results of Our Method without Optimization.** Green is the estimation and red is the target. The relative rotation errors are shown under point clouds.

A.5 DeepSDF as Pose Estimator

At the early stage of the thesis, we explored using the DeepSDF [6] decoder as a pose estimator. Here, we assume that we know the exact latent codes of the point cloud and experiment with objects from ShapeNet [66]. A simple baseline (FPFH + Ransac) is chosen for comparison (see Figure A.4). It shows that the implicit neural field is useful for registration by aligning the zero level-set of two or more fields.



(a) FPFH+RANSAC (b) DeepSDF

Figure A.4: **ECDF Curves of (a) FPFH and (b) DeepSDF**. Top: translational error. Bottom: rotational error (RRE).