TUM School of Computation, Information and Technology
Technische Universität München

# CAD Model Retrieval and Alignment in 3D Scans

**Armen Avetisyan**

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung des akademischen Grades eines

**Doktors der Naturwissenschaften (Dr. rer. nat.)**

genehmigten Dissertation.

**Vorsitzender:**
Prof. Dr. Nassir Navab

**Prüfende der Dissertation:**
1. Prof. Dr.-Ing. Matthias Nießner
2. Prof. Dr. Evgeny Burnaev,
   Skolkovo Institute of Science and Technology

Die Dissertation wurde am 21.02.2022 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 17.08.2022 angenommen.

# Acknowledgement

This dissertation is a product of a long journey during my PhD program that taught me that there is a difference between motivation and drive. The former is volatile & inconsistent but the latter makes you accomplish long-term goals! During the process, I experienced great challenges as well as a great success which helped me grow as a person. I would like to use this opportunity to express gratitude to all the people that accompanied me on that journey: The coffee breaks, meetups at the beer garden, and the spontaneous discussions at the whiteboard, all helped me to stay on track and to find a good balance between leisure & work.

I would like to thank my supervisor Prof. Dr. Matthias Nießner who believed in me and gave me the opportunity to join the PhD program. He taught me to push my boundaries, paved the way for exciting fields & cleared obstacles for me to focus efficiently on research. His guidance and availability for research discussions are remarkable. Furthermore, I would like to express my gratitude to Prof. Dr. Angela Dai who accompanied my initial years with useful advice to make sure I ramp up smoothly. A big thank you goes to Prof. Dr. Justus Thies, who was always available for entertaining discussions about and beyond research and with his sharp mind seemed to always be one step ahead of things.

My colleagues certainly sweetened the time and deserve great appreciation. Thanks to my clever office mate Aljaz Bozic and Ji Hou for their unique humor. I want to express my gratitude to Manuel Dahnert for his friendly nature, Andreas Roessler for adding enjoyment to our coffee breaks, and Dejan Azinović for always catching my inaccuracies when I was paraphrasing what I read. Many thanks to Guy Gafni and Yawar Siddiqui for gifting us with their expert-level memes. Norman Mueller and Johanna Wald deserve my gratitude for the entertaining time. Without my co-workers Dave Zhenyu Chen, Pablo Palafox, Shivangi Aneja, Hao Yu, Christian Diller, Alexey Bokhovkin, and Andrei Burov the time would be less enjoyable. Many thanks to Susanne Weitz and Assia Franzmann for their generous support!

Our interns Lei Han, Yang Li, Yizhak Ben-Shabat, Chris Choy, Shijie Li, Hassan Abu Alhaija, Jingwei Huang, Vassilis Choutas, Xiaochen Fan, and Cheng Lin certainly brought cultural richness, expertise, and fun to our summers in Munich. I owe gratitude to my manager Tigran Gasparian at Google X who consequently made time for me and gave excellent support in every situation.

Lastly, I deeply thank my parents for always being proud of me and my sisters and for their unreserved support!

# Abstract

Creating a digital copy of the real world could greatly benefit people, but remains a challenging goal in computer vision research. An accurately indexed indoor scene would allow users to search for objects within a given snapshot and replay or alter experienced situations. While 3D reconstructions provide geometric information about the observed scene, they often suffer from noise, incompleteness, lack of semantic information, and limited usability. In our work, we aim to digitize indoor scenes and convert them into a lightweight CAD model representation that has a high level of geometric quality, a low memory footprint, and is naturally ready for use in AR/VR applications.

The general process of CAD model alignment includes several tasks: detecting object instances, retrieving 3D CAD model candidates, and aligning the 3D CAD models with the scanned objects. Establishing correspondences between 3D scanned objects and 3D CAD models is challenging due to the significant differences in their underlying geometry. Artifacts like over-smoothed corners and missing geometry in the 3D scanned objects can lead to unreliable correspondences when using conventional 3D point descriptors. This dissertation examines various innovative data-driven techniques for the 3D CAD alignment task, ranging from the study of predicting robust sparse correspondences, estimating dense correspondences and finally investigating the connection between objects and architectural layout components in a scene. To address the discrepancy between the two domains, we build a large-scale dataset consisting of 97607 manually annotated keypoint pairs between 14225 CAD-to-scan alignments to serve as a foundation for using data-driven methods.

In Scan2CAD, we develop a learning-based method that utilizes the dataset we built. It predicts sparse keypoint correspondences between scan and CAD. The large corpus of our dataset enabled our method to reliably learn to predict heatmaps on the CAD model given a query point on the scan, while taking into account geometric and semantic similarity. These established correspondences are then used in a 9-DoF pose optimization to determine the final pose.

To improve runtime and accuracy, we introduce a new approach that focuses on dense correspondences between a scanned object and a candidate CAD model rather than sparse correspondences. A lightweight retrieval model finds a suitable CAD model and a 3D CNN predicts dense symmetry-aware object correspondences that are supervised by a specialized alignment loss. This formulation allowed a fast and efficient way to estimate the optimal pose.

Finally, we explored a more holistic approach that takes the relationships between objects in the scene and layout elements into account. This approach is effective because the placement of furniture and the overall scene layout are closely connected. By using a graph neural network to exploit the inter-dependency of entities, we can recognize the relationships between them, resulting in consistent alignment of objects within the scene. Our results demonstrate a significant improvement in CAD alignment accuracy when the global scene layout is taken into consideration.

# Zusammenfassung

Die Erstellung einer digitalen Kopie der realen Welt könnte die Menschen in vielerlei Hinsicht von Nutzen sein, sie stellt jedoch ein anspruchsvolles Ziel in der Computer Vision-Forschung dar. Eine präzise indexierte Innenraumszene würde es Benutzern ermöglichen, innerhalb eines gegebenen Schnappschusses nach Objekten zu suchen und erlebte Situationen erneut abzuspielen oder zu verändern. Obwohl 3D-Rekonstruktionen geometrische Informationen über die beobachtete Szene liefern, leiden sie oft unter Rauschen, Unvollständigkeit, fehlender semantischer Information und einer eingeschränkten Verwendbarkeit. In unserer Arbeit zielen wir darauf ab, Innenräume zu digitalisieren und sie in eine kompakte CAD-Modelldarstellung umzuwandeln, die eine hohe geometrische Qualität aufweist, einen geringen Speicherbedarf hat und für die Verwendung in AR/VR-Anwendungen geeignet ist.

Der Prozess der Ausrichtung von CAD-Modellen umfasst mehrere Aufgaben: Erkennen von Objektinstanzen, Abrufen von 3D-CAD-Modellkandidaten und Posebestimmung der 3D-CAD-Modelle mit den gescannten Objekten. Es ist schwierig, Korrespondenzen zwischen 3D-gescannten Objekten und 3D-CAD-Modellen herzustellen, aufgrund der erheblichen Unterschiede in ihrer Geometrie. Artefakte wie glättete Ecken und fehlende Geometrie in den 3D-gescannten Objekten können zu unzuverlässigen Korrespondenzen führen, wenn konventionelle 3D-Punktdeskriptoren verwendet werden.

Diese Dissertation untersucht verschiedene innovative datengesteuerte Techniken für die 3D-CAD-Lagebestimmung, die von der Vorhersage robuster Einzelkorrespondenzen, der Schätzung dichter Korrespondenzen und schließlich der Untersuchung der Verbindung zwischen Objekten und Layoutkomponenten in einer Szene reichen. Um die Diskrepanz zwischen den beiden Bereichen anzugehen, erstellen wir einen großen Datensatz bestehend aus 97607 manuell annotierten Keypoint-paaren zwischen 14225 CAD-Scan-Ausrichtungen, um als Grundlage für die Verwendung datengesteuerter Methoden zu dienen.

In Scan2CAD entwickeln wir eine lernbasierte Methode, die den von uns erstellten Datensatz nutzt. Sie prognostiziert Einzelkorrespondenzen zwischen Scan und CAD. Die große Datenmenge ermöglichte es unserer Methode, zuverlässig zu lernen, Heatmaps auf dem CAD-Modell anhand eines Abfragepunkts im Scan vorherzusagen, wobei die geometrische und semantische Ähnlichkeit berücksichtigt wird. Diese erstellten Korrespondenzen werden dann in einer 9-DoF-Poseoptimierung verwendet, um die endgültige Pose zu bestimmen.

Um Laufzeit und Genauigkeit zu verbessern, stellen wir einen neuen Ansatz vor, der sich auf dichte Korrespondenzen zwischen einem gescannten Objekt und einem Kandidaten-CAD-Modell statt auf Einzelkorrespondenzen konzentriert. Ein einfaches Abrufmodell findet ein geeignetes CAD-Modell und ein 3D-CNN schätzt dichte symmetiebewusste Objektkorrespondenzen, die von einer spezialisierten Ausrichtungsverlustfunktion überwacht werden. Diese Formulierung ermöglichte eine schnelle und effiziente Methode zur Schätzung der optimalen Pose.

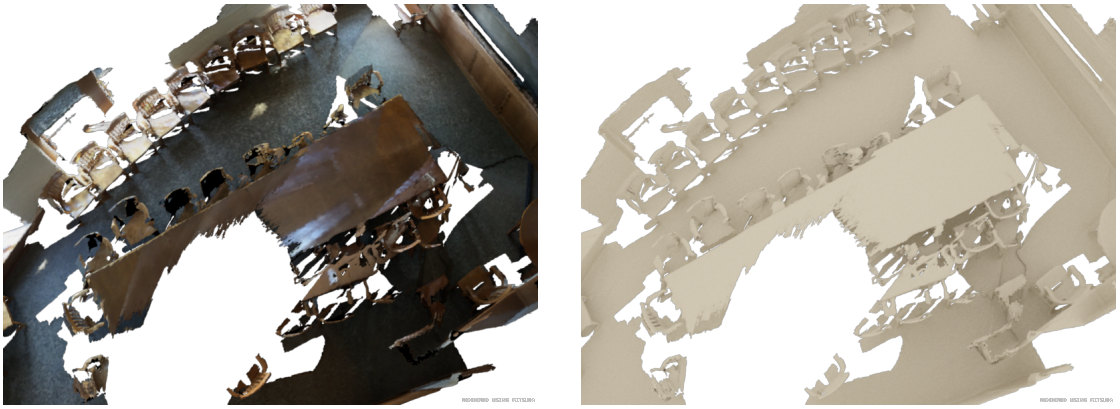
Schließlich haben wir einen holistischen Ansatz untersucht, der die Beziehungen zwischen Objekten in der Szene und Grundriss-Elementen berücksichtigt. Dieser Ansatz ist wirksam, da die Platzierung von Möbeln und der räumliche Grundriss der Szene eng miteinander verbunden sind. Durch die Verwendung eines Graphen Neural Network zur Identifizierung von Beziehungen zwischen allen Entitäten, können wir eine konsistente Ausrichtung der Objekte in der Szene erreichen. Unsere Ergebnisse zeigen eine deutliche Verbesserung der Genauigkeit der CAD-Ausrichtung, wenn der globale Grundriss der Szene berücksichtigt wird.

# Contents

# 1 Introduction

Over the past decade, researchers have increasingly turned to 3D data to explore new scene understanding techniques. For a long time, the primary source of data was photographs and images. The availability of consumer-grade depth sensors, such as the Microsoft Kinect or StructureIO sensor, has made it possible to create detailed and comprehensive reconstructions of indoor environments, resulting in a high-density and rich representation of the surroundings. When a 3D scene is reconstructed accurately, it opens up the possibility of placing and simulating objects in a virtual world and even blending them seamlessly into the real world. For virtual reality products, it is crucial to create an accurate map of the surrounding environment to prevent users from accidentally colliding with or running into obstacles. The fundamental issues with using 3D reconstruction algorithms for straightforward scene capture are that the results are often incomplete and contain noise (as shown in Figure 1.1). Additionally, objects in the scene are not segmented and instead are grouped together, as the entire scene is represented by triangles without any additional semantic information. As a result, the quality of such reconstructions is not on par with that of artistically modeled environments, making them only partially suitable for use in AR/VR applications. Scanners are often unable to thoroughly scan an entire room as they are not able to reach every point in space and capture objects from every viewpoint. Depth sensors usually have a limited resolution, and their accuracy decreases as the surface distance to the camera increases. As a result, sharp corners and edges may be smoothed out due to the discretization level and voxel resolution used. There is also a practical limitation on the voxel resolution that is determined by the memory capacity of the device running the reconstruction algorithm.

Given the many limitations of 3D reconstruction, one solution is to use post-processing techniques to improve the overall quality and enable a wider range of applications. One approach is to use 3D Computer Aided Design (CAD) models which are created with a high degree of precision and quality. There are millions of publicly available CAD models on various platforms that can be used to replace noisy and incomplete objects in scans with their high-quality CAD counterparts. Modeling software such as *Solidworks*, *Catia*, *AutoCAD* for the engineering industry or *Maya*, *Blender*, *3ds Max* allows engineers and hobby designers to create a vast number of high-quality models of everyday objects, such as vehicles, furniture, appliances, and even vegetables. Many objects found in typical households likely have a corresponding 3D model. CAD models often come with high-resolution textures and accurate *BRDF* materials for physically-based rendering. With accurate computer-assisted geometry and support for photo-realistic rendering, CAD models are suitable candidates for the replacement of objects in 3D reconstructions. Ideally, we would like to learn from and use existing collections of high-quality assets created by professional artists in order to improve reconstruction quality as illustrated in Figure 1.2.

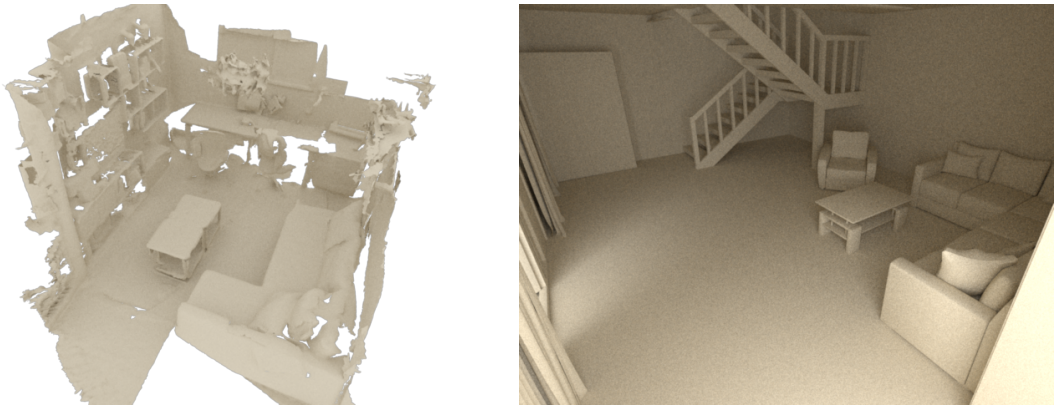


**(a)** With vertex colors. **(b)** Geometry only.

**Figure 1.1:** Noisy RGB-D indoor reconstruction with and without color illustrating incompleteness and artifacts of the final capture.

With advancements in semantic segmentation, particularly semantic instance segmentation, it is possible to understand the basic taxonomy of the objects in a scene. For example, one can determine the class of an object and its geometrical shape through segmentation. While this approach is fundamental for general scene understanding, it cannot by itself improve the quality of 3D reconstructions. This dissertation focuses on the digitization of 3D reconstructions, which is the process of transforming a scene-level RGB-D 3D reconstruction into a high-quality, lightweight CAD representation that has a higher level of geometric quality, lower memory footprint, and comes naturally with a set of attributes. Such a compact CAD representation of scenes allows for the virtual manipulation of object arrangements and is suitable for use in AR/VR applications. The challenges of this transformation are three-fold: 1) detecting objects in a 3D scene, 2) finding a suitable CAD candidate counterpart for replacement, and 3) estimating the pose of the CAD model. Each step plays a crucial role in the final CAD alignment performance. In this dissertation, we investigate three different aspects of the field of 3D CAD model alignment. Specifically, we:

- Introduce Scan2CAD, a learning-based method that uses pose optimization to match objects by predicting sparse keypoint correspondences between a scan and a CAD model.
- Present an end-to-end trained method that predicts dense correspondences between a scanned object and a CAD model, which can be used to efficiently determine the optimal pose.
- Introduce SceneCAD, which explores the benefits of jointly estimating the arrangement of CAD models and the layout elements of a scene.



**(a)** RGB-D reconstructed scene. **(b)** CAD-based scene.

**Figure 1.2:** Geometrical comparison between an RGB-D reconstruction and a CAD-based scene. The CAD-based scene is complete, maintains sharp edges/corners, and shows qualitatively a higher fidelity.

## 1.1 Cumulative Thesis Outline

The organization of this thesis is as follows: The main contributions and the publications upon which the thesis is based are presented in the introduction.

In chapter 2, background context is provided to facilitate understanding of the thesis, including information on common 3D geometry representations, RGB-D reconstruction techniques, on the creation and use of CAD models, CAD model alignment methods, and data-driven 3D scene understanding methods.

Our papers [1], [2], [3] are discussed in chapter 3, chapter 4, and chapter 5 respectively. In chapter 6, we point out the limitations of our methods. Finally, the thesis is concluded and potential future research directions are presented in chapter 7.

## 1.2 Contributions

In this dissertation we mainly work with Deep Learning (DL)-based techniques and propose various methods to transform a noisy RGB-D scan into a high-quality CAD scene representation by detecting objects in the scan, retrieving a matching CAD candidate, and aligning it onto the scan. The foundation of our work is the large-scale database that we build in order to learn shape retrieval & pose alignment of CAD models. We leverage the 3D model repository called ShapeNet [4] and a 3D scan dataset called ScanNet [5] in order to connect both domains for the CAD model alignment task.

- We build a large-scale cad-to-scan dataset comprising of 97607 manually annotated key-point correspondences between 14225 3D CAD models and scanned real-world object pairs on over 1500 indoor scenes. This dataset forms the basis for later data-driven learning methods addressing the task of aligning CAD models.

- Introduce a new 3D Convolutional Neural Network (CNN) that first predicts 3D correspondence heatmaps between a scanned object and a CAD model, which are then utilized in a second step by a variational Levenberg–Marquardt (LM) optimizer to estimate a final 9-DoF pose.
- Develop a neural network to align CAD models onto 3D scans, which is trained in an end-to-end manner. By jointly training a lightweight retrieval network and the dense correspondence network with an efficient alignment loss we outperform existing baselines by a significant margin.
- Propose a new approach that simultaneously estimates the arrangement of CAD models and the layout of a scene to create a globally consistent CAD scene representation. By taking into account the relationships between objects and the layout, this method enhances the alignment performance of CAD models and also generates a simple, lightweight scene layout using quadrilaterals.

## 1.3 List of Publications

**Authored**

- **Avetisyan, A.**, Dahnert, M., Dai, A., Savva, M., Chang, A. X., & Nießner, M. (2019). Scan2cad: Learning cad model alignment in rgb-d scans. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 2614-2623).
- **Avetisyan, A.**, Dai, A., & Nießner, M. (2019). End-to-end cad model retrieval and 9dof alignment in 3d scans. *In Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 2551-2560).
- **Avetisyan, A.**, Khanova, T., Choy, C., Dash, D., Dai, A., & Nießner, M. (2020). Scenecad: Predicting object alignments and layouts in rgb-d scans. *In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16* (pp. 596-612). Springer International Publishing.

**Co-Authored**

- Wald, J., **Avetisyan, A.**, Navab, N., Tombari, F., & Nießner, M. (2019). RIO: 3D object instance re-localization in changing indoor environments. *In Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 7658-7667).

# 2 Background

## 2.1 3D Geomtery Representation

On contrary to the 2D domain where high-resolution images can be stored on a regularly spaced 2D grid, where each cell is also called a pixel, in the 3D domain there are several options to encode the underlying 3D geometry.

Existing geometry representations can be broadly categorized into three categories: **point cloud**, **mesh**, and **voxel grid**:

**Point cloud** A point cloud is a point-based data representation that encodes the 3D surface of an object or an environment by a set of points. Each point is represented by a 3D coordinate $(x, y, z)$. Often points are decorated by additional features such as *RGB* values that represent the color information of the point or normal values $(n_x, n_y, n_z)$ which indicates the surface normal of the point. The points in a point cloud are typically unstructured which means that the point density in any region can be arbitrary due to variable spacing. However, they do not capture the topological structure of the 3D shape and are less suitable for tasks that require the manipulation of surface features or the representation of smooth, continuous surfaces.

Point clouds can be generated using different methods such as laser scanners (LiDAR), structure light scanners, stereo matching, point triangulation from multiscopic views, and surface sampling of 3D models.

**Mesh** A mesh is defined by a set of vertices and a set of faces that describes the surface of a 3D object or environment. A vertex is a discrete 3D point coordinate $(x, y, z)$ and a face contains indices of points that form a planar polygon (typically triangles). The discrete polygons interconnect the vertices and thus approximate the shape of the underlying object. Popular data format allow augmentation by vertex or faces properties such as color information, normal information, or UV coordinates. Meshes capture the topological structure of the 3D geometry and are suitable for tasks that require surface information. Meshes can be generated in several ways, for example, with 3D modeling software that allows direct manipulation of polygons or primitives by a set of operations, marching cubes, or by surface fitting such as the Poisson surface reconstruction method.

**Voxel grid** A voxel grid is a 3D data structure that divides the 3D space into equally sized voxels, where each voxel carries the value of one or multiple attributes. Typically for 3D shapes, voxel grids are discrete scalar fields that contain binary values or (signed) distance values.

A binary voxel grid is also called an occupancy grid where each voxel can take a 0 or 1 value indicating whether the voxel is occupied or free. For example, the value can represent whether

the volume of the voxel represents matter/geometry or free space. Since only binary values are allowed, boolean data types with a small memory footprint can be used to represent the values. Due to its simplicity, this is a popular choice in the computer vision community. However, in practice, a high resolution is required to sample the space in order to avoid aliasing effects.

A volumetric signed distance field is a representation of a 3D shape using a voxel grid, where each cell stores the distance between its center and the nearest surface point. The sign of the distance indicates whether the voxel is inside or outside of the shape. An unsigned distance field is used in situations where the distinction between inside or outside is not possible or unavailable. The advantage of signed distance fields over occupancy grids is that they provide a smooth and continuous representation of the surface.

## 2.2 3D Reconstruction of Indoor Environments

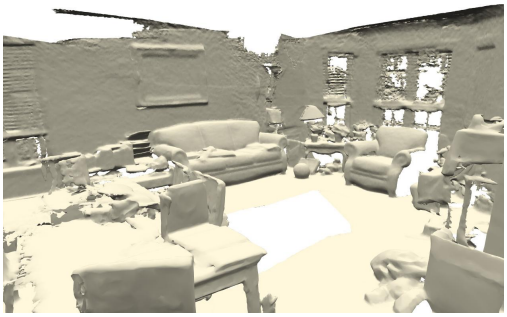
3D reconstruction is the process of creating a 3D model clone of a physical object or physical environment through a set of observations. In the past, several techniques for 3D reconstruction have been developed that vary in terms of the type of input, the algorithms used, and the output. These methods can be grouped into two broad categories: active and passive, which refer to the type of sensing method used for 3D reconstruction:

Active methods actively sense the object or environment with external light in order to build a depth map. For example, the depth of an object relative to a camera can be measured by emitting light in a particular pattern and measuring the reflectance from its surface. Examples of sensors in active reconstruction methods include structured light, time-of-light lasers, LiDAR, Radar, or ultrasounds [6]. The upcoming chapters of this thesis will concentrate on active reconstruction methods, and more specifically, on RGB-D reconstruction methods which acquire a color image and a depth image simultaneously.

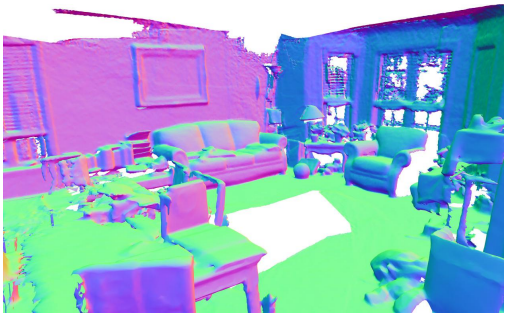
Passive methods do not involve any active interference with the object or environment being scanned, and thus do not rely on external stimuli. They measure the intensity of the light reflected or emitted by an object passively. Typically, passive methods employ a camera sensitive to visible light to capture a set of images and then deduce the geometry of the object or environment. Examples of passive reconstruction methods include photogrammetry, multi-view stereo, and structure from motion.

### 2.2.1 Active Methods

**Structured Light** A structured light sensor is a device with a light source that projects laser beams onto the scene. Typically, a near-infrared laser projects invisible dots or lines pattern in a grid structure. A separate receiver reads the reflected incoming light and estimates a depth map by the deformation characteristics of the projected pattern [8, 9]. The Microsoft Kinect has been a popular choice as a low-cost, hand-held sensing device in the computer vision community that works on the principle of structured light [10]. Some materials are categorically hard or impossible to scan. For example, shiny surfaces may deflect or dark objects may absorb the outgoing light making it for the receiver difficult to read the structured light. Transparent or semi-transparent objects cause difficulties as well because of complex scattering effects leading

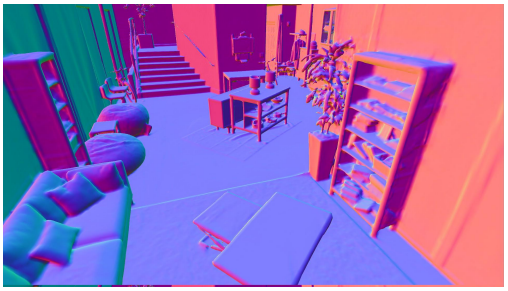


**(a)** Scanned geometry (noisy).



**(b)** Normals.

**(c)** Scanned geometry (high-fidelity).



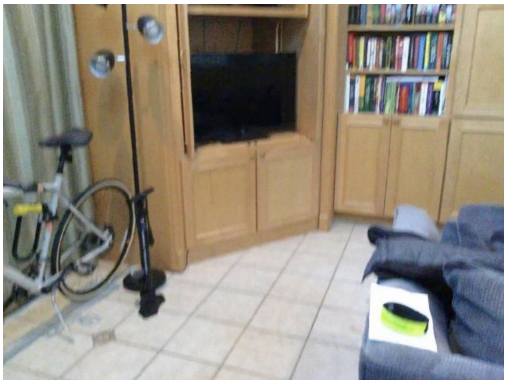
**(d)** Normals.

**Figure 2.1:** Comparison of a *low* quality (top) vs a *high* quality (bottom) scene reconstruction from the Replica dataset [7].

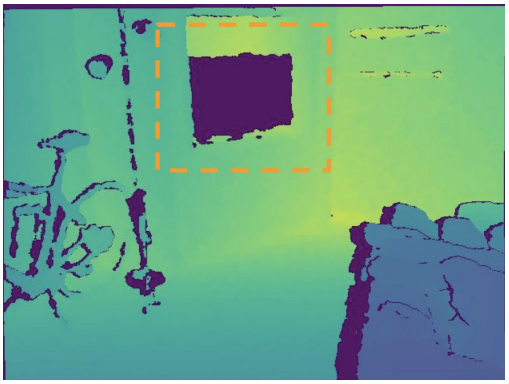
to unwanted deflections. Such materials can degrade the quality by creating noise or leaving holes in the depth map (see Figure 2.2).

**LiDAR** Light detection and ranging (LiDAR) is a technology that gained popularity in wide applications, including navigation, self-driving cars and robotic. A lidar sensor consists of a laser transmitter, a receiver and a detector. The laser emits a single laser pulse towards a directed mirror, which guides the beam onto the scene. The laser beam eventually hits the surface of an object and the resulting reflected laser beam is captured by a receiver. The global position is estimated by measuring the time and position of the emitted and received signal respectively. Note that moving LiDARs have a number of scanning patterns (e.g. rotating or pitching movement) which determine the sampling pattern of the points [11, 12]. Common challenges of LiDAR scans are inaccuracies due to a moving platform and restricted real-time performance from high scanning frequency which can lead to sparser point clouds [12].

**Time-of-Flight** A flash Time-of-Flight (TOF) emits short laser pulses onto the entire scene through an optical diffusor. An array of photodiodes measures the incoming time of flight of the reflected light for each pixel. Since a single laser pulse is bursted onto the scene, an active scanning movement is not needed [11]. A common challenge is the energy fall off of a single emitted light that is divided among multiple detectors after the reflection from the surface. A desirable property of a ToF camera is that the depth map is acquired in a 2D grid. Many



**(a)** RGB frame.



**(b)** Depth frame.

**Figure 2.2:** Reflective and very dark surfaces raise difficulties for accurate depth estimates. The emitted light is being either deflected away or absorbed leaving no signal in the region with the critical surface material as in this example with the TV from the ScanNet dataset [5] © 2017 IEEE

consumer-grade RGB-D cameras (e.g. Microsoft Kinect v2) use the ToF technology because of their low-cost and real-time scanning speed.

### 2.2.2 Fundamentals on Static RGB-D Scene Reconstruction

Many RGB-D scene reconstruction methods follow coarsely the following processing steps: acquisition of aligned RGB and depth frames, camera pose estimation, depth map fusion, and model update (see an overview Figure 2.3).

Differentiation between methods is in terms of scene representation, model data structure, frame registration & camera tracking, loop closure detection, and global optimization.

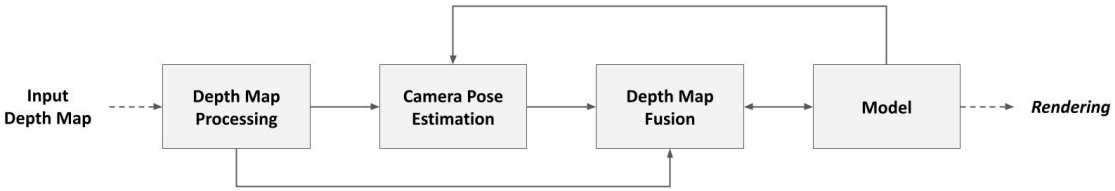



**Figure 2.3:** Typical system pipeline for many RGB-D reconstruction methods [10].

**Basic RGB-D Volumetric Fusion** The standard technique to generate RGB-D reconstruction is based on the volumetric fusion approach from Curless and Levoy [13]. The concept is to update an existing Signed Distance Function (SDF) represented by a discrete voxel grid by a new range image (depth map) at a time in a cumulative weighted manner (see Equation 2.1). The distances in the depth maps represent the nearest range surface to the camera sensor and the zero crossing is a point on the range image surface. In practice, the weights are necessary to handle noise in the depth maps. Finally, the surface mesh is extracted through face polygonization by a Marching

Cubes algorithm [14] or ray casting. The update rule of the SDF is

$$SDF_{i+1}(x) = \frac{W_i(x) \cdot SDF_i(x) + w_{i+1}(x) \cdot sdf_{i+1}(x)}{W_i(x) + w_{i+1}(x)} \tag{2.1}$$

where $x$ is the spatial position, $SDF_i$ is the weighted estimate of the signed distance value for iteration $i$, $sdf_{i+1}(x)$ is the new distance measurement, $W_i$ is the cumulative weight for iteration $i$ and $w_i$ is the new weight increment. In the simplest case, $w_i(x) = 1$ corresponds to a running average of the measurement stream. In practice, the weight function tends to be more sophisticated and tries to give *good* measurements a high weight and *bad* measurements a low weight respectively by employing various heuristics.

Curless and Levoy's formulation separates 3 regions: air or empty space, unseen or unobserved space, and near-surface space. The truncation region and isosurface are located in the near-surface space, where the signed distance function changes its sign as it approaches zero. For efficiency, the empty and unobserved spaces are often excluded from the map-building process, as they are assumed to contain little useful information.

**Model-based Dense TSDF-fusion** Newcombe et al. [15] fuses incoming depth images into a single global model by rigidly registering incoming depth frame to the surface from the latest version of the model via a coarse-to-fine ICP algorithm. The ICP alignment between the measured and the predicted surface represents the pose estimation. New depth maps are fused into a dense volumetric TSDF-grid which over time averages out noise and outliers in the depth maps. Unoptimized versions of this system exhibit almost quadratic memory growth with the scanning range.

**Fast Sparse TSDF-fusion** Nießner et al. [16] uses a model representation with an efficient data structure allowing fast data manipulations operations without excessive memory overhead. Only voxels that represent occupied geometry are stored, and pointers to the voxel are stored in a spatial hash. This system allows large-scale volumetric scene reconstruction in real-time. Hornung et al. [17] propose a compact octree-based data format that also explicitly models unknown and free space respectively. With the efficient data structure, the memory consumption is kept low, but map updates and rendering through ray-casting can still be performed at high frame rates. Chen et al. [18] use a semi-sparse hierarchical data structure with different branching factors at each level and a full grid at the nodes. This data structure maps successfully on the GPU and allows real-time reconstruction with a moving hierarchy scheme to capture larger scenes.

**Point-based Reconstruction** Keller at al. [19] uses a point-based surfel model representation that performs high-resolution reconstruction in real-time without an intermediate scene representation. This technique allows leveraging the commodity standard graphics pipeline for further speed up. Whelan et al. [20] further improved this method by adding a photometric model-to-frame tracking scheme and a non-rigid deformation graph on the map accompanied by a periodic loop closure.

**Global Optimization** Frame-to-frame tracking and to a lesser extent frame-to-model tracking suffer from pose drift over time due to accumulated local errors. Maier et al. [21] segments the pose graph & space in submaps and performs bundle adjustment locally within a submap. In the second step, submaps are globally aligned with each other. This technique significantly reduces computational burden compared to a full bundle adjustment operation while yielding globally consistent poses in the presence of noise. Dai et al. [22] proposed a technique where a global pose optimization is performed on every incoming frame, and the model surface is reintegrated by the updated pose. Coarse global alignment is established by sparse correspondences among all history frames, then the alignment is refined through a dense term. Consecutive frames are organized in chunks which are first locally optimized, and in the second step chunks are globally optimized.

**Lighting & Material Estimation** Guo et al. [23] estimate albedo, reflectance, and illumination parameters in a non-rigid RGB-D reconstruction setup. The illumination is modeled through a low-frequency spherical harmonics lighting model and the surface material is assumed to be Lambertian [24]. The explicit modeling of lighting & material parameters helped to improve tracking and reconstruction performance in scenarios with challenging illumination change.

### 2.2.3 Challenges of 3D Reconstruction

To prevent inaccuracies in simulations or virtual environments, it is important to create high-quality, high-fidelity reconstructions. In cases where interactions between agents and objects (such as sitting on a chair) or objects and objects (such as a ball bouncing off a wall) are simulated, small deviations from the true geometry can lead to incorrect intersections between the interacting volumes or implausible physics simulation results. The more precise the reconstructed scene is, the less likely downstream tasks will make errors. For instance, in light transport simulations, the surface normals of the scene play a significant role. Because the angle of deflection is calculated based on the normal at the incident point, relighting a noisy scene can greatly differ from the actual lighting. You can compare quality degrees in Figure 2.1.

Scanning technologies can only capture data for surfaces that are visible. In many situations, it is not feasible or possible to scan every part of an environment. The amount of scanning effort required to achieve high coverage can be substantial and influenced by factors such as the complexity of the shape (e.g. internal cavities) or access to the surface (e.g. under a bed). As a result, the final reconstruction is often partial and incomplete. The issue of unobserved space due to lack of access is a major limitation to achieving complete scene reconstructions.

**Occlusions** In typical indoor environments, objects often occlude other objects or parts of the layout. For example, a wardrobe covers a substantial part of the wall. In a non-intrusive scanning session, the wall behind the wardrobe will remain uncaptured.

**Complexity** Large environments and scenes with fine details and small features pose a challenge because of stark variations in scale. Due to practical resolution limitations, it may become difficult to maintain high accuracies over a large range of scales.

**Surface Material** Very reflective and/or absorptive surfaces and materials pose challenges for the popular structured light depth sensors because the emitted infrared rays are not properly sensed when hitting such surfaces. This leads to artifacts, noise, and errors in the depth map which negatively influences the final output (see Figure 2.2).

**Scene Lighting** Even if the scanner manages to attain a high surface coverage and thereby generate a complete scene resulting in a visually appealing scan, the lighting (e.g. shadows and specular highlights) is statically baked into the texture map of the scan as seen in Figure 2.4. In most 3D reconstructions stacks a uniform global illumination is assumed and typically lighting parameters (light position, light direction, light intensity, etc.) are not estimated. Particularly environments with complex lighting with multiple non-uniform light sources pose a challenge.

**Figure 2.4:** 3D scan aimed for high surface coverage. View-dependent lighting effects are statically baked into the texture map of the scene. It becomes difficult to realistically relight this scene and remove the shadows.

### 2.2.4 Applications

For example, autonomous agent navigation relies heavily on a reasonably well-reconstructed scene. Large holes or falsely bumpy surfaces lead to non-optimal trajectory planning. Depending on the quality of the reconstruction, certain targets during trajectory planning may be evaluated as inaccessible which can lead to catastrophic consequences. Other robots that grasp objects require a complete geometry of the underlying to estimate the optimal touching/contact points. Noise and clutter can negatively impact grasping performance and certain objects may be falsely evaluated as unpickable. Overall, for robotic applications, the underlying geometry often serves

as the backbone for various downstream tasks. Hence, a high scene reconstruction quality is crucial for the final task performance.

The key concept behind Augmented Reality (AR) applications is the ability to interact with the real world through virtual objects. The quality of the scene reconstruction is crucial to the user experience. For instance, if a virtual mug is placed on a real table, the user expects it to be stable and level. However, if there is noise or holes in the reconstruction, the mug may appear tilted or fall through the table. As augmented objects are superimposed on the video capture, any discrepancies or artifacts in the reconstruction will be particularly noticeable to the user (as shown in Figure 2.5).


**Figure 2.5:** Placement of a virtual object (pillow) on the sofa with an IPad [25]. Wrong geometry estimate will cause unrealistic contact points with the sofa and bears the risks of a deficient user experience.

## 2.3 Computer-Aided Design Models

CAD models are commonly created using 3D computer modeling software, such as 3ds Max in the entertainment industry, or SolidWorks in the engineering industry. These programs generally offer tools for simple 3D asset creation, including techniques like sweeping, revolving, extrusion, cutting, a variety of boolean operations, topological manipulations, and surface deformations. The software typically includes a graphical user interface with a main window displaying the 3D world and a toolbar with a set of features. While building a 3D model from scratch is a core function of these programs, many also include a *composition* or *assembly* mode, allowing users to place instances of objects to quickly compose a scene. This hierarchical, part-based approach can be used to create entire cities using a pre-existing model repository.

When designing a product, a modeler uses a computer display to control and alter a rendered representation of the product, which is generated from a mathematical model that must have sufficient capabilities to represent and manipulate free-form surfaces. The user does not directly

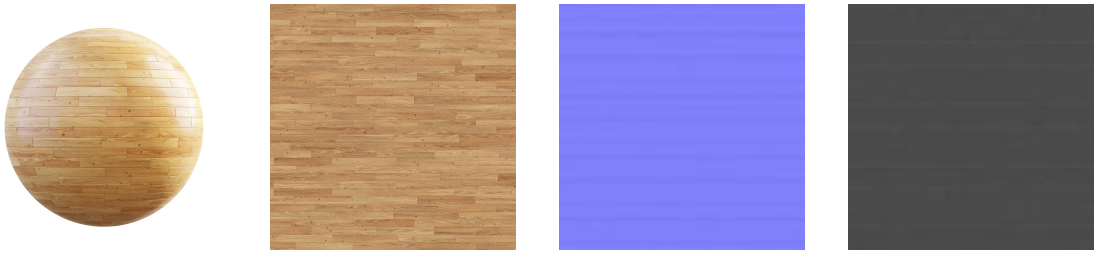
interact with the mathematical equations of the parametric surface models but instead works with geometric operations and constraints that affect the mathematical model and determine the shape [26].

In computer graphics, objects are primarily represented by polygons, usually in the form of triangles. The process of dividing a parametric surface representation into polygons that approximate the shape is called tessellation. Though this representation has its drawbacks, such as the need for a large number of triangles to approximate curved surfaces, it is the most widely used method in the industry. The combination with hardware accelerators like GPU enables the fast rendering of large numbers of triangles in real-time, making it the preferred format for further use in applications.

CAD models can have a wide range of features. Advanced CAD models may include material annotations, physical and mechanical properties, geometric constraints, annotations at the part level, functional annotations, and more. Many of these attributes are assigned during the creation process and are determined by the intended use. For example, computer games require CAD models with detailed attributes in order to create realistic renderings and physics simulations. A CAD model of a car tire, for example, would not only require the material properties of rubber for rendering, but also a friction coefficient that can be used by the simulation engine. Robotic frameworks like Robot Operating System (ROS) [27] commonly accept additional metadata, such as Universal Robot Description Format (URDF) files, for CAD models. These files contain information on physical properties like mass, inertia tensor, friction coefficient, stiffness coefficient, dampening coefficient, joint dynamics, and more [28]. With high-resolution geometry, high-quality material assets, and metadata about physical properties, CAD models are suitable for use in large-scale, realistic robot simulations.

In addition to creating and modifying geometry, 3D modeling software enables the assignment of realistic materials to different parts of an object. These materials are then used by simulation and rendering software to simulate realistic mechanics and generate photo-realistic renderings, respectively. The visual appearance of an object can be divided into two categories: changes in the direction of light and variations in material type. The Bidirectional Reflectance Distribution Function (BRDF) material model is able to model a wide range of appearances[29]. Given a specific light and camera configuration, as well as the parameters of a specific material such as wood, the BRDF model can realistically predict the amount of light that is reflected from a surface point. By using texture maps, variations in the surface of the same material, such as a wooden plank, can be modeled with high precision through the use of normal, roughness, and diffuse maps (see Figure 2.6). Assigning materials to surfaces or entire objects in the user interface is simple and efficient. This separation of shape and material provides more flexibility and freedom in the use of CAD models as shown in Figure 2.7.

CAD models are well-suited for use in synthetic simulations, as previously mentioned. To enhance the generalizability of data-driven models, it is possible to create arbitrary scene setups with automated, plausible positioning of CAD models. In recent years, indoor scene synthesis has been proven to be a potential candidate to provide plausible randomized scene compositions for data-hungry algorithms for scene understanding tasks or autonomous navigation [32, 33]. There



**(a)** Sample preview. **(b)** Diffuse map. **(c)** Normal map. **(d)** Roughness map.

**Figure 2.6:** Texture map samples of a commercially available wooden material [30]. This allow the evaluation of the BRDF for every point on the surface resulting in a visually appealing appearance.

**(a)** Slightly diffuse brown leather. **(b)** Slightly shiny black leather. **(c)** Underlying geometry of the 3D model.

**Figure 2.7:** A sample high-quality CAD model of an armchair [31] that contains realistic BRDF materials and high geometric fidelity. CAD models allow easy manipulation of appearance and geometry.

are various methods for implementing scene synthesis, but popular methods include stochastic grammar models and autoregressive models, as discussed in the survey [34, 35]. Stochastic grammar models use a set of rules to procedurally generate content, by repeatedly sampling from a distribution of parameters and attributes until a termination criterion is reached. Autoregressive methods, on the other hand, use neural networks to iteratively predict the category and pose of objects in the scene, without relying on strict production rules.

Methods for quickly and automatically generating 3D scenes can significantly improve the robustness of trained agents when they encounter new environments. For example, building interiors created by architects or designers can be quickly and easily filled with furniture for demonstration purposes. In the context of video games, these methods, which operate directly on CAD models, can serve as an endless source of randomization, as shown in Figure Figure 2.8.

CAD models have a wide range of qualities, but generally, they are known for their geometric completeness and high level of accuracy. Today, CAD models are widely available and commonly used. There are numerous websites that offer a wide variety of publicly accessible 3D models, such as GrabCAD `https://grabcad.com` and CGTrader `https://cgtrader.com`. The

**Figure 2.8:** Scene synthesis demonstrations (right) where CAD models are iteratively inserted inside the partial room (left) [33] © 2019 IEEE

research community released several large-scale CAD model repositories such as ShapeNet [4] and the ABC dataset [36].

In summary, CAD models and machine learning techniques can be utilized to enhance reconstruction quality through methods such as post-processing or incorporating CAD information into the volumetric fusion process.

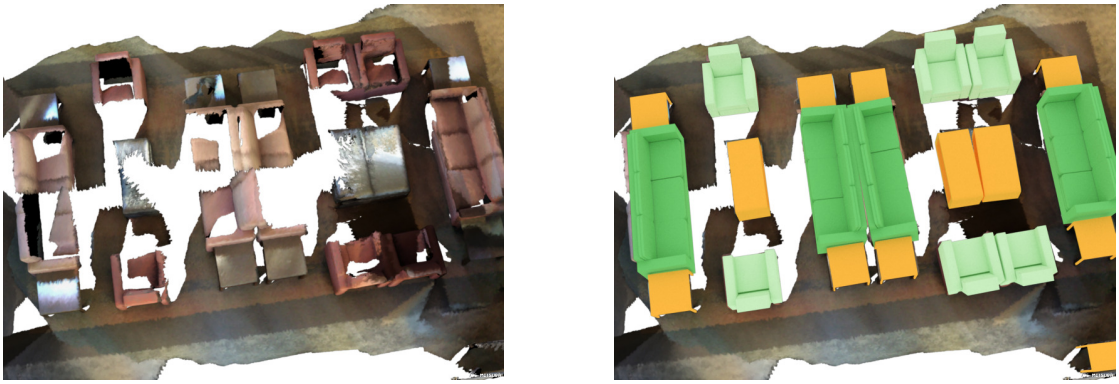
## 2.4 CAD Model Alignment

CAD models serve as blueprints for automating and standardizing production in many manufacturing processes. For example, a chair in a living room is directly created from a CAD model template. Instead of using a reconstruction process with high uncertainty to recreate the chair from observations, which often leads to poor results, it would be better to replace the scanned object with the actual CAD model. It is understood that CAD models are typically proprietary to a company and protected by copyright laws. However, for certain tasks, it may not be necessary to use an exact replica of the original model. In these cases, a visually similar model that is freely available can be used as a substitute. The process of aligning CAD models involves identifying objects in a 3D reconstruction and replacing them with the closest matching model within the same category. This includes determining the necessary adjustments for translation, rotation, and scale to ensure proper spatial alignment (see Figure 2.9 for illustration).

### 2.4.1 Handcrafted 3D Feature Descriptors

A popular approach for aligning two shapes is to calculate local feature descriptors across the surface geometry of the respective shapes and then in a second step match the features and ultimately estimate a final rigid pose. The assumption is that similar local regions of two different shapes will bear a similar descriptor.

**FPFH** Rusu et al. [37] proposed a new local feature descriptor from the family of Point Feature Histograms (PFH) that effectively describes a region spanned by a small radius around a point where distances between surfaces points, surface normals, and angular variations in normals are calculated. This step is repeated for neighboring points to construct the final feature descriptor.



**(a)** Top view of an RGB-D scan. **(b)** CAD model alignments overlayed onto the scan.

**Figure 2.9:** The CAD model alignment task aims to spatially align CAD models from a pre-defined input set onto an RGB-D scan. The end result is an object arrangement depicted on the right. Translation, rotation, and scale components are estimated to determine the final poses.

The discriminative power of the descriptor is demonstrated by effectively registering surfaces from various shapes and scales onto each other.

**SHOT** Tombari et al. [38] proposed an algorithm that estimates the eigenvalues of the spherical support of a given radius around a point to increase robustness in the presence of clutter and noise.

**PPF** Drost et al. [39, 40] introduce a global model surface descriptor called Point-Pair Features (PPF) to match against other shapes using a Hough voting scheme. PPF describe the relative position and orientation of two surface points in a four-dimensional vector from per-point positional and normal information. A global model descriptor is calculated by discretizing the distance and angular values from the PPF and grouping them together into a hash table. The final global pose is estimated by matching similar reference scene surface points to model points and finding an optimal local coordinate.

While these techniques work reasonably well for shapes from the same domain, it becomes challenging for inter-domain shapes because of different geometrical statistics. CAD models have sharp edges and perfectly flat & complete surfaces whereas 3D scans exhibit over-smoothed edges, holes, and rugged surfaces. The differences become more evident when comparing the distribution of surface normals. This leads to large distances in the feature-metric space even when describing the same semantic point in scan and CAD. The low-level geometrical features of CAD shapes and scanned shapes vary so much, that handcrafted conventional local feature descriptors purely based on geometric cues (e.g. normals, curvature, etc.) may not be capable to bridge the two domains.

### 2.4.2 Overview of Shape Alignment Methods

**Pre-Scanned 3D Model Alignment** The knowledge of existing objects in indoor environments can help to decrease the solution space for 3D model alignment. In many applications, this prior knowledge will be leveraged by pre-scanning occurring objects in the scene and thereby building a bespoke object database. A controlled object database can help to provide exact matching counterparts for detected object candidates or eliminate the need for scale estimates.

Kim et al. [41] proposed a method to transform point clouds from range scans into a lightweight high-level object model-based representation. The method works in two stages where in the first stage a primitive-based 3D model from a small number of object scans is learned. Scanned objects are segmented into parts and for each part, a primitive-based representation (box or cylinder) is found or fitted by a set of heuristics. Relationships between parts are established through 1-DoF junction entities (rotation or translation). In the second stage objects in the scene are detected and matched against the acquired 3D models. Parts of objects are segmented in the point cloud and then matched against parts of the model through handcrafted geometric 3D features. The final pose is estimated by determining the deformation parameters and the rigid transformation through Iterative Closest Point (ICP). Since deformation modes are estimated, the method is capable of aligning models onto articulated objects.

Salas-Moreno et al. [42] aligns high-quality pre-scanned 3D models onto the 3D map of a Simultaneous Localization and Mapping (SLAM) system in order to perform effective object-based loop closure and relocalization. They detect known and pre-scanned objects in the scene and estimate the 6DoF pose through a Hough Voting scheme of correspondences from PPF of oriented points on the surface of the object similar to the method of Drost et al. [39]. This method demonstrates beneficial joint effects when performing object recognition and alignment in the loop with SLAM.

Zeng et al. [43] developed a learned 3D local feature descriptor to find correspondences between RGB-D depth frames for surface registration. A 3D CNN ingests volumetric patches around points of interest and outputs feature vectors which are supervised by minimizing the $l_2$ distance for matching patch pairs and maximized for non-matching patch pairs respectively in a contrastive manner. The training data is extracted from 3D reconstructions where randomly sampled points are projected into the camera frustum of the corresponding views where positive patch pairs come from different views of the same underlying point and negative patch pairs are randomly cropped patches. For final surface registration, the Random Sample Consensus (RANSAC) algorithm is performed to find a suitable transformation between two input point clouds. The authors also demonstrated 6-DoF model alignment capabilities between pre-scanned object models and partial scans.

**CAD Model Alignment** In many cases, an external 3D model database exists already and hence a cumbersome pre-scanning effort is not necessary. A number of researchers have focused on scenarios where common objects indoors can be matched with *similar* objects from a large-scale database [44]. Besides the elimination of the cumbersome pre-scanning stage, an existing 3D model database can provide higher quality standards as CAD models usually source from professional modeling software.

Nan et al. [45] detect objects in a scanned scene by a search-classify approach where classification and segmentation are done in an alternating fashion. The scene is over-segmented into patches and the region around a seed point is gradually grown by using the classification score of its neighboring patches. A randomized decision forest is trained to perform a multi-label classification on the feature set of partial real-world objects. After this step, several template 3D models belonging to the same class are non-rigidly aligned onto the object point cloud by minimizing the point-to-template distance in a ICP manner. Finally, the model that has the lowest distance metric among the candidates is selected. This method showed appealing classification and 3D model alignment results on several input scans.

Shao et al. [46] interactively segments an input RGB-D frame into semantic regions and then aligns 3D models from a database into the semantic segments of the frame. A Conditional Random Field (CRF) solves the ten-class labeling problem through an appearance term that exploits local color information and a geometry term that exploits local depth information. A random regression forest uses patches of the segmented and labeled regions of the depth map to estimate the model index in the database and transformation parameters of a candidate model. For each patch, a geometrical feature descriptor is calculated which is ingested by the random regression forest to estimate the final class label by averaging over segment patches and trees in the forest. The final 6-DoF transformation is refined by fitting the model into the depth map through a point-to-point distance minimization.

Li et al. [47] presented a real-time system that performed RGB-D reconstruction and simultaneous retrieval and alignment of 3D CAD models onto the scan. During reconstruction, the system detects keypoints in the point cloud and in the 3D model through a Harris corner response value. A distance value between two keypoints is determined by projecting surface points from the local neighborhood of the scanned object into the local SDF grid of the model keypoint and then accumulating up to a final value. The assumption is that for a correct match, the occupied and known free space between the keypoint pair must align well and hence yield a low distance value. With a 1-Point RANSAC a correspondence set from the keypoint pairs between a candidate model and the unsegmented scan is iteratively found which determines the rigid transformation. In a final verification step, the transformation among multiple candidate models is picked that attains the highest surface coverage with the scanned scene.

While sparse correspondences tend to be compact and memory efficient, they typically rely on keypoint detectors to provide candidate points. On the other hand, dense correspondences provide enough signal for robust alignment, however, require an object mask for pruning. Dense correspondences provide associations for every point in the source domain to another point in the target domain. This implies, the number of associations comes in abundance which reduces the risks of catastrophic mismatches.

### 2.4.3 ICP

The simple approach for aligning two shapes is the ICP alignment algorithm. The key idea is that, given two sets of points, e.g. source point cloud and target point cloud, a rigid transformation from the source to the target can be estimated by iteratively determining correspondences through the nearest-neighbor distance metric. Upon determination of correspondences, an optimal closed-form solution for rotation and translation can be calculated. The process is repeated until

convergence. The main disadvantage of this method is that it is sensitive to initialization and hence can converge quickly to local minima because the correspondence finding is purely based on a geometric heuristic.

Correspondence finding remains a key challenge in the alignment process. Correspondences between two shapes can be established in various types (common examples are illustrated in Figure 2.10). Many handcrafted feature descriptors struggle to handle noisy, incomplete, and low-resolution real-world data. Discrepancies in the shape geometries can lead to false matches when selecting feature descriptors by a nearest-neighbor distance. A family of algorithms that addresses this is RANSAC [48]. This method samples a set of putative keypoint correspondences and iteratively eliminates false matches through keypoint rejection. After a number of iterations, a set of hypothetical inliers and outliers can be determined and with a Procrustes-like method a robust alignment can be estimated in the presence of outliers. However, the disadvantage of this method is that for reliable estimation a large number of trials are needed and the performance drastically degrades with a very low inlier ratio.

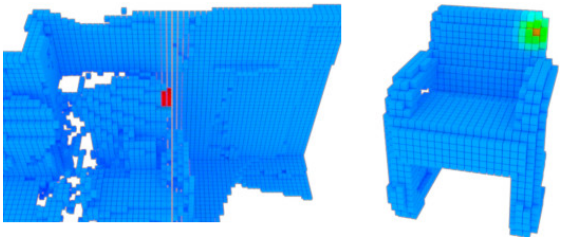
Assuming a set of already established correspondences between two point clouds $X \in \mathbb{R}^{3\times N}, Y \in \mathbb{R}^{3\times N}$, then the objective is to find a translation and rotation component that minimizes a rigid alignment problem such as:

$$f = \min_{R,t} \quad \sum_{i}^{N} \|R(x_i + t) - y_i\|^2 \tag{2.2}$$

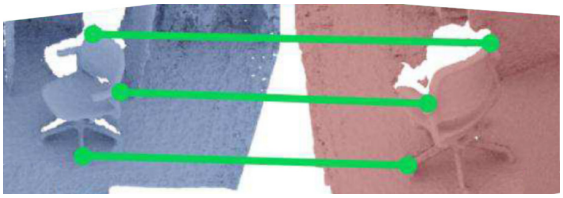
where $R \in SO(3)$ is the rotation component, $t \in \mathbb{R}^3$ is the translation component, $x_i \in X$ points of the source and $y_i \in Y$ points of the target. A approach that solves the minimization problem $f$ optimally is called *Procrustes superimposition algorithm* or *Kabsch algorithm* by applying following procedure [49]:

1. Recenter $X$ by its centroid: $x_i \leftarrow x_i - \bar{x}$
2. Recenter $Y$ by its centroid: $y_i \leftarrow y_i - \bar{y}$
3. Calculate covariance matrix: $C = XY^T$
4. Singular Value Decomposition (SVD) on $C$ as: $C = VSW^T$
5. Calculate $d = sign(det(C))$
6. Calculate $R = W \cdot diag(1, 1, d) \cdot V^T$
7. Calculate $t = -\bar{x} + \bar{y}$

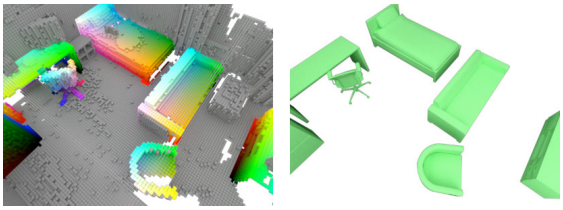
This algorithm is often seen at the end of various alignment pipelines because of its compactness and efficiency. While this formulation offers many advantages, it dictates the shape representation to be in point form which narrows down design choices.



**(a)** Sparse heatmap-based correspondences with a query point in red.



**(b)** Sparse correspondences established with a local feature descriptor [43] © 2017 IEEE



**(c)** Dense correspondences (RGB color-coded) cover the entire object from the source domain (left) and range between 0 and 1 representing the normalized coordinate in the target domain (right).

**Figure 2.10:** Correspondences can assume various types. The choice of the correspondence type will further narrow down the downstream alignment algorithm. This figure illustrates three different types of correspondences, each of which requires distinct optimization strategies.

### 2.4.4 Shape Retrieval

Shape retrieval plays an important part in the CAD model alignment task. The task requires the selection of correct candidate CAD models for later pose fitting. The goal is to find one or multiple plausible in-category candidate CAD models for a query object. Retrieval methods face challenges in inferring strong informational cues from 3D data in the presence of noise, clutter, and missing data. Depending on the scenario geometry-based properties, perceptual properties, and semantic cues may be necessary to allow retrieval with a reasonable tolerance in similarity as illustrated in Figure 2.11.

Since shape similarity has a heavy perceptual factor, it remains difficult to get a mathematical grasp on it. The SHREC challenges [50, 51] tackle the problem by manually pairing RGB-D objects with CAD models based on category and sub-category annotations. Due to the lack of

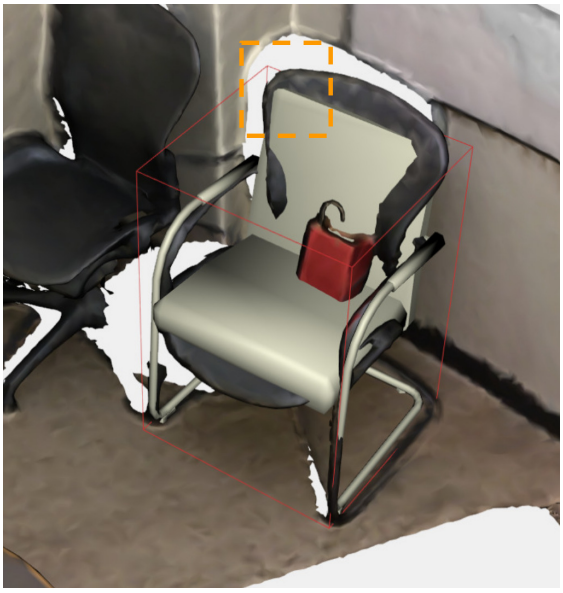


**Figure 2.11:** This overlay demonstrates the difficulty of shape retrieval and correspondence finding. On a high level, both chairs are very similar to each other. Low-level geometric features differ drastically, however. For example, to reliably describe the corner in the orange box both geometric *and* semantic features must be considered.

standardized shape similarity metrics, coarse proxies such as in-category matches are used to bootstrap the formalization of the problem. CAD model retrieval is considered to be correct if the category and sub-category match with the query scan object. Even though a large variety of shape descriptors have been developed in the past, the complexity of shapes in real-world objects remains a challenge.

A popular approach is based on Light Field Descriptors [52], where objects are rendered from several viewpoints and then Fourier descriptors are calculated on the shape silhouette. This method works well for shape classification, however, is unable to describe color appearance. Most hand-crafted shape descriptors fail to assess robustly shape similarities across different domains. Recently, neural networks trained with 3D data for shape classification tasks have been used to provide shape descriptors.

Dahnert et al. [53] learned a joint-embedding space with real-world scanned objects and CAD models with a distance metric indicating the perceptual similarity between a real and synthetic object (see Figure 2.12). An annotated dataset of 5102 scan-CAD similarity rankings between scanned objects and CAD models has been established to train a series of stacked hourglass 3D CNN to map object features into a latent embedding space. The network architecture ingests an occupancy grid of a scanned real-world object and performs a foreground-background segmentation in the first stage where the foreground object is completed into a volumetric grid in a subsequent step. Lastly, the completed object is encoded by a siamese network into a feature vector where it is supervised by the features of a positive matching CAD counterpart and a randomized negative non-matching CAD model in a contrastive manner. The method showed plausible retrieval results on challenging partial and coarse query scans.

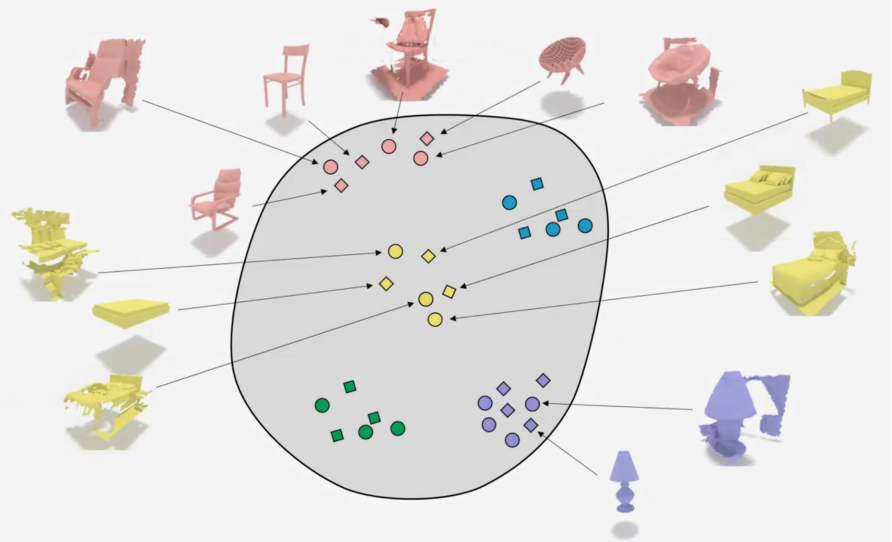


**Figure 2.12:** Illustration of a learned similarity metric space containing synthetic and real elements by [53] © 2019 IEEE. Similarity distances are learned from human annotators based on perceptual similarity.

### 2.4.5 Image-Shape Alignment

Analogously to the 3D CAD model alignment task, the goal of image-shape alignment is to detect, retrieve and estimate the pose of a CAD model on single images. Limited information in the monocular images due to the scale ambiguity or occlusions and partial observations make this task a challenging problem. Since the camera frustum only covers a relatively small part of the whole scene, there is only a limited number of objects seen per image instead of all objects at the same time.

Gupta et al. [54] aligned 3D CAD models onto RGB-D frames by feeding a normal map into a CNN-based segmentation network that isolates the models and then coarsely estimates the heading angle through a multi-label classification module. A small number of representative CAD models per category is exhaustively fitted into the segmentation mask of the candidate object instance using ICP to refine initial translation, rotation, and scale components. During the alignment process, the current transformation estimate is used to render the CAD model candidate as a depth map, which was then used to find correspondences with the points inside the segmentation mask. To select the best-explaining CAD model, a linear classifier is used to score alignments from a set of hand-crafted features, including the overlap ratios between the rendered model and the instance mask.

Sun et al. [55] worked towards a dataset containing pixel-accurate shape-to-image alignments between 10069 image to CAD model pairs (see example in Figure 2.13). The alignment annotations are remarkably precise and exceed existing 2D-3D shape alignment datasets by quality and quantity. In the Pix3D dataset, captured objects are centered in the image, and the existence of an exact matching CAD counterpart is assumed for retrieval and CAD alignment. While the

pose alignment consists of a 3D translation $t \in \mathbb{R}$ and a 3D rotation component $R \in SO(3)$, the evaluation is carried out in image space. The scale estimate remains ambiguous because the focal length and a 3D translation are estimated. Furthermore, the authors provided several qualitative evaluation metrics for the image-shape alignment task.

Georgakis et al. [56] presented a method where the pose of a candidate CAD model with respect to an input RGB frame is estimated through a learned keypoint detector and a learned keypoint descriptor. A base 2D CNN is used to predict a keypoint score map which is supervised by the keypoint score weighted reprojection error when transforming a keypoint set from one depth map into another depth map. The keypoint descriptor branch from the same 2D CNN is trained by a triplet loss to ensure distinctiveness for correspondence finding. Another 2D CNN ingesting RGB frames generates keypoint score maps and feature descriptors that are supervised by the teacher 2D CNN to ensure cross-modality inference during testing. The final pose is estimated by the RANSAC and Perspective-n-Point (PnP) algorithm.

**Figure 2.13:** Image-to-shape pair samples from Pix3D [55] © 2018 IEEE

## 2.5 Data-driven 3D Scene Understanding

### 2.5.1 3D Semantic Instance Segmentation

The 3D Semantic Instance Segmentation (SIS) task is a process that aims to detect and segment individual objects in a 3D scene, while also assigning semantic labels to each object. This task does not alter or enhance the quality of the scene, but it provides a deeper understanding of the objects and their relationships within the scene. It also serves as a foundation for further analysis and algorithms.

Hou et al. [57] developed a system for 3D semantic instance segmentation that encodes color images and the 3D geometry of a scene into a feature volume. This feature volume is then used by a new 3D region proposal network to detect objects and their bounding boxes, along with class labels. The system also includes a mask head that predicts per-voxel masks within the predicted bounding boxes. By backprojecting multiple RGB views into a 3D volume, the authors were able to improve overall prediction accuracy. A sample prediction is seen in Figure 2.14.

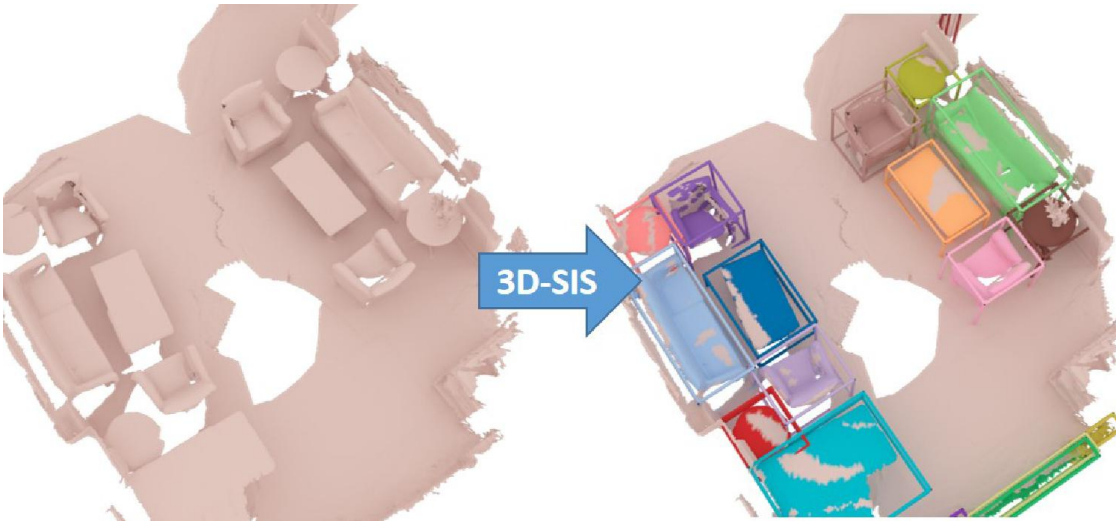



**Figure 2.14:** Illustration of 3D Semantic Instance Segmentation where the spanned bounding box corresponds to the extent of the detected objects and individual colors highlight the different instances. A class label is assigned to each detected object. [57] © 2019 IEEE

Qi et al. [58] introduced *VoteNet*, a neural network that operates on point clouds to detect objects in the scene and predict a bounding box and class label for each object. A point cloud is encoded by a PointNet++ [59]-style feature learning backbone into a new set of seed points that cast votes to the bounding box center of the object they belong to. The votes are grouped into K clusters and used by a proposal module to predict the size and heading angle of the bounding box, as well as estimate a class label. Despite its compact network size and simple architecture, VoteNet surpasses other methods in terms of speed and accuracy.

Lahoud et al. [60] approach the 3D semantic instance segmentation task as a multi-task metric learning problem. A 3D CNN encodes an input voxel grid and produces two sets of latent features. The first set uses an appropriate loss function in a contrastive manner to bring feature vectors of voxels belonging to the same instance closer together in the embedding space while pushing those belonging to different instances farther apart. The second set uses its feature vectors to estimate the direction from a voxel to its corresponding object center. Finally, a mean-shift algorithm clusters the embedding spaces and combines them to produce the final result.

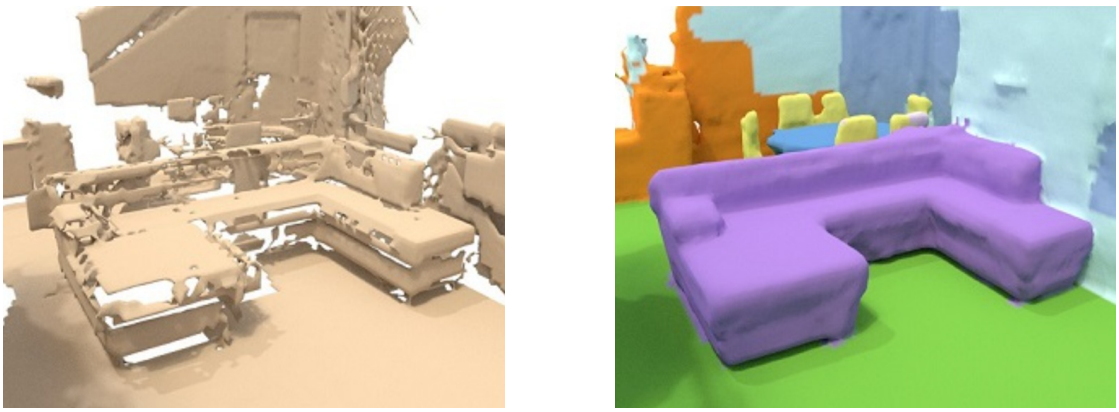
Jiang et al. [61] propose a method that ingests a point cloud by a 3D UNet sparse convolution backbone [62] to generate per-point features which are used to estimate offset vectors and semantic labels. The point positions, offset vectors and semantic labels are then used in a clustering module to group the points into two complementary candidate sets. A module called *ScoreNet* assigns a score value to each instance candidate from both sets. Lastly, the clusters and their respective scores are used in a Non-maximum Suppression (NMS) step to identify the final object instances.

### 2.5.2 Semantic Scene Completion

RGB-D reconstruction is subject to a variety of error sources, and not all of them can be addressed through advanced sensing technology alone. To enhance the quality of scene reconstruction, it is necessary to consider factors beyond pure geometry. Despite the inherent uncertainty associated with predicting unobserved space, utilizing learned scene priors by plausible completion procedures has emerged as a popular approach to improve scene quality as comprehensively surveyed by Roldao et al. [63]. Recent advancements in 3D deep learning have expanded the capabilities of scene completion methods by allowing for the prediction of semantic labels, as it is assumed that semantics and geometry are closely interconnected.

Song et al. [64] presented a system that uses a 3D CNN to convert a 3D TSDF grid generated by backprojecting a depth map of a view frustum into a complete 3D occupancy grid of the scene. The resulting output includes voxel-wise semantic labels. Despite being trained on a synthetic dataset with rendered depth maps and access to complete ground truth scene data, the method demonstrates strong performance on real-world data. Chen et al. [65] extended this method by adding an adversarial loss to increase the realism of the generated scenes.

Dai et al. [66] introduced an autoregressive method that predicts both the completed geometry and semantic segmentation from an input scan encoded as a TSDF grid. The method operates on hierarchies of resolutions where at each hierarchy an input partial scan and the previous low-resolution prediction are fed to produce a finer completion. The use of a fully-convolutional architecture allows the system to effectively complete large scenes during inference without the need to divide them into smaller subvolumes. See sample prediction in Figure 2.15.



**(a)** Partial input scan.

**(b)** Completed geometry with semantic labels.

**Figure 2.15:** Semantic scene completion results from ScanComplete [66] © 2018 IEEE

Rist et al. [67] used a deep implicit function to tackle the semantic scene completion task. They encoded an input LIDAR point cloud into a latent representation and then, together with a query position, decoded it into a probability vector representing the semantic class. To address the sparsity of a single sweep of a LIDAR point cloud, multiple sweeps were merged into a common reference frame, which served as the ground truth target for supervised training. Due to

its architecture, the system was able to not only achieve strong results but also process a larger spatial extent for prediction.

Wang et al. [68] proposed an octree-based CNN approach to complete objects and scenes. The octree-based encoder-decoder utilizes a UNet [69] design where partial shapes are encoded by applying convolution operations on every octree node. During decoding, a specialized layer at each level estimates the occupancy of a node, which recursively activates its children and enables further deconvolutions. At the leaf node, a local plane patch spanning the surface or a class label is predicted, depending on the task.

# 3 CAD Model Alignment in RGB-D Scans Using Sparse Heatmap Correspondences

This chapter introduces the following paper:
**Avetisyan, A.**, Dahnert, M., Dai, A., Savva, M., Chang, A. X., & Nießner, M. (2019). Scan2cad: Learning cad model alignment in rgb-d scans. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 2614-2623).

**Summary** We present a new method for aligning 3D CAD models from a database to the noisy and incomplete geometry of an RGB-D scan, in order to create a clean and compact CAD representation of the scan. Specifically, for indoor scene 3D reconstruction, our method inputs a set of CAD models and predicts a 9-DoF pose that aligns each model with the underlying scan geometry. To tackle this problem, we create a new scan-to-CAD alignment dataset based on 1506 ScanNet scans with 97607 manually annotated keypoint pairs between 14225 CAD models from ShapeNet and their counterpart objects in the scans. A web-based application has been created to assign annotation tasks to our expert annotators. To train our correspondence prediction network, the manually annotated keypoint pairs and new augmented keypoint pairs, by utilizing the estimated rigid transformation, are used in a 3D CNN to predict correspondences in form of heatmaps. Ground-truth heatmap targets are created by blurring the surface point with a fixed-size Gaussian kernel. For a given crop from the scan around a query keypoint, the 3D CNN predicts a heatmap on the voxelized geometry of a candidate CAD model. Additionally, a binary score is estimated whether the input scan and candidate CAD model match semantically and a 3D scale is regressed for the normalized CAD model. The binary score helps to discard non-matching CAD models early in the process. Based on these heatmap correspondences, we formulate a variational energy minimization that aligns a given set of CAD models to the scan. In the alignment stage, a Levenberg–Marquardt algorithm finds an optimal 9-DoF pose that minimizes the heatmap response function where the estimate from the CNN initializes the scale. From the set of aligned candidate CAD models, the one is selected that reaches the lowest cost objective. We evaluate our approach on our newly introduced benchmark where we outperform both handcrafted feature descriptors and state-of-the-art CNN-based methods by 21.39%.

**Contributions** The first author created the 3D annotation tool using javascript and WebGL for data collection, implemented the heatmap correspondence prediction method in PyTorch, and the 9-DoF alignment algorithm in C++ with the Ceres library. Manuel Dahnert developed the baseline methods. The co-authors assisted with data annotation and contributed to discussions that resulted in the final publication.

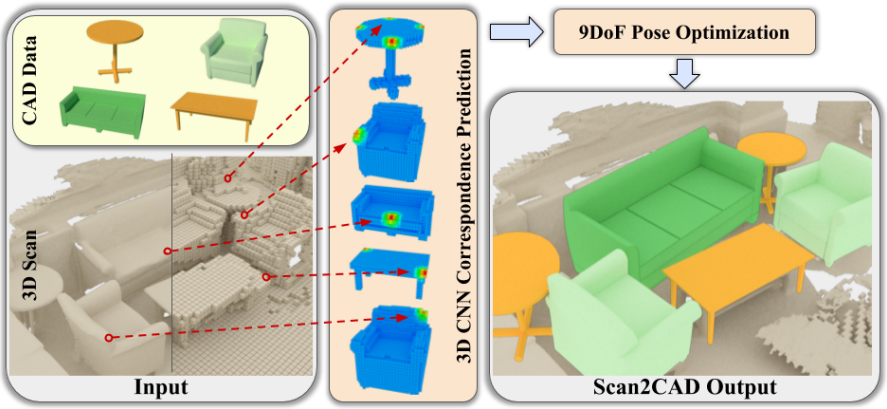



**Figure 3.1:** Scan2CAD takes as input an RGB-D scan and a set of 3D CAD models (left). We then propose a novel 3D CNN approach to predict heatmap correspondences between the scan and the CAD models (middle). From these predictions, we formulate an energy minimization to find optimal 9 DoF object poses for CAD model alignment to the scan (right).

## 3.1 Introduction

In recent years, the wide availability of consumer-grade RGB-D sensors, such as the Microsoft Kinect, Intel Real Sense, or Google Tango, has led to significant progress in RGB-D reconstruction. We now have 3D reconstruction frameworks, often based on volumetric fusion [13], that achieve impressive reconstruction quality [70, 15, 16, 20, 71] and reliable global pose alignment [20, 72, 22]. At the same time, deep learning methods for 3D object classification and semantic segmentation have emerged as a primary consumer of large-scale annotated reconstruction datasets [5, 73]. These developments suggest great potential in the future of 3D digitization, for instance, in applications for virtual and augmented reality.

Despite these improvements in reconstruction quality, the geometric completeness and fine-scale detail of indoor scene reconstructions remain a fundamental limitation. In contrast to artist-created computer graphics models, 3D scans are noisy and incomplete, due to sensor noise, motion blur, and scanning patterns. Learning-based approaches for object and scene completion [74, 64, 66] cannot reliably recover sharp edges or planar surfaces, resulting in quality far from artist-modeled 3D content.

One direction to address this problem is to retrieve a set of CAD models from a shape database and align them to an input scan, in contrast to a bottom-up reconstruction of the scene geometry. If all objects are replaced in this way, we obtain a clean and compact scene representation, precisely serving the requirements for many applications ranging from AR/VR scenarios to architectural design. Unfortunately, matching CAD models to scan geometry is an extremely challenging problem: While high-level geometric structures might be similar, the low-level geometric features differ significantly (e.g., surface normal distributions). This severely limits the applicability of handcrafted geometric features, such as FPFH [37], SHOT [75], point-pair-features [40], or SDF-based feature descriptors [47]. While learning-based approaches like random forests [45, 46] exist, their model capacity remains relatively low, especially in comparison to more modern

methods based on deep learning, which can achieve significantly higher accuracy, but remain at their infancy. We believe this is in large part attributed to the lack of appropriate training data.

In this paper, we make the following contributions:

- We introduce the Scan2CAD dataset, a large-scale dataset comprising 97607 pairwise keypoint correspondences and 9 Degrees of Freedom (DoF) alignments between 14225 instances of 3049 unique synthetic models, between ShapeNet [4] and reconstructed scans in ScanNet [5], as well as oriented bounding boxes for each object.
- We propose a novel 3D CNN architecture that learns a joint embedding between real and synthetic 3D objects to predict accurate correspondence heatmaps between the two domains.
- We present a new variational optimization formulation to minimize the distance between scan keypoints and their correspondence heatmaps, thus obtaining robust 9DoF scan-to-CAD alignments.

## 3.2 Related work

**RGB-D Scanning and Reconstruction** The availability of low-cost RGB-D sensors has led to significant research progress in RGB-D 3D reconstruction. A very prominent line of research is based on volumetric fusion [13], where depth data is integrated in a volumetric signed distance function. Many modern real-time reconstruction methods, such as KinectFusion [70, 15], are based on this surface representation. In order to make the representation more memory-efficient, octree [18] or hash-based scene representations have been proposed [16, 71]. An alternative fusion approach is based on points [19]; the reconstruction quality is slightly lower, but it has more flexibility when handling scene dynamics and can be adapted on-the-fly for loop closures [20]. Very recent RGB-D reconstruction frameworks combine efficient scene representations with global pose estimation [72], and can even perform online updates with global loop closures [22]. A closely related direction to ours (and a possible application) is recognition of objects as a part of a SLAM method, and using the retrieved objects as part of a global pose graph optimization [42, 76].

**3D Features for Shape Alignment and Retrieval** Geometric features have a long-established history in computer vision, such as Spin Images [77], Fast Point Feature Histograms (FPFH) [37], or Point-Pair Features (PPF) [40]. Based on these descriptors or variations of them, researchers have developed shape retrieval and alignment methods. For instance, Kim et al. [41] learn a shape prior in the form of a deformable part model from input scans to find matches at test time; or AA2h [78] use a similar approach to PPF, where a histogram of normal distributions of sample points is used for retrieval. Li et al. [47] propose a formulation based on a hand-crafted TSDF feature descriptor to align CAD models in real-time to RGB-D scans. While these retrieval approaches based on hand-crafted geometric features show initial promise, they struggle to generalize matching between the differing data characteristics of clean CAD models and noisy, incomplete real-world data.

An alternative direction is learned geometric feature descriptors. For example, Nan et al. [45] use a random decision forest to classify objects on over-segmented input geometry from high-quality scans. Shao et al. [46] introduce a semi-automatic system to resolve segmentation ambiguities, where a user first segments a scene into semantic regions, and then shape retrieval is applied. 3DMatch [43] leverage a Siamese neural network to match keypoints in 3D scans for pose estimation. Zhou et al. [79] is of similar nature, proposing a view consistency loss for 3D keypoint prediction network on RGB-D image data. Inspired by such approaches, we develop a 3D CNN-based approach targeting correspondences between the synthetic domain of CAD models and the real domain of RGB-D scan data.

Other approaches retrieve and align CAD models given single RGB [80, 81, 55, 82] or RGB-D [54, 83] images. These methods are related, but our focus is on geometric alignment independent of RGB information, rather than CAD-to-image.

**Shape Retrieval Challenges and RGB-D Datasets** Shape retrieval challenges have recently been organized as part of the Eurographics 3DOR [50, 84]. Here, the task was formulated as matching of object instances from ScanNet [5] and SceneNN [85] to CAD models from the ShapeNetSem dataset [4]. Evaluation only considered binary in-category vs out-of-category (and sub-category) match as the notion of relevance. As such, this evaluation does not address the alignment quality between scan objects and CAD models, which is our focus.

ScanNet [5] provides aligned CAD models for a small subset of the annotated object instances (for only 200 objects out of the total 36000). Moreover, the alignment quality is low with many object category mismatches and alignment errors, as the annotation task was performed by crowdsourcing. The PASCAL 3D+ [86] dataset annotates 13898 objects in the PASCAL VOC images with coarse 3D poses defined against representative CAD models. ObjectNet3D [87] provides a dataset of CAD models aligned to 2D images, approximately 200K object instances in 90K images. The IKEA objects [80] and Pix3D [55] datasets similarly provide alignments of a small set of identifiable CAD models to 2D images of the same objects in the real world; the former has 759 images annotated with 90 models, the latter has 10069 annotated with 395 models.

No existing dataset provides fine-grained object instance alignments at the scale of our Scan2CAD dataset with 14225 CAD models (3049 unique instances) annotated to their scan counterpart distributed on 1506 3D scans.

## 3.3 Overview

**Task** We address alignment between clean CAD models and noisy, incomplete 3D scans from RGB-D fusion, as illustrated in Figure 3.1. Given a 3D scene $\mathbb{S}$ and a set of 3D CAD models $\mathbb{M} = \{m_i\}$, the goal is to find a 9DoF transformation $T_i$ (3 degrees for translation, rotation, and scale each) for every CAD model $m_i$ such that it aligns with a semantically matching object $\mathbb{O} = \{o_j\}$ in the scan. One important note is that we cannot guarantee the existence of 3D models which exactly matches the geometry of the scan objects.

**Dataset and Benchmark** In section 3.4, we introduce the construction of our Scan2CAD dataset. We propose an annotation pipeline designed for use by trained annotators. An annotator first inspects a 3D scan and selects a model from a CAD database that is geometrically similar to a target object in the scan. Then, for each model, the annotator defines corresponding keypoint pairs between the model and the object in the scan. From these keypoints, we compute ground truth 9DoF alignments. We annotate the entire ScanNet dataset and use the original training, validation, and test splits to establish our alignment benchmark.

**Heatmap Prediction Network** In section 3.5, we propose a 3D CNN taking as input a volume around a candidate keypoint in a scan and a volumetric representation of a CAD model. The network is trained to predict a correspondence heatmap over the CAD volume, representing the likelihood that the input keypoint in the scan is matching with each voxel. The heatmap prediction is formulated as a classification problem, which is easier to train than regression, and produces sparse correspondences needed for pose optimization.

**Alignment Optimization** section 3.6 describes our variational alignment optimization. To generate candidate correspondence points in the 3D scan, we detect Harris keypoints, and predict correspondence heatmaps for each Harris keypoint and CAD model. Using the predicted heatmaps we find optimal 9DoF transformations. False alignments are pruned via a geometric confidence metric.

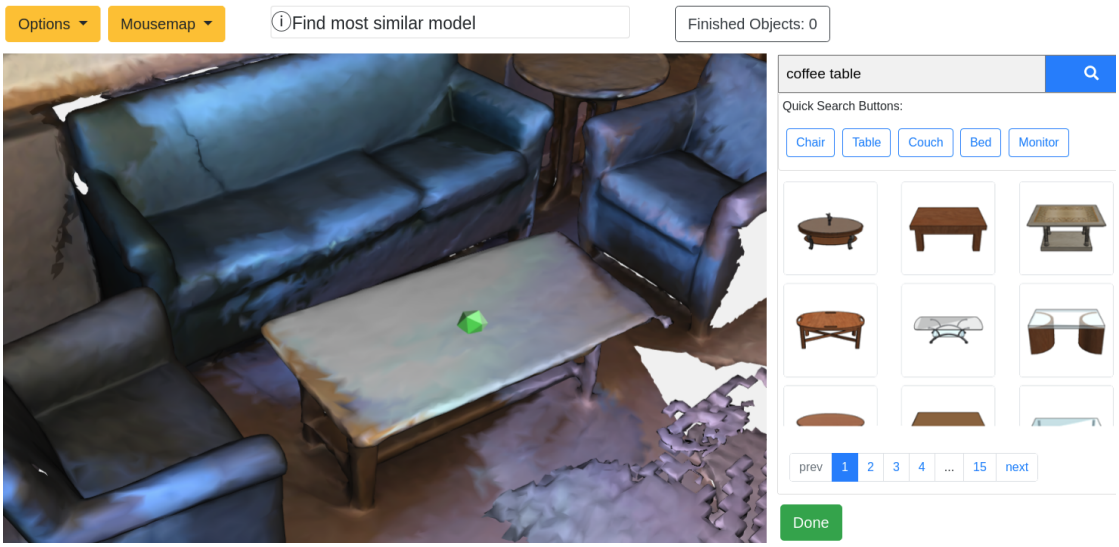
## 3.4 Dataset

Our Scan2CAD dataset builds upon the 3D scans from ScanNet [5] and CAD models from ShapeNet [4]. Each *scene* $\mathbb{S}$ contains multiple *objects* $\mathbb{O} = \{o_i\}$, where each *object* $o_i$ is matched with a ShapeNet CAD model $m_i$ and both share multiple keypoint pairs (correspondences) and one transformation matrix $T_i$ defining the alignment. Note that ShapeNet CAD models have a consistently defined front and upright orientation which induces an amodal tight oriented bounding box for each scan object, see Figure 3.3.

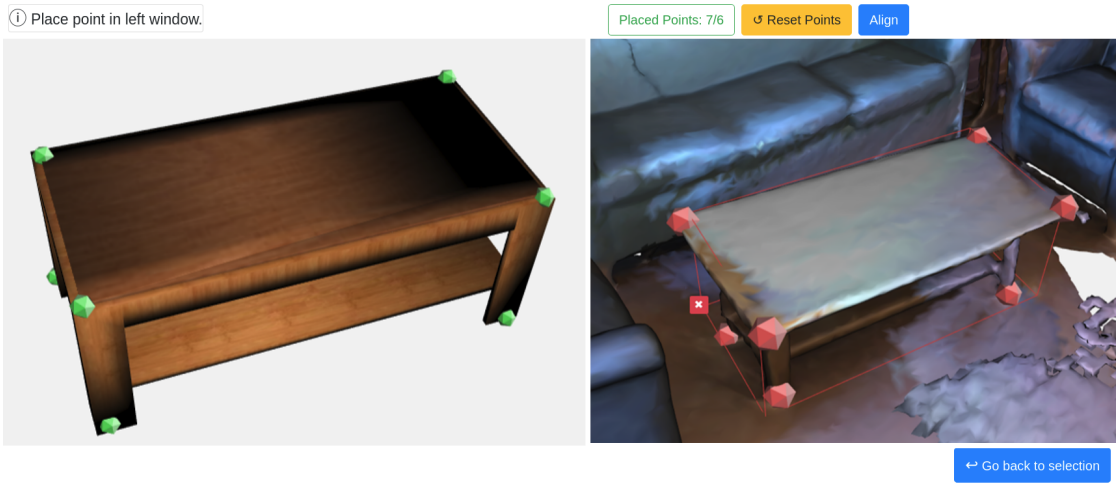
### 3.4.1 Data Annotation

The annotation is done via a web application that allows for simple scaling and distribution of annotation jobs; see Figure 3.2. The annotation process is separated into two steps. The first step is object *retrieval*, where the user clicks on a point on the 3D scan surface, implicitly determining an object category label from the ScanNet object instance annotations. We use the instance category label as query text in the ShapeNet database to retrieve and display all matching CAD models in a separate window as illustrated in Figure 3.2a. After selecting a CAD model the user performs *alignment*.

In the alignment step, the user sees two separate windows in which the CAD model (left) and the scan object (right) are shown (see Figure 3.2b). Keypoint correspondences are defined by alternately clicking paired points on the CAD model and scan object. We require users to specify at least 6 keypoint pairs to determine a robust ground truth transformation. After keypoint




**(a)** First step: Retrieval view.




**(b)** Second step: Alignment view.

**Figure 3.2:** Our annotation web interface is a two-step process. (a) After the user places an anchor on the scan surface, class-matching CAD models are displayed on the right. (b) Then the user annotates keypoint pairs between the scan and CAD model from which we derive the ground truth 9DoF transformation.

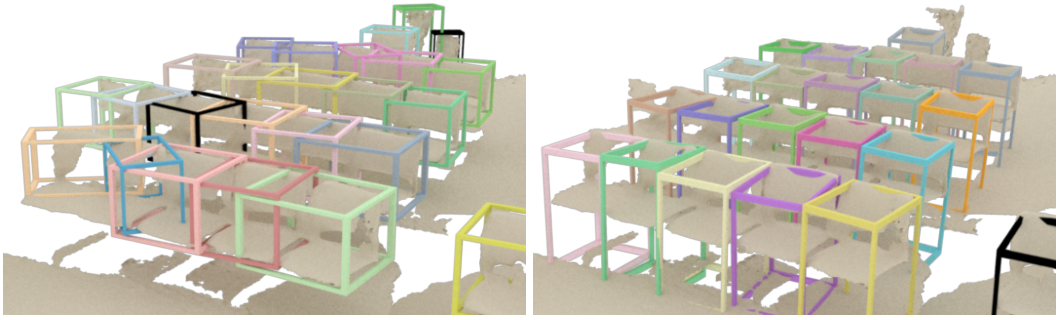


**Figure 3.3:** (Left) Oriented bounding boxes (OBBs) computed from the instance segmentation of ScanNet [5] are often incomplete due to missing geometry (e.g., in this case, missing chair legs). (Right) Our OBBs are derived from the aligned CAD models and are thus complete.

pairs are specified, the alignment computation is triggered by clicking a button. This alignment (given exact 1-to-1 correspondences) is solved with the genetic algorithm *CMA-ES* [88, 89] that minimizes the point-to-point distance over 9 parameters. In comparison to gradient-based methods or Procrustes superimposition method, we found this approach to perform significantly better in reliably returning high-quality alignments regardless of initialization.

The quality of these keypoint pairs and alignments was verified in several verification passes, with re-annotations performed to ensure a high quality of the dataset. The verification passes were conducted by the authors of this work.

A subset of the ShapeNet CAD models have symmetries that play an important role in making correspondences. Hence, we annotated all ShapeNet CAD models used in our dataset with their rotational symmetries to prevent false negatives in evaluations. We defined 2-fold ($C_2$), 4-fold ($C_4$) and infinite ($C_\infty$) rotational symmetries around a canonical axis of the object.

### 3.4.2 Dataset Statistics

The annotation process yielded 97607 keypoint pairs on 14225 (3049 unique) CAD models with their respective scan counterpart distributed on a total of 1506. Approximately $28\%$ out of the 3049 CAD models have a symmetry tag (either $C_2$, $C_4$ or $C_\infty$).

Given the complexity of the task and to ensure high quality annotations, we employed 7 part-time annotators (in contrast to crowd-sourcing). On average, each scene has been edited $1.76$ times throughout the re-annotation cycles. The top 3 annotated model classes are chairs, tables and cabinets which arises due to the nature of indoor scenes in ScanNet. The number of objects aligned per scene ranges from $1$ to $40$ with an average of $9.3$. It took annotators on average of $2.48$min to align each object, where the time to find an appropriate CAD model dominated the time for keypoint placement. The average annotation time for an entire scene is $20.52$min.

It is interesting to note that manually placed keypoint correspondences between scans and CAD models differ significantly from those extracted from a Harris corner detector. Here, we

compare the mean distance from the annotated CAD keypoint to: (1) the corresponding annotated scan keypoint ($= 3.5cm$) and (2) the nearest Harris keypoint in the scan ($= 12.8cm$).

### 3.4.3 Benchmark

Using our annotated dataset, we designed a benchmark to evaluate scan-to-CAD alignment methods. A model alignment is considered successful only if the category of the CAD model matches that of the scan object *and* the pose error is within translation, rotational, and scale bounds relative to the ground truth CAD. We do not enforce strict instance matching (i.e., matching the exact CAD model of the ground truth annotation) as ShapeNet models typically do not identically match real-world scanned objects. Instead, we treat CAD models of the same category as interchangeable (according to the ShapeNetCorev2 *top-level synset*).

Once a CAD model is determined to be aligned correctly, the ground truth counterpart is removed from the candidate pool in order to prevent multiple alignments to the same object. Alignments are fully parameterized by 9 pose parameters. A quantitative measure based on bounding box overlap (IoU) can be readily calculated with these parameters as CAD models are defined on the unit box. The error thresholds for a successful alignment are set to $\epsilon_t \leq 20$cm, $\epsilon_r \leq 20°$, and $\epsilon_s \leq 20\%$ for translation, rotation, and scale respectively (for extensive error analysis please see the supplemental). The rotation error calculation takes $C_2$, $C_4$ and $C_\infty$ rotated versions into account.

The Scan2CAD dataset and associated symmetry annotations are available to the community. For standardized comparison of future approaches, we operate an automated test script on a hidden test set that can be found under `www.Scan2CAD.org`.

## 3.5 Correspondence Prediction Network

### 3.5.1 Data Representation

Scan data is represented by its signed distance field (SDF) encoded in a volumetric grid and generated through *volumetric fusion* [13] from the depth maps of the RGB-D reconstruction (voxel resolution = $3cm$, truncation = $15cm$). For the CAD models, we compute unsigned distance fields (DF) using the level-set generation toolkit by Batty [90].

### 3.5.2 Network Architecture

Our architecture takes as input a pair of voxel grids: A SDF centered at a point in the scan with a large receptive field at $64^3$ size, and a DF of a particular CAD model at $32^3$ size. We use a series of convolutional layers to separately encode each input stream (see Figure 3.4). The two encoders compress the volumetric representation into compact feature volumes of $4^3 \times 64$ (scan) and $4^3 \times 8$ (CAD) which are then concatenated before passing to the decoder stage. The decoder stage predicts three output targets, heatmap, compatibility, and scale, described as follows:

**Heatmap** The first output is a heatmap $H : \Omega \rightarrow [0, 1]$ over the $32^3$ voxel domain $\Omega \subset \mathbb{N}^3$ of the CAD model producing the voxel-wise correspondence probability. This indicates the

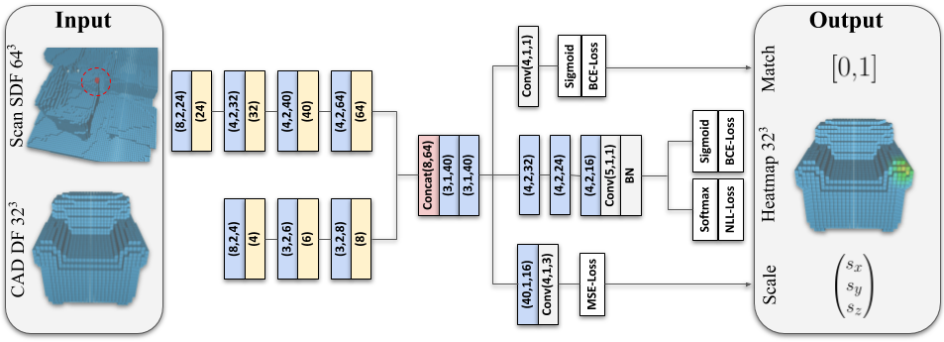



**Figure 3.4:** 3D CNN architecture of our Scan2CAD approach: we take as input SDF chunks around a given keypoint from a 3D scan and the DF of a CAD model. These are encoded with 3D CNNs to learn a shared embedding between the synthetic and real data; from this, we classify whether there is semantic compatibility between both inputs (top), predict a correspondence heatmap in the CAD space (middle) and the scale difference between the inputs (bottom).

probability of matching each voxel in $\Omega$ to the center point of the scan SDF. We train our network using a combined binary cross-entropy (BCE) loss and a negative log-likelihood (NLL) to predict the final heatmap $H$. The raw output $S : \Omega \rightarrow \mathbb{R}$ of the last layer in the decoder is used to generate the heatmaps:

$$
\begin{aligned}
&H_1 : \Omega \rightarrow [0,1], \quad x \mapsto \text{sigmoid}(S(x)) \\
&H_2 : \Omega \rightarrow [0,1], \quad x \mapsto \text{softmax}(S(x)) \\
&\mathcal{L}_H = \sum_{x \in \Omega} w(x) \cdot \text{BCE}(H_1, H_{\text{GT}}) + \sum_{x \in \Omega} v \cdot \text{NLL}(H_2, H_{\text{GT}})
\end{aligned}
$$

where $w(x) = 64.0$ if $H_{\text{GT}}(x) > 0.0$ else $1.0, v = 64$ are weighting factors to increase the signal of the few sparse positive keypoint voxels in the voxel grid ($\approx 99\%$ of the target voxels have a value equal to 0). The combination of the sigmoid and softmax terms is a compromise between high recall but low precision using sigmoid, and more locally sharp keypoint predictions using softmax over all voxels. The final target heatmap, used later for alignment, is constructed with an element-wise multiplication of both heatmap variations: $H = H_1 \circ H_2$.

**Compatibility** The second prediction target is a single probability score $\in [0, 1]$ indicating semantic compatibility between scan and CAD. This category equivalence score is 0 when the category labels are different (e.g., scan table and CAD chair) and 1 when the category labels match (e.g., scan chair and CAD chair). The loss function for this output is a sigmoid function followed by a BCE loss:

$$
\mathcal{L}_{\text{compat.}} = \text{BCE}(\text{sigmoid}(x), x_{\text{GT}})
$$

**Scale** The third output predicts the scale $\in \mathbb{R}^3$ of the CAD model to the respective scan. Note that we do not explicitly enforce positivity of the predictions. This loss term is a mean-squared-

error (MSE) for a prediction $x \in \mathbb{R}^3$:

$$\mathcal{L}_{\text{scale}} = \text{MSE}(x, x_{\text{GT}}) = \|x - x_{\text{GT}}\|_2^2$$

Finally, to train our network, we use a weighted combination of the presented losses:

$$\mathcal{L} = 1.0\mathcal{L}_{\text{H}} + 0.1\mathcal{L}_{\text{compat.}} + 0.2\mathcal{L}_{\text{scale}}$$

where the weighting of each loss component was empirically determined for balanced convergence.

### 3.5.3 Training Data Generation

**Voxel Grids** Centered scan volumes are generated by projecting the annotated keypoint into the scan voxel grid and then cropping around it with a crop window of $63^3$. Ground truth heatmaps are generated by projecting annotated keypoints (and any symmetry-equivalent keypoints) into the CAD voxel grid. We then use a Gaussian blurring kernel ($\sigma = 2.0$) on the voxel grid to account for small keypoint annotation errors and to avoid sparsity in the loss residuals.

**Training Samples** With our annotated dataset we generate $N_{P,\text{ann.}} = 97607$ positive training pairs where one pair consists of an annotated scan keypoint and the corresponding CAD model. Additionally, we create $N_{P,\text{aug.}} = 10 \cdot N_{P,\text{ann.}}$, augmented positive keypoint pairs by randomly sampling points on the CAD surface, projecting them to the scan via the ground truth transformation and rejecting if the distance to the surface in the scan $\geq 3cm$. In total we generate $N_P = N_{P,\text{ann.}} + N_{P,\text{aug.}}$ positive training pairs.

Negative pairs are generated in two ways: (1) Randomly choosing a voxel point in the scan and a random CAD model (likelihood of false negative is exceedingly low). (2) Taking an annotated scan keypoint and pairing it with a random CAD model of different class. We generate $N_N = N_P$ negative samples with (1) and $N_{HN} = N_P$ with (2).

Hence, the training set has a positives-to-negatives ratio of 1:2 ($N_P : N_N + N_{HN}$). We found an over-representation of negative pairs gives satisfactory performance on the compatibility prediction.

### 3.5.4 Training Process

We use an SGD optimizer with a batch size of 32 and an initial learning rate of $0.01$, which is decreased by $1/2$ every $50$K iterations. We train for 250K iterations ($\approx 62.5$ hours). The weights are initialized randomly. The losses of the heatmap prediction stream and the scale prediction stream are masked such that only positive samples make up the residuals for back-propagation.

The CAD encoder is pre-trained with an auto-encoder on ShapeNet models with a reconstruction task and a $MSE$ as loss function. All models of ShapeNetCore ($\approx 55K$) are used for pre-training and the input and output dimensions are $32^3$ distance field grids. The network is trained with SGD until convergence ($\approx 50$ epochs).

## 3.6 Alignment Optimization

**Filtering** The input to our alignment optimization is a representative set of Harris keypoints $\mathbb{K} = \{p_j\},\ j = 1 \dots N_0$ from a scene $\mathbb{S}$ and a set of CAD models $\mathbb{M} = \{m_i\}$. The correspondences between $\mathbb{K}$ and $\mathbb{M}$ were established by the correspondence prediction from the previous stage (see section 3.5) where each keypoint $p_j$ is tested against every model $m_i$.

Since not every keypoint $p_j$ semantically matches to every CAD model $m_i$, we reject correspondences based on the compatibility prediction of our network. The threshold for rejecting $p_j$ is determined by the Otsu thresholding scheme [91]. In practice this method turned out to be much more effective than a fixed threshold. After the filtering there are $N \leq N_0$ (usually $N \approx 0.1 N_0$) correspondence pairs to be used for the alignment optimization.

**Variational Optimization** From the remaining $\mathbb{K}_{\text{filter.}} \subset \mathbb{K}$ Harris keypoints, we construct *point-heatmap* pairs $(p_j, H_j)$ for each CAD model $m_i$, with $p_j \in \mathbb{R}^3$ a point in the scan and $H_j : \Omega \to [0, 1]$ a heatmap.

In order to find an optimal pose we construct the following minimization problem:

$$\begin{aligned} \mathrm{c}_{\text{vox}} &= T_{\text{world}\to\text{vox}} \cdot T_{m_i}(a, s) \cdot p_j \\ f &= \min_{a,s} \quad \sum_{j}^{N} (1 - H_j(\mathrm{c}_{\text{vox}}))^2 + \lambda_s \|s\|_2^2 \end{aligned} \tag{3.1}$$

where $\mathrm{c}_{\text{vox}}$ is a voxel coordinate, $T_{\text{world}\to\text{vox}}$ denotes a transformation that maps world points into the voxel grid for look-ups, $a$ denotes the coordinates of the Lie algebra (for rotation and translation), $s$ defines the scale, and $\lambda_s$ defines the scale regularization strength. $a, s$ compose a transformation matrix $T_{m_i} = \psi(a_{m_i}, s_{m_i})$:

$$\begin{aligned} \psi :\ & \mathbb{R}^6 \times \mathbb{R}^3 \to \mathbb{R}^{4\times 4}, \\ & a, s \mapsto \operatorname{expm}\left( \begin{bmatrix} \Gamma(a_{1,2,3}) & a_{4,5,6} \\ 0 & 0 \end{bmatrix} \right) \cdot \begin{bmatrix} s & 0 \\ 0 & 1 \end{bmatrix} \end{aligned}$$

where $\Gamma$ is the hat map, expm is the matrix exponential.

We solve Equation 3.1 using the Levenberg-Marquardt (LM) algorithm. As we can suffer from zero-gradients (especially at bad initialization), we construct a scale-pyramid from the heatmaps which we solve in coarse-to-fine fashion.

In each LM step we optimize over the incremental change and update the parameters as following: $T_{m_i}^{k+1} \leftarrow \phi(a^*, s^*) \cdot T_{m_i}^{k}$ where $a^*, s^*$ are the optimal parameters. As seen in Equation 3.1, we add a regularization on the scale in order to prevent degenerate solutions which can appear for very large scales.

By restarting the optimization with different translation parameters (i.e., varying initializations), we obtain multiple alignments per CAD model $m_i$. We then generate as many CAD model alignments as required for a given scene in the evaluation. Note, in a ground truth scene one unique CAD model $m_i$ can appear in multiple locations e.g., chairs in conference rooms.

**Pruning** Finally, there will be alignments of various CAD models into a scene where a subset will be misaligned. In order to select only the best alignments and prune potential misalignments we use a confidence metric similar to [47]; for more detail, we refer to the appendix.

| base [+variations, ...] | bath | bookshelf | cabinet | chair | display | sofa | table | trash bin | other | class avg. | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| +sym | 46.88 | 44.39 | 40.49 | 64.46 | 26.85 | 56.26 | 47.15 | 38.43 | 24.68 | 43.29 | 48.01 |
| +sym,+scale | 51.35 | 45.46 | 45.24 | 66.94 | 29.88 | 64.78 | 48.30 | 38.00 | 28.65 | 46.51 | 50.85 |
| +sym,+CP | 59.32 | 51.93 | 55.11 | 70.99 | 41.58 | 66.77 | 53.74 | 43.39 | 42.93 | 53.97 | 60.44 |
| +scale,+CP | 45.24 | 45.85 | 47.16 | 61.55 | 27.65 | 51.92 | 41.21 | 31.13 | 29.62 | 42.37 | 47.64 |
| +sym,+scale,+CP | 56.05 | 51.28 | **57.45** | 72.64 | 36.36 | 70.63 | 52.28 | 46.80 | 43.32 | 54.09 | 60.43 |
| +sym,+scale,+CP,+PT (3/3 fix) | 57.03 | 50.63 | 56.76 | 70.39 | 39.74 | 65.00 | 52.03 | **46.87** | 41.83 | 53.36 | 58.61 |
| +sym,+scale,+CP,+PT (1/3 fix) | **60.08** | **58.62** | 56.35 | **73.92** | **44.19** | **75.08** | **56.80** | 45.78 | **46.53** | **57.48** | **63.94** |

**Table 3.1:** Correspondence prediction F1-scores in % for variations of our correspondence prediction network. We evaluate the effect of symmetry (sym), predicting scale (scale), predicting compatibility (CP), encoder pre-training (PT), and pre-training with parts of the encoder fixed (#fix), see section 3.5 for more detail regarding our network design and training scheme.

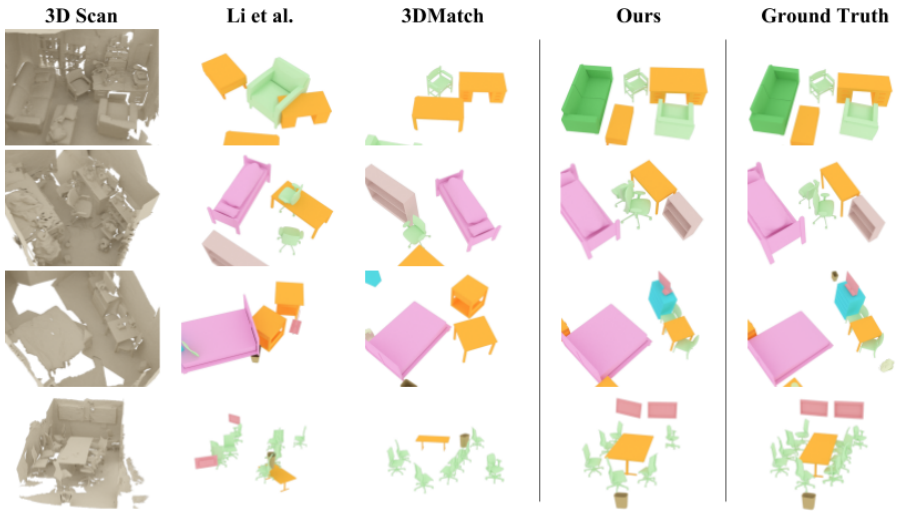



**Figure 3.5:** Qualitative comparison of alignments on four different test ScanNet [5] scenes. Our approach to learning geometric features between real and synthetic data produce much more reliable keypoint correspondences, which coupled with our alignment optimization, produces significantly more accurate alignments.

## 3.7 Results

### 3.7.1 Correspondence Prediction

To quantify the performance of correspondence heatmap predictions, we evaluate the voxel-wise F1-score for a prediction and its Gaussian-blurred target. The task is challenging and by design $\frac{2}{3}$ test samples are false correspondences, $\approx 99\%$ of the target voxels are 0-valued, and only a

single 1-valued voxel out of $32^3$ voxels exists. The F1-score will increase only by identifying true correspondences. As seen in Table 3.1, our best 3D CNN achieves $63.94\%$.

Table 3.1 additionally addressed our design choices; in particular, we evaluate the effect of using pre-training (PT), using compatibility (CP) as a proxy loss (defined in subsection 3.5.2), enabling symmetry awareness (sym), and predicting scale (scale). Here, a pre-trained network reduces overfitting, enhancing generalization capability. Optimizing for compatibility strongly improves heatmap prediction as it efficiently detects false correspondences. While predicting scale only slightly influences the heatmap predictions, it becomes very effective for the later alignment stage. Additionally, incorporating symmetry enables significant improvement by explicitly disambiguating symmetric keypoint matches.

### 3.7.2 Alignment

In the following, we compare our approach to other handcrafted feature descriptors: FPFH [37], SHOT [38], Li et al. [47] and a learned feature descriptor: 3DMatch [43] (trained on our Scan2CAD dataset). We combine these descriptors with a RANSAC outlier rejection method to obtain pose estimations for an input set of CAD models. A detailed description of the baselines can be found in the appendix. As seen in Table 3.2, our best method achieves $31.68\%$ and outperforms all other methods by a significant margin. We additionally show qualitative results in Figure 3.5. Compared to state-of-the-art handcrafted feature descriptors, our learned approach powered by our Scan2CAD dataset produces considerably more reliable correspondences and CAD model alignments. Even compared to the learned descriptor approach of 3DMatch, our explicit learning across the synthetic and real domains coupled with our alignment optimization produces notably improved CAD model alignment.

Figure 3.6 shows the capability of our method to align in an unconstrained real-world setting where ground truth CAD models are not given, we instead provide a set of 400 random CAD models from ShapeNet [4].

| | bath | bookshelf | cabinet | chair | display | sofa | table | trash bin | other | class avg. | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| FPFH (Rusu et al. [37]) | 0.00 | 1.92 | 0.00 | 10.00 | 0.00 | 5.41 | 2.04 | 1.75 | 2.00 | 2.57 | 4.45 |
| SHOT (Tombari et al. [38]) | 0.00 | 1.43 | 1.16 | 7.08 | 0.59 | 3.57 | 1.47 | 0.44 | 0.75 | 1.83 | 3.14 |
| Li et al. [47] | 0.85 | 0.95 | 1.17 | 14.08 | 0.59 | 6.25 | 2.95 | 1.32 | 1.50 | 3.30 | 6.03 |
| 3DMatch (Zeng et al. [43]) | 0.00 | 5.67 | 2.86 | 21.25 | 2.41 | 10.91 | 6.98 | 3.62 | 4.65 | 6.48 | 10.29 |
| Ours: +sym | 24.30 | 10.61 | 5.97 | 9.49 | 3.90 | 25.26 | 12.34 | 10.74 | 3.58 | 11.80 | 8.772 |
| Ours: +sym,+scale | 18.99 | 13.61 | 7.24 | 14.73 | 9.76 | 41.05 | 14.04 | 5.26 | 6.29 | 14.55 | 11.48 |
| Ours: +sym,+CP | 35.90 | 32.35 | 28.64 | 40.48 | 18.85 | 60.00 | 33.11 | 28.42 | 16.89 | 32.74 | 29.42 |
| Ours: +scale,+CP | 34.18 | 31.76 | 21.82 | 37.02 | 14.75 | 50.53 | 32.31 | **31.05** | 11.59 | 29.45 | 26.75 |
| Ours: +sym,+scale,+CP | 36.20 | **36.40** | **34.00** | **44.26** | 17.89 | **70.63** | 30.66 | 30.11 | 20.60 | **35.64** | **31.68** |
| Ours: +sym,+scale,+CP,+PT (3/3 fix) | **37.97** | 30.15 | 28.64 | 41.55 | 19.51 | 57.89 | 33.85 | 20.00 | 17.22 | 31.86 | 29.27 |
| Ours: +sym,+scale,+CP,+PT (1/3 fix) | 34.81 | **36.40** | 29.00 | 40.60 | **23.25** | 66.00 | **37.64** | 24.32 | **22.81** | 34.98 | 31.22 |

**Table 3.2:** Accuracy comparison ($\%$) on our CAD alignment benchmark. While handcrafted feature descriptors can achieve some alignment on more featureful objects (e.g., chairs, sofas), they do not tolerate well the geometric discrepancies between scan and CAD data – which remains difficult for the learned keypoint descriptors of 3DMatch. Scan2CAD directly addresses this problem of learning features that generalize across these domains, thus significantly outperforming state of the art.

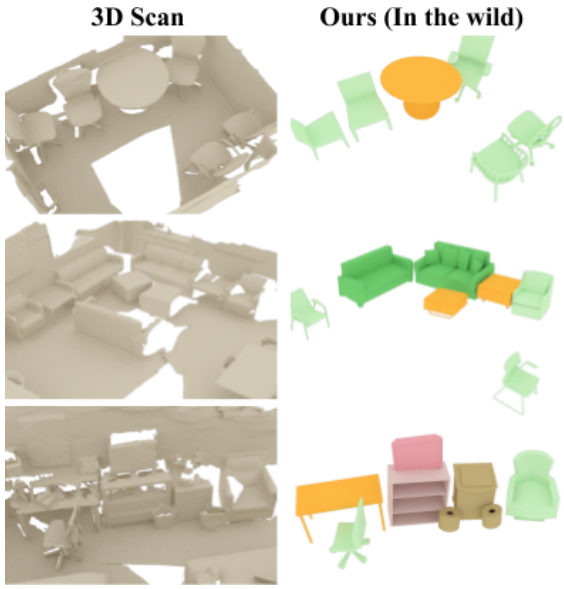



**Figure 3.6:** Unconstrained scenario where instead of having a ground truth set of CAD models given, we use a set of 400 randomly selected CAD models from ShapeNetCore [4], more closely mimicking a real-world application scenario.

## 3.8 Limitations

While the focus of this work is mainly on the alignment between 3D scans and CAD models, we only provide a basic algorithmic component for retrieval (finding the most similar model). This necessitates an exhaustive search over a set of CAD models. We believe that one of the immediate next steps in this regard would be designing a neural network architecture that is specifically trained on shape similarity between scan and CAD geometry to introduce more efficient CAD model retrieval. Additionally, we currently only consider geometric information, and it would also be intresting to introduce learned color features into the correspondence prediction, as RGB data is typically higher-resolution than depth or geometry, and could potentially improve alignment results.

## 3.9 Conclusion

In this work, we presented Scan2CAD, which aligns a set of CAD models to 3D scans by predicting correspondences in form of heatmaps and then optimizes over these correspondence predictions. First, we introduce a new dataset of 9DoF CAD-to-scan alignments with 97607 pairwise keypoint annotations defining the alignment of 14225 objects. Based on this new dataset, we design a 3D CNN to predict correspondence heatmaps between a CAD model and a 3D scan. From these predicted heatmaps, we formulate a variational cost minimization that then finds the optimal 9DoF pose alignments between CAD models and the scan, enabling effective transformation of noisy, incomplete RGB-D scans into a clean, complete CAD model

representation. This enables us to achieve significantly more accurate results than state-of-the-art approaches, and we hope that our dataset and benchmark will inspire future work towards bringing RGB-D scans to CAD or artist-modeled quality.

## 3.10 Appendix

In this appendix, we detail statistics regarding the Scan2CAD dataset in Sec. 3.10.1. In Sec. 3.10.2, we detail our evaluation metric for the alignment models. We show additional details for our keypoint correspondence prediction network in Sec. 3.10.3 and we show example correspondence predictions. We provide additional detail for our alignment algorithm in Sec 3.10.5. In Sec. 3.10.7, we describe the implementation details of the baseline approaches.

### 3.10.1 Dataset

A compilation of our dataset is presented in Figure 3.15. As a full coverage was aimed during the annotation, we can see the variety and richness of the aligned objects.

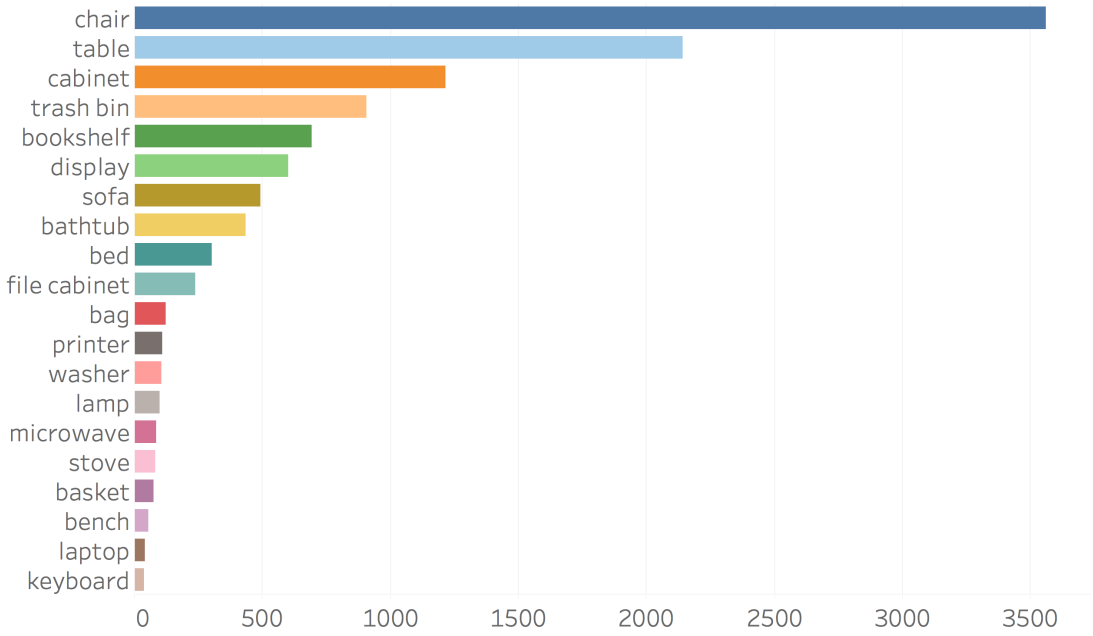



**Figure 3.7:** Distribution of top 20 categories of annotated objects in our Scan2CAD dataset.

**Statistics** We show the object category statistics of our dataset in Figure 3.7. Since our dataset is constructed on scans of indoor environments, it contains many furniture categories (e.g., chairs, tables, and sofas). In addition, it also provides alignments for a wide range of other objects such as backpacks, keyboards, and monitors.

**Timings** The annotation timings per object and per scan are illustrated in Figure 3.8 (top) and Figure 3.8 (bottom). On an object level, the timings are relatively consistent with little variance in time. On a scan level, however, the variation in annotation time is larger which is due to variation in scene size. Larger scenes are likely to contain more objects and hence require longer annotation times.

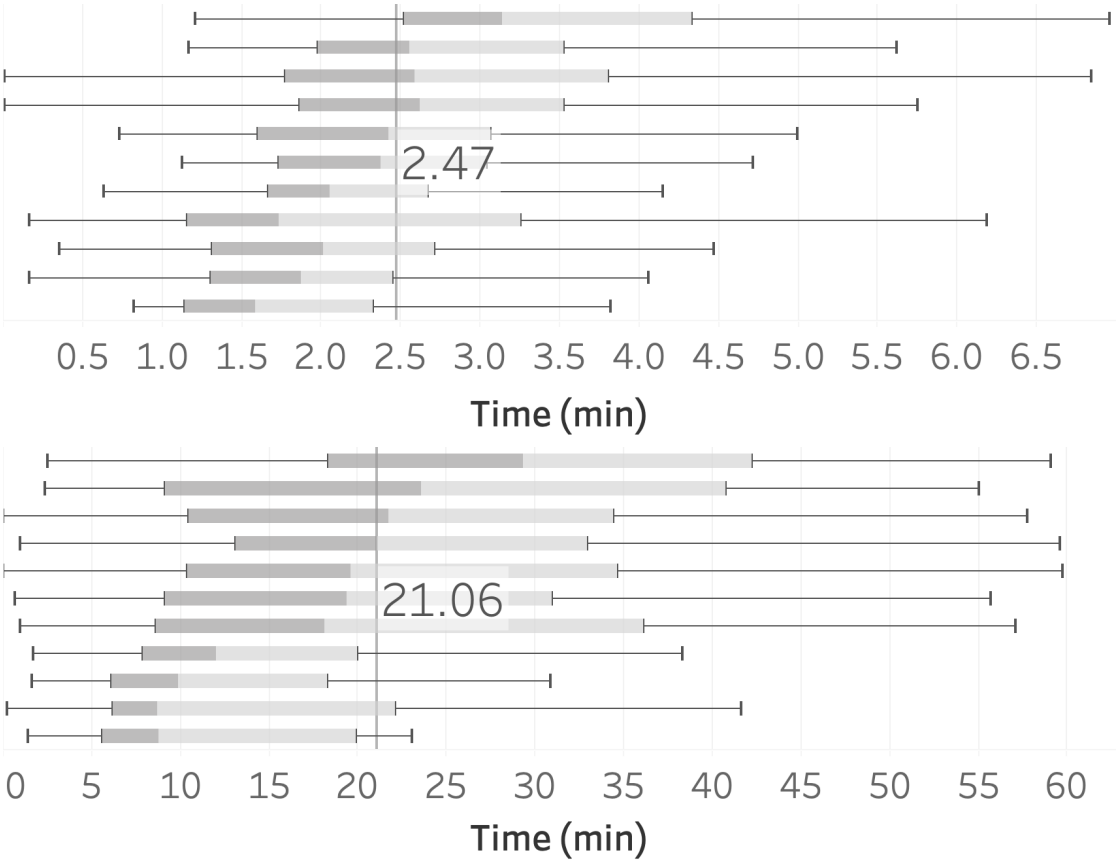



**Figure 3.8:** Annotation timing distributions for each annotated object (top) and for each annotated scene (bottom). Each row shows a box-whisker plot with the median time and interquartile range for an annotator. The vertical rule shows the overall median across annotators.


**Figure 3.9:** Examples of symmetry annotations.

**Symmetries** In order to take into account the natural symmetries of many object categories during our training and evaluation, we collected a set of symmetry type annotations for all instances of CAD models. Figure 3.9 shows examples and total counts for all rotational symmetry annotations.

### 3.10.2 Evaluation Metric

In this subsection, we describe the details of the algorithm for computing the alignment accuracy. To compute the accuracy, we do a greedy matching of aligned CAD models to the ground truth CAD models.

For a given aligned scene **id-scan** with $N$ aligned CAD models, we query the ground truth alignment for the given scene. The evaluation script then iterates through all aligned candidate models and checks whether there is a ground truth CAD model of the same class where the alignment error is below the given bounds; if one is found, then the counter (of positive alignments)

**Data:** 1 **id-scan**, $N$ CADs (**id**, **cat**, **pose**)
**Result:** accuracy in $\%$
**Init:**
Get $N$ GT-CADs from database with *key*=**id-scan**
Set thresholds $t_t = 20cm, t_r = 20°, t_s = 20\%$
counter = 0;
**for** *c* **in** *CADs* **do**
  **id**, **cat**, **pose** = c
  **for** *c-gt* **in** *GT-CADs* **do**
    $\mathbf{id}_{\text{GT}}$, $\mathbf{cat}_{\text{GT}}$, $\mathbf{pose}_{\text{GT}}$ = c-gt
    **if** ***cat*** == $\boldsymbol{cat}_{GT}$ **then**
      $\epsilon_t$ = `Distance` (**pose**.t, $\mathbf{pose}_{\text{GT}}$.t)
      $\epsilon_r$ = `Distance` (**pose**.r, $\mathbf{pose}_{\text{GT}}$.r, $\mathbf{sym}_{\text{GT}}$)
      $\epsilon_s$ = `Distance` (**pose**.s, $\mathbf{pose}_{\text{GT}}$.s)
      **if** $\epsilon_t \leq t_t$ *and* $\epsilon_r \leq t_r$ *and* $\epsilon_s \leq t_s$ **then**
        counter ++
        remove $\mathbf{id}_{\text{GT}}$ from GT-CADs
        break
      **end**
    **end**
  **end**
**end**
**Output: accuracy** = counter/N

**Algorithm 1:** Pseudo code of our evaluation benchmark. **id, cat, pose** denotes the id, category label and 9DoF alignment transformation for a particular CAD model. Note that the rotation distance function takes symmetries into account.

is incremented and the respective ground truth CAD model is removed from the ground truth pool. See algorithm 1 for the pseudo-code.

### 3.10.3 Correspondence Prediction Network

**Network details** The details of the building blocks for our correspondence prediction network are depicted in Figure 3.10. See Figure 4 of the main paper for the full architecture. We introduce the following blocks:

- **ConvBlocks** are the most atomic blocks and consist of a sequence of **Conv3-BatchNorm-ReLU** layers as commonly found in other literature.
- **ResBlocks** are essentially residual skip connecting layers.
- **BigBlocks** contain two **ResBlocks** in succession.

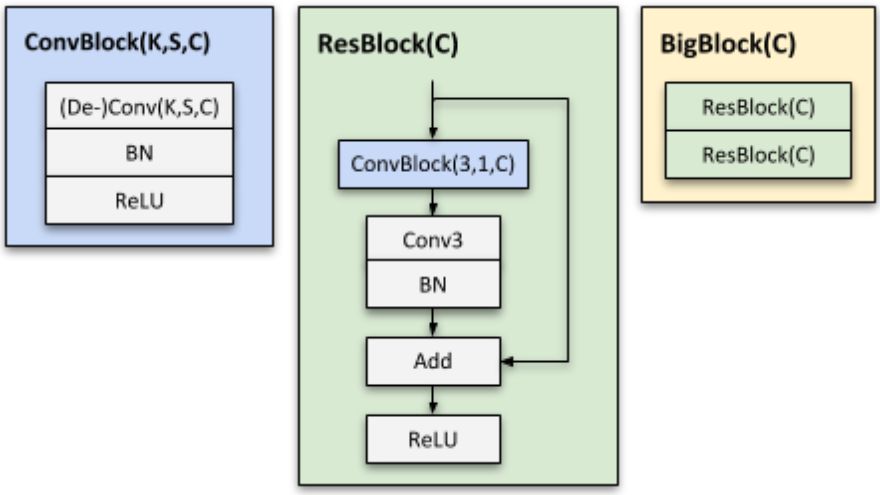



**Figure 3.10:** CNN building blocks for our Scan2CAD architecture. **K, S, C** stand for *kernel-size*, *stride* and *num-channels* respectively.

**Training curves** Figure 3.11 shows how much data is required for training the alignment approach. The curves show predicted compatibility scores of our network. We train our 3D CNN approach with different numbers of training samples (full, half and quarter of the dataset), and show both training and validation curves for each of the three experiments. When using only a quarter or half of the dataset, we see severe overfitting. This implies that our entire dataset provides significantly better generalization.

In Figure 3.12, we show the Precision-recall curve of the compatibility prediction of a our ablations (see Sec. 7.1 in the main paper). The PR-curves underline the strength of our best preforming network variation.

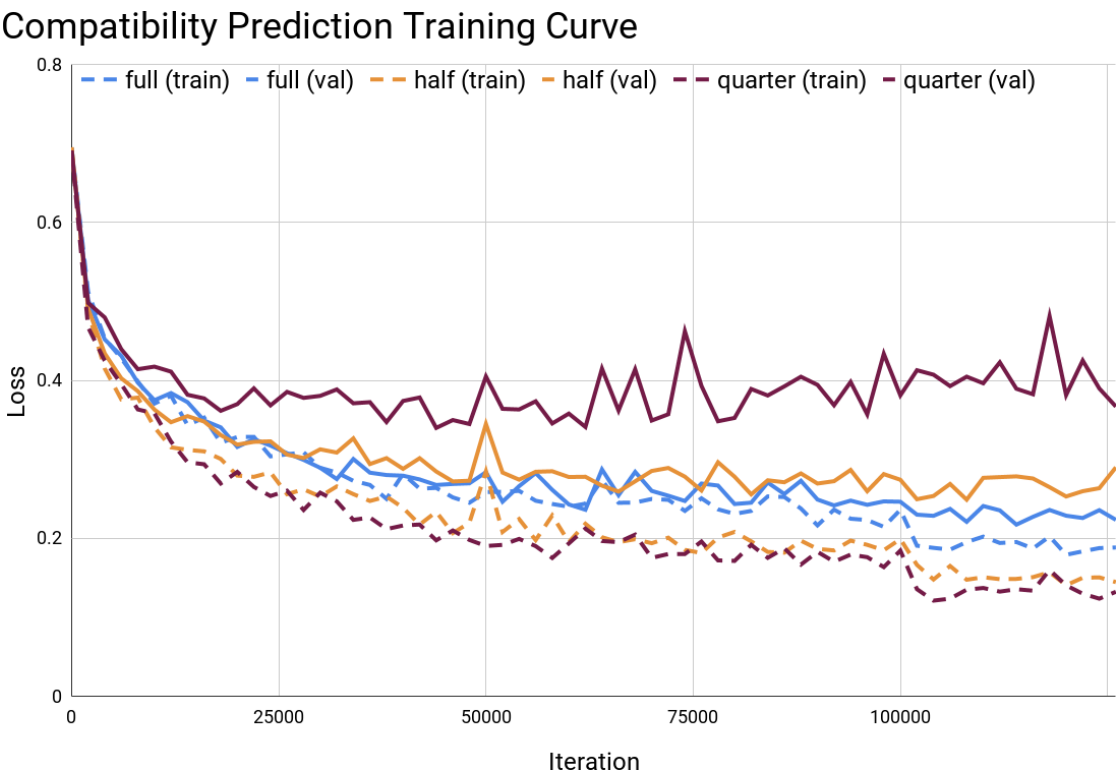



**Figure 3.11:** Training and validation curves for varying training data sizes showing the probability score predictions. Experiments are carried out with full, half, and a quarter of the data set size. We see severe overfitting for half and quarter dataset training experiments, while our full training corpus mitigates overfitting.

**Correspondence predictions** Visual results of the correspondence prediction are shown in Figure 3.14. One can see that our correspondence prediction network predicts as well symmetry-equivalent correspondences. The scan input with a voxel resolution of 3cm and a grid dimension of 64 can cover 1.92m per dimension. A larger receptive field is needed for large objects in order infer correspondences from a more global semantic context (see left-hand side first and second row.).

### 3.10.4 Alignment Error Analysis

Our alignment results have different sensibility for each parameter block (translation, rotation, scale). In order to gauge the stringency of each parameter block we varied the threshold for one parameter block and held the other two constant at the default value (see Figure 3.13). We observe that for the default thresholds $\epsilon_t = 0.2\text{m}, \epsilon_r = 20°, \epsilon_s = 20\%$ all thresholds

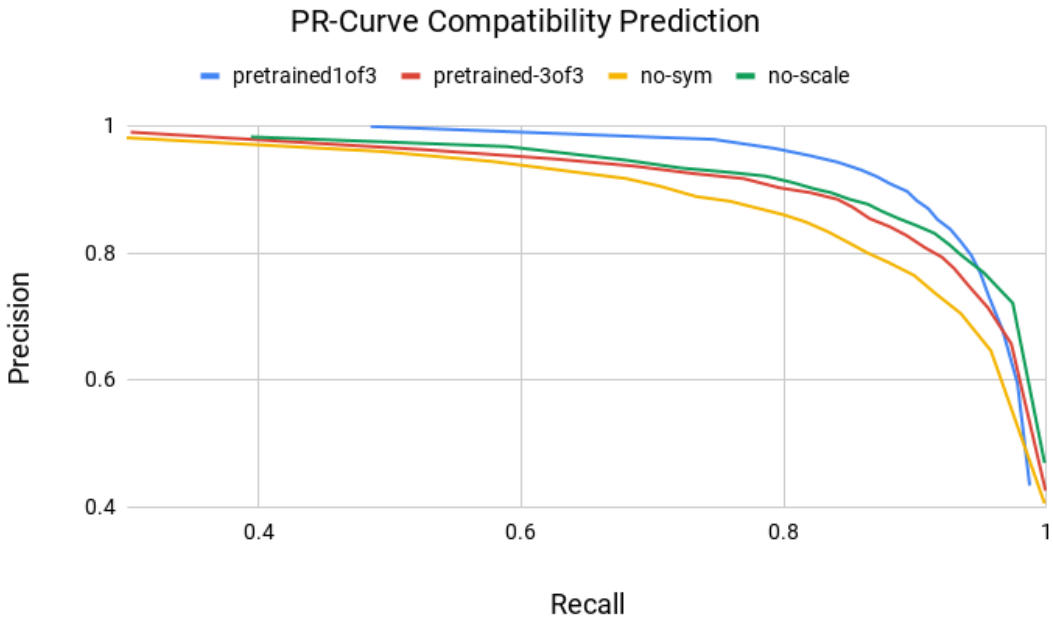



**Figure 3.12:** Precision-recall curve of our compatibility score predictions.

### 3.10.5 Alignment Algorithm Details

In order to remove misaligned objects, we prune objects after the alignment optimization based on the known free space of the given input scan. This is particularly important for the unconstrained ('in-the-wild') scenario where the set of ground truth CAD models to be aligned is not given as part of the input. For a given candidate transformation $T_m$ (as described in Sec. 6 in the main paper), we compute:

$$
\begin{aligned}
c &= \frac{\sum_x^{\Omega_{\text{CAD}}^{\text{occupied}}} \mathcal{O}_{\text{scan}}^{\text{seen}}(T_{\text{world}\to\text{vox,scan}} \cdot T_m^{-1} \cdot T_{\text{vox}\to\text{world,CAD}} \cdot x)^2}{|\Omega_{\text{CAD}}^{\text{occupied}}|} \\
\Omega_{\text{CAD}}^{\text{occupied}} &= \{x \in \Omega_{\text{CAD}} \mid \mathcal{O}_{\text{CAD}}(x) < 1\} \\
\Omega_{\text{scan}}^{\text{seen}} &= \{x \in \Omega_{\text{scan}} \mid \mathcal{O}_{\text{scan}}(x) > -\tau\} \\
\mathcal{O}_{\text{scan}}^{\text{seen}}(x) &= \mathcal{O}_{\text{scan}}(x) \text{ if } x \in \Omega_{\text{scan}}^{\text{seen}} \text{ else } 0
\end{aligned}
$$

where $T_m^{-1}$ defines the transformation from CAD to scan, $\Omega$ defines a voxel grid space ($\subset \mathbb{N}^3$), $\tau$ is the truncation distance used in volumetric fusion (we use $\tau = 15$cm), and $\mathcal{O}$ are look-ups into the signed distance function or distance functions for the scan or CAD model. We also require that at least $30\%$ of the CAD surface voxels $\Omega_{\text{CAD}}^{\text{occupied}}$ project into seen space of the scan voxel grid $\Omega_{\text{scan}}^{\text{seen}}$. Finally, we rank all alignments (of various models) per scene w.r.t. their confidence and prune all lower ranked models that are closer than $0.3$m to a higher ranked model.

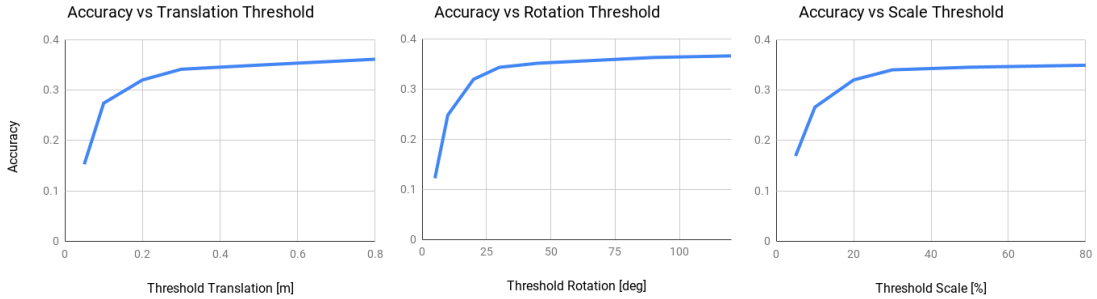



**Figure 3.13:** Accuracy vs. varying thresholds for translation (left), rotation (middle) and scale (right). Only one threshold is varied whereas the remaining ones were held constant at their default value either $\epsilon_t = 0.2\text{m}, \epsilon_r = 20°, \epsilon_s = 20\%$.

### 3.10.6 Alignment Optimization Analysis: Comparison to RANSAC

In Table 3.3, we additionally demonstrate the efficacy of our new alignment approach compared to alignment by RANSAC (using our predicted heatmap correspondences). Our alignment via heatmap optimization is more robust to outliers while also incorporating symmetries, resulting in significantly improved performance.

| Method | avg. acc. in % |
|---|---|
| Our Heatmap CNN + RANSAC | 18.27 |
| Our Heatmap CNN + Heatmap optim. | 31.68 |

**Table 3.3:** Our heatmap optimization for alignment in comparison to RANSAC. The input correspondences for RANSAC are provided by the maximum response of the predicted heatmap.

### 3.10.7 Baseline Method Details

In the following, we provide additional details for the used baseline approaches. FPFH and SHOT work on point clouds and compute geometric properties between points within a support region around a keypoint. We use the implementation provided in the Point Cloud Library [92].

The method presented by Li et al. [47] takes the free space around a keypoint into account to compute a descriptor distance between a keypoint in scan and another keypoint in a CAD object. Here, we use the original implementation from the authors and modified it such that it works within a consistent evaluation framework together with the other methods. However, since we are not restricted to real-time constraints, we neglect the computation of the geometric primitives around the keypoints, which helps to find good initial rotation estimations. Instead, we computed all 36 rotation variants to find the smallest distance. We also replace the original 1-point RANSAC with another RANSAC as described below.

3DMatch [43] takes as input a 3D volumetric patch from a TDF around a keypoint and computes via a series of 3D convolutions and max-poolings a 512 dimensional feature vector. In order to train 3DMatch, we assemble a correspondence dataset as described in Sec. 5.3 in

the main paper. We train the network for 25 epochs using the original contrastive loss with a margin of 1. During test time, we extract the 3D patch around a detected Harris keypoint of both CAD object and scan and separately compute their feature vector. In addition to the evaluation in the main paper, for 3DMatch, we additionally show the performance of 3DMatch when trained only on real only (scan-scan correspondences from ScanNet), as shown in Table 3.4. This suffers dramatically in matching the different characteristics of scan-CAD at test time. Our approach to predict scan-CAD heatmap correspondences results in significantly higher alignment accuracy compared to both 3DMatch trained on scan-CAD as well as scan-scan.

For each method, we compute the feature descriptors for all keypoints in the scan and the CAD objects, respectively. We then find correspondences between pairs of keypoints if their height difference is less than $0.8m$ and if the L2 distance between the descriptors is below a certain threshold. Due to potential re-occurring structures in scan and CAD we select the top-8 correspondences with the smallest descriptor distances for each keypoint in the scan.

After establishing potential correspondences between the scan and a CAD object, we use a RANSAC outlier rejection method to filter out wrong correspondences and find a suitable transformation to align the CAD object within the scene. During each RANSAC iteration, we estimate the translation parameters and the up-right rotation by selecting 3 random correspondences. If the transformation estimate gives a higher number of inliers than previous estimates, we keep this transformation. The threshold of the Euclidean distance for which a correspondence is considered as an inlier is set to $0.20m$. We use a fixed scale determined by the class average scale from our Scan2CAD train set. For a given registration for a specific CAD model, we mark off all keypoints in the scan which were considered as inliers as well as all scan keypoints which are located inside the bounding box of the aligned CAD model. These marked keypoints will be ignored for the registration of later CAD models.

To find optimal parameter for FPFH, SHOT, and Li et al., we construct an additional correspondence benchmark and ran a hyperparameter search based on the validation set.

| Method | avg. acc. in % |
|---|---|
| 3D Match + ScanNet (only real data) | 0.26 |
| 3D Match + our dataset | 10.29 |
| Our method + our dataset | 31.68 |

**Table 3.4:** Comparison to 3DMatch trained with only real data, trained on our data, and our result; evaluation on our test set.

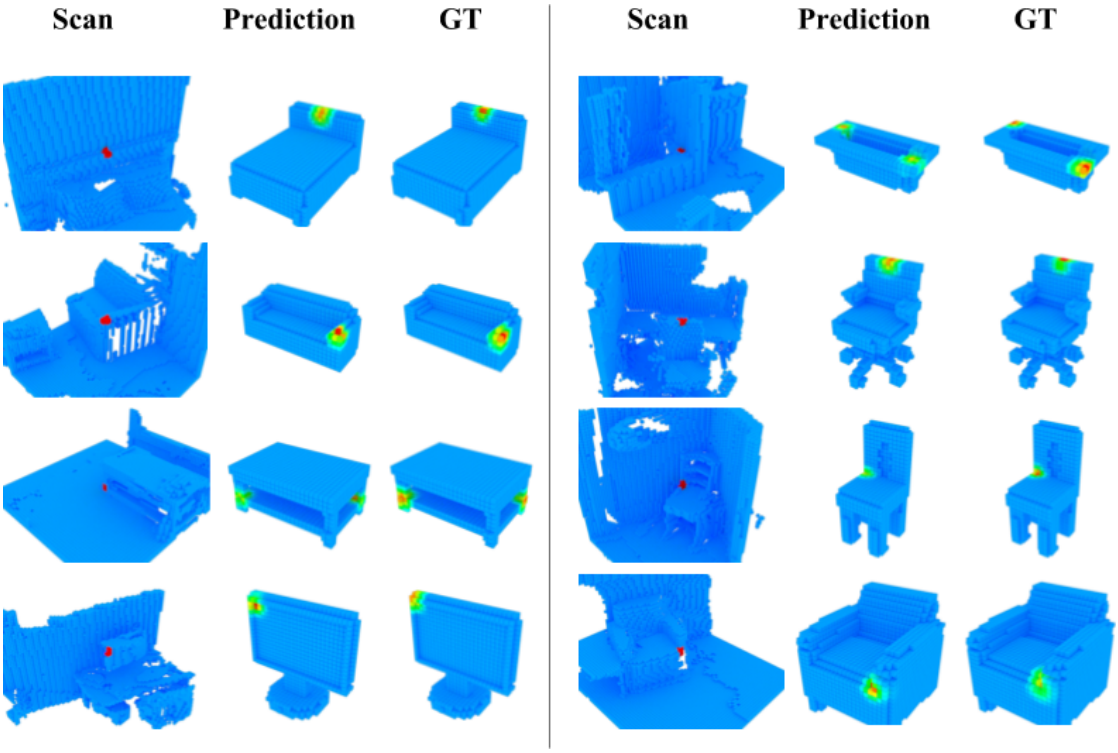



**Figure 3.14:** Sample correspondence predictions over a range of various CAD models. Heatmaps contain symmetry-equivalent correspondences.

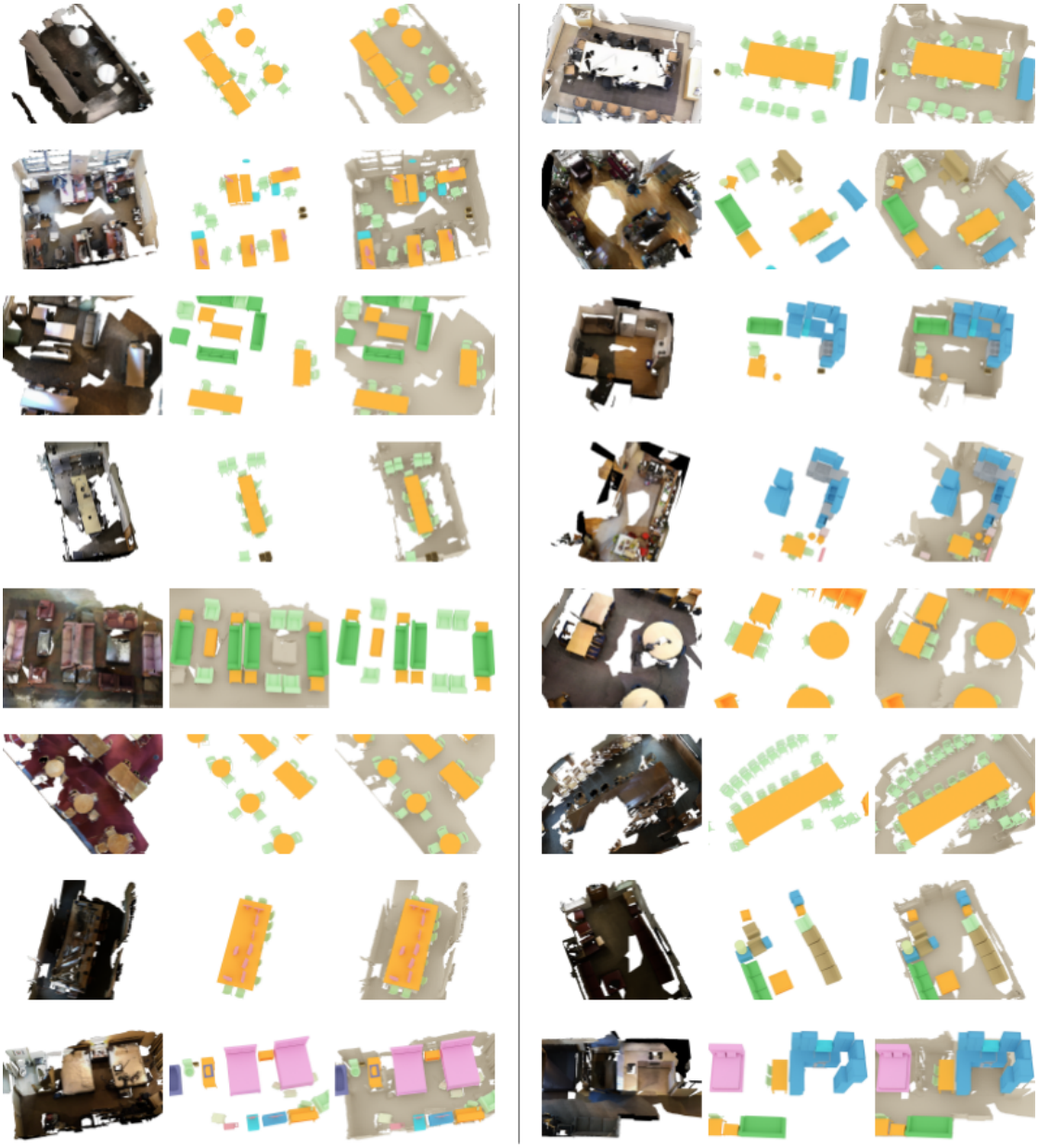


**Figure 3.15:** Samples of annotated scenes. Left: 3D scan. Center: annotated CAD model arrangement; right: overlay CAD models onto scan.

# 4 End-to-End CAD Model Retrieval and 9-DoF Alignment Using Dense Object Correspondences

This chapter introduces the following paper:
**Avetisyan, A.**, Dai, A., & Nießner, M. (2019). End-to-end cad model retrieval and 9dof alignment in 3d scans. *In Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 2551-2560).

**Summary** In this publication we propose a novel, end-to-end approach to align CAD models to a 3D scan of a scene, enabling the transformation of a noisy, incomplete 3D scan to a compact, CAD reconstruction with clean, complete object geometry. Our main contribution lies in formulating a differentiable Procrustes alignment that is paired with a symmetry-aware dense object correspondence (SOCs) prediction. To simultaneously align CAD models to all the objects of a scanned scene, our approach detects object locations, then predicts symmetry-aware dense object correspondences between scan and CAD geometry in a unified object space, as well as a nearest neighbor CAD model candidate. The dense correspondences are used in a differentiable Procrustes alignment step to predict a final pose. Objects in the scan are detected by predicting a voxel-wise objectness probability over the volumetric grid of the whole scene where discrete object locations are extracted by applying NMS on the heatmap. For each potential object location, the 3D bounding box extent is predicted which is used to crop out a feature volume from the backbone of the network. We utilize this feature volume to predict symmetric aware dense object correspondences between the geometry inside the bounding box and a retrieved CAD model candidate. For retrieval, shape descriptors are created by using the feature vectors of the bottleneck layer of an auto-encoder trained with a reconstruction loss on all available CAD models. Lastly, an optimal pose can be calculated by using the Procrustes alignment method with the predicted dense correspondences. Our approach operates in a fully-convolutional fashion, enabling the alignment of CAD models to the objects of a scan in a single forward pass. This enables our method to outperform state-of-the-art approaches by $19.04\%$ for CAD model alignment to scans, with $\approx 250\times$ faster runtime than previous data-driven approaches.

**Contributions** The first author developed and implemented the main method, including the heatmap-based object detection component, the calculation of shape descriptors using an auto-encoder, and the Procrustes alignment method with PyTorch routines. The final paper was the result of discussions with the co-authors.

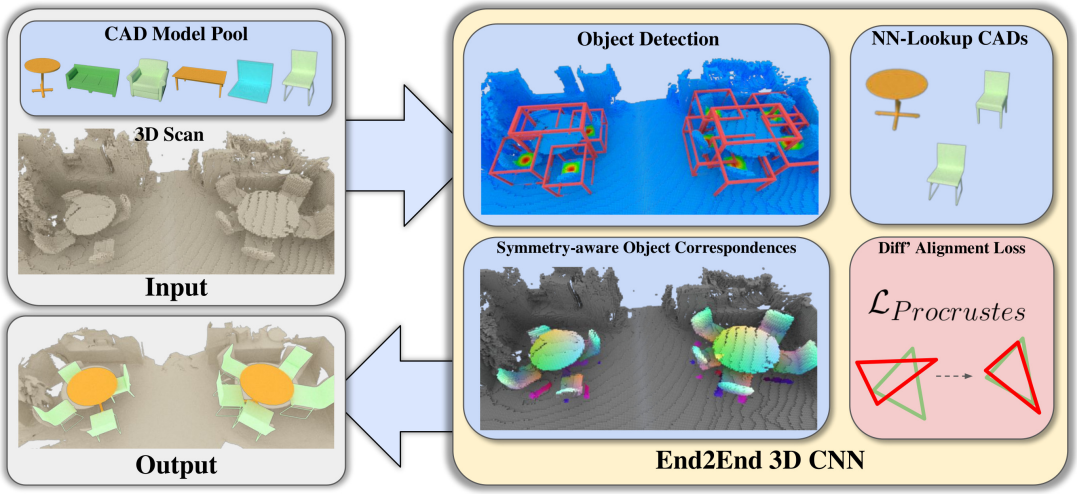



**Figure 4.1:** From a 3D scan and a set of CAD models, our method learns to predict 9DoF CAD model alignments to the objects of the scan in a fully-convolutional, end-to-end fashion. Our proposed 3D CNN first detects objects in the scan, then uses the regressed object bounding boxes to establish symmetry-aware object correspondences between a scan object and CAD model, which inform our differentiable Procrustes alignment loss, enabling learning of alignment-informed correspondences and producing CAD model alignment to a scan in a single forward pass.

## 4.1 Introduction

In recent years, RGB-D scanning and reconstruction has seen significant advances, driven by the increasing availability of commodity range sensors such as the Microsoft Kinect, Intel RealSense, or Google Tango. State-of-the-art 3D reconstruction approaches can now achieve impressive capture and reconstruction of real-world environments [70, 15, 16, 20, 20, 72, 22], spurring forth many potential applications of this digitization, such as content creation, or augmented or virtual reality.

Such advances in 3D scan reconstruction have nonetheless remained limited towards these use scenarios, due to geometric incompleteness, noise and oversmoothing, and lack of fine-scale sharp detail. In particular, there is a notable contrast in such reconstructed scan geometry in comparison to the clean, sharp 3D models created by artists for visual and graphics applications.

With the increasing availability of synthetic CAD models [4], we have the opportunity to reconstruct a 3D scan through CAD model shape primitives; that is, finding and aligning similar CAD models from a database to each object in a scan. Such a scan-to-CAD transformation enables construction of a clean, compact representation of a scene, more akin to artist-created 3D models to be consumed by mixed reality or design applications. Here, a key challenge lies in finding and aligning similar CAD models to scanned objects, due to strong low-level differences between CAD model geometry (clean, complete) and scan geometry (noisy, incomplete). Current approaches towards this problem thus often operate in a sparse correspondence-based fashion [47, 1] in order to establish reasonable robustness under such differences.

Unfortunately, such approaches, in order to find and align CAD models to an input scan, thus involve several independent steps of correspondence finding, correspondence matching, and finally an optimization over potential matching correspondences for each candidate CAD model. With such decoupled steps, there is a lack of feedback through the pipeline; e.g., correspondences can be learned, but they are not informed by the final alignment task. In contrast, we propose to predict symmetry-aware dense object correspondences between scan and CADs in a global fashion. For an input scan, we leverage a fully-convolutional 3D neural network to first detect object locations, and then from each object location predict a uniform set of dense object correspondences and object symmetry are predicted, along with a nearest neighbor CAD model; from these, we introduce a differentiable Procrustes alignment, producing a final set of CAD models and 9DoF alignments to the scan in an end-to-end fashion. Our approach outperforms state-of-the-art methods for CAD model alignment by $19.04\%$ for real-world 3D scans.

Our approach is the first, to the best of our knowledge, to present an end-to-end scan-to-CAD alignment, constructing a CAD model reconstruction of a scene in a single forward pass. In summary, we propose an end-to-end approach for scan-to-CAD alignment featuring:

- a novel differentiable Procrustes alignment loss, enabling end-to-end CAD model alignment to a 3D scan,
- symmetry-aware dense object correspondence prediction, enabling robust alignment even under various object symmetries, and
- CAD model alignment for a scan of a scene in a single forward pass, enabling very efficient runtime ($<$ 3s on real-world scan evaluation)

## 4.2 Related work

**RGB-D Scanning and Reconstruction** 3D scanning methods have a long research history across several communities, ranging from offline to real-time techniques. In particular, RGB-D scanning has become increasingly popular, due to the increasing availability of commodity range sensors. A very popular reconstruction technique is the volumetric fusion approach by Curless and Levoy [13], which has been materialized in many real-time reconstruction frameworks such as KinectFusion [70, 15], Voxel Hashing [16] or BundleFusion [22], as well as in the context of state-of-the-art offline reconstruction methods [72]. An alternative to these voxel-based scene representations is based on surfels [19], which has been used by ElasticFusion [20] to realize loop closure updates. These works have led to RGB-D scanning methods that feature robust, global tracking and can capture very large 3D environments. However, though these methods can achieve stunning results in RGB-D capture and tracking, the quality of reconstructed 3D geometry nonetheless remains far from from artist-created 3D content, as the reconstructed scans are partial, and contain noise or oversmoothing from sensor quality or small camera tracking errors.

**3D Features for Shape Alignment and Retrieval** An alternative to bottom-up 3D reconstruction from RGB-D scanning techniques is to find high-quality CAD models that can

replace the noisy and incomplete geometry from a 3D scan. Finding and aligning these CAD models inevitably requires 3D feature descriptors to find geometric matches between the scan and the CAD models. Traditionally, these descriptors were hand-crafted, and often based on a computation of histograms (e.g., point normals), such as FPFH [37], SHOT [93], or point-pair features [40].

More recently, with advances in deep neural networks, these descriptors can be learned, for instance based on an implicit signed distance field representation [43, 94, 95]. A typical pipeline for CAD-to-scan alignments builds on these descriptors; i.e., the first step is to find 3D feature matches and then use a variant of RANSAC or PnP to compute 6DoF or 9Dof CAD alignments. This two-step strategy has been used by Slam++ [42], Li et al. [47], Shao et al. [46], the data-driven work by Nan et al. [45] and the recent Scan2CAD approach [1]. One potential approach to combine correspondence prediction and alignment is through differentiable RANSAC [96], which has been applied for camera localization. Our approach is designed to learn robust dense correspondences through a differentiable Procrustes alignment where correspondences and their relative weights are jointly optimized together without requiring multiple hypothesis generation. Other approaches rely only on single RGB(-D) frame input, but use a similar two-step alignment strategy [80, 81, 55, 82, 54, 83]. While these methods are related, their focus is different as we address geometric alignment independent of RGB information.

While promising results have been achieved by these two-step approaches, there remains a fundamental limitation in the decoupled nature of feature matching and alignment computation. This inherently limits the ability of data-driven descriptors, as they remain unaware of the used optimization algorithm.

In our work, we propose an end-to-end alignment algorithm where correspondences are trained through gradients from an differentiable Procrustes optimizer.

**Shape Retrieval Challenges and RGB-D Datasets** In the context of 2D object alignment methods several datasets provide alignment annotations between RGB images and CAD models, including the PASCAL 3D+ [86], ObjectNet3D [87], the IKEA objects [80], and Pix3D [55]; however, no geometric information is given in the query images. SHREC provides a very popular series of 3D shape retrieval challenges, organized as part of Eurographics 3DOR [50, 84]; the tasks include matching objects from ScanNet [5] and SceneNN [85] to ShapeNet models [4].

More recently, Scan2CAD [1] provides accurate CAD alignment annotations on top of ScanNet [5] using ShapeNet models [4], based on roughly 100k manually annotated correspondences. In addition to evaluating our method on the Scan2CAD test dataset, we also evaluate on the synthetic SUNCG [64] dataset.

## 4.3 Overview

For an input 3D scan along with a set of candidate CAD models, our method aims to align similar CAD models to each object instance in the scan. Object locations in the scan are detected, and for each detected object, a similar CAD model is retrieved and a 9DoF transformation (3 degrees each for translation, rotation, and scale) computed to align it to the scan geometry. Thus we can

transform a noisy, incomplete 3D scan into a compact, CAD-based representation with clean, complete geometry, as shown in Figure Figure 4.1.

To this end, we propose an end-to-end 3D CNN-based approach to simultaneously retrieve and align CAD models to the objects of a scan in a single pass, for scans of varying sizes. This end-to-end formulation enables the final alignment process to inform learning of scan-CAD correspondences. To enable effective learning of scan-CAD object correspondences, we propose to use *symmetry-aware object correspondences* (SOCs), which establish dense correspondences between scan objects and CAD models, and are trained by our differentiable Procrustes alignment loss.

Then for an input scan $\mathbb{S}$ represented by volumetric grid encoding a truncated signed distance field, our model first detects object center locations as heatmap predictions over the volumetric grid and corresponding bounding box sizes for each object location. The bounding box represents the extent of the underlying object. From these detected object locations, we use the estimated bounding box size to crop out the neighborhood region around the object center from the learned feature space in order to predict our SOC correspondences to CAD models.

From this neighborhood of feature information, we then predict SOCs. These densely establish correspondences for each voxel in the object neighborhood to CAD model space. In order to be invariant to potential reflection and rotational symmetries, which could induce ambiguity in the correspondences, we simultaneously estimate the symmetry type of the object. We additionally predict a binary mask to segment the object instance from background clutter in the neighborhood, thus informing the set of correspondences to be used for the final alignment. To find a CAD model corresponding to the scan object, we jointly learn an object descriptor which is used to retrieve a semantically similar CAD model from a database.

Finally, we introduce a differentiable Procrustes alignment, enabling a fully end-to-end formulation, where learned scan object-CAD SOC correspondences can be informed by the final alignment process, achieving efficient and accurate 9DoF CAD model alignment for 3D scans.

## 4.4 Method

### 4.4.1 Network Architecture

Our network architecture is shown in Figure Figure 4.2. It is designed to operate on 3D scans of varying sizes, in a fully-convolutional manner. An input scan is given by a volumetric grid encoding a truncated signed distance field, representing the scan geometry. We design our network backbone to learn features for detecting objects in a scan, establishing SOCs, and aligning CAD models to them. The end-to-end formulation enables the learned SOCs to be informed by the alignment performance.

The network backbone is structured in an encoder-decoder fashion, and composed of a series of ResNet blocks [97]. The bottleneck volume is spatially reduced by a factor of 16 from the input volume, and is decoded to the original resolution through transpose convolutions. The decoder is structured symmetrically to the encoder, but with half the feature channels, which we empirically found to produce faster convergence and more accurate performance. The output of the decoder is used to predict an objectness heatmap, identifying potential object locations, which is employed to inform bounding box regression for object detection. The predicted object

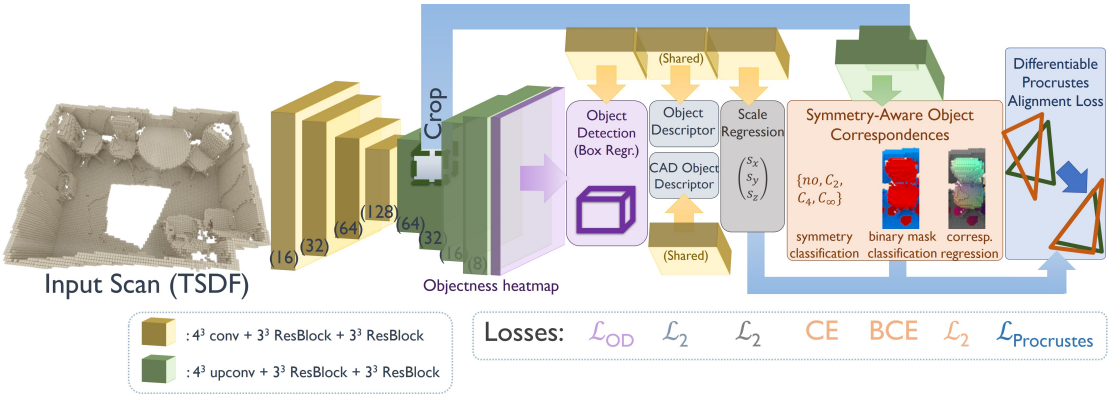



**Figure 4.2:** Network architecture for our end-to-end approach for CAD model alignment. An input TSDF scan represented in a volumetric grid is input to an encoder-decoder backbone constructed with residual blocks. Objects are detected through objectness prediction and bounding box regression; these predicted object boxes are then used to crop features from the decoder to inform CAD model alignment to a detected object. The cropped features are processed to simultaneously predict an object descriptor constrained to be similar to a corresponding CAD object descriptor (used for retrieving CAD models) and a 3-dimensional scale. Our symmetry-aware object correspondences (SOCs) informs directly our differentiable Procrustes alignment loss.

bounding boxes are used to crop and extract features from the output of the second decoder layer, which then inform the SOC predictions. The features used to inform the SOC correspondence are extracted from the second block of the decoder, whose feature map spatial dimensions are $1/4$ of the original input dimension.

**Object Detection** We first detect objects, predicting bounding boxes for the objects in a scan, which then inform the SOC predictions. The output of the backbone decoder predicts heatmaps representing objectness probability over the full volumetric grid (whether a voxel is a center of an object). We then regress object bounding boxes corresponding to these potential object centers. For object bounding boxes predictions, we regress a 3-channel feature map, with each 3-dimensional vector corresponding to the bounding box extent size, and regressed using an $\ell_2$ loss.

Objectness is predicted as a heatmap, encoding voxel-wise probabilities as to whether each voxel is a center of an object. Note that $\Omega \subset \mathbb{N}^3$ is the discretized space (i.e. voxel grid). To predict a location heatmap $H_1$, we additionally employ two proxy losses, using a second heatmap prediction $H_2$ as well as a predicted offset field $O$. $H_1$ and $H_2$ are two 1-channel heatmaps designed to encourage high recall and precision, respectively, and $O$ is a 3-channel grid representing an offset field to the nearest object center. The objectness heatmap loss is:

$$\mathcal{L}_{OD} = 2.0 \cdot \mathcal{L}_{\text{recall}} + 10.0 \cdot \mathcal{L}_{\text{precision}} + 10.0 \cdot \mathcal{L}_{\text{offset}}$$

The weights for each component in the loss are designed to bring the losses numerically to approximately the same order of magnitude. Here, $\mathcal{L}_{\text{recall}}$ and $\mathcal{L}_{\text{precision}}$ are inspired from the conditional keypoint correspondence heatmap predictions of Scan2CAD [1].

$\mathcal{L}_{\text{recall}}$ aims to achieve high recall. It operates on the prediction $H_1$, on which we apply a sigmoid and calculate the loss via binary-cross entropy (BCE). This loss on its own tends to establish a high recall, but also blurry predictions.

$$\mathcal{L}_{\text{recall}} = \sum_{x \in \Omega} \text{BCE}(\sigma(H_1(x)), H_{\text{GT}}(x)) \tag{4.1}$$

$$H_1 : \Omega \rightarrow [0, 1], \quad \sigma : \text{sigmoid} \tag{4.2}$$

$\mathcal{L}_{\text{precision}}$ aims to achieve high precision. It operates on the prediction $H_2$, on which we apply a softmax and calculate the loss via negative log-likelihood (NLL). Due to the softmax, this loss encourages highly localized predictions in the output volume, which helps to attain high precision.

$$\mathcal{L}_{\text{precision}} = \sum_{x \in \Omega} \text{NLL}(\sigma(H_2(x)), H_{\text{GT}}(x)) \tag{4.3}$$

$$H_2 : \Omega \rightarrow [0, 1], \quad \sigma : \text{softmax} \tag{4.4}$$

$\mathcal{L}_{\text{offset}}$ is a regression loss on the predicted 3D offset field $O$, following [98]. Each voxel of $O$ represents a 3-dimensional vector that points to the nearest object center. This regression loss is used as a proxy loss to support the other two classification losses.

$$\mathcal{L}_{\text{offset}} = \sum_{x \in \Omega} \|O(x) - O_{\text{GT}}(x)\|_2^2 \tag{4.5}$$

$$O : \Omega \rightarrow \mathbb{R}^3$$

**Predicting SOCs** SOCs are dense, voxel-wise correspondences from scan geometry to CAD models. They are defined as $\text{SOC} : \Omega \rightarrow [-0.5, 0.5]^3$, the normalized space of the CAD models.

In order to account for symmetry ambiguities, ground truth SOCs are generated such that the front-facing axis of the CAD model maintains minimal angle with the x-axis of the scan voxel grid. Thus for symmetric objects, the SOCs are generated in a consistent fashion, i.e., always aligned with the x-axis of the scan coordinate system.

SOCs are predicted using features cropped from the network backbone. For each detected object, we crop a region with the extend of the predicted bounding box volume $\mathcal{F}$ from the feature map of the second upsampling layer to inform our dense, symmetry-aware object correspondences. This feature volume $\mathcal{F}$ is first fitted through tri-linear interpolation into a uniform voxel grid of size $48^3$ before streaming into different prediction heads. SOCs incorporate several output predictions: a volume of dense correspondences from scan space to CAD object space, an instance segmentation mask, and a symmetry classification.

The dense correspondences, which map to CAD object space, implicitly contain CAD model alignment information. These correspondences are regressed as CAD object space coordinates, similar to [99], with the CAD object space defined as a uniform grid centered around the object, with coordinates normalized to $[-0.5, 0.5]$. These coordinates are regressed using an $\ell_2$ loss.

We also introduce a proxy symmetry loss to encourage correct SOC prediction by predicting the symmetry class of the object for common symmetry classes for furniture objects: two-fold rotational symmetry, four-fold rotational symmetry, infinite rotational symmetry, and no symmetry.

**Retrieval** To retrieve a similar CAD model to the detected object, we use the cropped feature neighborhood $\mathcal{F}$ to train an object descriptor for the scan region, using a series of 3D convolutions to reduce the feature dimensionality. This resulting $512$-dimensional object descriptor is then constrained to match the latent vector of an autoencoder trained on the CAD model dataset, with latent spaces constrained by an $\ell_2$ loss. This enables retrieval of a semantically similar CAD model at test time through a nearest neighbor search using the object descriptor.

**Scale** Similarly to the retrieval head, the scale is predicted per detected object (i.e. per crop). We regress the $\mathbb{R}^3$ scale vector with an $\ell_2$ loss. At train and test time this estimate is used as final scale estimate with further post processing.

**9DoF Alignment** Our differentiable 9DoF alignment enables training for CAD model alignment in an end-to-end fashion, thereby informing learned correspondences of the final alignment objective. To this end, we leverage a differentiable Procrustes loss on the masked correspondences given by the SOC predictions to find the rotation alignment. That is, we aim to find a rotation matrix $R$ which brings together the CAD and scan correspondence points $P_c, P_s$:

$$R^* = \text{argmin}_R ||RP_c - P_s||_F, \qquad R \in SO_3$$

This is solved through a differentiable SVD of $P_s P_c^T = U\Sigma V^T$, with $R = U \begin{bmatrix} 1 & & \\ & 1 & \\ & & d \end{bmatrix} V^T$, $d = \det(VU^T)$. Here, the SVD is computed by solving the non-linear characteristic polynomial of the $3 \times 3$ matrix $P_s P_c^T$ iteratively, giving the final rotation. For scale and translation, we directly regress the scale using two 3D downsampling convolutions on $\mathcal{F}$, and the translation is predicted from the detected object centers. Note that an object center is the geometric center of the bounding box.

### 4.4.2 Training

**Data** Input scan data is represented by its truncated signed distance field (TSDF) encoded in a volumetric grid and generated through volumetric fusion [13] (we use voxel size = 3cm, truncation = 15cm). The CAD models used to train the autoencoder to produce a latent space for scan object descriptor training are represented as unsigned distance fields (DF), using the level-set generation toolkit by Batty [90].

To train our model for CAD model alignment for real scan data, we use the Scan2CAD dataset introduced by [1]. These Scan2CAD annotations provide 1506 scenes for training. Using upright rotation augmentation, we augment the number of training samples by 4 ($90^\circ$ increments with $20^\circ$ random jitter). We train our network using full scenes as input, with batch size of 1. For SOC prediction at train time the batch size is equal to the number of groundtruth objects in the given

| | bath | bookshelf | cabinet | chair | display | sofa | table | trash bin | other | class avg. | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| FPFH (Rusu et al. [37]) | 0.00 | 1.92 | 0.00 | 10.00 | 0.00 | 5.41 | 2.04 | 1.75 | 2.00 | 2.57 | 4.45 |
| SHOT (Tombari et al. [38]) | 0.00 | 1.43 | 1.16 | 7.08 | 0.59 | 3.57 | 1.47 | 0.44 | 0.75 | 1.83 | 3.14 |
| Li et al. [47] | 0.85 | 0.95 | 1.17 | 14.08 | 0.59 | 6.25 | 2.95 | 1.32 | 1.50 | 3.30 | 6.03 |
| 3DMatch (Zeng et al. [43]) | 0.00 | 5.67 | 2.86 | 21.25 | 2.41 | 10.91 | 6.98 | 3.62 | 4.65 | 6.48 | 10.29 |
| Scan2CAD (Avetisyan et al. [1]) | 36.20 | 36.40 | 34.00 | 44.26 | 17.89 | **70.63** | 30.66 | 30.11 | 20.60 | 35.64 | 31.68 |
| Direct 9DoF | 5.88 | 13.89 | 13.48 | 21.94 | 2.78 | 8.04 | 10.53 | 13.01 | 17.65 | 11.91 | 15.12 |
| Ours (no symmetry) | 11.11 | 29.27 | 29.29 | 68.26 | 20.41 | 16.26 | 41.03 | 40.12 | 14.29 | 30 | 40.51 |
| Ours (no SOCs) | 11.11 | 21.95 | 7.07 | 61.77 | 8.16 | 9.76 | 28.21 | 17.9 | 19.48 | 20.6 | 29.97 |
| Ours (no anchor) | **45.24** | **45.85** | 47.16 | 61.55 | **27.65** | 51.92 | 41.21 | 31.13 | **29.62** | 42.37 | 47.64 |
| Ours (no Procrustes) | 33.33 | 36.59 | 28.28 | 50.51 | 14.29 | 13.01 | 58.97 | 35.19 | 28.57 | 33.19 | 35.74 |
| **Ours (final)** | 38.89 | 41.46 | **51.52** | **73.04** | 26.53 | 26.83 | **76.92** | **48.15** | 18.18 | **44.61** | **50.72** |

**Table 4.1:** Accuracy comparison (%) on Scan2CAD [1]. We compare to state-of-the-art handcrafted feature descriptors (FPFH [37], SHOT [38], Li et al. [47]) as well as learned descriptors (3DMatch [43], Scan2CAD [1]) for CAD model alignment. These approaches consider correspondence finding and pose alignment optimization independently, while our end-to-end formulation can learn correspondences informed by alignment, achieving significantly higher CAD model alignment accuracy.

scene as crops are only performed around groundtruth object centers. Only large scenes during training are randomly cropped to $400 \times 400 \times 64$ to meet memory requirements. We found that training using 1 scene per batch generally yields stable convergence behavior.

For CAD model alignment to synthetic scan data, we use the SUNCG dataset [64], where we virtually scan the scenes following [66, 57] to produce input partial TSDF scans. The training process for synthetic SUNCG scan data is identical to training with real data. See supplemental material for further details.

**Optimization** We use an SGD optimizer with a batch size of 1 scene and an initial learning rate of 0.002, which is decayed by 0.5 every $20K$ iterations. We train for $50K$ iterations until convergence, which takes $\approx 48$ hours.

We train our model from scratch with the exception of the object retrieval descriptors. For object retrieval, we pre-train an autoencoder on all ShapeNetCore CAD models, trained to reconstruct their distance fields at $32^3$. This CAD autoencoder is trained with a batch size of 16 for $30K$ iterations. We then train the full model with pre-trained object descriptors for all ShapeNet models for CAD model alignment, with the CAD autoencoder latent space constraining the object descriptor training for retrieval.

## 4.5 Results

We evaluate our proposed end-to-end approach for CAD model alignment in comparison to the state of the art as well as with an ablation study analyzing our differentiable Procrustes alignment loss and various design choices. We evaluate on real-world scans using the Scan2CAD dataset [1]. We use the evaluation metric proposed by Scan2CAD [1]; that is, the ground truth CAD model pool is available as input, and a CAD model alignment is considered to be successful if the category of the CAD model matches that of the scan object and the alignment falls within 20cm, $20^\circ$, and $20\%$ for translation, rotation, and scale, respectively. For further evaluation on synthetic scans, we refer to the supplemental material.

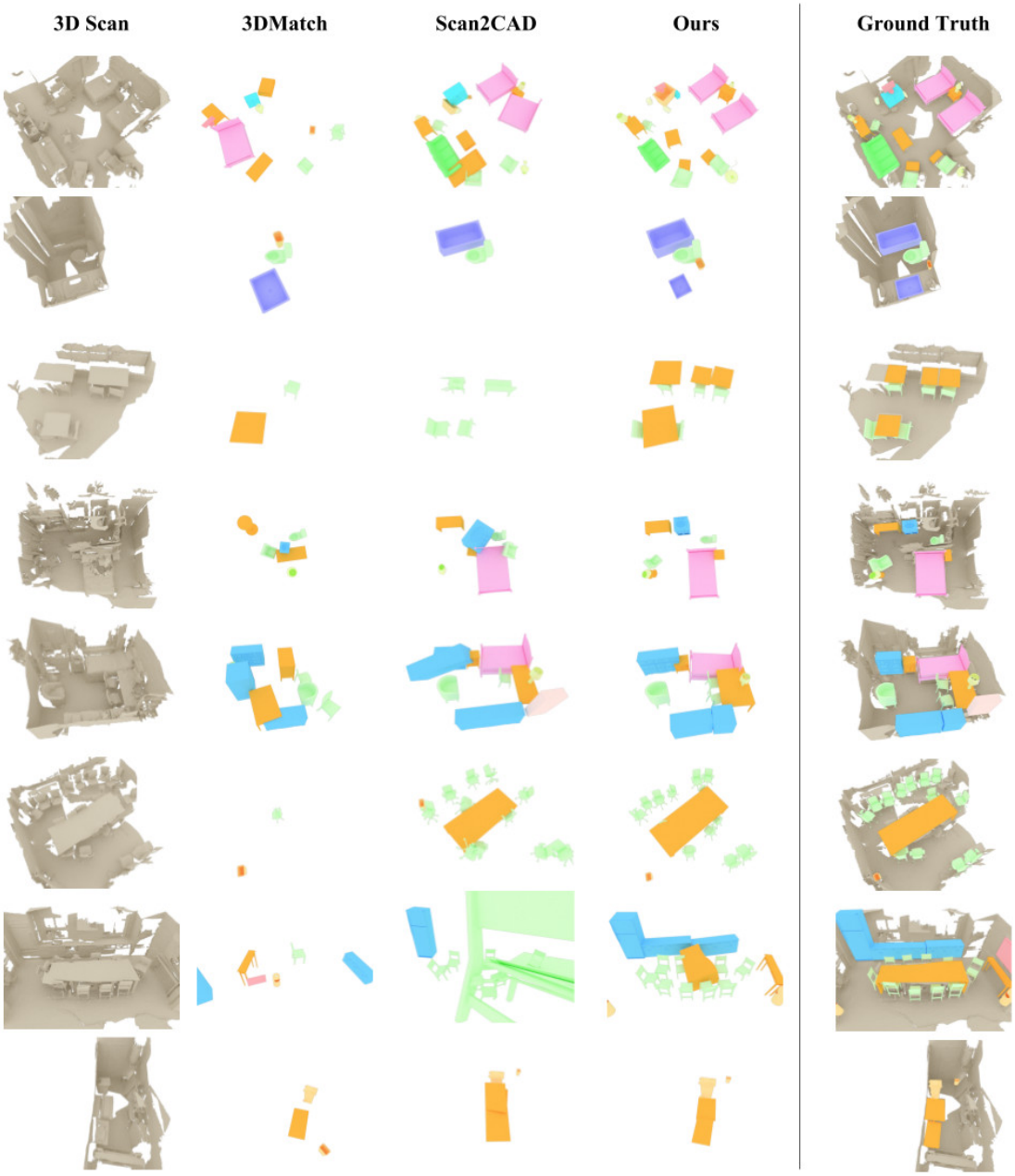



**Figure 4.3:** Qualitative comparison of CAD model alignment to ScanNet [5] scans. Our joint formulation of SOC correspondence prediction and differentiable Procrustes alignment enable both more accurate and robust CAD model alignment estimation across varying scene types and sizes.

| Scene size | small | medium | large |
|---|---|---|---|
| Scene dim | $128 \times 96 \times 48$ | $144 \times 128 \times 64$ | $256 \times 320 \times 64$ |
| # objects | 7 | 16 | 20 |
| Scan2CAD [1] | 288.60s | 565.86s | 740.34s |
| Ours | **0.62s** | **1.11s** | **2.60s** |

**Table 4.2:** Runtime (seconds) of our approach on varying-sized scenes. Our end-to-end approach predicts CAD model alignment in a single forward pass, enabling very efficient CAD model alignment – several hundred times faster than previous data-driven approaches.

In addition to evaluating CAD model alignment using the Scan2CAD [1] evaluation metrics, we also evaluate our approach on an unconstrained scenario with 3000 random CAD models as a candidate pool, shown in Figure Figure 4.4. In this scenario, we maintain robust CAD model alignment accuracy with a much larger set of possible CAD models.

**Comparison to state of the art.** Table Table 4.1 evaluates our approach against several state-of-the-art methods for CAD model alignment, which establish correspondences and alignment independently of each other. In particular, we compare to several approaches leveraging handcrafted feature descriptors: FPFH [37], SHOT [93], Li et al. [47], as well as learned feature descriptors: 3DMatch [43], Scan2CAD [1]. We follow these descriptors with RANSAC to obtain final alignment estimation, except for Scan2CAD, where we use the proposed alignment optimization. Our end-to-end formulation, where correspondence learning can be informed by the alignment, outperforms these decoupled approaches by over $19.04\%$. Figure Figure 4.3 shows qualitative visualizations of our approach in comparison to these methods.

**How much does the differentiable Procrustes alignment loss help?** We additionally analyze the effect of our differentiable Procrustes loss. In Table Table 4.1, we compare several different alignment losses. As a baseline, we train our model to directly regress the 9DoF alignment parameters with an $\ell_2$ loss. We then evaluate our approach with (final) and without (no Procrustes) our differentiable Procrustes loss. For CAD model alignment to 3D scans, our differentiable Procrustes alignment notably improves performance, by over $14.98\%$.

**How much does SOC prediction help?** We evaluate our SOC prediction on CAD model alignment in Table Table 4.1. We train our model with (final) and without (no SOCs) SOC prediction as well as with coordinate correspondence prediction but without symmetry (no symmetry). We observe that our SOC prediction significantly improves performance, by over $20.75\%$. Establishing SOCs is fundamental to our approach, as dense correspondences can produce more reliable alignment, and unresolved symmetries can lead to ambiguities and inconsistencies in finding object correspondences. In particular, we also evaluate the effect of symmetry classification in our SOCs; explicitly predicting symmetry yields a performance improvement of $10.21\%$.

**What is the effect of using an anchor mechanism for object detection?** In Table Table 4.1, we also compare our CAD model alignment approach with (final) and without (no anchor) using anchors for object detection, where without anchors we predict only object

center locations as a probability heatmap over the volumetric grid of the scan, but do not regress bounding boxes, and thus only crop a fixed neighborhood for the following SOCs and alignment. We observe that by employing bounding box regression, we can improve CAD model alignment performance, as this facilitates scale estimation and allows correspondence features to encompass the full object region.

### 4.5.1 Limitations

Although our approach shows significant improvements compared to state of the art, we believe there are directions for improvement. Currently, we focus on the objects in a scan, but do not consider structural components such as walls and floors. We believe, however, that our method could be expanded to detect and match plane segments in the spirit of structural layout detection such as PlaneRCNN [100]. In addition, we currently only consider the geometry of the scan or CAD; however, it is an interesting direction to consider finding matching textures in order to better visually match the appearance of a scan. Finally, we hope to incorporate our alignment algorithm in an online system that can work at interactive rates and give immediate feedback to the scanning operator.

## 4.6 Conclusion

We have presented an end-to-end approach that automatically aligns CAD models with commodity 3D scans, which is facilitated with symmetry-aware correspondences and a differentiable Procrustes algorithm. We show that by jointly training the correspondence prediction with direct, end-to-end alignment, our method is able to outperform existing state of the art by over $19.04\%$ in alignment accuracy. In addition, our approach is roughly $250\times$ faster than previous data-driven approaches and thus could be easily incorporated into an online scanning system. Overall, we believe that this is an important step towards obtaining clean and compact representations from 3D scans, and we hope it will open up future research in this direction.

| | bath | bookshelf | cabinet | chair | display | sofa | table | trash bin | other | class avg. | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| SHOT (Tombari et al. [38]) | 0 | 1.8 | 0 | 8.8 | 0.0 | 1.2 | 0 | 0 | 2.2 | 1.5 | 2.8 |
| FPFH (Rusu et al. [37]) | 0 | 0 | 1.5 | 10.7 | 0 | 1.2 | 2.1 | 2.9 | 0 | 2.0 | 3.7 |
| Li et al. [47] | 0 | 1.8 | 2.3 | 1.11 | 0 | 2.8 | 6.4 | 2.7 | 0 | 3.0 | 4.6 |
| 3DMatch (Zeng et al. [43]) | 0 | 5.3 | 3.8 | 19.5 | 1.7 | 5.2 | 17.0 | 6.0 | 6.5 | 7.2 | 9.2 |
| Scan2CAD (Avetisyan et al. [1]) | 25.0 | 28.1 | 30.8 | 39.7 | 20.3 | 14.3 | 51.1 | 31.5 | 19.6 | 28.9 | 28.8 |
| **Ours** | **40.6** | **38.6** | **36.2** | **68.1** | **25.4** | **27.0** | **63.8** | **38.0** | **40.2** | **42.0** | **44.1** |

**Table 4.3:** Performance comparison (%) on the hidden test set of the Scan2CAD alignment benchmark [1]. We outperform existing methods by a significant margin on all classes; the last two rows provide class and average instance alignment accuracy, respectively.

| | bed | cabinet | chair | desk | dresser | other | shelves | sofa | table | class avg. | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| SHOT (Tombari et al. [38]) | 13.43 | 3.23 | 10.18 | 2.78 | 0 | 0 | 1.75 | 3.61 | 11.93 | 5.21 | 6.3 |
| FPFH (Rusu et al. [37]) | 38.81 | 3.23 | 7.64 | 11.11 | 3.85 | 13.21 | 0 | 21.69 | 11.93 | 12.39 | 9.94 |
| Scan2CAD (Avetisyan et al. [1]) | 52.24 | 17.97 | 36 | **30.56** | 3.85 | 20.75 | 7.89 | 40.96 | 43.12 | 28.15 | 29.23 |
| Ours (No Procrustes) | **71.64** | 29.95 | 39.27 | 23.61 | 30.77 | 20.75 | 9.65 | **69.88** | 40.37 | 37.32 | 36.42 |
| **Ours (final)** | **71.64** | **32.72** | **48.73** | 27.78 | **38.46** | **37.74** | **14.04** | 67.47 | **45.87** | **42.72** | **41.83** |

**Table 4.4:** CAD alignment accuracy comparison (%) on SUNCG [64]. We compare to state-of-the-art handcrafted feature descriptors FPFH [37], SHOT [38] as well as a learning based method Scan2CAD [1] for CAD model alignment. Note that the Procrustes loss considerably improves overall alignment accuracy.

## 4.7 Appendix

### 4.7.1 SUNCG

We conduct experiments on the SUNCG dataset [64] to verify the effectiveness of our method. For training and evaluation, we create virtual scans of the synthetic scenes, where we simulate a large-scale indoor 3D reconstruction by using rendered depth frames similar to [57, 66] with the distinction that we add noise to the synthetic depth frames in the fusion process. The voxel resolution for the generated SDF grids is at $4.68cm$. The ground truth models are provided by the SUNCG scenes, where we discard any objects that have not been seen during the virtual scanning (no occupancy in the scanned SDF). We show a quantitative evaluation in Tab. Table 4.4, where we outperform the current state-of-the-art method Scan2CAD [1] by a significant margin. We show that our method can align CAD models robustly through all classes. Additionally, we see that our Procrustes loss notably improves overall alignment accuracy. In particular, for less frequent CAD models (e.g., those summarized in *other*), we observe a considerable improvement in alignment accuracy.

Fig. Figure 4.5 shows qualitative results on scanned SUNCG scenes. Our end-to-end approach is able to handle large indoor scenes with complex furniture arrangements.

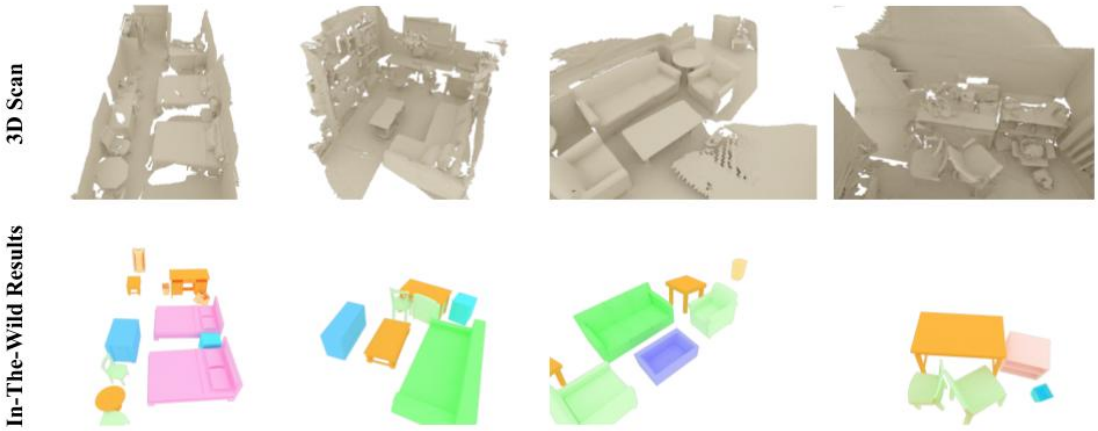



**Figure 4.4:** Our end-to-end CAD model alignment approach applied to an unconstrained set of candidate CAD models; here, we use a set of 3000 randomly selected CAD models from ShapeNet-Core [4]. The results of our approach (bottom) show robust CAD model alignment performance in a scenario which is often reflected in real-world applications.

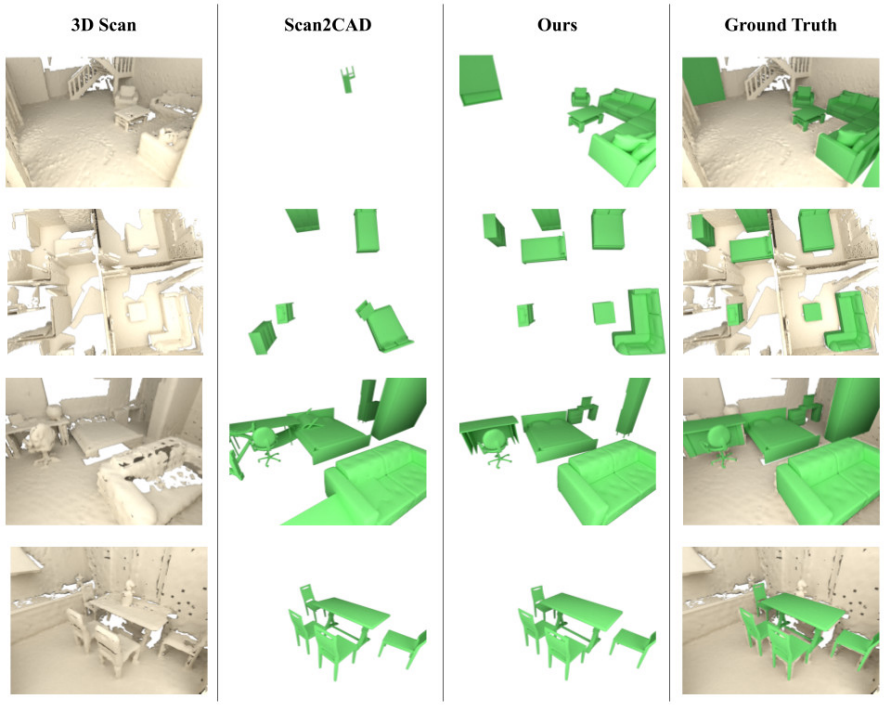



**Figure 4.5:** Qualitative results on virtual scans from SUNCG. Note that our method handles complex CAD arrangements better than Scan2CAD.

# 5 Learning CAD Model Alignments and Scene Layouts in RGB-D Scans

This chapter introduces the following paper. Reproduced with permission from Springer Nature: **Avetisyan, A.**, Khanova, T., Choy, C., Dash, D., Dai, A., & Nießner, M. (2020). Scenecad: Predicting object alignments and layouts in rgb-d scans. *In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16* (pp. 596-612). Springer International Publishing.

**Summary** In this publication we propose a novel approach to reconstructing lightweight, CAD-based representations of 3D environments scanned from commodity RGB-D sensors. Our approach involves optimizing both the alignment of the CAD model and the layout of the scanned scene, taking into account the relationships between objects and their placement within the scene. By treating the problem of object arrangement and scene layout as interconnected, we are able to generate more accurate and consistent representations of the scanned environment. We align the object CAD models to the scene by predicting dense correspondences with the scene geometry and use a hierarchical layout prediction technique to determine the layout planes from the corners and edges of the scene. To accomplish this, we employ a message-passing graph neural network that learns the connections between objects and the layout, resulting in a globally accurate alignment of objects in the scene. Our lightweight layout prediction module builds structural components in three steps. First, it identifies corner positions and generates a feature vector for each. Next, it pairs each corner's feature vector with every other corner to create a list of all potential edges, which is then input into an MLP layer to determine valid connections between corners. The surviving links form a 3D wireframe. In the final step, planar cycles in the adjacency graph of the 3D wireframe are detected and filtered by another MLP. Detected objects and layout components in the scene are fed into a graph neural network to estimate the relationships between objects and layout by predicting object-object relative poses as well as object-layout support relationships. This formulation guides both object and layout arrangement to be consistent with each other. Our method, which takes into account the overall layout of the scene, leads to a significant enhancement in CAD alignments compared to existing techniques. On SUNCG, alignment accuracy increases from 41.83% to 58.41%, and on ScanNet, it improves from 50.05% to 61.24%. As a result, the CAD-based representations generated by our method are particularly suitable for use in content creation such as AR/VR.

**Contributions** The first author implemented the main method, including the relationship graph neural network and the lightweight layout estimation component. Tatiana Khanova implemented the baseline methods of *Scan2BIM* and *SemSeg + RANSAC*, and led the creation of the layout dataset. All the co-authors participated in discussions that led to the final publication.

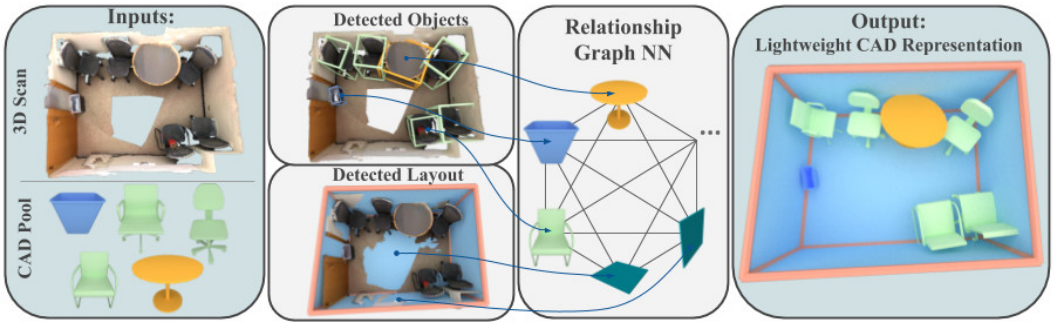



**Figure 5.1:** Our method takes as input a 3D scan and a set of CAD models. We jointly detect objects and layout elements in the scene. Each detected object or layout component then forms a node in a graph neural network which estimates object-object relationships and object-layout relationships. This holistic understanding of the scene enables results in a lightweight CAD-based representation of the scene.

## 5.1 Introduction

The recent progress of 3D reconstruction of real-world environments from commodity range sensors has spurred interest in using such captured 3D data for applications across many fields, such as content creation, mixed reality, or robotics. State-of-the-art 3D reconstruction approaches can now produce impressively-robust camera tracking and surface reconstruction [15, 16, 72, 22].

Unfortunately, the resulting 3D reconstructions are not well-suited for direct use with many applications, as the geometric reconstructions remain incomplete (e.g., due to occlusions and sensor limitations), are often noisy or oversmoothed, and often consume a large memory footprint due to high density of triangles or points used to represent a surface at high resolution. There still remains a notable gap between these reconstructions and artist-modeled 3D content, which are clean, complete, and lightweight [54].

Inspired by these attributes of artist-created 3D content, we aim to construct a CAD-based scene representation of an input RGB-D scan, with objects represented by individual CAD models and scene layout represented by lightweight meshes. In contrast to previous approaches which have individually tackled the tasks of CAD model alignment [47, 1, 2] and of layout estimation [101, 102, 103], we observe that object arrangement is typically tightly correlated with the scene layout. We thus propose to jointly optimize for CAD model alignment and scene layout to produce a globally-consistent CAD-based representation of the scene.

From an input RGB-D scan along with a CAD model pool, we align CAD models to the scanned scene by establishing dense correspondences. To estimate the scene layout, we characterize the layout into planar elements, and propose a hierarchical layout prediction by first detecting corner locations, then predicting scene edges, and from sets of edges potentially presenting a layout plane, predicting the final layout. We then propose a graph neural network architecture for optimizing the relationships between objects and layout by predicting object-object relative poses as well as object-layout support relationships. This optimization guides both object and layout arrangement to be consistent with each other. Our approach is fully-convolutional and trained

end-to-end, generating a CAD-based scene representation of a scan in a single forward pass.

In summary, we present the following contributions:

- We formulate a lightweight heuristic-free 3D layout prediction algorithm that hierarchically predicts corners, edges and then planes in an end-to-end fashion consisting of only $\approx 1M$ trainable parameters generating satisfactory layouts without the need for extensive heuristics.
- We present a scene graph network that learns relationships between objects and scene layout, enabling globally consistent CAD model alignments and results in a significant increase in prediction performance in both synthetic as well as real-world datasets.
- We introduce a new richly-annotated real-world scene layout dataset consisting of 1151 CAD shells and wireframes on top of the ScanNet RGB-D dataset, allowing large-scale data-driven training for layout estimation.

## 5.2 Related Work

**CAD model alignment** Aligning an expert-generated 3D model or a 3D template to 3D scan data has been studied widely due to its wide range of applications, for instance motion capture [104], 3D object detection and localization [40, 105, 106], and scene registration [107]. Our aim is to leverage large-scale datasets of CAD models to reconstruct a lightweight, semantically-informed, high-quality CAD representation of an RGB-D scan of a scene. Several approaches have been developed to retrieve and align CAD models from a shape database and align them in real time to a scan during the 3D scanning process [78, 47], although their use of handcrafted features for geometric scan-to-CAD matching limit robustness.

Zeng et al. [43] developed a learned feature extractor using a siamese network design for geometric feature matching, which can be employed for scan-to-CAD feature matching, though this remains difficult due to the domain gap between synthetic CAD models and real-world scans. Avetisyan et al. [1] proposed a scan-to-CAD retrieval and alignment approach leveraging learned features to detect objects in a 3D scan and establish correspondences across the domain gap of scan and CAD. They later built upon this work to develop a fully end-to-end trainable approach for this CAD alignment task [2]. For such approaches, each object is considered independently, whereas our approach exploits contextual information from object-object and object-layout to produce globally consistent CAD model alignment and layout estimation.

Other approaches retrieve and align CAD models to RGB images [80, 108, 55]; our work instead focuses on geometric alignment of CAD models and layout.

**Graph neural networks and relational inference in 3D.** Recent developments in graph inference and graph neural networks have shown significant promise for inference on 3D data. Recently, various approaches have viewed 3D meshes as graphs in order find correspondences between 3D shapes [109], deform a template mesh to fit an image observation of a shape [110], or generate a mesh model of an object [111], among other applications. Learning on graphs

has also shown promise for estimating higher-level relational information in scenes, as a scene graph. 3D-RelNet [112] predicts 3D shapes and poses from single RGB images and establish pairwise pose constraints between objects to improve overall prediction quality. Our approach is similarly inspired to establish relationships between objects; we additionally employ relationships between objects and structural components (i.e., walls, floors, and ceilings), which considerably inform object arrangement. Armeni et al. [113] propose a unified hierarchical structure that hosts building, room, and object relationships into one 3D scene graph. They leverage this graph structure to generate scene graphs from 2D images. Our approach focuses on leveraging relational information to reconstruct imperfect scans with a CAD-based representation for each object and layout element.

**Layout estimation.** Various layout estimation approaches have been developed to infer structural information from RGB and RGB-D data. Scan2BIM [101] generates building information models (BIM) from 3D scans by detecting planes and finding plausible intersections to produce room-level segmentation of floors, ceilings and walls under Manhattan-style constraints. PlaneR-CNN [100] and PlaneNet [114] propose deep neural network architectures to detect planes from RGB images and estimate their 3D parameters. FloorNet [102] estimates a 2D Manhattan-style floorplan representation for an input RGB-D scan using a point-based neural network architecture. Floor-SP [103] relaxes the Manhattan constraints with an integer programming formulation, and produces more robust floorplan estimation. In contrast to these layout estimation approaches, our focus lies in leveraging global scene relations between objects as well as structural elements in order to produce a CAD-based representation of the scene.

**Single view 3D reconstruction.** Holistic 3D Scene Parsing [82] parses a single RGB image and reconstruct a holistic 3D arrangements of CAD models jointly optimizing for 3D object detection, scene layout and hidden human context. Zou et al. [115] infers a complete interpretation of the scene from a single RGBD frame where objects and scene layout are predicted in data-driven fashion. In contrast to single view reconstruction, our approach aims towards holistic scene understanding that can operate on large-scale 3D scenes while consuming only a few seconds of runtime at test time.

## 5.3 SceneCAD: Joint Object Alignment and Layout Estimation

The input scan is represented as a sparse 3D voxel grid of the occupied surface geometry carrying fused RGB data. The scan is first encoded by a series of sparse 3D convolutional layers [116] to produce a feature volume $F'$. The sparse output $F'$ is then densified into a dense 3D feature grid $F \in \mathbb{R}^{N_f \times N_x \times N_y \times N_z}$ where $N_F$ is the number of channels in the feature and $N_x$, $N_y$, and $N_z$ are the resolution of the feature along x, y, and z axis respectively. Note that the encoder serves as backbone for proceeding modules. Hence, $F$ is the input to the CAD alignment module as well as the layout estimation module.

Based on $F$, we detect objects along with their bounding box in the object detection module and layout planes in the layout detection module. We then establish our relational inference by formulating a message-passing graph neural network on the predicted objects and layout

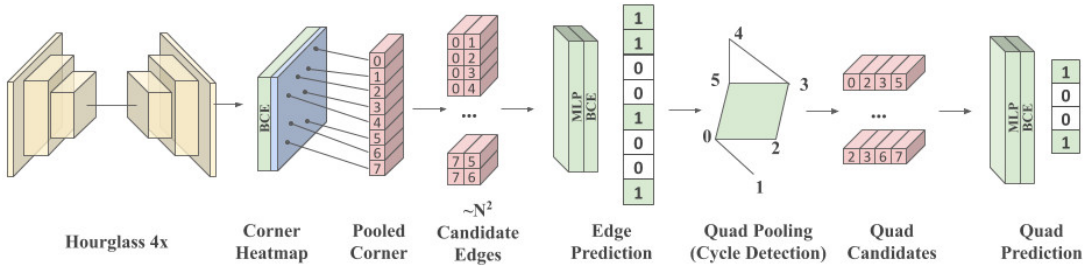



**Figure 5.2:** Layout estimation as planar quad structures. Layout components are characterized as planar elements which are detected hierarchically. From an input scene, corners of these layout elements are predicted in heatmap fashion leveraging non-maximum suppression. From these predicted corners, edges are then predicted for each possible pair of corners as a binary classification task. From the predicted edge candidates, valid quads of four connected edges are considered as candidate layout elements, with a binary classification used to produce the final layout prediction.

planes, where each node represents an object or layout plane, with losses on edge relationships representing relative poses and support. Finally, we predict a set of retrieved CAD models along with their 9-DoF poses (3 translation, 3 rotation, and 3 scale) for every detected object.

The message-passing graph neural network helps to inform objects of both relations between other objects as well as with the scene layout, e.g., certain types of furniture such as beds and chairs are typically directly supported by a floor, chairs near a table often face the table. This joint optimization thus helps to enable globally consistent CAD model alignment in the final output.

### 5.3.1 Layout Prediction

The indoor scene of interest in our problem consists of planar or quadrilateral components such as walls, floors, and ceilings. However, some of these planar elements create complex geometry such as bars, beams, or other structures that effectively make template-matching approach to find the room layout challenging. Thus, we propose a bottom-up approach that predicts corners, edges, and planar elements sequentially to predict the room layout. Our layout prediction pipeline is structured hierarchically: first predicting the corner locations, then predicting edges between the corners, and finally extracting quads from the predicted edges. We visualize the overview of the pipeline on Figure Figure 5.2.

Corners are predicted by a convolutional network that decodes $F$ to its original dimension by predicting a heatmap; i.e. a voxel-wise score that indicates a *cornerness* likeliness. The loss for this predicted heatmap is a voxel-wise binary cross-entropy classification loss in conjunction with a softmax and a negative log-likelihood over the entire voxel grid where the problem is formulated as a spatial multi-class problem. This is structured as an encoder-decoder, where the bottleneck lies at a spatial reduction of $4\times$. Note that we make predictions for corners which have not been observed in the input scan (e.g., due to occlusions, c.f.). See supplemental material for a visual illustration of the layout prediction pipeline. From the output corner heatmap, we apply a

non-maximum suppression to filter out weak responses, and define the final corner predictions as a set of xyz coordinates $\mathcal{V} = \{\mathbf{v}_i\}_i$, $\mathbf{v}_i = [x_i, y_i, z_i]$.

We the predict the layout edges from the predicted corners $\mathcal{V}$. We construct the candidate set of edges by taking all pair-wise combinations of corners $\mathbf{e}_{ij} = (\mathbf{v}_i, \mathbf{v}_j)$ for all $i \in [1, ..., |\mathcal{V}|]$ and $j \in [1, ..., i-1]$. We denote all edges as $\mathcal{E} = \{\mathbf{e}_{ij}\}_{ij}$. From the pool of candidate edges we predict a set of edges that belongs to the scene structure using a graph neural network. Specifically, for each potential edge $\mathbf{e}_{ij} = (\mathbf{v}_i, \mathbf{v}_j)$, we extract corresponding features from the vertex prediction convolutional network, $F[\mathbf{v}_i], F[\mathbf{v}_j]$ where $F[\cdot]$ denotes the feature vector at the specified $x, y, z$ coordinate. We concatenate these features along with the normalized coordinates to form an input feature vector for each edge $\mathbf{f}_{\mathbf{e}_{ij}} = [F[\mathbf{v}_i], F[\mathbf{v}_j], \mathbf{N}(\mathbf{v}_i), \mathbf{N}(\mathbf{v}_j)]$. For each edge we construct two feature descriptors with alternating order of corner features $\mathbf{f}_{\mathbf{e}_{ji}}$ to mitigate the effect of order dependency. We feed these concatenated features into a graph network, which we train with edge-wise binary cross entropy loss against ground truth edges. As the vertex predictions have uncertainty, we label edges with predicted vertices within a certain radius from the ground truth layout vertices to be positives. This edge prediction limits the set of candidate layout quads which would otherwise be $O\left(\binom{|\mathcal{V}|}{4}\right)$.

From these predicted edges, we then compute the set of candidate layout quads as the set of planar, valid 4-cycles within these edges $\mathbf{q}_{ijkl} = \{\mathbf{e}_{ij}, \mathbf{e}_{jk}, \mathbf{e}_{kl}, \mathbf{e}_{li}\}$. To detect valid cycles, we use the depth-first-search cycle detection algorithm We predict the final set of layout quads as either positive or negative where the positive predictions constitute the scene layout, decomposed as quads. The feature descriptor for a candidate quad is constructed by concatenating the features from $F$ corresponding to the corner locations of its vertices and normalized corner locations, $\mathbf{q}_{ijkl} = [F[\mathbf{v}_i], F[\mathbf{v}_j], F[\mathbf{v}_k], F[\mathbf{v}_l], \mathbf{N}(\mathbf{v}_i), \mathbf{N}(\mathbf{v}_j), \mathbf{N}(\mathbf{v}_k), \mathbf{N}(\mathbf{v}_l)]$. Similar to the edge features, every quad feature descriptor is 4-way permuted $\mathbf{q}_{jkli}, \mathbf{q}_{klij}$, and $\mathbf{q}_{lijk}$ in order to mitigate order-dependency. This feature is input to an MLP followed by a binary cross entropy loss. From these predicted quads, we recover the scene layouts without heuristic post-processing.

### 5.3.2 CAD Model Alignment

Along with the room layout, we aim to find and align light-weight CAD models to objects in the scanned scene. To this end, we propose a CAD model alignment pipeline that detects objects, retrieves CAD models, and finds transformations that aligns the CAD model to the scanned scene. First, we use a single-shot anchor-based object detector to identify objects [57], using the features from the backbone we extracted ($\mathbf{F}$) from the previous stage. We then filter the predicted anchors with non-maximum suppression following the standard single-shot object detection pipeline [117]. Given this set of object bounding boxes $\mathcal{B}$, we extract $N_d \times N_d \times N_d$ feature volume $F_o$ for all $o \in [1, ..., |\mathcal{B}|]$ from the feature map $F$ around the object anchor $a_o$. We use this feature volume for CAD model retrieval and alignment. A corresponding CAD model is retrieved by calculating an object descriptor of length 512 and searching the nearest neighbor CAD model from an shared embedding space. This shared embedding space is established by minimizing the distance between descriptors of scanned objects and their CAD counterpart with an L1 loss during training.

Finally, given the nearest CAD model for all object anchors, we find dense correspondences between the CAD model and the feature volume $F_o$. Dense correspondences are trained through

an explicit voxel-wise L1 regression loss. We use Procrutes [118] to estimate a rotation matrix and an L1 distance loss with respect to the groundtruth rotation matrix to further enhance correspondence quality. Note that the Procrutes method yields a transformation matrix through the Singular Value Decomposition which is differentiable, allowing for end-to-end training.

### 5.3.3 Learning Object and Layout Relationships

From our layout prediction and CAD model alignment, we obtain a set of layout quads and aligned CAD models, both obtained independently from the same backbone features. However, this can result in globally inconsistent arrangements; for instance, objects passing through the ground floor, or shelves misaligned with walls. We thus propose to learn the object-layout as well as object-object relationships as a proxy loss used to guide the CAD model alignments and layout quads into a globally consistent arrangement.

We construct this relationship learning as a graph problem, where the set of objects and layout quads form the nodes of the graph. Edges are constructed between every object-object node-pair and every object-quad node-pair, forming a graph on which we formulate a message-passing graph neural network.

Each node of the graph is characterized by a feature vector of length 128. For objects this feature vector is obtained by pooling the object feature volume to $8^3$ resolution, followed by linearization. For layout quads, this feature vector is constructed by concatenating the features from $F$ or the associated corner locations, upon which an MLP is applied to obtain a 128-dimensional vector.

Figure Figure 5.3 shows an overview of our message-passing network. Messages are passed from nodes to edges for a graph $G = (V, E)$, with nodes $v_i \in V$ and edges $e_{j,k} = (v_j, v_k) \in E$. We define the message passing similar to [119, 120, 111, 121]:

$$v \rightarrow e : \mathbf{h}_{i,j}^{t+1} = f_e(\text{concat}(\mathbf{h}_i^t, \mathbf{h}_j^t - \mathbf{h}_i^t))$$

where $\mathbf{h}_i^t$ is the feature corresponding to vertex $v_i$ at message passing step $t$, $\mathbf{h}_{i,j}^t$ is the feature corresponding edge $e_{i,j}$ at step $t$, and $f_e$ represents an MLP. That is, edges features are computed as the concatenation of its constituent vertices.

We then take these output edge features from the message passing and perform a classification of various relationships using a cross entropy loss. We describe the relationships as follows, which we chose as they do not require extra manual annotation effort given existing ground truth CAD alignments and scene layout; see Section subsection 5.4.2 for more detail regarding extraction of ground truth object and layout relationships. For object-layout relationships, we formulate a 3-class classification task for support relations, predicting *horizontal support*, *vertical support*, or no support. Only one relationship per object-layout pair is allowed. For object-object relationships, we predict the angular difference between the front-facing vectors of the respective objects, in order to recognize common relative arrangements of objects (e.g., chairs often face tables). This is trained with a 6-class cross entropy loss where the angular deviation up to $180\circ$ is discretized into 6 bins.

Here, the relationship prediction adds a proxy loss to the model in Figure Figure 5.2 which inter-correlates object and layout alignments, implicitly guiding the CAD model alignment and layout quad estimation to become more globally consistent.

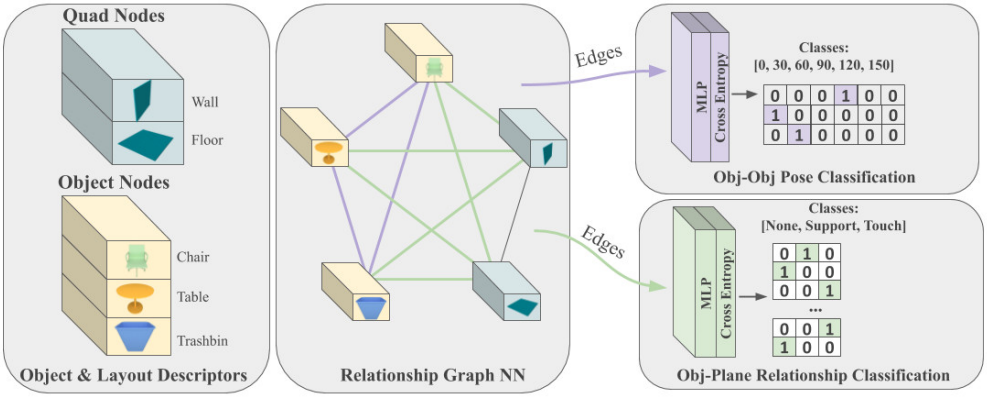



**Figure 5.3:** Object and layout relational prediction. We establish a message-passing neural network in order to predict object-object and object-layout relations. The inputs are feature descriptors of detected objects and quads pooled to the same size, and the output is relationship classification between objects and layout elements, as well as pose relations between objects. Note this relational inference is fully differentiable, enabling end-to-end prediction.

## 5.4 Object+Layout Dataset

To train and evaluate our method, we introduce a new dataset of 1151 CAD layout annotations to the real-world RGB-D scans of the ScanNet dataset [5]. These layout annotations, in addition to the CAD annotations of Scan2CAD [1] to ScanNet scenes, inform our method and evaluation on real-world scan data.

In order to obtain these room layout annotations, we use a semi-automated annotation process. We then automatically extract the object-object and object-layout relations.

### 5.4.1 Extraction of Scene Layouts

We performed a semi-automatic layout annotation for ScanNet scene data. First, large planar surfaces are detected using RANSAC on the reconstructed scans. We then employ a manual refinement step to modify potential errors in the automatic extraction. The surface extraction is preceded by a semantic instance segmentation to obtain wall, floor, ceiling, window, door, etc. instances. RANSAC is then applied to extract 3D planes from each instance. Planes that fall below a threshold will be merged or connected. All planes are projected onto the floor plane and through a set of various heuristics the most plausible intersection points are selected to ultimately become corner points for the final layout. The room height is either estimated by the maximum height of the detected wall instances or is spanned by the ceiling.

Following the proposals given by RANSAC, we then manually verified which proposals were plausible. This step is relatively quick ($\approx$ 2min per scene) and indicated that the RANSAC produced 1151 plausible initial layouts. These layouts were then refined through a manual annotation process. We developed a Blender[1]-based tool was introduced for the layout refinement,

[1] https://www.blender.org

allowing annotators to edit/merge/delete corner junctions as well as add or modify edges and planes. All automatically generated layouts were verified and refined by two student annotators ($\approx 15min$ per scene). An illustration of layouts annotation samples on ScanNet can be found in the supplemental.

### 5.4.2 Extraction of Object and Layout Relationships

To support learning global scene relationships, we extract object and layout relations to supervised our message-passing approach to learning relationships. We opt to learn relations which can be automatically extracted from given CAD model and layout annotations.

We extract object-object and object-layout relationships. For the object-object case, we compute the angular difference between the front-facing vectors of each object where symmetrical properties are ignored; in practice, we compute this on-the-fly during the training process.

Relationships between objects and layout elements are established by support:

- A vertical **support** relationship between a layout element and an object is valid if the bottom side of the bounding box of the object within close proximity to and close to parallel to the layout element.
- A horizontal **touch** relationship is valid if the left, right, front or back side of the bounding box of the object is within close proximity to and close to parallel to the layout element.

These relations are extracted through an exhaustive search. That is, each pair of object-layout is checked for vertical support or horizontal touch. To estimate proximity of objects, we expand the bounding box of the objects by $\tau_p$, and expand the sides of the bounding boxes of the layout elements by $\tau_p$. We then consider the object and layout element to be in close proximity if their expanded bounding boxes overlap. We use $\tau_p = 0.2$ meters for all experiments.

### 5.4.3 Synthetic Data

We additionally evaluate our approach on synthetic data, where CAD object and layout ground truth are provided in the construction of the synthetic 3D scenes. We use synthetic scenes from the SUNCG dataset [122]. SUNCG contains models of indoor building environments including CAD models and room layouts. Layout components are given and hence extraction into planar quads can be performed automatically. To generate the input partial scans, we virtually scan the scenes to produce input scans similar to real-world scenarios, following previous approaches to generate synthetic partial scan data [57].

Object and layout relational information was extracted following the same procedure for ScanNet data.

## 5.5 Results

### 5.5.1 CAD Alignment Performance

We evaluate our method on synthetic SUNCG [122] scans as well as real-world ScanNet [5] scans in Tables Table 5.3 and Table 5.1, respectively. We follow the CAD alignment evaluation

| | bathtub | bookshelf | cabinet | chair | display | other | sofa | table | trashbin | class avg. | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| FPFH (Rusu et al. [37]) | 0.00 | 1.92 | 0.00 | 10.00 | 0.00 | 5.41 | 2.04 | 1.75 | 2.00 | 2.57 | 4.45 |
| SHOT (Tombari et al. [38]) | 0.00 | 1.43 | 1.16 | 7.08 | 0.59 | 3.57 | 1.47 | 0.44 | 0.75 | 1.83 | 3.14 |
| Li et al. [47] | 0.85 | 0.95 | 1.17 | 14.08 | 0.59 | 6.25 | 2.95 | 1.32 | 1.50 | 3.30 | 6.03 |
| 3DMatch (Zeng et al. [43]) | 0.00 | 5.67 | 2.86 | 21.25 | 2.41 | 10.91 | 6.98 | 3.62 | 4.65 | 6.48 | 10.29 |
| Scan2CAD (Avetisyan et al. [1]) | 36.20 | 36.40 | 34.00 | 44.26 | 17.89 | **70.63** | 30.66 | 30.11 | 20.60 | 35.64 | 31.68 |
| End2End (Avetisyan et al. [2]) | 38.89 | 41.46 | 51.52 | 73.04 | 26.53 | 26.83 | 76.92 | **48.15** | 18.18 | 44.61 | 50.72 |
| Ours (dense) | 33.33 | 39.39 | **58.62** | 70.76 | 28.57 | 33.72 | 50.00 | 34.55 | 23.73 | 41.41 | 51.05 |
| Ours (dense) + obj-obj | 44.44 | **54.55** | 49.15 | 68.05 | 37.50 | 36.05 | 61.11 | 42.01 | 27.12 | 46.66 | 52.97 |
| Ours (dense) + layout | **54.55** | 47.37 | 38.33 | 71.11 | 32.88 | 28.05 | 62.86 | 37.91 | **32.26** | 45.04 | 52.06 |
| Ours (dense) full | 39.39 | 42.11 | 48.33 | 74.32 | 42.47 | 36.59 | 62.86 | 36.26 | 30.65 | 45.89 | 54.33 |
| Ours (sparse) | 42.42 | 39.47 | 51.67 | 77.28 | 45.21 | 28.05 | 77.14 | 37.91 | 25.81 | 47.22 | 55.77 |
| Ours (sparse) + obj-obj | 42.42 | 44.74 | 50.00 | 77.53 | 43.84 | 30.49 | 74.29 | 39.56 | **32.26** | 48.35 | 56.70 |
| Ours (sparse) + layout | 45.45 | 42.11 | 48.33 | 78.27 | 42.47 | 31.71 | 77.14 | 37.36 | 27.42 | 47.81 | 56.29 |
| Ours (sparse) full | 42.42 | 36.84 | 58.33 | **81.23** | **50.68** | 40.24 | **82.86** | 45.60 | **32.26** | **52.27** | **61.24** |

**Table 5.1:** CAD alignment evaluation on ScanNet Scan2CAD data [5, 1]. Our final method (last row), incorporating contextual information from both object-object relationships and object-layout relationships, outperforms the baseline by a notable margin of $10.52\%$.

metric proposed by [1], which measures alignment accuracy where an alignment is considered successful if it falls within 20cm, $20^{\circ}$, and $20\%$ scale of the ground truth. On both SUNCG and ScanNet scans we compare to several state-of-the-art handcrafted geometric feature matching approaches [37, 38, 47] and learned approaches [43, 1, 2]. We additionally show qualitative comparisons in Figures Figure 5.6 and Figure 5.7. On synthetic scan data we outperform the strongest baseline by $16.58\%$, and improve by $10.52\%$ on real scan data. This demonstrates the benefit of leveraging global information regarding object and layout relations in improving object alignments.

We also perform an ablation study on the various design choices and impact of relation information. We evaluate a dense convolutional backbone for our network architecture (*dense*) in contrast to our final sparse convolutional backbone leveraging the sparse convolutions proposed by [116]. We additionally show that the object-to-object relational inference (*obj-obj*) as well as layout estimation (*layout*) improve upon no relational inference, and our full method incorporating both object and layout relational inference, the most contextual information, yields the best performance.

### 5.5.2 Layout Prediction

For the final quad prediction we achieve a F1-score of $37.9\%$ on ScanNet and $69.6\%$ on SUNCG. Corners are considered as successfully detected if the predicted corner is within a radius of $40cm$ from the ground truth corner. Edges are considered as correctly predicted if they connect the same corners as the ground truth edges. Similarly, correctly predicted quads are spanned by the same 4 corners as the associated ground truth quad. We aim to achieve a high recall for corners and edges due to our hierarchical prediction. We achieve robust results on both datasets, although ScanNet is notably more difficult as many scenes can miss views of entire layout components (e.g., missing ceilings).

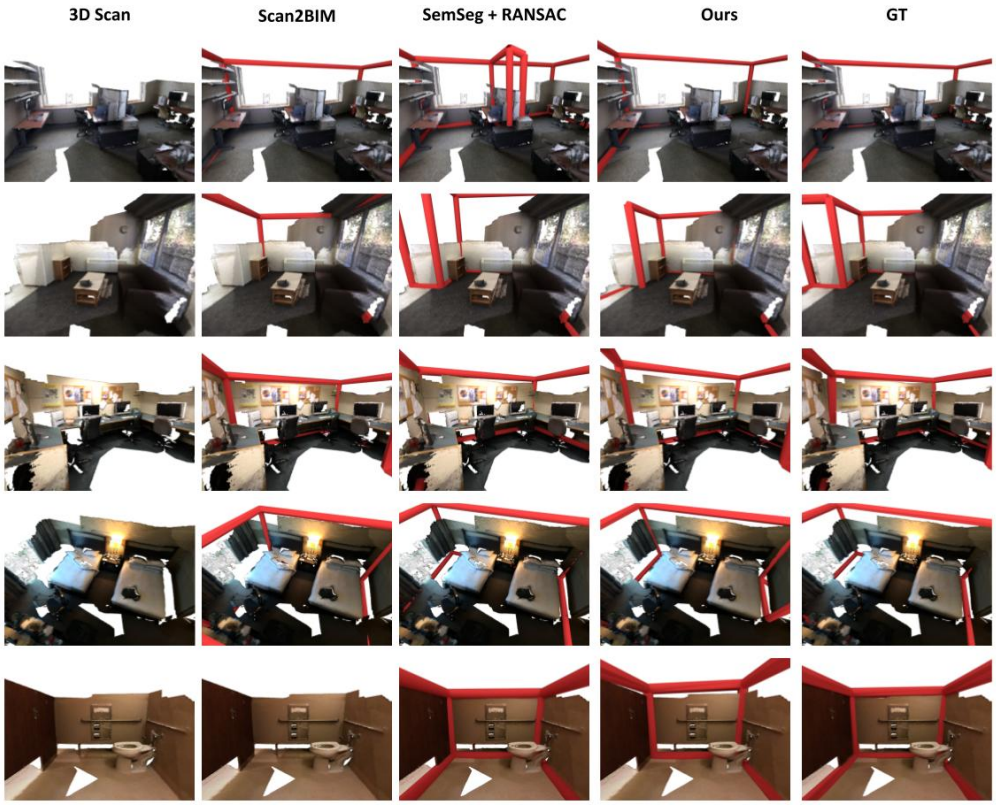



**Figure 5.4:** Qualitative comparison of our layout estimation on the ScanNet dataset [5]. Layout elements are highlighted with their wireframes. Our method provides a very lightweight, learned approach ($\approx 1M$ trainable parameters) for layout estimation.

## 5.6 Limitations

While the focus of this work was to show improved scene understanding through joint prediction of objects **and** layouts, we believe there is potential for further achievements. For instance, our layout prediction method is bound to predict quad planes only and hence more sophisticated methods could be used for more accurate layout estimation. Also, we used a very lightweight graph neural network for message passing. One could use a more sophisticated method for more accurate relationship prediction and a richer set of relationships that may contain functionality relationships, spatial relationships or room semantic relationships.

## 5.7 Conclusion

In this work we formulated a method to digitize 3D scans that goes beyond the focus of objects in the scene. We propose a novel method that estimates the layout of the scene by sequentially predicting corners, then edges and finally quads in a fully differentiable way. The estimated layout is used in conjunction with an object detector to predict contact relationships between

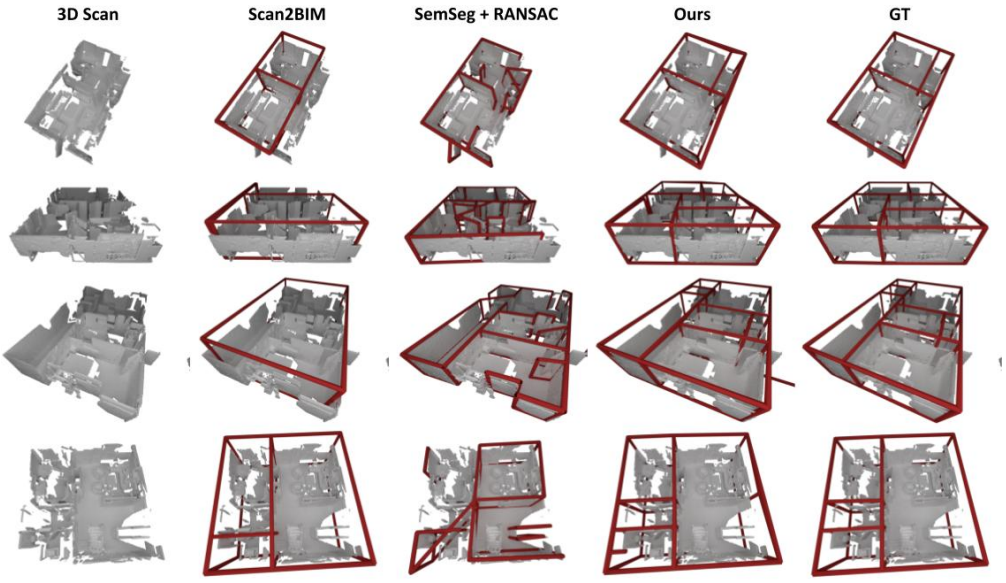



**Figure 5.5:** Layout estimation on SUNCG [122] scans. Layout elements are highlighted with their wireframes. Our method excels with its simplicity, especially for very large and complex scenes where heuristics to determine intersections tend to struggle.

| # voxels | 18K | 42K | 71K |
|---|---|---|---|
| Scene extent | $2.6m^2 \times 2.4m^2$ | $3.2m^2 \times 3.5m^2$ | $7.5m^2 \times 6.2m^2$ |
| # objects | 1 | 5 | 26 |
| Timing | **1.9s** | **2.0s** | **2.60s** |

**Table 5.2:** Runtime (seconds) of our approach on different test scenes categorized into small, medium and large.

| | bed | cabinet | chair | desk | dresser | other | shelves | sofa | table | class avg. | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| SHOT (Tombari et al. [38]) | 13.43 | 3.23 | 10.18 | 2.78 | 0.00 | 0.00 | 1.75 | 3.61 | 11.93 | 5.21 | 6.30 |
| FPFH (Rusu et al. [37]) | 38.81 | 3.23 | 7.64 | 11.11 | 3.85 | 13.21 | 0.00 | 21.69 | 11.93 | 12.39 | 9.94 |
| Scan2CAD (Avetisyan et al. [1]) | 52.24 | 17.97 | 36.00 | 30.56 | 3.85 | 20.75 | 7.89 | 40.96 | 43.12 | 28.15 | 29.23 |
| End2End (Avetisyan et al. [2]) | 71.64 | 32.72 | 48.73 | 27.78 | 38.46 | 37.74 | 14.04 | 67.47 | 45.87 | 42.72 | 41.83 |
| Ours (dense) | 63.89 | 35.16 | 56.82 | 39.02 | 30.00 | 38.85 | 29.17 | **76.67** | 31.03 | 44.51 | 44.48 |
| Ours (dense) + obj-obj | 77.78 | 36.26 | 53.03 | 41.46 | 40.00 | 47.48 | 20.83 | **76.67** | 25.86 | 46.60 | 46.41 |
| Ours (dense) + layout | 75.00 | 37.04 | 60.68 | 37.14 | 38.89 | 45.53 | **33.33** | 72.41 | 32.08 | 48.01 | 48.33 |
| Ours (dense) full | **81.25** | 40.00 | 51.92 | 45.45 | 41.18 | 49.17 | 31.58 | 75.86 | **46.00** | 51.38 | 50.41 |
| Ours (sparse) | 54.29 | 42.55 | 66.67 | 48.57 | **44.44** | 57.60 | 27.27 | 57.89 | 36.84 | 48.46 | 52.31 |
| Ours (sparse) + obj-obj | 74.29 | 40.43 | 70.09 | **65.71** | 27.78 | **60.80** | 27.27 | 55.26 | 38.60 | 51.14 | 55.27 |
| Ours (sparse) + layout | 65.71 | 42.55 | **77.78** | 54.29 | 38.89 | **60.80** | 22.73 | 57.89 | 45.61 | 51.81 | 57.12 |
| Ours (sparse) full | 71.43 | **43.62** | **77.78** | 54.29 | 38.89 | **60.80** | 22.73 | 68.42 | 45.61 | **53.73** | **58.41** |

**Table 5.3:** CAD alignment accuracy on SUNCG [122] scans. Our final method (last row) goes beyond considering only objects and jointly estimates room layout and object and layout relationships, resulting in significantly improved performance.

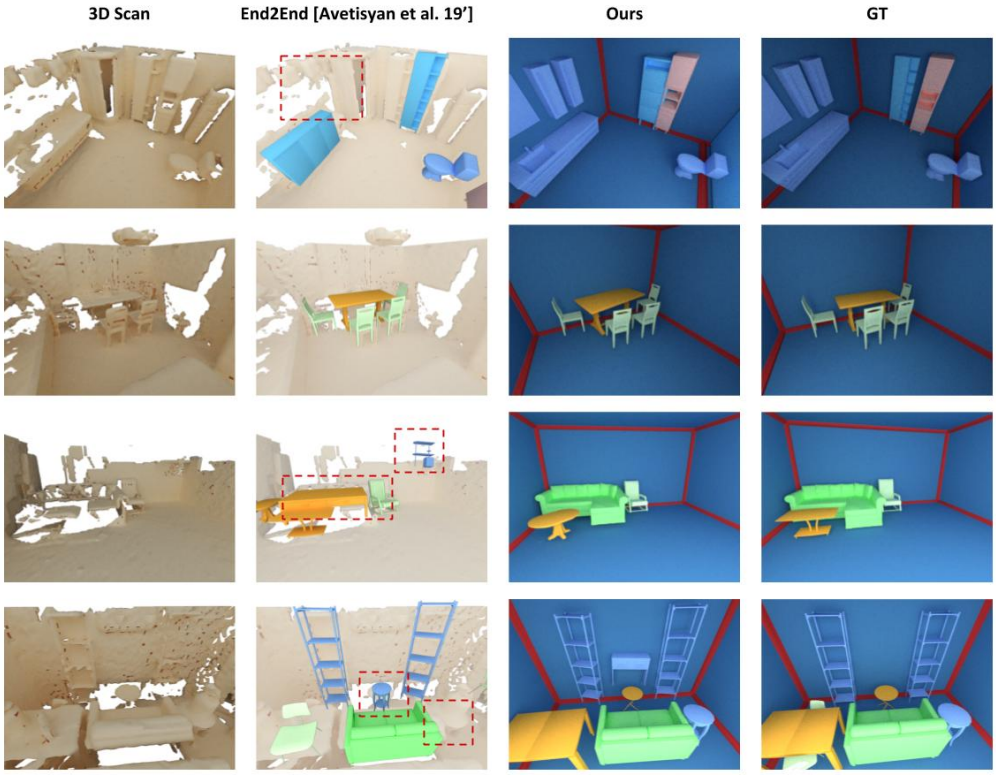



**Figure 5.6:** Qualitative CAD alignment and layout estimation results on SUNCG [122] scans. Our joint estimation approach produces more globally consistent CAD alignments and generates additionally room layout applicable for VR/AR applications.

objects and the layout and ultimately to predict a CAD arrangement of the scene. We can show that objects and the surrounding (scene layout) go hand in hand and are a crucial factor towards full scene digitization and scene understanding. Objects in the scene are often not arbitrarily arranged, for instance often cabinets are leaned at walls or a table is surrounded by chairs in a dining room, hence we leverage the inherent coupling between objects and layout structure in the learning process. Our approach improves global CAD alignment accuracy by learning those patterns on both real and synthetic scans. We hope that we can encourage further research towards this avenue, and see as next immediate steps for future work the necessity of texturing digitized shapes in order to enhance the immersive experience in VR environments.

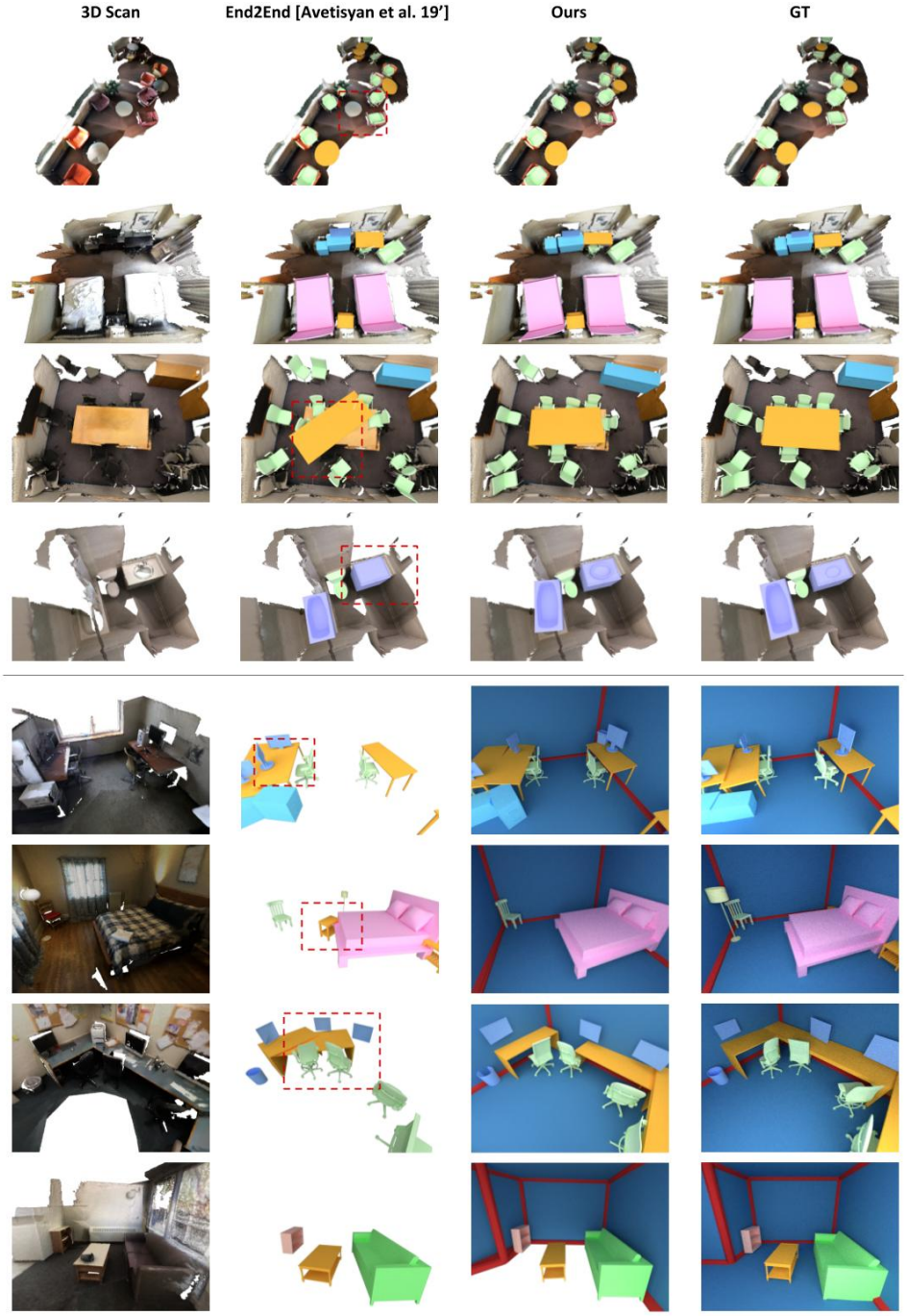



**Figure 5.7:** Qualitative CAD alignment and layout estimation results on ScanNet [5] scans (zoomed in views on the bottom). Our approach incorporating object and layout relationships produces globally consistent alignments along with the room layout.

## 5.8 Appendix

### 5.8.1 Dataset

In this supplemental document, we provide additional details for our layout dataset that is used for training. Figure Figure 5.8 shows an illustration of the real-world layout annotations comprising of more than 1000 individual ScanNet [5] scenes. In addition to layouts from real-world scans, we also extract layouts from the synthetic SUNCG dataset [122]; see Figure Figure 5.9. From these dataset annotations, we create ground truth targets. as visualized in Figure Figure 5.10. for our hierarchical layout estimation training.

In Figure Figure 5.11, we illustrate visually the different kinds of object-to-layout relationship classes. Note that an object can have relationships with multiple layout elements.

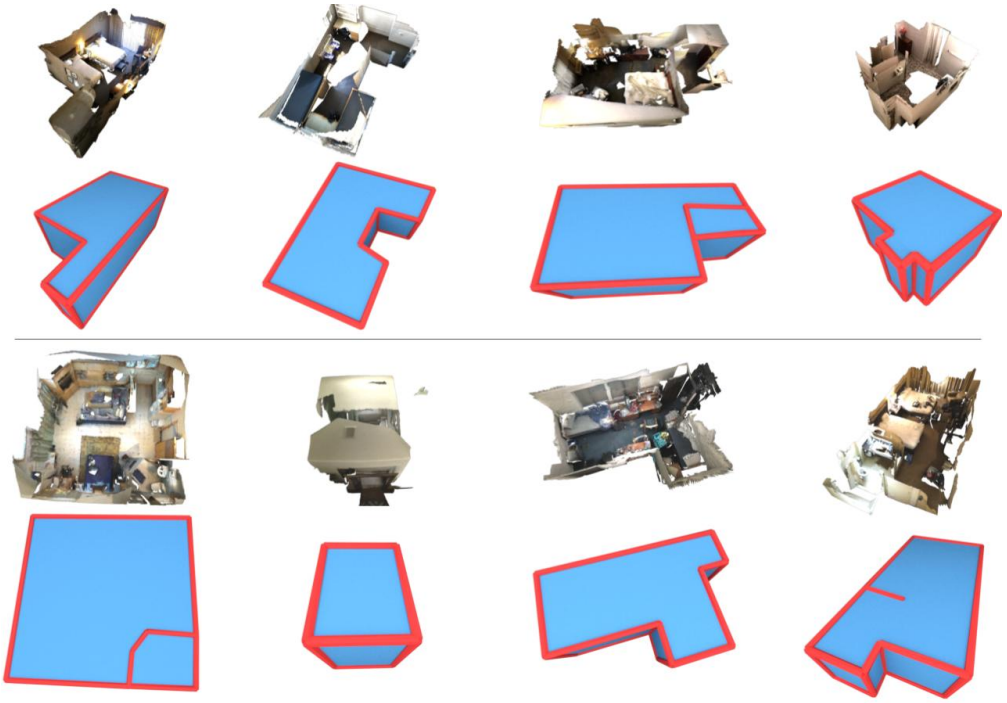


**Figure 5.8:** Samples of manually annotated layouts on ScanNet [5]. Annotations include wireframes and room-level CAD shells of the scenes representing walls, floors, and ceilings.

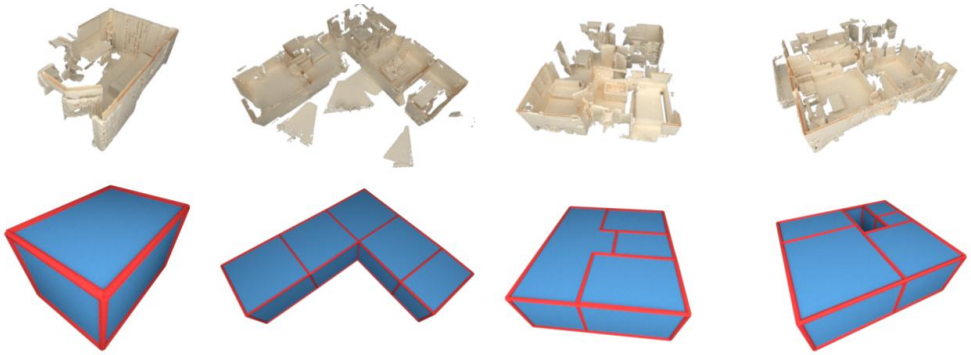


**Figure 5.9:** Samples of automatically parsed layouts from SUNCG [122]. Layouts include wireframes and room-level CAD shells.

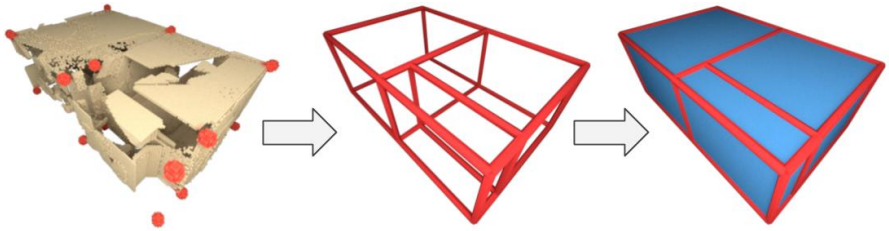


**Figure 5.10:** Sample targets of the layout estimation. The pipeline starts with a corner point estimation (left). Then, valid edges are estimated from the detected corners producing a wireframe (middle). Finally, valid layout quads are predicted from the edge candidates.

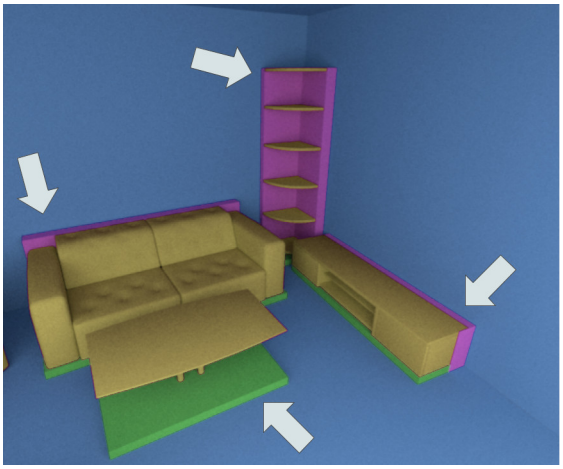


**Figure 5.11:** Groundtruth sample from the dataset. Green tile correspond to *vertical support* relationship and pink tile correspond to *horizontal touch* relationship.

# 6 Limitations

The various approaches in our works contributed to a significant leap towards the digitization of 3D reconstructed scenes. We demonstrated capabilities to parse a 3D reconstructed and replace its object with high-quality CAD models. While the CAD model alignment was aimed and evaluated with emphasis on alignment accuracy, the retrieval performance was treated looser. That is, a retrieved CAD model was considered to be correct when its high-level class label matched the class label of the ground truth annotation. This decision was made because it was often difficult or impossible to find a CAD model that exactly matched the scanned object. However, this looser evaluation metric for retrieval may result in a lower perceived quality of the CAD model alignments.

As previously noted, CAD models offer a great deal of flexibility and compatibility for physically realistic materials. In our work, we removed the texture and color data from the CAD models and only used the geometry for further training. While this design choice allowed for simplifications and reduced complexity in our methodologies, it also had a drawback. It is clear that neural networks have the ability to use additional input data to improve performance. Distinctive textures that many CAD models have, could provide extra information that leads to more accurate correspondence predictions or better retrievals based on meaningful visual similarity. Not only could materials enhance correspondence training, but they could also create more impressive renderings for the final CAD arrangement. We discussed the extensive exploration of digitizing scanned geometry in this field. However, our current method does not allow for digitizing the appearance of 3D scans, which is a significant limitation. Digitizing 2D content, such as book pages, can be done with high quality using a regular page scanner and an Optical Character Recognition (OCR) process. However, digitizing both the geometry and appearance of 3D scans, such as creating a mesh with realistic materials, remains a challenging task.

In this work, the classification and segmentation of objects have not been a primary focus. However, it is an important aspect with significant consequences. In essence, most objects are composed of multiple sub-parts. For example, a chair is made up of legs, a seat, arms, and a backrest, but is typically detected and classified as a single entity. Our approach detects and aligns CAD models to high-level instances without distinguishing between their parts. Although the search space for part-based alignment is larger, it also offers several benefits. For instance, a cabinet in both its open and closed states can be reconstructed through part-based alignment, whereas in a conventional whole entity-based formulation, separate CAD models would be required for each state. Additionally, objects with moving parts, such as joints and hinges, as well as objects undergoing non-rigid deformations such as pillows and blankets, pose challenges for our current formulation. Our CAD model repository and alignment algorithm are not equipped to handle these non-rigid correspondences, making it a limitation that still needs to be addressed.

# 7 Conclusion & Future Work

With Scan2CAD, we established a foundation for data-driven techniques using sparse keypoint correspondences to replace scanned objects with high-quality CAD models. We created a large-scale dataset with CAD-to-scan annotations and trained a 3D CNN to predict correspondences between different domains. The CNN predicts a heatmap on the CAD model for a query keypoint on the scan, indicating which voxel is most likely to be the true point association. Even with variations in low-level geometry, noise, and symmetry ambiguity, we can predict robust correspondences. In the second stage, an LM optimizer uses the correspondences to optimize the translation, rotation, and scale parameters, resulting in accurate pose estimation for each CAD.

Scan2CAD is a successful first approach for aligning CAD models, however, it has some limitations. It is slow because it fits every CAD model to all query points in the scene, without using sophisticated segmentation methods to reduce the number of candidate query points. Additionally, it lacks an object retrieval method, making the brute-force approach necessary. In our second work, we aim to improve the run-time and accuracy by introducing a 3D backbone with an object detection module that can parse the scene as a whole without extensive pre-processing. For each detected object, an object descriptor is calculated to retrieve the most similar CAD model counterpart with a nearest-neighbor distance metric. We also introduce dense correspondences between every voxel on the scan surface and the normalized volume of the CAD model, and use a differentiable alignment loss function to train the retrieval and alignment in an end-to-end fashion. These changes improve the run-time by two orders of magnitude and increase CAD alignment accuracy by a significant margin.

In SceneCAD, our latest work, we aim to address alignment issues caused by uncontextualized predictions, such as aligning a bathtub onto a sofa. Additionally, to achieve more complete digitization, we predict a layout estimate which is used to refine final CAD pose estimates. SceneCAD uses a graph neural network to learn relationships between objects and layout to predict globally consistent CAD model alignments. We also developed a novel and lightweight 3D layout prediction method that sequentially predicts scene corners, edges between corners, resulting in a 3D wireframe, and finally planes that represent structural elements. By learning object relationships and injecting context from structural elements by explicitly modeling the scene layout, we can considerably improve CAD alignment accuracy compared to state-of-the-art methods.

There are limitations and challenges to achieving sophisticated scene digitization as discussed in chapter 6. A promising approach to resolving this is by transferring materials and textures from real-world scanned objects to CAD models which would enhance both the geometry and quality of the CAD models. Additionally, for VR applications, the CAD scene representations should also

include the appearance information from the original scans. A possible solution is to transfer material information from camera observations to generate texture and UV maps for CAD surfaces. This would also address issues with baked lighting and other effects that vary with viewing angle in 3D reconstructions. Similarly, estimating lighting information would enable the digitized scene to be relit with the same lighting as in the real-world scene. However, separating the surface material information from lighting information is a major challenge and would require estimating both.

Our work primarily focuses on static and rigid objects, but there are situations in the real world where more flexibility is needed to digitize semi-rigid objects with moving parts (such as drawers, office chairs, or laptops). One solution could be to use parametric CAD models that have a range of degrees of freedom that allow for the manipulation of their shape. These models can change shape based on shape parameters, for example, the lid state of a laptop can be controlled with a single parameter for better alignment with scanned objects. Additionally, using integer parameters allows for discrete shape changes, such as controlling the number of legs. While this approach makes the CAD alignment task more complex, it offers benefits such as compact models in the CAD database and less stringent requirements for exact matches between scanned objects and CAD models. In the extreme case, a single parametric model per class could be developed that can assume a wide range of shapes with a small number of parameters.

# Bibliography


[1] A. Avetisyan, M. Dahnert, A. Dai, M. Savva, A. X. Chang, and M. Nießner. Scan2cad: Learning cad model alignment in rgb-d scans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2614–2623, 2019.

[2] A. Avetisyan, A. Dai, and M. Nießner. End-to-end cad model retrieval and 9dof alignment in 3d scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2551–2560, 2019.

[3] A. Avetisyan, T. Khanova, C. Choy, D. Dash, A. Dai, and M. Nießner. Scenecad: Predicting object alignments and layouts in rgb-d scans. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 596–612. Springer, 2020.

[4] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. ShapeNet: An Information-Rich 3D Model Repository. *arXiv preprint arXiv:1512.03012*, 2015.

[5] 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.

[6] M. Aharchi and M. Ait Kbir. A review on 3d reconstruction techniques from 2d images. In *The Proceedings of the Third International Conference on Smart City Applications*, pages 510–522. Springer, 2020.

[7] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.

[8] J. Batlle, E. Mouaddib, and J. Salvi. Recent progress in coded structured light as a technique to solve the correspondence problem: a survey. *Pattern recognition*, 31(7):963–982, 1998.

[9] D. Fofi, T. Sliwa, and Y. Voisin. A comparative survey on invisible structured light. In *Machine vision applications in industrial inspection XII*, volume 5303, pages 90–98. SPIE, 2004.

[10] M. Zollhöfer, P. Stotko, A. Görlitz, C. Theobalt, M. Nießner, R. Klein, and A. Kolb. State of the art on 3d reconstruction with rgb-d cameras. In *Computer graphics forum*, volume 37, pages 625–652. Wiley Online Library, 2018.

[11] R. Horaud, M. Hansard, G. Evangelidis, and C. Ménier. An overview of depth cameras and range scanners based on time-of-flight technologies. *Machine vision and applications*, 27(7):1005–1020, 2016.

[12] S. Bi, C. Yuan, C. Liu, J. Cheng, W. Wang, and Y. Cai. A survey of low-cost 3d laser scanning technology. *Applied Sciences*, 11(9):3938, 2021.

[13] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312. ACM, 1996.

[14] 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.

[15] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011.

[16] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (TOG)*, 2013.

[17] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous robots*, 34(3):189–206, 2013.

[18] J. Chen, D. Bautembach, and S. Izadi. Scalable real-time volumetric surface reconstruction. *ACM Transactions on Graphics (TOG)*, 32(4):113, 2013.

[19] M. Keller, D. Lefloch, M. Lambers, S. Izadi, T. Weyrich, and A. Kolb. Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *3D Vision-3DV 2013, 2013 International Conference on*, pages 1–8. IEEE, 2013.

[20] T. Whelan, S. Leutenegger, R. F. Salas-Moreno, B. Glocker, and A. J. Davison. Elasticfusion: Dense slam without a pose graph. *Proc. Robotics: Science and Systems, Rome, Italy*, 2015.

[21] R. Maier, J. Sturm, and D. Cremers. Submap-based bundle adjustment for 3d reconstruction from rgb-d data. In *German Conference on Pattern Recognition*, pages 54–65. Springer, 2014.

[22] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (TOG)*, 36(3):24, 2017.

[23] K. Guo, F. Xu, T. Yu, X. Liu, Q. Dai, and Y. Liu. Real-time geometry, albedo, and motion reconstruction using a single rgb-d camera. *ACM Transactions on Graphics (ToG)*, 36(4):1, 2017.

[24] C. Wu, K. Varanasi, Y. Liu, H.-P. Seidel, and C. Theobalt. Shading-based dynamic shape refinement from multi-view video under general illumination. In *2011 International Conference on Computer Vision*, pages 1108–1115. IEEE, 2011.

[25] Augmented reality ipad demo. https://www.apple.com/uk/augmented-reality/. Accessed: 2021-11-26 8PM.

[26] S. J. Schoonmaker. *The CAD guidebook: A basic manual for understanding and improving computer-aided design*. CRC Press, 2002.

[27] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, et al. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

[28] A. Munawar, Y. Wang, R. Gondokaryono, and G. S. Fischer. A real-time dynamic simulator and an associated front-end representation format for simulating complex robots and environments. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1875–1882. IEEE, 2019.

[29] B. Burley and W. D. A. Studios. Physically-based shading at disney. In *ACM SIGGRAPH*, volume 2012, pages 1–7. vol. 2012, 2012.

[30] Wooden pbr material. https://cgaxis.com/product/elm-belved-floor-pbr-texture-2/. Accessed: 2021-11-15 6PM.

[31] 3d model of an armchair. https://www.cgtrader.com/3d-models/furniture/chair/chesterfield-traditional-tufted-classic-armchair. Accessed: 2021-11-08 6PM.

[32] A. Das, S. Datta, G. Gkioxari, S. Lee, D. Parikh, and D. Batra. Embodied question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–10, 2018.

[33] D. Ritchie, K. Wang, and Y.-a. Lin. Fast and flexible indoor scene synthesis via deep convolutional generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6182–6190, 2019.

[34] S.-H. Zhang, S.-K. Zhang, Y. Liang, and P. Hall. A survey of 3d indoor scene synthesis. *Journal of Computer Science and Technology*, 34(3):594–608, 2019.

[35] S. Qi, Y. Zhu, S. Huang, C. Jiang, and S.-C. Zhu. Human-centric indoor scene synthesis using stochastic grammar. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5899–5908, 2018.

[36] S. Koch, A. Matveev, Z. Jiang, F. Williams, A. Artemov, E. Burnaev, M. Alexa, D. Zorin, and D. Panozzo. Abc: A big cad model dataset for geometric deep learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[37] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *ICRA*, pages 3212–3217. IEEE, 2009.

[38] F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. In *European conference on computer vision*, pages 356–369. Springer, 2010.

[39] B. Drost, M. Ulrich, N. Navab, and S. Ilic. Model globally, match locally: Efficient and robust 3d object recognition. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 998–1005. Ieee, 2010.

[40] B. Drost and S. Ilic. 3d object detection and localization using multimodal point pair features. In *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, pages 9–16. IEEE, 2012.

[41] Y. M. Kim, N. J. Mitra, D.-M. Yan, and L. Guibas. Acquiring 3d indoor environments with variability and repetition. *ACM Transactions on Graphics (TOG)*, 31(6):138, 2012.

[42] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1352–1359, 2013.

[43] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. 3DMatch: Learning local geometric descriptors from RGB-D reconstructions. In *CVPR*, 2017.

[44] K. Chen, Y.-K. Lai, and S.-M. Hu. 3d indoor scene modeling from rgb-d data: a survey. *Computational Visual Media*, 1(4):267–278, 2015.

[45] L. Nan, K. Xie, and A. Sharf. A search-classify approach for cluttered indoor scene understanding. *ACM Transactions on Graphics (TOG)*, 31(6):137, 2012.

[46] T. Shao, W. Xu, K. Zhou, J. Wang, D. Li, and B. Guo. An interactive approach to semantic modeling of indoor scenes with an rgbd camera. *ACM Transactions on Graphics (TOG)*, 31(6):136, 2012.

[47] Y. Li, A. Dai, L. Guibas, and M. Nießner. Database-assisted object retrieval for real-time 3d reconstruction. In *Computer Graphics Forum*, volume 34. Wiley Online Library, 2015.

[48] 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*, 24(6):381–395, 1981.

[49] W. Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976.

[50] B. S. Hua, Q. T. Truong, M. K. Tran, Q. H. Pham, A. Kanezaki, T. Lee, H. Y. Chiang, W. Hsu, B. Li, Y. Lu, et al. Shrec'17: Rgb-d to cad retrieval with objectnn dataset. In *10th Eurographics Workshop on 3D Object Retrieval, 3DOR 2017*, pages 25–32. Eurographics Association, 2017.

[51] 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. Shrec'18: Rgb-d object-to-cad retrieval. *Proc. 3DOR*, 2, 2018.

[52] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, volume 22, pages 223–232. Wiley Online Library, 2003.

[53] M. Dahnert, A. Dai, L. J. Guibas, and M. Nießner. Joint embedding of 3d scan and cad objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8749–8758, 2019.

[54] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Aligning 3d models to rgb-d images of cluttered scenes. In *CVPR*, pages 4731–4740, 2015.

[55] X. Sun, J. Wu, X. Zhang, Z. Zhang, C. Zhang, T. Xue, J. B. Tenenbaum, and W. T. Freeman. Pix3D: Dataset and methods for single-image 3D shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2974–2983, 2018.

[56] G. Georgakis, S. Karanam, Z. Wu, and J. Kosecka. Learning local rgb-to-cad correspondences for object pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8967–8976, 2019.

[57] J. Hou, A. Dai, and M. Nießner. 3D-SIS: 3D Semantic Instance Segmentation of RGB-D Scans. In *CVPR*, 2019.

[58] C. R. Qi, O. Litany, K. He, and L. J. Guibas. Deep hough voting for 3d object detection in point clouds. *arXiv preprint arXiv:1904.09664*, 2019.

[59] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5105–5114, 2017.

[60] J. Lahoud, B. Ghanem, M. Pollefeys, and M. R. Oswald. 3d instance segmentation via multi-task metric learning. In *ICCV*, 2019.

[61] L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, and J. Jia. PointGroup: Dual-Set Point Grouping for 3D Instance Segmentation. In *CVPR*, 2020.

[62] B. Graham, M. Engelcke, and L. van der Maaten. 3D semantic segmentation with submanifold sparse convolutional networks. In *CVPR*, 2018.

[63] L. Roldao, R. De Charette, and A. Verroust-Blondet. 3d semantic scene completion: a survey. *International Journal of Computer Vision*, pages 1–28, 2022.

[64] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser. Semantic scene completion from a single depth image. *Proceedings of 30th IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[65] Y.-T. Chen, M. Garbade, and J. Gall. 3d semantic scene completion from a single depth image using adversarial training. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 1835–1839. IEEE, 2019.

[66] A. Dai, D. Ritchie, M. Bokeloh, S. Reed, J. Sturm, and M. Nießner. Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans. *arXiv preprint arXiv:1712.10215*, 2018.

[67] C. B. Rist, D. Emmerichs, M. Enzweiler, and D. M. Gavrila. Semantic scene completion using local deep implicit functions on lidar data. *IEEE transactions on pattern analysis and machine intelligence*, 44(10):7205–7218, 2021.

[68] P.-S. Wang, Y. Liu, and X. Tong. Deep octree-based cnns with output-guided skip connections for 3d shape and scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 266–267, 2020.

[69] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

[70] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568. ACM, 2011.

[71] O. Kähler, V. A. Prisacariu, C. Y. Ren, X. Sun, P. Torr, and D. Murray. Very high frame rate volumetric integration of depth images on mobile devices. *IEEE transactions on visualization and computer graphics*, 21(11):1241–1250, 2015.

[72] S. Choi, Q.-Y. Zhou, and V. Koltun. Robust reconstruction of indoor scenes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5556–5565. IEEE, 2015.

[73] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang. Matterport3D: Learning from RGB-D data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017.

[74] A. Dai, C. R. Qi, and M. Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[75] 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.

[76] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger. Fusion++: Volumetric object-level slam. In *2018 International Conference on 3D Vision (3DV)*, pages 32–41. IEEE, 2018.

[77] A. E. Johnson. Spin-images: a representation for 3-d surface matching. 1997.

[78] Y. M. Kim, N. J. Mitra, Q. Huang, and L. Guibas. Guided real-time scanning of indoor objects. In *Computer Graphics Forum*, volume 32, pages 177–186. Wiley Online Library, 2013.

[79] X. Zhou, A. Karpur, C. Gan, L. Luo, and Q. Huang. Unsupervised domain adaptation for 3d keypoint estimation via view consistency. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 137–153, 2018.

[80] J. J. Lim, H. Pirsiavash, and A. Torralba. Parsing ikea objects: Fine pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2992–2999, 2013.

[81] H. Izadinia, Q. Shan, and S. M. Seitz. Im2cad. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 2422–2431. IEEE, 2017.

[82] S. Huang, S. Qi, Y. Zhu, Y. Xiao, Y. Xu, and S.-C. Zhu. Holistic 3D scene parsing and reconstruction from a single RGB image. In *European Conference on Computer Vision*, pages 194–211. Springer, 2018.

[83] C. Zou, R. Guo, Z. Li, and D. Hoiem. Complete 3D scene parsing from an RGBD image. *International Journal of Computer Vision (IJCV)*, 2018.

[84] M. Savva, F. Yu, H. Su, M. Aono, B. Chen, D. Cohen-Or, W. Deng, H. Su, S. Bai, X. Bai, et al. Shrec'16 track large-scale 3d shape retrieval from shapenet core55.

[85] B.-S. Hua, Q.-H. Pham, D. T. Nguyen, M.-K. Tran, L.-F. Yu, and S.-K. Yeung. Scenenn: A scene meshes dataset with annotations. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 92–101. IEEE, 2016.

[86] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on*, pages 75–82. IEEE, 2014.

[87] Y. Xiang, W. Kim, W. Chen, J. Ji, C. Choy, H. Su, R. Mottaghi, L. Guibas, and S. Savarese. Objectnet3d: A large scale database for 3D object recognition. In *European Conference on Computer Vision*, pages 160–176. Springer, 2016.

[88] N. Hansen, S. D. Müller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary computation*, 11(1):1–18, 2003.

[89] N. Hansen. Benchmarking a bi-population cma-es on the bbob-2009 function testbed. In *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, pages 2389–2396. ACM, 2009.

[90] C. Batty. Sdfgen. https://github.com/christopherbatty/SDFGen.

[91] N. Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

[92] R. B. Rusu and S. Cousins. 3d is here: Point cloud library (pcl). In *Robotics and automation (ICRA), 2011 IEEE International Conference on*, pages 1–4. IEEE, 2011.

[93] F. Tombari, S. Salti, and L. Di Stefano. A combined texture-shape descriptor for enhanced 3d feature matching. In *2011 18th IEEE International Conference on Image Processing*, pages 809–812. IEEE, 2011.

[94] H. Deng, T. Birdal, and S. Ilic. Ppf-foldnet: Unsupervised learning of rotation invariant 3D local descriptors. In *ECCV*, 2018.

[95] H. Deng, T. Birdal, and S. Ilic. 3d local features for direct pairwise registration. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019.

[96] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother. Dsac-differentiable ransac for camera localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6684–6692, 2017.

[97] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[98] G. Papandreou, T. Zhu, N. Kanazawa, A. Toshev, J. Tompson, C. Bregler, and K. Murphy. Towards accurate multi-person pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4903–4911, 2017.

[99] 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. *arXiv preprint arXiv:1901.02970*, 2019.

[100] C. Liu, K. Kim, J. Gu, Y. Furukawa, and J. Kautz. Planercnn: 3d plane detection and reconstruction from a single image. *arXiv preprint arXiv:1812.04072*, 2018.

[101] S. Murali, P. Speciale, M. R. Oswald, and M. Pollefeys. Indoor scan2bim: Building information models of house interiors. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6126–6133. IEEE, 2017.

[102] C. Liu, J. Wu, and Y. Furukawa. Floornet: A unified framework for floorplan reconstruction from 3d scans. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 201–217, 2018.

[103] J. Chen, C. Liu, J. Wu, and Y. Furukawa. Floor-sp: Inverse cad for floorplans by sequential room-wise shortest path. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2661–2670, 2019.

[104] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black. Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In *European Conference on Computer Vision*, pages 561–578. Springer, 2016.

[105] F. Engelmann, J. Stückler, and B. Leibe. Joint object pose estimation and shape reconstruction in urban street scenes using 3d shape priors. In *German Conference on Pattern Recognition*, pages 219–230. Springer, 2016.

[106] S. Zakharov, I. Shugurov, and S. Ilic. Dpod: 6d pose object detector and refiner. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1941–1950, 2019.

[107] J. Wald, A. Avetisyan, N. Navab, F. Tombari, and M. Nießner. Rio: 3d object instance re-localization in changing indoor environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7658–7667, 2019.

[108] Y. Xiang, W. Kim, W. Chen, J. Ji, C. B. Choy, H. Su, R. Mottaghi, L. J. Guibas, and S. Savarese. Objectnet3d: A large scale database for 3d object recognition. In *ECCV*, 2016.

[109] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[110] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018.

[111] A. Dai and M. Nießner. Scan2mesh: From unstructured range scans to 3d meshes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5574–5583, 2019.

[112] N. Kulkarni, I. Misra, S. Tulsiani, and A. Gupta. 3d-relnet: Joint object and relational network for 3d prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2212–2221, 2019.

[113] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5664–5673, 2019.

[114] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa. Planenet: Piece-wise planar reconstruction from a single rgb image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2579–2588, 2018.

[115] C. Zou, R. Guo, Z. Li, and D. Hoiem. Complete 3d scene parsing from an rgbd image. *International Journal of Computer Vision*, 127(2):143–162, 2019.

[116] C. Choy, J. Gwak, and S. Savarese. 4D spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, 2019.

[117] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *ECCV*, pages 21–37. Springer, 2016.

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

[119] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural message passing for quantum chemistry. *arXiv preprint arXiv:1704.01212*, 2017.

[120] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel. Neural relational inference for interacting systems. *arXiv preprint arXiv:1802.04687*, 2018.

[121] 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)*, 38(5):1–12, 2019.

[122] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser. Semantic scene completion from a single depth image. 2017.

# Appendix

# Original Publications

## Scan2CAD: Learning CAD Model Alignment in RGB-D Scans

Armen Avetisyan[1], Manuel Dahnert[1], Angela Dai[1]
Manolis Savva[2], Angel X. Chang[2], Matthias Nießner[1]

[1]Technical University of Munich [2]Simon Fraser University

This is the accepted but not the published version of the paper due to copyright restrictions.
Published version: https://doi.org/10.1109/CVPR.2019.00272

# Scan2CAD: Learning CAD Model Alignment in RGB-D Scans

Armen Avetisyan[1] Manuel Dahnert[1] Angela Dai[1] Manolis Savva[2]
Angel X. Chang[2] Matthias Nießner[1]

[1]Technical University of Munich [2]Simon Fraser University

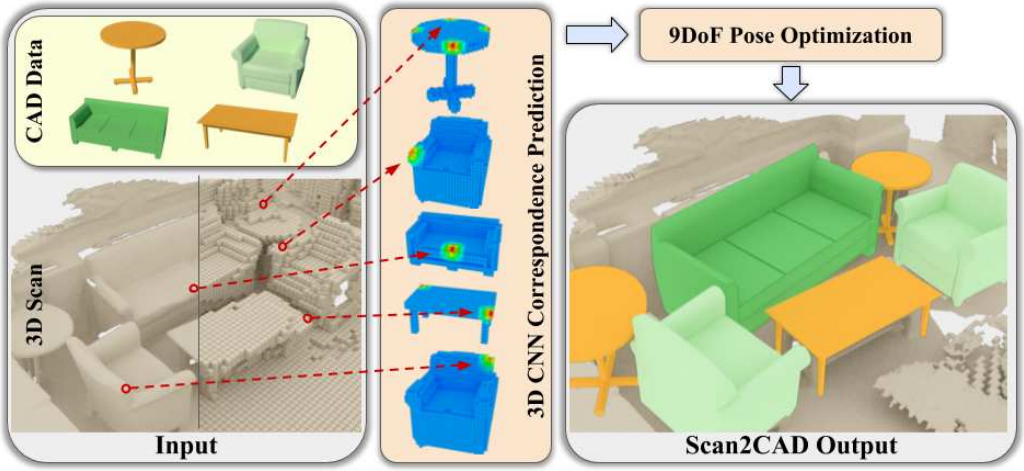



Figure 1: Scan2CAD takes as input an RGB-D scan and a set of 3D CAD models (left). We then propose a novel 3D CNN approach to predict heatmap correspondences between the scan and the CAD models (middle). From these predictions, we formulate an energy minimization to find optimal 9 DoF object poses for CAD model alignment to the scan (right).

## Abstract

*We present Scan2CAD*[1]*, a novel data-driven method that learns to align clean 3D CAD models from a shape database to the noisy and incomplete geometry of an RGB-D scan. For a 3D reconstruction of an indoor scene, our method takes as input a set of CAD models, and predicts a 9DoF pose that aligns each model to the underlying scan geometry. To tackle this problem, we create a new scan-to-CAD alignment dataset based on 1506 ScanNet scans with 97607 annotated keypoint pairs between 14225 CAD models from ShapeNet and their counterpart objects in the scans. Our method selects a set of representative keypoints in a 3D scan for which we find correspondences to the CAD geometry. To this end, we design a novel 3D CNN architecture to learn a joint embedding between real and synthetic objects, and thus predict a correspondence heatmaps. Based on these correspondence heatmaps, we formulate a variational energy minimization that aligns a given set of CAD models to the reconstruction. We evaluate our approach on our newly introduced Scan2CAD benchmark where we outperform both handcrafted feature descriptor as well as state-of-the-art CNN based methods by* 21.39%.

[1]The Scan2CAD dataset is publicly released along with an automated benchmark script for testing under www.Scan2CAD.org

## 1. Introduction

In recent years, the wide availability of consumer-grade RGB-D sensors, such as the Microsoft Kinect, Intel Real Sense, or Google Tango, has led to significant progress in RGB-D reconstruction. We now have 3D reconstruction frameworks, often based on volumetric fusion [6], that achieve impressive reconstruction quality [18, 29, 30, 40, 21] and reliable global pose alignment [40, 5, 8]. At the same time, deep learning methods for 3D object classification and semantic segmentation have emerged as a primary consumer of large-scale annotated reconstruction datasets [7, 2]. These developments suggest great potential in the future of 3D digitization, for instance, in applications for virtual and augmented reality.

Despite these improvements in reconstruction quality, the geometric completeness and fine-scale detail of indoor scene reconstructions remain a fundamental limitation. In contrast to artist-created computer graphics models, 3D scans are noisy and incomplete, due to sensor noise, motion blur, and scanning patterns. Learning-based approaches for object and scene completion [9, 37, 10] cannot reliably recover sharp edges or planar surfaces, resulting in quality far from artist-modeled 3D content.

One direction to address this problem is to retrieve a set of CAD models from a shape database and align them to an input scan, in contrast to a bottom-up reconstruction of

the scene geometry. If all objects are replaced in this way, we obtain a clean and compact scene representation, precisely serving the requirements for many applications ranging from AR/VR scenarios to architectural design. Unfortunately, matching CAD models to scan geometry is an extremely challenging problem: While high-level geometric structures might be similar, the low-level geometric features differ significantly (e.g., surface normal distributions). This severely limits the applicability of handcrafted geometric features, such as FPFH [33], SHOT [35], point-pair-features [11], or SDF-based feature descriptors [25]. While learning-based approaches like random forests [28, 36] exist, their model capacity remains relatively low, especially in comparison to more modern methods based on deep learning, which can achieve significantly higher accuracy, but remain at their infancy. We believe this is in large part attributed to the lack of appropriate training data.

In this paper, we make the following contributions:

- We introduce the Scan2CAD dataset, a large-scale dataset comprising 97607 pairwise keypoint correspondences and 9DoF alignments between 14225 instances of 3049 unique synthetic models, between ShapeNet [3] and reconstructed scans in ScanNet [7], as well as oriented bounding boxes for each object.
- We propose a novel 3D CNN architecture that learns a joint embedding between real and synthetic 3D objects to predict accurate correspondence heatmaps between the two domains.
- We present a new variational optimization formulation to minimize the distance between scan keypoints and their correspondence heatmaps, thus obtaining robust 9DoF scan-to-CAD alignments.

## 2. Related work

**RGB-D Scanning and Reconstruction** The availability of low-cost RGB-D sensors has led to significant research progress in RGB-D 3D reconstruction. A very prominent line of research is based on volumetric fusion [6], where depth data is integrated in a volumetric signed distance function. Many modern real-time reconstruction methods, such as KinectFusion [18, 29], are based on this surface representation. In order to make the representation more memory-efficient, octree [4] or hash-based scene representations have been proposed [30, 21]. An alternative fusion approach is based on points [22]; the reconstruction quality is slightly lower, but it has more flexibility when handling scene dynamics and can be adapted on-the-fly for loop closures [40]. Very recent RGB-D reconstruction frameworks combine efficient scene representations with global pose estimation [5], and can even perform online updates with global loop closures [8]. A closely related direction to ours (and a possible application) is recognition of objects as a part of a SLAM method, and using the retrieved objects as part of a global pose graph optimization [34, 27].

**3D Features for Shape Alignment and Retrieval** Geometric features have a long-established history in computer vision, such as Spin Images [20], Fast Point Feature Histograms (FPFH) [33], or Point-Pair Features (PPF) [11]. Based on these descriptors or variations of them, researchers have developed shape retrieval and alignment methods. For instance, Kim et al. [24] learn a shape prior in the form of a deformable part model from input scans to find matches at test time; or AA2h [23] use a similar approach to PPF, where a histogram of normal distributions of sample points is used for retrieval. Li et al. [25] propose a formulation based on a hand-crafted TSDF feature descriptor to align CAD models in real-time to RGB-D scans. While these retrieval approaches based on hand-crafted geometric features show initial promise, they struggle to generalize matching between the differing data characteristics of clean CAD models and noisy, incomplete real-world data.

An alternative direction is learned geometric feature descriptors. For example, Nan et al. [28] use a random decision forest to classify objects on over-segmented input geometry from high-quality scans. Shao et al. [36] introduce a semi-automatic system to resolve segmentation ambiguities, where a user first segments a scene into semantic regions, and then shape retrieval is applied. 3DMatch [43] leverage a Siamese neural network to match keypoints in 3D scans for pose estimation. Zhou et al. [44] is of similar nature, proposing a view consistency loss for 3D keypoint prediction network on RGB-D image data. Inspired by such approaches, we develop a 3D CNN-based approach targeting correspondences between the synthetic domain of CAD models and the real domain of RGB-D scan data.

Other approaches retrieve and align CAD models given single RGB [26, 19, 38, 17] or RGB-D [12, 45] images. These methods are related, but our focus is on geometric alignment independent of RGB information, rather than CAD-to-image.

**Shape Retrieval Challenges and RGB-D Datasets** Shape retrieval challenges have recently been organized as part of the Eurographics 3DOR [16, 32]. Here, the task was formulated as matching of object instances from ScanNet [7] and SceneNN [15] to CAD models from the ShapeNetSem dataset [3]. Evaluation only considered binary in-category vs out-of-category (and sub-category) match as the notion of relevance. As such, this evaluation does not address the alignment quality between scan objects and CAD models, which is our focus.

ScanNet [7] provides aligned CAD models for a small subset of the annotated object instances (for only 200 objects out of the total 36000). Moreover, the alignment

quality is low with many object category mismatches and alignment errors, as the annotation task was performed by crowdsourcing. The PASCAL 3D+ [42] dataset annotates 13898 objects in the PASCAL VOC images with coarse 3D poses defined against representative CAD models. ObjectNet3D [41] provides a dataset of CAD models aligned to 2D images, approximately 200K object instances in 90K images. The IKEA objects [26] and Pix3D [38] datasets similarly provide alignments of a small set of identifiable CAD models to 2D images of the same objects in the real world; the former has 759 images annotated with 90 models, the latter has 10069 annotated with 395 models.

No existing dataset provides fine-grained object instance alignments at the scale of our Scan2CAD dataset with 14225 CAD models (3049 unique instances) annotated to their scan counterpart distributed on 1506 3D scans.

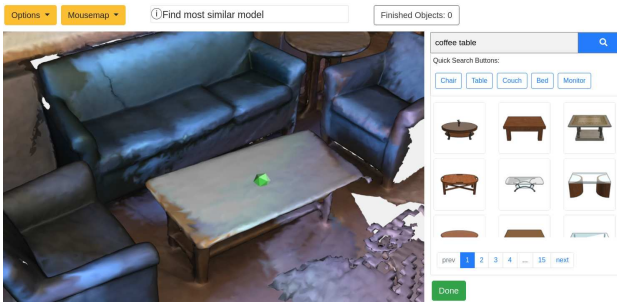
## 3. Overview

**Task** We address alignment between clean CAD models and noisy, incomplete 3D scans from RGB-D fusion, as illustrated in Fig. 1. Given a 3D scene $\mathbb{S}$ and a set of 3D CAD models $\mathbb{M} = \{m_i\}$, the goal is to find a 9DoF transformation $T_i$ (3 degrees for translation, rotation, and scale each) for every CAD model $m_i$ such that it aligns with a semantically matching object $\mathbb{O} = \{o_j\}$ in the scan. One important note is that we cannot guarantee the existence of 3D models which exactly matches the geometry of the scan objects.

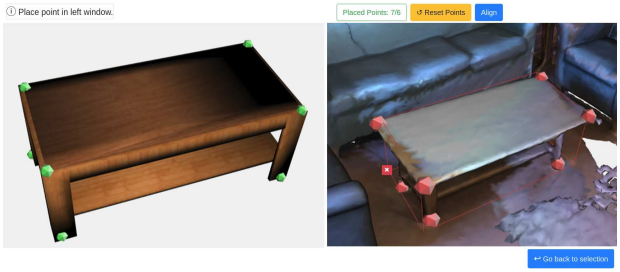
**Dataset and Benchmark** In Sec. 4, we introduce the construction of our Scan2CAD dataset. We propose an annotation pipeline designed for use by trained annotators. An annotator first inspects a 3D scan and selects a model from a CAD database that is geometrically similar to a target object in the scan. Then, for each model, the annotator defines corresponding keypoint pairs between the model and the object in the scan. From these keypoints, we compute ground truth 9DoF alignments. We annotate the entire ScanNet dataset and use the original training, validation, and test splits to establish our alignment benchmark.

**Heatmap Prediction Network** In Sec. 5, we propose a 3D CNN taking as input a volume around a candidate keypoint in a scan and a volumetric representation of a CAD model. The network is trained to predict a correspondence heatmap over the CAD volume, representing the likelihood that the input keypoint in the scan is matching with each voxel. The heatmap prediction is formulated as a classification problem, which is easier to train than regression, and produces sparse correspondences needed for pose optimization.

**Alignment Optimization** Sec. 6 describes our variational alignment optimization. To generate candidate correspondence points in the 3D scan, we detect Harris keypoints, and predict correspondence heatmaps for each Harris keypoint and CAD model. Using the predicted heatmaps we find optimal 9DoF transformations. False alignments are pruned via a geometric confidence metric.




(a) First step: Retrieval view.




(b) Second step: Alignment view.

Figure 2: Our annotation web interface is a two-step process. (a) After the user places an anchor on the scan surface, class-matching CAD models are displayed on the right. (b) Then the user annotates keypoint pairs between the scan and CAD model from which we derive the ground truth 9DoF transformation.

## 4. Dataset

Our Scan2CAD dataset builds upon the 3D scans from ScanNet [7] and CAD models from ShapeNet [3]. Each *scene* $\mathbb{S}$ contains multiple *objects* $\mathbb{O} = \{o_i\}$, where each *object* $o_i$ is matched with a ShapeNet CAD model $m_i$ and both share multiple keypoint pairs (correspondences) and one transformation matrix $T_i$ defining the alignment. Note that ShapeNet CAD models have a consistently defined front and upright orientation which induces an amodal tight oriented bounding box for each scan object, see Fig. 3.

### 4.1. Data Annotation

The annotation is done via a web application that allows for simple scaling and distribution of annotation jobs; see Fig. 2. The annotation process is separated into two steps. The first step is object *retrieval*, where the user clicks on a point on the 3D scan surface, implicitly determining an object category label from the ScanNet object instance annotations. We use the instance category label as query text in the ShapeNet database to retrieve and display all matching

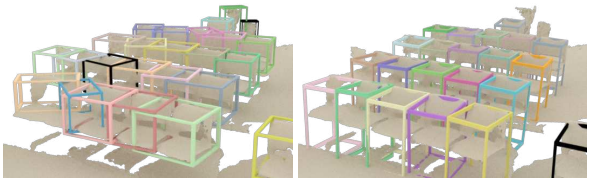


Figure 3: (Left) Oriented bounding boxes (OBBs) computed from the instance segmentation of ScanNet [7] are often incomplete due to missing geometry (e.g., in this case, missing chair legs). (Right) Our OBBs are derived from the aligned CAD models and are thus complete.

CAD models in a separate window as illustrated in Fig. 2a. After selecting a CAD model the user performs *alignment*.

In the alignment step, the user sees two separate windows in which the CAD model (left) and the scan object (right) are shown (see Fig. 2b). Keypoint correspondences are defined by alternately clicking paired points on the CAD model and scan object. We require users to specify at least 6 keypoint pairs to determine a robust ground truth transformation. After keypoint pairs are specified, the alignment computation is triggered by clicking a button. This alignment (given exact 1-to-1 correspondences) is solved with the genetic algorithm *CMA-ES* [14, 13] that minimizes the point-to-point distance over 9 parameters. In comparison to gradient-based methods or Procrustes superimposition method, we found this approach to perform significantly better in reliably returning high-quality alignments regardless of initialization.

The quality of these keypoint pairs and alignments was verified in several verification passes, with re-annotations performed to ensure a high quality of the dataset. The verification passes were conducted by the authors of this work.

A subset of the ShapeNet CAD models have symmetries that play an important role in making correspondences. Hence, we annotated all ShapeNet CAD models used in our dataset with their rotational symmetries to prevent false negatives in evaluations. We defined 2-fold ($C_2$), 4-fold ($C_4$) and infinite ($C_\infty$) rotational symmetries around a canonical axis of the object.

### 4.2. Dataset Statistics

The annotation process yielded 97607 keypoint pairs on 14225 (3049 unique) CAD models with their respective scan counterpart distributed on a total of 1506. Approximately 28% out of the 3049 CAD models have a symmetry tag (either $C_2$, $C_4$ or $C_\infty$).

Given the complexity of the task and to ensure high quality annotations, we employed 7 part-time annotators (in contrast to crowd-sourcing). On average, each scene has been edited 1.76 times throughout the re-annotation cycles. The top 3 annotated model classes are chairs, tables and cabinets which arises due to the nature of indoor scenes in ScanNet. The number of objects aligned per scene ranges from 1 to 40 with an average of 9.3. It took annotators on average of 2.48min to align each object, where the time to find an appropriate CAD model dominated the time for keypoint placement. The average annotation time for an entire scene is 20.52min.

It is interesting to note that manually placed keypoint correspondences between scans and CAD models differ significantly from those extracted from a Harris corner detector. Here, we compare the mean distance from the annotated CAD keypoint to: (1) the corresponding annotated scan keypoint ($= 3.5cm$) and (2) the nearest Harris keypoint in the scan ($= 12.8cm$).

### 4.3. Benchmark

Using our annotated dataset, we designed a benchmark to evaluate scan-to-CAD alignment methods. A model alignment is considered successful only if the category of the CAD model matches that of the scan object *and* the pose error is within translation, rotational, and scale bounds relative to the ground truth CAD. We do not enforce strict instance matching (i.e., matching the exact CAD model of the ground truth annotation) as ShapeNet models typically do not identically match real-world scanned objects. Instead, we treat CAD models of the same category as interchangeable (according to the ShapeNetCorev2 *top-level synset*).

Once a CAD model is determined to be aligned correctly, the ground truth counterpart is removed from the candidate pool in order to prevent multiple alignments to the same object. Alignments are fully parameterized by 9 pose parameters. A quantitative measure based on bounding box overlap (IoU) can be readily calculated with these parameters as CAD models are defined on the unit box. The error thresholds for a successful alignment are set to $\epsilon_t \leq 20$cm, $\epsilon_r \leq 20°$, and $\epsilon_s \leq 20\%$ for translation, rotation, and scale respectively (for extensive error analysis please see the supplemental). The rotation error calculation takes $C_2$, $C_4$ and $C_\infty$ rotated versions into account.

The Scan2CAD dataset and associated symmetry annotations are available to the community. For standardized comparison of future approaches, we operate an automated test script on a hidden test set that can be found under `www.Scan2CAD.org`.

## 5. Correspondence Prediction Network

### 5.1. Data Representation

Scan data is represented by its signed distance field (SDF) encoded in a volumetric grid and generated through *volumetric fusion* [6] from the depth maps of the RGB-D reconstruction (voxel resolution = $3cm$, truncation = $15cm$). For the CAD models, we compute unsigned distance fields (DF) using the level-set generation toolkit by Batty [1].

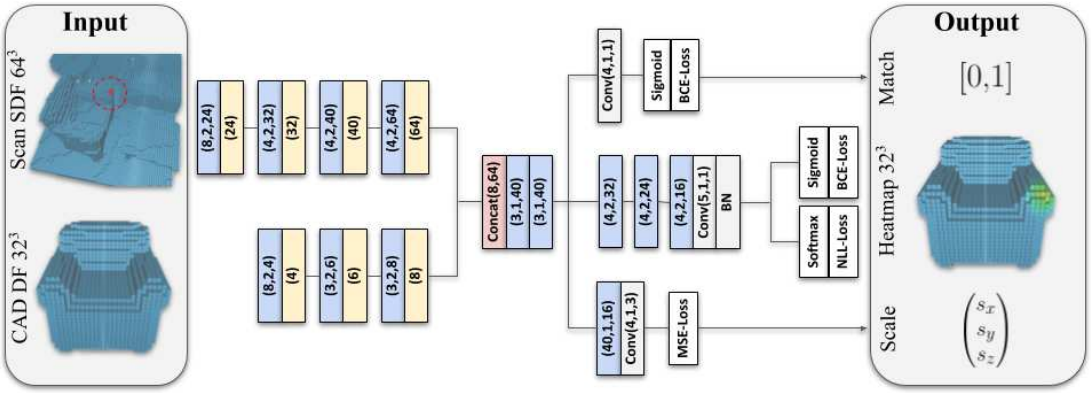



Figure 4: 3D CNN architecture of our Scan2CAD approach: we take as input SDF chunks around a given keypoint from a 3D scan and the DF of a CAD model. These are encoded with 3D CNNs to learn a shared embedding between the synthetic and real data; from this, we classify whether there is semantic compatibility between both inputs (top), predict a correspondence heatmap in the CAD space (middle) and the scale difference between the inputs (bottom).

### 5.2. Network Architecture

Our architecture takes as input a pair of voxel grids: A SDF centered at a point in the scan with a large receptive field at $64^3$ size, and a DF of a particular CAD model at $32^3$ size. We use a series of convolutional layers to separately encode each input stream (see Fig. 4). The two encoders compress the volumetric representation into compact feature volumes of $4^3 \times 64$ (scan) and $4^3 \times 8$ (CAD) which are then concatenated before passing to the decoder stage. The decoder stage predicts three output targets, heatmap, compatibility, and scale, described as follows:

**Heatmap** The first output is a heatmap $H : \Omega \rightarrow [0, 1]$ over the $32^3$ voxel domain $\Omega \in \mathbb{N}^3$ of the CAD model producing the voxel-wise correspondence probability. This indicates the probability of matching each voxel in $\Omega$ to the center point of the scan SDF. We train our network using a combined binary cross-entropy (BCE) loss and a negative log-likelihood (NLL) to predict the final heatmap $H$. The raw output $S : \Omega \rightarrow \mathbb{R}$ of the last layer in the decoder is used to generate the heatmaps:

$$H_1 : \Omega \rightarrow [0, 1], \quad x \mapsto \text{sigmoid}(S(x))$$
$$H_2 : \Omega \rightarrow [0, 1], \quad x \mapsto \text{softmax}(S(x))$$
$$\mathcal{L}_H = \sum_{x \in \Omega} w(x) \cdot \text{BCE}(H_1, H_{\text{GT}}) + \sum_{x \in \Omega} v \cdot \text{NLL}(H_2, H_{\text{GT}})$$

where $w(x) = 64.0$ if $H_{\text{GT}}(x) > 0.0$ else $1.0, v = 64$ are weighting factors to increase the signal of the few sparse positive keypoint voxels in the voxel grid ($\approx 99\%$ of the target voxels have a value equal to 0). The combination of the sigmoid and softmax terms is a compromise between high recall but low precision using sigmoid, and more locally sharp keypoint predictions using softmax over all voxels. The final target heatmap, used later for alignment, is constructed with an element-wise multiplication of both heatmap variations: $H = H_1 \circ H_2$.

**Compatibility** The second prediction target is a single probability score $\in [0, 1]$ indicating semantic compatibility between scan and CAD. This category equivalence score is 0 when the category labels are different (e.g., scan table and CAD chair) and 1 when the category labels match (e.g., scan chair and CAD chair). The loss function for this output is a sigmoid function followed by a BCE loss:

$$\mathcal{L}_{\text{compat.}} = \text{BCE}(\text{sigmoid}(x), x_{\text{GT}})$$

**Scale** The third output predicts the scale $\in \mathbb{R}^3$ of the CAD model to the respective scan. Note that we do not explicitly enforce positivity of the predictions. This loss term is a mean-squared-error (MSE) for a prediction $x \in \mathbb{R}^3$:

$$\mathcal{L}_{\text{scale}} = \text{MSE}(x, x_{\text{GT}}) = \|x - x_{\text{GT}}\|_2^2$$

Finally, to train our network, we use a weighted combination of the presented losses:

$$\mathcal{L} = 1.0\mathcal{L}_{\text{H}} + 0.1\mathcal{L}_{\text{compat.}} + 0.2\mathcal{L}_{\text{scale}}$$

where the weighting of each loss component was empirically determined for balanced convergence.

### 5.3. Training Data Generation

**Voxel Grids** Centered scan volumes are generated by projecting the annotated keypoint into the scan voxel grid and then cropping around it with a crop window of $63^3$. Ground truth heatmaps are generated by projecting annotated keypoints (and any symmetry-equivalent keypoints) into the CAD voxel grid. We then use a Gaussian blurring kernel ($\sigma = 2.0$) on the voxel grid to account for small keypoint annotation errors and to avoid sparsity in the loss residuals.

**Training Samples** With our annotated dataset we generate $N_{P,\text{ann.}} = 97607$ positive training pairs where one pair consists of an annotated scan keypoint and the corresponding CAD model. Additionally, we create $N_{P,\text{aug.}} = 10 \cdot N_{P,\text{ann.}}$, augmented positive keypoint pairs by randomly sampling points on the CAD surface, projecting them to the scan via the ground truth transformation and rejecting if the distance to the surface in the scan $\geq 3cm$. In total we generate $N_P = N_{P,\text{ann.}} + N_{P,\text{aug.}}$ positive training pairs.

Negative pairs are generated in two ways: (1) Randomly choosing a voxel point in the scan and a random CAD model (likelihood of false negative is exceedingly low). (2) Taking an annotated scan keypoint and pairing it with a random CAD model of different class. We generate $N_N = N_P$ negative samples with (1) and $N_{HN} = N_P$ with (2).

Hence, the training set has a positives-to-negatives ratio of 1:2 ($N_P : N_N + N_{HN}$). We found an over-representation of negative pairs gives satisfactory performance on the compatibility prediction.

### 5.4. Training Process

We use an SGD optimizer with a batch size of 32 and an initial learning rate of $0.01$, which is decreased by $1/2$ every $50$K iterations. We train for 250K iterations ($\approx 62.5$ hours). The weights are initialized randomly. The losses of the heatmap prediction stream and the scale prediction stream are masked such that only positive samples make up the residuals for back-propagation.

The CAD encoder is pre-trained with an auto-encoder on ShapeNet models with a reconstruction task and a $MSE$ as loss function. All models of ShapeNetCore ($\approx 55K$) are used for pre-training and the input and output dimensions are $32^3$ distance field grids. The network is trained with SGD until convergence ($\approx 50$ epochs).

## 6. Alignment Optimization

**Filtering** The input to our alignment optimization is a representative set of Harris keypoints $\mathbb{K} = \{p_j\}$, $j = 1 \dots N_0$ from a scene $\mathbb{S}$ and a set of CAD models $\mathbb{M} = \{m_i\}$. The correspondences between $\mathbb{K}$ and $\mathbb{M}$ were established by the correspondence prediction from the previous stage (see Sec. 5) where each keypoint $p_j$ is tested against every model $m_i$.

Since not every keypoint $p_j$ semantically matches to every CAD model $m_i$, we reject correspondences based on the compatibility prediction of our network. The threshold for rejecting $p_j$ is determined by the Otsu thresholding scheme [31]. In practice this method turned out to be much more effective than a fixed threshold. After the filtering there are $N \leq N_0$ (usually $N \approx 0.1 N_0$) correspondence pairs to be used for the alignment optimization.

**Variational Optimization** From the remaining $\mathbb{K}_{\text{filter.}} \subset \mathbb{K}$ Harris keypoints, we construct *point-heatmap* pairs $(p_j, H_j)$ for each CAD model $m_i$, with $p_j \in \mathbb{R}^3$ a point in the scan and $H_j : \Omega \to [0, 1]$ a heatmap.

In order to find an optimal pose we construct the following minimization problem:

$$\mathrm{c}_{\text{vox}} = T_{\text{world}\to\text{vox}} \cdot T_{m_i}(a, s) \cdot p_j$$

$$f = \min_{a,s} \sum_{j}^{N} (1 - H_j(\mathrm{c}_{\text{vox}}))^2 + \lambda_s \|s\|_2^2 \qquad (1)$$

where $\mathrm{c}_{\text{vox}}$ is a voxel coordinate, $T_{\text{world}\to\text{vox}}$ denotes a transformation that maps world points into the voxel grid for look-ups, $a$ denotes the coordinates of the Lie algebra (for rotation and translation), $s$ defines the scale, and $\lambda_s$ defines the scale regularization strength. $a, s$ compose a transformation matrix $T_{m_i} = \psi(a_{m_i}, s_{m_i})$:

$$\psi : \mathbb{R}^6 \times \mathbb{R}^3 \to \mathbb{R}^{4\times4},$$

$$a, s \mapsto \text{expm}\left(\begin{bmatrix} \Gamma(a_{1,2,3}) & a_{4,5,6} \\ 0 & 0 \end{bmatrix}\right) \cdot \begin{bmatrix} s & 0 \\ 0 & 1 \end{bmatrix}$$

where $\Gamma$ is the hat map, expm is the matrix exponential.

We solve Eq. 1 using the Levenberg-Marquardt (LM) algorithm. As we can suffer from zero-gradients (especially at bad initialization), we construct a scale-pyramid from the heatmaps which we solve in coarse-to-fine fashion.

In each LM step we optimize over the incremental change and update the parameters as following: $T_{m_i}^{k+1} \leftarrow \phi(a^*, s^*) \cdot T_{m_i}^{k}$ where $a^*, s^*$ are the optimal parameters. As seen in Eq. 1, we add a regularization on the scale in order to prevent degenerate solutions which can appear for very large scales.

By restarting the optimization with different translation parameters (i.e., varying initializations), we obtain multiple alignments per CAD model $m_i$. We then generate as many CAD model alignments as required for a given scene in the evaluation. Note, in a ground truth scene one unique CAD model $m_i$ can appear in multiple locations e.g., chairs in conference rooms.

**Pruning** Finally, there will be alignments of various CAD models into a scene where a subset will be misaligned. In order to select only the best alignments and prune potential misalignments we use a confidence metric similar to [25]; for more detail, we refer to the appendix.

## 7. Results

### 7.1. Correspondence Prediction

To quantify the performance of correspondence heatmap predictions, we evaluate the voxel-wise F1-score for a prediction and its Gaussian-blurred target. The task is challenging and by design $\frac{2}{3}$ test samples are false correspondences, $\approx 99\%$ of the target voxels are 0-valued, and only a

| base [+variations, ...] | bath | bookshelf | cabinet | chair | display | sofa | table | trash bin | other | class avg. | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| +sym | 46.88 | 44.39 | 40.49 | 64.46 | 26.85 | 56.26 | 47.15 | 38.43 | 24.68 | 43.29 | 48.01 |
| +sym,+scale | 51.35 | 45.46 | 45.24 | 66.94 | 29.88 | 64.78 | 48.30 | 38.00 | 28.65 | 46.51 | 50.85 |
| +sym,+CP | 59.32 | 51.93 | 55.11 | 70.99 | 41.58 | 66.77 | 53.74 | 43.39 | 42.93 | 53.97 | 60.44 |
| +scale,+CP | 45.24 | 45.85 | 47.16 | 61.55 | 27.65 | 51.92 | 41.21 | 31.13 | 29.62 | 42.37 | 47.64 |
| +sym,+scale,+CP | 56.05 | 51.28 | **57.45** | 72.64 | 36.36 | 70.63 | 52.28 | 46.80 | 43.32 | 54.09 | 60.43 |
| +sym,+scale,+CP,+PT (3/3 fix) | 57.03 | 50.63 | 56.76 | 70.39 | 39.74 | 65.00 | 52.03 | **46.87** | 41.83 | 53.36 | 58.61 |
| +sym,+scale,+CP,+PT (1/3 fix) | **60.08** | **58.62** | 56.35 | **73.92** | **44.19** | **75.08** | **56.80** | 45.78 | **46.53** | **57.48** | **63.94** |

Table 1: Correspondence prediction F1-scores in % for variations of our correspondence prediction network. We evaluate the effect of symmetry (sym), predicting scale (scale), predicting compatibility (CP), encoder pre-training (PT), and pre-training with parts of the encoder fixed (#fix), see Sec. 5 for more detail regarding our network design and training scheme.

single 1-valued voxel out of $32^3$ voxels exists. The F1-score will increase only by identifying true correspondences. As seen in Tab. 1, our best 3D CNN achieves 63.94%.

Tab. 1 additionally addressed our design choices; in particular, we evaluate the effect of using pre-training (PT), using compatibility (CP) as a proxy loss (defined in Sec. 5.2), enabling symmetry awareness (sym), and predicting scale (scale). Here, a pre-trained network reduces overfitting, enhancing generalization capability. Optimizing for compatibility strongly improves heatmap prediction as it efficiently detects false correspondences. While predicting scale only slightly influences the heatmap predictions, it becomes very effective for the later alignment stage. Additionally, incorporating symmetry enables significant improvement by explicitly disambiguating symmetric keypoint matches.

### 7.2. Alignment

In the following, we compare our approach to other handcrafted feature descriptors: FPFH [33], SHOT [39], Li et al. [25] and a learned feature descriptor: 3DMatch [43] (trained on our Scan2CAD dataset). We combine these descriptors with a RANSAC outlier rejection method to obtain pose estimations for an input set of CAD models. A detailed description of the baselines can be found in the appendix. As seen in Tab. 2, our best method achieves 31.68% and outperforms all other methods by a significant margin. We additionally show qualitative results in Fig. 5. Compared to state-of-the-art handcrafted feature descriptors, our learned approach powered by our Scan2CAD dataset produces considerably more reliable correspondences and CAD model alignments. Even compared to the learned descriptor approach of 3DMatch, our explicit learning across the synthetic and real domains coupled with our alignment optimization produces notably improved CAD model alignment.

Fig. 6 shows the capability of our method to align in an unconstrained real-world setting where ground truth CAD models are not given, we instead provide a set of 400 random CAD models from ShapeNet [3].

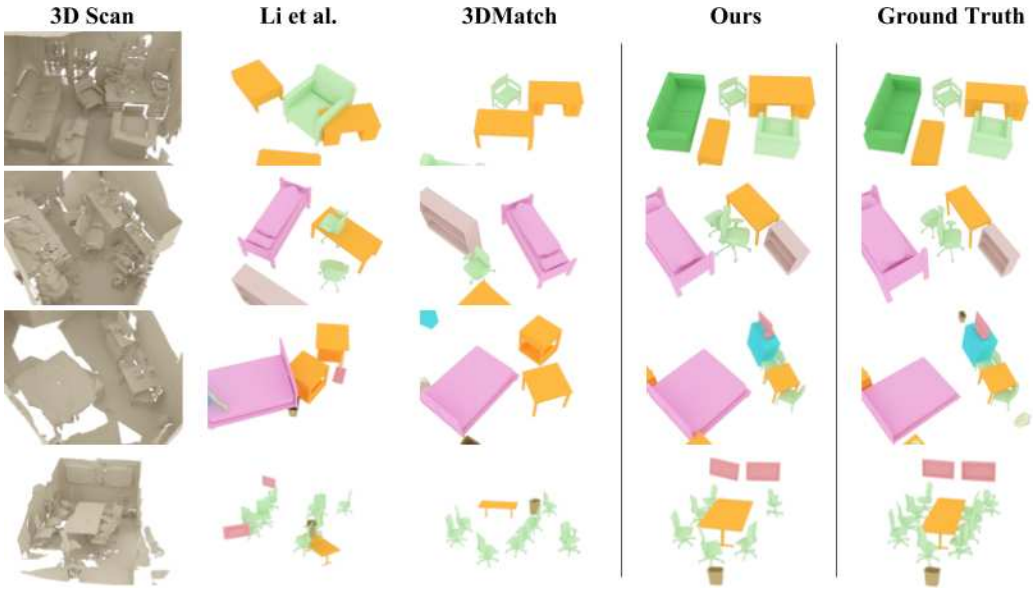



Figure 5: Qualitative comparison of alignments on four different test ScanNet [7] scenes. Our approach to learning geometric features between real and synthetic data produce much more reliable keypoint correspondences, which coupled with our alignment optimization, produces significantly more accurate alignments.

| | bath | bookshelf | cabinet | chair | display | sofa | table | trash bin | other | class avg. | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| FPFH (Rusu et al. [33]) | 0.00 | 1.92 | 0.00 | 10.00 | 0.00 | 5.41 | 2.04 | 1.75 | 2.00 | 2.57 | 4.45 |
| SHOT (Tombari et al. [39]) | 0.00 | 1.43 | 1.16 | 7.08 | 0.59 | 3.57 | 1.47 | 0.44 | 0.75 | 1.83 | 3.14 |
| Li et al. [25] | 0.85 | 0.95 | 1.17 | 14.08 | 0.59 | 6.25 | 2.95 | 1.32 | 1.50 | 3.30 | 6.03 |
| 3DMatch (Zeng et al. [43]) | 0.00 | 5.67 | 2.86 | 21.25 | 2.41 | 10.91 | 6.98 | 3.62 | 4.65 | 6.48 | 10.29 |
| Ours: +sym | 24.30 | 10.61 | 5.97 | 9.49 | 3.90 | 25.26 | 12.34 | 10.74 | 3.58 | 11.80 | 8.772 |
| Ours: +sym,+scale | 18.99 | 13.61 | 7.24 | 14.73 | 9.76 | 41.05 | 14.04 | 5.26 | 6.29 | 14.55 | 11.48 |
| Ours: +sym,+CP | 35.90 | 32.35 | 28.64 | 40.48 | 18.85 | 60.00 | 33.11 | 28.42 | 16.89 | 32.74 | 29.42 |
| Ours: +scale,+CP | 34.18 | 31.76 | 21.82 | 37.02 | 14.75 | 50.53 | 32.31 | **31.05** | 11.59 | 29.45 | 26.75 |
| Ours: +sym,+scale,+CP | 36.20 | **36.40** | **34.00** | **44.26** | 17.89 | **70.63** | 30.66 | 30.11 | 20.60 | **35.64** | **31.68** |
| Ours: +sym,+scale,+CP,+PT (3/3 fix) | **37.97** | 30.15 | 28.64 | 41.55 | 19.51 | 57.89 | 33.85 | 20.00 | 17.22 | 31.86 | 29.27 |
| Ours: +sym,+scale,+CP,+PT (1/3 fix) | 34.81 | **36.40** | 29.00 | 40.60 | **23.25** | 66.00 | **37.64** | 24.32 | **22.81** | 34.98 | 31.22 |

Table 2: Accuracy comparison (%) on our CAD alignment benchmark. While handcrafted feature descriptors can achieve some alignment on more featureful objects (e.g., chairs, sofas), they do not tolerate well the geometric discrepancies between scan and CAD data – which remains difficult for the learned keypoint descriptors of 3DMatch. Scan2CAD directly addresses this problem of learning features that generalize across these domains, thus significantly outperforming state of the art.

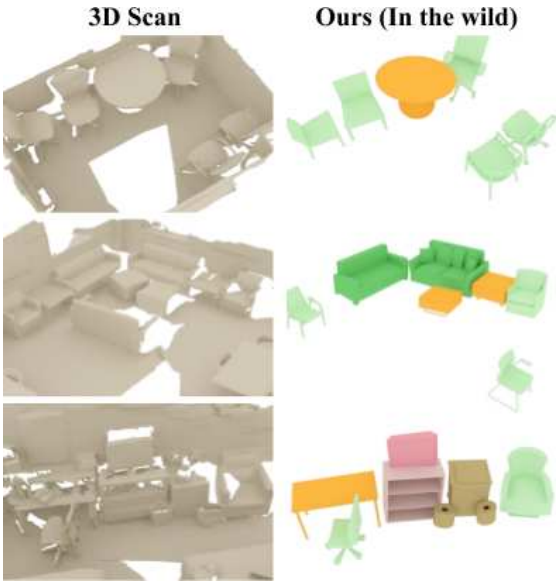



Figure 6: Unconstrained scenario where instead of having a ground truth set of CAD models given, we use a set of 400 randomly selected CAD models from ShapeNetCore [3], more closely mimicking a real-world application scenario.

## 8. Limitations

While the focus of this work is mainly on the alignment between 3D scans and CAD models, we only provide a basic algorithmic component for retrieval (finding the most similar model). This necessitates an exhaustive search over a set of CAD models. We believe that one of the immediate next steps in this regard would be designing a neural network architecture that is specifically trained on shape similarity between scan and CAD geometry to introduce more efficient CAD model retrieval. Additionally, we currently only consider geometric information, and it would also be intresting to introduce learned color features into the correspondence prediction, as RGB data is typically higher-resolution than depth or geometry, and could potentially improve alignment results.

## 9. Conclusion

In this work, we presented Scan2CAD, which aligns a set of CAD models to 3D scans by predicting correspondences in form of heatmaps and then optimizes over these correspondence predictions. First, we introduce a new dataset of 9DoF CAD-to-scan alignments with 97607 pairwise keypoint annotations defining the alignment of 14225 objects. Based on this new dataset, we design a 3D CNN to predict correspondence heatmaps between a CAD model and a 3D scan. From these predicted heatmaps, we formulate a variational cost minimization that then finds the optimal 9DoF pose alignments between CAD models and the scan, enabling effective transformation of noisy, incomplete RGB-D scans into a clean, complete CAD model representation. This enables us to achieve significantly more accurate results than state-of-the-art approaches, and we hope that our dataset and benchmark will inspire future work towards bringing RGB-D scans to CAD or artist-modeled quality.

## Acknowledgements

We would like to thank the expert annotators Soh Yee Lee, Rinu Shaji Mariam, Suzana Spasova, Emre Taha, Sebastian Thekkekara, and Weile Weng for their efforts in building the Scan2CAD dataset. We thank valuable discussions with Jürgen Sturm. This work is supported by Occipital, the ERC Starting Grant Scan2CAD (804724), a Google Faculty Award, and the ZD.B. We would also like to thank the support of the TUM-IAS, funded by the German Excellence Initiative and the European Union Seventh Framework Programme under grant agreement n 291763, for the TUM-IAS Rudolf Mößbauer Fellowship and Hans-Fisher Fellowship (Focus Group Visual Computing).

## References


[1] C. Batty. SDFGen. https://github.com/christopherbatty/SDFGen. 4

[2] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang. Matterport3D: Learning from RGB-D data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017. 1

[3] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 2, 3, 7, 8

[4] J. Chen, D. Bautembach, and S. Izadi. Scalable real-time volumetric surface reconstruction. *ACM Transactions on Graphics (TOG)*, 32(4):113, 2013. 2

[5] S. Choi, Q.-Y. Zhou, and V. Koltun. Robust reconstruction of indoor scenes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5556–5565. IEEE, 2015. 1, 2

[6] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312. ACM, 1996. 1, 2, 4

[7] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 1, 2, 3, 4, 7

[8] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (TOG)*, 36(3):24, 2017. 1, 2

[9] A. Dai, C. R. Qi, and M. Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 1

[10] A. Dai, D. Ritchie, M. Bokeloh, S. Reed, J. Sturm, and M. Nießner. Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans. *arXiv preprint arXiv:1712.10215*, 2018. 1

[11] B. Drost and S. Ilic. 3d object detection and localization using multimodal point pair features. In *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, pages 9–16. IEEE, 2012. 2

[12] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Aligning 3D models to RGB-D images of cluttered scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4731–4740, 2015. 2

[13] N. Hansen. Benchmarking a bi-population cma-es on the bbob-2009 function testbed. In *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, pages 2389–2396. ACM, 2009. 4

[14] N. Hansen, S. D. Müller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary computation*, 11(1):1–18, 2003. 4

[15] B.-S. Hua, Q.-H. Pham, D. T. Nguyen, M.-K. Tran, L.-F. Yu, and S.-K. Yeung. Scenenn: A scene meshes dataset with annotations. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 92–101. IEEE, 2016. 2

[16] B.-S. Hua, Q.-T. Truong, M.-K. Tran, Q.-H. Pham, A. Kanezaki, T. Lee, H. Chiang, W. Hsu, B. Li, Y. Lu, et al. Shrec17: Rgb-d to cad retrieval with objectnn dataset. 2

[17] S. Huang, S. Qi, Y. Zhu, Y. Xiao, Y. Xu, and S.-C. Zhu. Holistic 3D scene parsing and reconstruction from a single RGB image. In *European Conference on Computer Vision*, pages 194–211. Springer, 2018. 2

[18] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568. ACM, 2011. 1, 2

[19] H. Izadinia, Q. Shan, and S. M. Seitz. Im2cad. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 2422–2431. IEEE, 2017. 2

[20] A. E. Johnson. Spin-images: a representation for 3-d surface matching. 1997. 2

[21] O. Kähler, V. A. Prisacariu, C. Y. Ren, X. Sun, P. Torr, and D. Murray. Very high frame rate volumetric integration of depth images on mobile devices. *IEEE transactions on visualization and computer graphics*, 21(11):1241–1250, 2015. 1, 2

[22] M. Keller, D. Lefloch, M. Lambers, S. Izadi, T. Weyrich, and A. Kolb. Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *3D Vision-3DV 2013, 2013 International Conference on*, pages 1–8. IEEE, 2013. 2

[23] Y. M. Kim, N. J. Mitra, Q. Huang, and L. Guibas. Guided real-time scanning of indoor objects. In *Computer Graphics Forum*, volume 32, pages 177–186. Wiley Online Library, 2013. 2

[24] Y. M. Kim, N. J. Mitra, D.-M. Yan, and L. Guibas. Acquiring 3D indoor environments with variability and repetition. *ACM Transactions on Graphics (TOG)*, 31(6):138, 2012. 2

[25] Y. Li, A. Dai, L. Guibas, and M. Nießner. Database-assisted object retrieval for real-time 3D reconstruction. In *Computer Graphics Forum*, volume 34, pages 435–446. Wiley Online Library, 2015. 2, 6, 7, 8

[26] J. J. Lim, H. Pirsiavash, and A. Torralba. Parsing ikea objects: Fine pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2992–2999, 2013. 2, 3

[27] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger. Fusion++: Volumetric object-level slam. In *2018 International Conference on 3D Vision (3DV)*, pages 32–41. IEEE, 2018. 2

[28] L. Nan, K. Xie, and A. Sharf. A search-classify approach for cluttered indoor scene understanding. *ACM Transactions on Graphics (TOG)*, 31(6):137, 2012. 2

[29] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and

A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011. 1, 2

[30] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (TOG)*, 2013. 1, 2

[31] N. Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979. 6

[32] 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. 2

[33] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 3212–3217. Citeseer, 2009. 2, 7, 8

[34] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1352–1359, 2013. 2

[35] 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. 2

[36] T. Shao, W. Xu, K. Zhou, J. Wang, D. Li, and B. Guo. An interactive approach to semantic modeling of indoor scenes with an RGBD camera. *ACM Transactions on Graphics (TOG)*, 31(6):136, 2012. 2

[37] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser. Semantic scene completion from a single depth image. *Proceedings of 30th IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1

[38] X. Sun, J. Wu, X. Zhang, Z. Zhang, C. Zhang, T. Xue, J. B. Tenenbaum, and W. T. Freeman. Pix3D: Dataset and methods for single-image 3D shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2974–2983, 2018. 2, 3

[39] F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Computer Vision – ECCV 2010*, pages 356–369, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. 7, 8

[40] T. Whelan, S. Leutenegger, R. F. Salas-Moreno, B. Glocker, and A. J. Davison. Elasticfusion: Dense slam without a pose graph. *Proc. Robotics: Science and Systems, Rome, Italy*, 2015. 1, 2

[41] Y. Xiang, W. Kim, W. Chen, J. Ji, C. Choy, H. Su, R. Mottaghi, L. Guibas, and S. Savarese. Objectnet3d: A large scale database for 3D object recognition. In *European Conference on Computer Vision*, pages 160–176. Springer, 2016. 3

[42] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on*, pages 75–82. IEEE, 2014. 3

[43] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 199–208. IEEE, 2017. 2, 7, 8

[44] X. Zhou, A. Karpur, C. Gan, L. Luo, and Q. Huang. Unsupervised domain adaptation for 3d keypoint estimation via view consistency. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 137–153, 2018. 2

[45] C. Zou, R. Guo, Z. Li, and D. Hoiem. Complete 3D scene parsing from an RGBD image. *International Journal of Computer Vision (IJCV)*, 2018. 2

# End-to-End CAD Model Retrieval and 9DoF Alignment in 3D Scans

Armen Avetisyan, Angela Dai, Matthias Nießner

Technical University of Munich

This is the accepted but not the published version of the paper due to copyright restrictions.
Published version: https://doi.org/10.1109/ICCV.2019.00264

# End-to-End CAD Model Retrieval and 9DoF Alignment in 3D Scans

Armen Avetisyan Angela Dai Matthias Nießner
Technical University of Munich

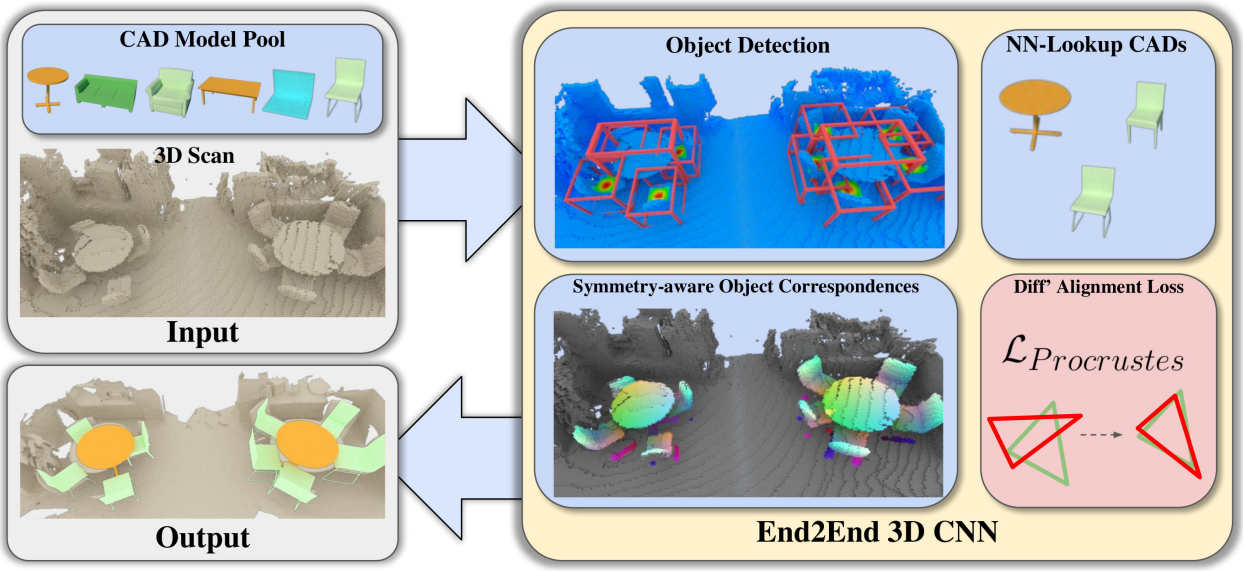



Figure 1: From a 3D scan and a set of CAD models, our method learns to predict 9DoF CAD model alignments to the objects of the scan in a fully-convolutional, end-to-end fashion. Our proposed 3D CNN first detects objects in the scan, then uses the regressed object bounding boxes to establish symmetry-aware object correspondences between a scan object and CAD model, which inform our differentiable Procrustes alignment loss, enabling learning of alignment-informed correspondences and producing CAD model alignment to a scan in a single forward pass.

## Abstract

*We present a novel, end-to-end approach to align CAD models to an 3D scan of a scene, enabling transformation of a noisy, incomplete 3D scan to a compact, CAD reconstruction with clean, complete object geometry. Our main contribution lies in formulating a differentiable Procrustes alignment that is paired with a symmetry-aware dense object correspondence prediction. To simultaneously align CAD models to all the objects of a scanned scene, our approach detects object locations, then predicts symmetry-aware dense object correspondences between scan and CAD geometry in a unified object space, as well as a nearest neighbor CAD model, both of which are then used to inform a differentiable Procrustes alignment. Our approach operates in a fully-convolutional fashion, enabling alignment of CAD models to the objects of a scan in a single forward pass. This enables our method to outperform state-of-the-art approaches by* $19.04\%$ *for CAD model alignment to scans, with* $\approx 250\times$ *faster runtime than previous data-driven approaches.*

## 1. Introduction

In recent years, RGB-D scanning and reconstruction has seen significant advances, driven by the increasing availability of commodity range sensors such as the Microsoft Kinect, Intel RealSense, or Google Tango. State-of-the-art 3D reconstruction approaches can now achieve impressive capture and reconstruction of real-world environments [19, 26, 27, 38, 38, 5, 8], spurring forth many potential applications of this digitization, such as content creation, or augmented or virtual reality.

Such advances in 3D scan reconstruction have nonetheless remained limited towards these use scenarios, due to geometric incompleteness, noise and oversmoothing, and lack of fine-scale sharp detail. In particular, there is a notable contrast in such reconstructed scan geometry in comparison to the clean, sharp 3D models created by artists for visual and graphics applications.

With the increasing availability of synthetic CAD models [4], we have the opportunity to reconstruct a 3D scan

through CAD model shape primitives; that is, finding and aligning similar CAD models from a database to each object in a scan. Such a scan-to-CAD transformation enables construction of a clean, compact representation of a scene, more akin to artist-created 3D models to be consumed by mixed reality or design applications. Here, a key challenge lies in finding and aligning similar CAD models to scanned objects, due to strong low-level differences between CAD model geometry (clean, complete) and scan geometry (noisy, incomplete). Current approaches towards this problem thus often operate in a sparse correspondence-based fashion [22, 1] in order to establish reasonable robustness under such differences.

Unfortunately, such approaches, in order to find and align CAD models to an input scan, thus involve several independent steps of correspondence finding, correspondence matching, and finally an optimization over potential matching correspondences for each candidate CAD model. With such decoupled steps, there is a lack of feedback through the pipeline; e.g., correspondences can be learned, but they are not informed by the final alignment task. In contrast, we propose to predict symmetry-aware dense object correspondences between scan and CADs in a global fashion. For an input scan, we leverage a fully-convolutional 3D neural network to first detect object locations, and then from each object location predict a uniform set of dense object correspondences and object symmetry are predicted, along with a nearest neighbor CAD model; from these, we introduce a differentiable Procrustes alignment, producing a final set of CAD models and 9DoF alignments to the scan in an end-to-end fashion. Our approach outperforms state-of-the-art methods for CAD model alignment by $19.04\%$ for real-world 3D scans.

Our approach is the first, to the best of our knowledge, to present an end-to-end scan-to-CAD alignment, constructing a CAD model reconstruction of a scene in a single forward pass. In summary, we propose an end-to-end approach for scan-to-CAD alignment featuring:

- a novel differentiable Procrustes alignment loss, enabling end-to-end CAD model alignment to a 3D scan,
- symmetry-aware dense object correspondence prediction, enabling robust alignment even under various object symmetries, and
- CAD model alignment for a scan of a scene in a single forward pass, enabling very efficient runtime ($<$ 3s on real-world scan evaluation)

## 2. Related work

**RGB-D Scanning and Reconstruction** 3D scanning methods have a long research history across several communities, ranging from offline to real-time techniques. In particular, RGB-D scanning has become increasingly popular, due to the increasing availability of commodity range sensors. A very popular reconstruction technique is the volumetric fusion approach by Curless and Levoy [6], which has been materialized in many real-time reconstruction frameworks such as KinectFusion [19, 26], Voxel Hashing [27] or BundleFusion [8], as well as in the context of state-of-the-art offline reconstruction methods [5]. An alternative to these voxel-based scene representations is based on surfels [21], which has been used by ElasticFusion [38] to realize loop closure updates. These works have led to RGB-D scanning methods that feature robust, global tracking and can capture very large 3D environments. However, though these methods can achieve stunning results in RGB-D capture and tracking, the quality of reconstructed 3D geometry nonetheless remains far from from artist-created 3D content, as the reconstructed scans are partial, and contain noise or oversmoothing from sensor quality or small camera tracking errors.

**3D Features for Shape Alignment and Retrieval** An alternative to bottom-up 3D reconstruction from RGB-D scanning techniques is to find high-quality CAD models that can replace the noisy and incomplete geometry from a 3D scan. Finding and aligning these CAD models inevitably requires 3D feature descriptors to find geometric matches between the scan and the CAD models. Traditionally, these descriptors were hand-crafted, and often based on a computation of histograms (e.g., point normals), such as FPFH [30], SHOT [36], or point-pair features [12].

More recently, with advances in deep neural networks, these descriptors can be learned, for instance based on an implicit signed distance field representation [41, 10, 11]. A typical pipeline for CAD-to-scan alignments builds on these descriptors; i.e., the first step is to find 3D feature matches and then use a variant of RANSAC or PnP to compute 6DoF or 9Dof CAD alignments. This two-step strategy has been used by Slam++ [31], Li et al. [22], Shao et al. [32], the data-driven work by Nan et al. [25] and the recent Scan2CAD approach [1]. One potential approach to combine correspondence prediction and alignment is through differentiable RANSAC [3], which has been applied for camera localization. Our approach is designed to learn robust dense correspondences through a differentiable Procrustes alignment where correspondences and their relative weights are jointly optimized together without requiring multiple hypothesis generation. Other approaches rely only on single RGB(-D) frame input, but use a similar two-step alignment strategy [23, 20, 34, 18, 13, 42]. While these methods are related, their focus is different as we address geometric alignment independent of RGB information.

While promising results have been achieved by these two-step approaches, there remains a fundamental limita-

tion in the decoupled nature of feature matching and alignment computation. This inherently limits the ability of data-driven descriptors, as they remain unaware of the used optimization algorithm.

In our work, we propose an end-to-end alignment algorithm where correspondences are trained through gradients from an differentiable Procrustes optimizer.

**Shape Retrieval Challenges and RGB-D Datasets** In the context of 2D object alignment methods several datasets provide alignment annotations between RGB images and CAD models, including the PASCAL 3D+ [40], ObjectNet3D [39], the IKEA objects [23], and Pix3D [34]; however, no geometric information is given in the query images. SHREC provides a very popular series of 3D shape retrieval challenges, organized as part of Eurographics 3DOR [17, 29]; the tasks include matching objects from ScanNet [7] and SceneNN [16] to ShapeNet models [4].

More recently, Scan2CAD [1] provides accurate CAD alignment annotations on top of ScanNet [7] using ShapeNet models [4], based on roughly 100k manually annotated correspondences. In addition to evaluating our method on the Scan2CAD test dataset, we also evaluate on the synthetic SUNCG [33] dataset.

## 3. Overview

For an input 3D scan along with a set of candidate CAD models, our method aims to align similar CAD models to each object instance in the scan. Object locations in the scan are detected, and for each detected object, a similar CAD model is retrieved and a 9DoF transformation (3 degrees each for translation, rotation, and scale) computed to align it to the scan geometry. Thus we can transform a noisy, incomplete 3D scan into a compact, CAD-based representation with clean, complete geometry, as shown in Figure 1.

To this end, we propose an end-to-end 3D CNN-based approach to simultaneously retrieve and align CAD models to the objects of a scan in a single pass, for scans of varying sizes. This end-to-end formulation enables the final alignment process to inform learning of scan-CAD correspondences. To enable effective learning of scan-CAD object correspondences, we propose to use *symmetry-aware object correspondences* (SOCs), which establish dense correspondences between scan objects and CAD models, and are trained by our differentiable Procrustes alignment loss.

Then for an input scan $\mathbb{S}$ represented by volumetric grid encoding a truncated signed distance field, our model first detects object center locations as heatmap predictions over the volumetric grid and corresponding bounding box sizes for each object location. The bounding box represents the extent of the underlying object. From these detected object locations, we use the estimated bounding box size to crop out the neighborhood region around the object center from the learned feature space in order to predict our SOC correspondences to CAD models.

From this neighborhood of feature information, we then predict SOCs. These densely establish correspondences for each voxel in the object neighborhood to CAD model space. In order to be invariant to potential reflection and rotational symmetries, which could induce ambiguity in the correspondences, we simultaneously estimate the symmetry type of the object. We additionally predict a binary mask to segment the object instance from background clutter in the neighborhood, thus informing the set of correspondences to be used for the final alignment. To find a CAD model corresponding to the scan object, we jointly learn an object descriptor which is used to retrieve a semantically similar CAD model from a database.

Finally, we introduce a differentiable Procrustes alignment, enabling a fully end-to-end formulation, where learned scan object-CAD SOC correspondences can be informed by the final alignment process, achieving efficient and accurate 9DoF CAD model alignment for 3D scans.

## 4. Method

### 4.1. Network Architecture

Our network architecture is shown in Figure 2. It is designed to operate on 3D scans of varying sizes, in a fully-convolutional manner. An input scan is given by a volumetric grid encoding a truncated signed distance field, representing the scan geometry. We design our network backbone to learn features for detecting objects in a scan, establishing SOCs, and aligning CAD models to them. The end-to-end formulation enables the learned SOCs to be informed by the alignment performance.

The network backbone is structured in an encoder-decoder fashion, and composed of a series of ResNet blocks [14]. The bottleneck volume is spatially reduced by a factor of 16 from the input volume, and is decoded to the original resolution through transpose convolutions. The decoder is structured symmetrically to the encoder, but with half the feature channels, which we empirically found to produce faster convergence and more accurate performance. The output of the decoder is used to predict an objectness heatmap, identifying potential object locations, which is employed to inform bounding box regression for object detection. The predicted object bounding boxes are used to crop and extract features from the output of the second decoder layer, which then inform the SOC predictions. The features used to inform the SOC correspondence are extracted from the second block of the decoder, whose feature map spatial dimensions are $1/4$ of the original input dimension.

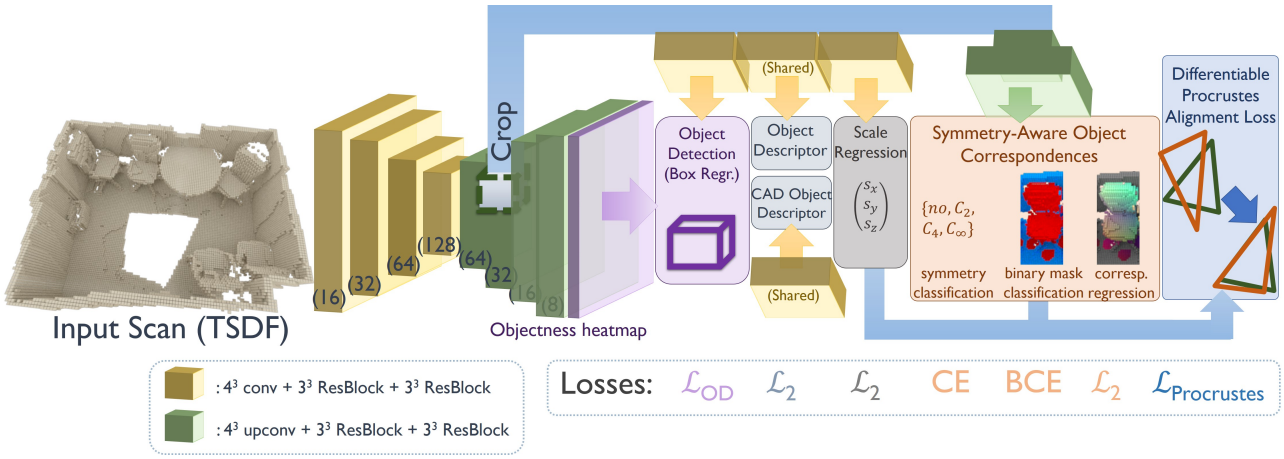



Figure 2: Network architecture for our end-to-end approach for CAD model alignment. An input TSDF scan represented in a volumetric grid is input to an encoder-decoder backbone constructed with residual blocks. Objects are detected through objectness prediction and bounding box regression; these predicted object boxes are then used to crop features from the decoder to inform CAD model alignment to a detected object. The cropped features are processed to simultaneously predict an object descriptor constrained to be similar to a corresponding CAD object descriptor (used for retrieving CAD models) and a 3-dimensional scale. Our symmetry-aware object correspondences (SOCs) informs directly our differentiable Procrustes alignment loss.

**Object Detection** We first detect objects, predicting bounding boxes for the objects in a scan, which then inform the SOC predictions. The output of the backbone decoder predicts heatmaps representing objectness probability over the full volumetric grid (whether a voxel is a center of an object). We then regress object bounding boxes corresponding to these potential object centers. For object bounding boxes predictions, we regress a 3-channel feature map, with each 3-dimensional vector corresponding to the bounding box extent size, and regressed using an $\ell_2$ loss.

Objectness is predicted as a heatmap, encoding voxel-wise probabilities as to whether each voxel is a center of an object. Note that $\Omega \subset \mathbb{N}^3$ is the discretized space (i.e. voxel grid). To predict a location heatmap $H_1$, we additionally employ two proxy losses, using a second heatmap prediction $H_2$ as well as a predicted offset field $O$. $H_1$ and $H_2$ are two 1-channel heatmaps designed to encourage high recall and precision, respectively, and $O$ is a 3-channel grid representing an offset field to the nearest object center. The objectness heatmap loss is:

$$\mathcal{L}_{OD} = 2.0 \cdot \mathcal{L}_{\text{recall}} + 10.0 \cdot \mathcal{L}_{\text{precision}} + 10.0 \cdot \mathcal{L}_{\text{offset}}$$

The weights for each component in the loss are designed to bring the losses numerically to approximately the same order of magnitude. Here, $\mathcal{L}_{\text{recall}}$ and $\mathcal{L}_{\text{precision}}$ are inspired from the conditional keypoint correspondence heatmap predictions of Scan2CAD [1].

$\mathcal{L}_{\text{recall}}$ aims to achieve high recall. It operates on the prediction $H_1$, on which we apply a sigmoid and calculate the loss via binary-cross entropy (BCE). This loss on its own tends to establish a high recall, but also blurry predictions.

$$\mathcal{L}_{\text{recall}} = \sum_{x \in \Omega} \text{BCE}(\sigma(H_1(x)), H_{\text{GT}}(x)) \quad (1)$$

$$H_1 : \Omega \to [0, 1], \quad \sigma : \text{sigmoid} \quad (2)$$

$\mathcal{L}_{\text{precision}}$ aims to achieve high precision. It operates on the prediction $H_2$, on which we apply a softmax and calculate the loss via negative log-likelihood (NLL). Due to the softmax, this loss encourages highly localized predictions in the output volume, which helps to attain high precision.

$$\mathcal{L}_{\text{precision}} = \sum_{x \in \Omega} \text{NLL}(\sigma(H_2(x)), H_{\text{GT}}(x)) \quad (3)$$

$$H_2 : \Omega \to [0, 1], \quad \sigma : \text{softmax} \quad (4)$$

$\mathcal{L}_{\text{offset}}$ is a regression loss on the predicted 3D offset field $O$, following [28]. Each voxel of $O$ represents a 3-dimensional vector that points to the nearest object center. This regression loss is used as a proxy loss to support the other two classification losses.

$$\mathcal{L}_{\text{offset}} = \sum_{x \in \Omega} \|O(x) - O_{\text{GT}}(x)\|_2^2 \quad (5)$$

$$O : \Omega \to \mathbb{R}^3$$

**Predicting SOCs** SOCs are dense, voxel-wise correspondences from scan geometry to CAD models. They are de-

fined as SOC : $\Omega \rightarrow [-0.5, 0.5]^3$, the normalized space of the CAD models.

In order to account for symmetry ambiguities, ground truth SOCs are generated such that the front-facing axis of the CAD model maintains minimal angle with the x-axis of the scan voxel grid. Thus for symmetric objects, the SOCs are generated in a consistent fashion, i.e., always aligned with the x-axis of the scan coordinate system.

SOCs are predicted using features cropped from the network backbone. For each detected object, we crop a region with the extend of the predicted bounding box volume $\mathcal{F}$ from the feature map of the second upsampling layer to inform our dense, symmetry-aware object correspondences. This feature volume $\mathcal{F}$ is first fitted through tri-linear interpolation into a uniform voxel grid of size $48^3$ before streaming into different prediction heads. SOCs incorporate several output predictions: a volume of dense correspondences from scan space to CAD object space, an instance segmentation mask, and a symmetry classification.

The dense correspondences, which map to CAD object space, implicitly contain CAD model alignment information. These correspondences are regressed as CAD object space coordinates, similar to [37], with the CAD object space defined as a uniform grid centered around the object, with coordinates normalized to $[-0.5, 0.5]$. These coordinates are regressed using an $\ell_2$ loss.

We also introduce a proxy symmetry loss to encourage correct SOC prediction by predicting the symmetry class of the object for common symmetry classes for furniture objects: two-fold rotational symmetry, four-fold rotational symmetry, infinite rotational symmetry, and no symmetry.

**Retrieval** To retrieve a similar CAD model to the detected object, we use the cropped feature neighborhood $\mathcal{F}$ to train an object descriptor for the scan region, using a series of 3D convolutions to reduce the feature dimensionality. This resulting 512-dimensional object descriptor is then constrained to match the latent vector of an autoencoder trained on the CAD model dataset, with latent spaces constrained by an $\ell_2$ loss. This enables retrieval of a semantically similar CAD model at test time through a nearest neighbor search using the object descriptor.

**Scale** Similarly to the retrieval head, the scale is predicted per detected object (i.e. per crop). We regress the $\mathbb{R}^3$ scale vector with an $\ell_2$ loss. At train and test time this estimate is used as final scale estimate with further post processing.

**9DoF Alignment** Our differentiable 9DoF alignment enables training for CAD model alignment in an end-to-end fashion, thereby informing learned correspondences of the final alignment objective. To this end, we leverage a differentiable Procrustes loss on the masked correspondences given by the SOC predictions to find the rotation alignment. That is, we aim to find a rotation matrix $R$ which brings together the CAD and scan correspondence points $P_c, P_s$:

$$R^* = \text{argmin}_R ||RP_c - P_s||_F, \qquad R \in SO_3$$

This is solved through a differentiable SVD of $P_s P_c^T = U\Sigma V^T$, with $R = U \begin{bmatrix} 1 & & \\ & 1 & \\ & & d \end{bmatrix} V^T$, $d = \det(VU^T)$. Here, the SVD is computed by solving the non-linear characteristic polynomial of the $3 \times 3$ matrix $P_s P_c^T$ iteratively, giving the final rotation. For scale and translation, we directly regress the scale using two 3D downsampling convolutions on $\mathcal{F}$, and the translation is predicted from the detected object centers. Note that an object center is the geometric center of the bounding box.

### 4.2. Training

**Data** Input scan data is represented by its truncated signed distance field (TSDF) encoded in a volumetric grid and generated through volumetric fusion [6] (we use voxel size = 3cm, truncation = 15cm). The CAD models used to train the autoencoder to produce a latent space for scan object descriptor training are represented as unsigned distance fields (DF), using the level-set generation toolkit by Batty [2].

To train our model for CAD model alignment for real scan data, we use the Scan2CAD dataset introduced by [1]. These Scan2CAD annotations provide 1506 scenes for training. Using upright rotation augmentation, we augment the number of training samples by 4 ($90^\circ$ increments with $20^\circ$ random jitter). We train our network using full scenes as input, with batch size of 1. For SOC prediction at train time the batch size is equal to the number of groundtruth objects in the given scene as crops are only performed around groundtruth object centers. Only large scenes during training are randomly cropped to $400 \times 400 \times 64$ to meet memory requirements. We found that training using 1 scene per batch generally yields stable convergence behavior.

For CAD model alignment to synthetic scan data, we use the SUNCG dataset [33], where we virtually scan the scenes following [9, 15] to produce input partial TSDF scans. The training process for synthetic SUNCG scan data is identical to training with real data. See supplemental material for further details.

**Optimization** We use an SGD optimizer with a batch size of 1 scene and an initial learning rate of 0.002, which is decayed by 0.5 every $20K$ iterations. We train for $50K$ iterations until convergence, which takes $\approx 48$ hours.

We train our model from scratch with the exception of the object retrieval descriptors. For object retrieval, we pretrain an autoencoder on all ShapeNetCore CAD models, trained to reconstruct their distance fields at $32^3$. This CAD autoencoder is trained with a batch size of 16 for $30K$ iterations. We then train the full model with pre-trained object

| | bath | bookshelf | cabinet | chair | display | sofa | table | trash bin | other | class avg. | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| FPFH (Rusu et al. [30]) | 0.00 | 1.92 | 0.00 | 10.00 | 0.00 | 5.41 | 2.04 | 1.75 | 2.00 | 2.57 | 4.45 |
| SHOT (Tombari et al. [35]) | 0.00 | 1.43 | 1.16 | 7.08 | 0.59 | 3.57 | 1.47 | 0.44 | 0.75 | 1.83 | 3.14 |
| Li et al. [22] | 0.85 | 0.95 | 1.17 | 14.08 | 0.59 | 6.25 | 2.95 | 1.32 | 1.50 | 3.30 | 6.03 |
| 3DMatch (Zeng et al. [41]) | 0.00 | 5.67 | 2.86 | 21.25 | 2.41 | 10.91 | 6.98 | 3.62 | 4.65 | 6.48 | 10.29 |
| Scan2CAD (Avetisyan et al. [1]) | 36.20 | 36.40 | 34.00 | 44.26 | 17.89 | **70.63** | 30.66 | 30.11 | 20.60 | 35.64 | 31.68 |
| Direct 9DoF | 5.88 | 13.89 | 13.48 | 21.94 | 2.78 | 8.04 | 10.53 | 13.01 | 17.65 | 11.91 | 15.12 |
| Ours (no symmetry) | 11.11 | 29.27 | 29.29 | 68.26 | 20.41 | 16.26 | 41.03 | 40.12 | 14.29 | 30 | 40.51 |
| Ours (no SOCs) | 11.11 | 21.95 | 7.07 | 61.77 | 8.16 | 9.76 | 28.21 | 17.9 | 19.48 | 20.6 | 29.97 |
| Ours (no anchor) | **45.24** | **45.85** | 47.16 | 61.55 | **27.65** | 51.92 | 41.21 | 31.13 | **29.62** | 42.37 | 47.64 |
| Ours (no Procrustes) | 33.33 | 36.59 | 28.28 | 50.51 | 14.29 | 13.01 | 58.97 | 35.19 | 28.57 | 33.19 | 35.74 |
| **Ours (final)** | 38.89 | 41.46 | **51.52** | **73.04** | 26.53 | 26.83 | **76.92** | **48.15** | 18.18 | **44.61** | **50.72** |

Table 1: Accuracy comparison (%) on Scan2CAD [1]. We compare to state-of-the-art handcrafted feature descriptors (FPFH [30], SHOT [35], Li et al. [22]) as well as learned descriptors (3DMatch [41], Scan2CAD [1]) for CAD model alignment. These approaches consider correspondence finding and pose alignment optimization independently, while our end-to-end formulation can learn correspondences informed by alignment, achieving significantly higher CAD model alignment accuracy.

| Scene size | small | medium | large |
|---|---|---|---|
| Scene dim | $128 \times 96 \times 48$ | $144 \times 128 \times 64$ | $256 \times 320 \times 64$ |
| # objects | 7 | 16 | 20 |
| Scan2CAD [1] | 288.60s | 565.86s | 740.34s |
| Ours | **0.62s** | **1.11s** | **2.60s** |

Table 2: Runtime (seconds) of our approach on varying-sized scenes. Our end-to-end approach predicts CAD model alignment in a single forward pass, enabling very efficient CAD model alignment – several hundred times faster than previous data-driven approaches.

descriptors for all ShapeNet models for CAD model alignment, with the CAD autoencoder latent space constraining the object descriptor training for retrieval.

## 5. Results

We evaluate our proposed end-to-end approach for CAD model alignment in comparison to the state of the art as well as with an ablation study analyzing our differentiable Procrustes alignment loss and various design choices. We evaluate on real-world scans using the Scan2CAD dataset [1]. We use the evaluation metric proposed by Scan2CAD [1]; that is, the ground truth CAD model pool is available as input, and a CAD model alignment is considered to be successful if the category of the CAD model matches that of the scan object and the alignment falls within 20cm, 20°, and 20% for translation, rotation, and scale, respectively. For further evaluation on synthetic scans, we refer to the supplemental material.

In addition to evaluating CAD model alignment using the Scan2CAD [1] evaluation metrics, we also evaluate our approach on an unconstrained scenario with 3000 random CAD models as a candidate pool, shown in Figure 4. In this scenario, we maintain robust CAD model alignment accuracy with a much larger set of possible CAD models.

**Comparison to state of the art.** Table 1 evaluates our approach against several state-of-the-art methods for CAD model alignment, which establish correspondences and alignment independently of each other. In particular, we compare to several approaches leveraging handcrafted feature descriptors: FPFH [30], SHOT [36], Li et al. [22], as well as learned feature descriptors: 3DMatch [41], Scan2CAD [1]. We follow these descriptors with RANSAC to obtain final alignment estimation, except for Scan2CAD, where we use the proposed alignment optimization. Our end-to-end formulation, where correspondence learning can be informed by the alignment, outperforms these decoupled approaches by over 19.04%. Figure 3 shows qualitative visualizations of our approach in comparison to these methods.

**How much does the differentiable Procrustes alignment loss help?** We additionally analyze the effect of our differentiable Procrustes loss. In Table 1, we compare several different alignment losses. As a baseline, we train our model to directly regress the 9DoF alignment parameters with an $\ell_2$ loss. We then evaluate our approach with (final) and without (no Procrustes) our differentiable Procrustes loss. For CAD model alignment to 3D scans, our differentiable Procrustes alignment notably improves performance, by over 14.98%.

**How much does SOC prediction help?** We evaluate our SOC prediction on CAD model alignment in Table 1. We train our model with (final) and without (no SOCs) SOC prediction as well as with coordinate correspondence prediction but without symmetry (no symmetry). We observe that our SOC prediction significantly improves performance, by over 20.75%. Establishing SOCs is fundamental to our approach, as dense correspondences can produce more reliable alignment, and unresolved symmetries can lead to ambiguities and inconsistencies in finding ob-

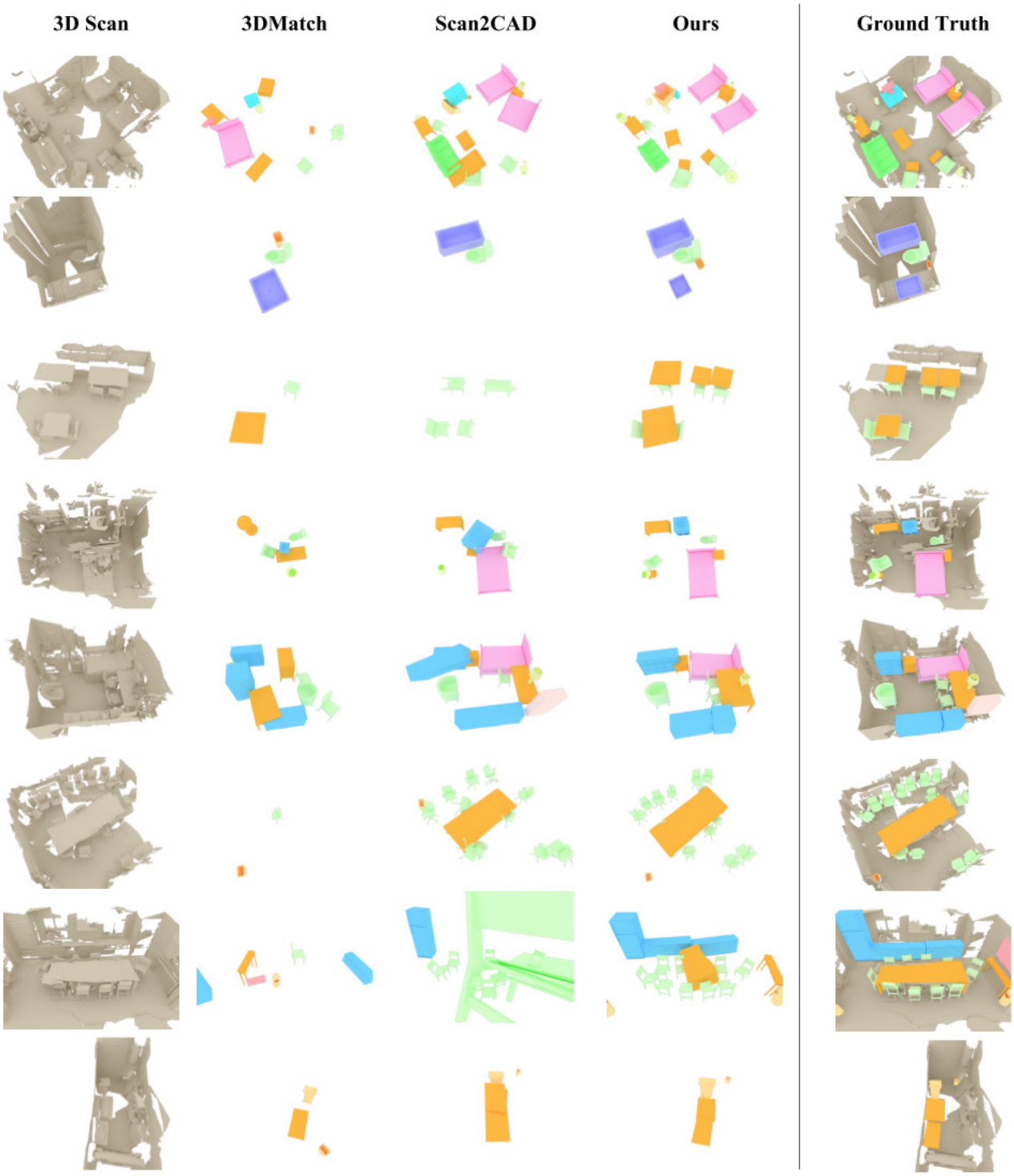



Figure 3: Qualitative comparison of CAD model alignment to ScanNet [7] scans. Our joint formulation of SOC correspondence prediction and differentiable Procrustes alignment enable both more accurate and robust CAD model alignment estimation across varying scene types and sizes.

ject correspondences. In particular, we also evaluate the effect of symmetry classification in our SOCs; explicitly predicting symmetry yields a performance improvement of $10.21\%$.

**What is the effect of using an anchor mechanism for object detection?** In Table 1, we also compare our CAD model alignment approach with (final) and without (no anchor) using anchors for object detection, where without anchors we predict only object center locations as a probability heatmap over the volumetric grid of the scan, but do not regress bounding boxes, and thus only crop a fixed neighborhood for the following SOCs and alignment. We observe that by employing bounding box regression, we can improve CAD model alignment performance, as this facilitates scale estimation and allows correspondence features to encompass the full object region.

### 5.1. Limitations

Although our approach shows significant improvements compared to state of the art, we believe there are directions for improvement. Currently, we focus on the objects in a scan, but do not consider structural components such as walls and floors. We believe, however, that our method could be expanded to detect and match plane segments in the spirit of structural layout detection such as PlaneRCNN [24]. In addition, we currently only consider the geometry of the scan or CAD; however, it is an interesting direction to consider finding matching textures in order to better visually match the appearance of a scan. Finally, we hope to incorporate our alignment algorithm in an online system that can work at interactive rates and give immediate feedback to the scanning operator.

## 6. Conclusion

We have presented an end-to-end approach that automatically aligns CAD models with commodity 3D scans, which is facilitated with symmetry-aware correspondences and a differentiable Procrustes algorithm. We show that by jointly training the correspondence prediction with direct, end-to-end alignment, our method is able to outperform existing state of the art by over $19.04\%$ in alignment accuracy. In addition, our approach is roughly $250\times$ faster than previous data-driven approaches and thus could be easily incorporated into an online scanning system. Overall, we believe that this is an important step towards obtaining clean and compact representations from 3D scans, and we hope it will open up future research in this direction.

## Acknowledgements

We would like to thank Justus Thies and Jürgen Sturm for valuable feedback. This work is supported by Occipital, the ERC Starting Grant Scan2CAD (804724), a Google Faculty Award, an Nvidia Professorship Award, and the ZD.B. We would also like to thank the support of the TUM-IAS, funded by the German Excellence Initiative and the European Union Seventh Framework Programme under grant agreement n° 291763, for the TUM-IAS Rudolf Mößbauer Fellowship.

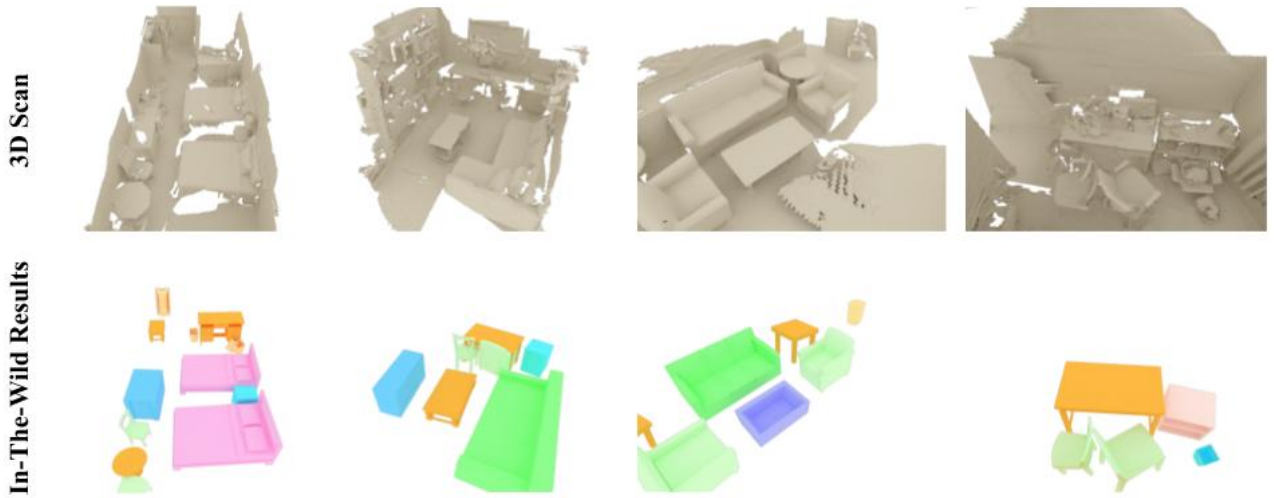



Figure 4: Our end-to-end CAD model alignment approach applied to an unconstrained set of candidate CAD models; here, we use a set of 3000 randomly selected CAD models from ShapeNetCore [4]. The results of our approach (bottom) show robust CAD model alignment performance in a scenario which is often reflected in real-world applications.

## References


[1] Armen Avetisyan, Manuel Dahnert, Angela Dai, Manolis Savva, Angel X. Chang, and Matthias Nießner. Scan2cad: Learning cad model alignment in rgb-d scans. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2019. 2, 3, 4, 5, 6

[2] Christopher Batty. SDFGen. https://github.com/christopherbatty/SDFGen. 5

[3] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. Dsac-differentiable ransac for camera localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6684–6692, 2017. 2

[4] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 1, 3, 8

[5] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. Robust reconstruction of indoor scenes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5556–5565. IEEE, 2015. 1, 2

[6] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312. ACM, 1996. 2, 5

[7] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 3, 7

[8] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (TOG)*, 36(3):24, 2017. 1, 2

[9] Angela Dai, Daniel Ritchie, Martin Bokeloh, Scott Reed, Jürgen Sturm, and Matthias Nießner. Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2018. 5

[10] Haowen Deng, Tolga Birdal, and Slobodan Ilic. Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors. In *European Conference on Computer Vision (ECCV)*. Springer, 2018. 2

[11] Haowen Deng, Tolga Birdal, and Slobodan Ilic. 3d local features for direct pairwise registration. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019. 2

[12] Bertram Drost and Slobodan Ilic. 3d object detection and localization using multimodal point pair features. In *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*, pages 9–16. IEEE, 2012. 2

[13] Saurabh Gupta, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Aligning 3D models to RGB-D images of cluttered scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4731–4740, 2015. 2

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[15] Ji Hou, Angela Dai, and Matthias Nießner. 3d-sis: 3d semantic instance segmentation of rgb-d scans. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2019. 5

[16] Binh-Son Hua, Quang-Hieu Pham, Duc Thanh Nguyen, Minh-Khoi Tran, Lap-Fai Yu, and Sai-Kit Yeung. Scenenn: A scene meshes dataset with annotations. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 92–101. IEEE, 2016. 3

[17] Binh-Son Hua, Quang-Trung Truong, Minh-Khoi Tran, Quang-Hieu Pham, Asako Kanezaki, Tang Lee, HungYueh Chiang, Winston Hsu, Bo Li, Yijuan Lu, et al. Shrec'17: Rgb-d to cad retrieval with objectnn dataset. 3

[18] Siyuan Huang, Siyuan Qi, Yixin Zhu, Yinxue Xiao, Yuanlu Xu, and Song-Chun Zhu. Holistic 3D scene parsing and reconstruction from a single RGB image. In *European Conference on Computer Vision*, pages 194–211. Springer, 2018. 2

[19] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568. ACM, 2011. 1, 2

[20] Hamid Izadinia, Qi Shan, and Steven M Seitz. Im2cad. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 2422–2431. IEEE, 2017. 2

[21] Maik Keller, Damien Lefloch, Martin Lambers, Shahram Izadi, Tim Weyrich, and Andreas Kolb. Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *3D Vision-3DV 2013, 2013 International Conference on*, pages 1–8. IEEE, 2013. 2

[22] Yangyan Li, Angela Dai, Leonidas Guibas, and Matthias Nießner. Database-assisted object retrieval for real-time 3D reconstruction. In *Computer Graphics Forum*, volume 34, pages 435–446. Wiley Online Library, 2015. 2, 6

[23] Joseph J Lim, Hamed Pirsiavash, and Antonio Torralba. Parsing ikea objects: Fine pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2992–2999, 2013. 2, 3

[24] Chen Liu, Kihwan Kim, Jinwei Gu, Yasutaka Furukawa, and Jan Kautz. Planercnn: 3d plane detection and reconstruction from a single image. *arXiv preprint arXiv:1812.04072*, 2018. 8

[25] Liangliang Nan, Ke Xie, and Andrei Sharf. A search-classify approach for cluttered indoor scene understanding. *ACM Transactions on Graphics (TOG)*, 31(6):137, 2012. 2

[26] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th*

*IEEE international symposium on*, pages 127–136. IEEE, 2011. 1, 2

[27] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (TOG)*, 2013. 1, 2

[28] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. Towards accurate multi-person pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4903–4911, 2017. 4

[29] Quang-Hieu Pham, Minh-Khoi Tran, Wenhui Li, Shu Xiang, Heyu Zhou, Weizhi Nie, Anan Liu, Yuting Su, Minh-Triet Tran, Ngoc-Minh Bui, et al. Shrec'18: Rgb-d object-to-cad retrieval. 3

[30] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 3212–3217. Citeseer, 2009. 2, 6

[31] Renato F Salas-Moreno, Richard A Newcombe, Hauke Strasdat, Paul HJ Kelly, and Andrew J Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1352–1359, 2013. 2

[32] Tianjia Shao, Weiwei Xu, Kun Zhou, Jingdong Wang, Dongping Li, and Baining Guo. An interactive approach to semantic modeling of indoor scenes with an RGBD camera. *ACM Transactions on Graphics (TOG)*, 31(6):136, 2012. 2

[33] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. *Proceedings of 30th IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 3, 5

[34] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3D: Dataset and methods for single-image 3D shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2974–2983, 2018. 2, 3

[35] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique signatures of histograms for local surface description. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 356–369, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. 6

[36] Federico Tombari, Samuele Salti, and Luigi Di Stefano. A combined texture-shape descriptor for enhanced 3d feature matching. In *2011 18th IEEE International Conference on Image Processing*, pages 809–812. IEEE, 2011. 2, 6

[37] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. *arXiv preprint arXiv:1901.02970*, 2019. 5

[38] Thomas Whelan, Stefan Leutenegger, Renato F Salas-Moreno, Ben Glocker, and Andrew J Davison. Elasticfusion: Dense slam without a pose graph. *Proc. Robotics: Science and Systems, Rome, Italy*, 2015. 1, 2

[39] Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. Objectnet3d: A large scale database for 3D object recognition. In *European Conference on Computer Vision*, pages 160–176. Springer, 2016. 3

[40] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on*, pages 75–82. IEEE, 2014. 3

[41] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 199–208. IEEE, 2017. 2, 6

[42] Chuhang Zou, Ruiqi Guo, Zhizhong Li, and Derek Hoiem. Complete 3D scene parsing from an RGBD image. *International Journal of Computer Vision (IJCV)*, 2018. 2

# SceneCAD: Predicting Object Alignments and Layouts in RGB-D Scans

Armen Avetisyan[1], Tatiana Khanova[3], Christopher Choy[2]
Denver Dash[3], Angela Dai[1], Matthias Nießner[1]

[1]Technical University of Munich [2]Stanford University [3]Occipital Inc.

This is the accepted but not the published version of the paper due to copyright restrictions.
Published version: https://doi.org/10.1007/978-3-030-58542-6_36

# SceneCAD: Predicting Object Alignments and Layouts in RGB-D Scans

Armen Avetisyan[1], Tatiana Khanova[3], Christopher Choy[2], Denver Dash[3], Angela Dai[1], and Matthias Nießner[1]

[1] Technical University of Munich [2] Stanford University [3] Occipital Inc.

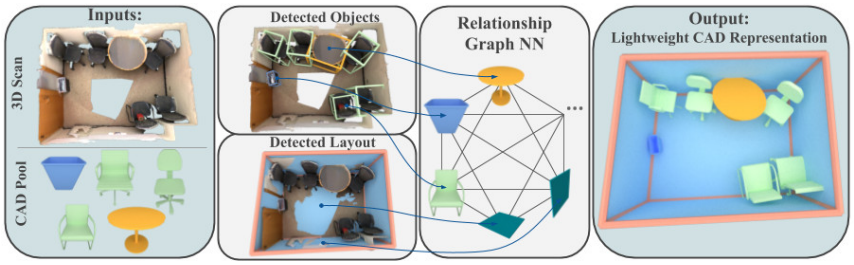



**Fig. 1.** Our method takes as input a 3D scan and a set of CAD models. We jointly detect objects and layout elements in the scene. Each detected object or layout component then forms a node in a graph neural network which estimates object-object relationships and object-layout relationships. This holistic understanding of the scene enables results in a lightweight CAD-based representation of the scene.

**Abstract.** We present a novel approach to reconstructing lightweight, CAD-based representations of scanned 3D environments from commodity RGB-D sensors. Our key idea is to jointly optimize for both CAD model alignments as well as layout estimations of the scanned scene, explicitly modeling inter-relationships between objects-to-objects and objects-to-layout. Since object arrangement and scene layout are intrinsically coupled, we show that treating the problem jointly significantly helps to produce globally-consistent representations of a scene. Object CAD models are aligned to the scene by establishing dense correspondences between geometry, and we introduce a hierarchical layout prediction approach to estimate layout planes from corners and edges of the scene. To this end, we propose a message-passing graph neural network to model the inter-relationships between objects and layout, guiding generation of a globally object alignment in a scene. By considering the global scene layout, we achieve significantly improved CAD alignments compared to state-of-the-art methods, improving from 41.83% to 58.41% alignment accuracy on SUNCG and from 50.05% to 61.24% on ScanNet, respectively. The resulting CAD-based representations makes our method well-suited for applications in content creation such as augmented- or virtual reality.

## 1 Introduction

The recent progress of 3D reconstruction of real-world environments from commodity range sensors has spurred interest in using such captured 3D data for applications across many fields, such as content creation, mixed reality, or robotics. State-of-the-art 3D reconstruction approaches can now produce impressively-robust camera tracking and surface reconstruction [29, 30, 7, 11].

Unfortunately, the resulting 3D reconstructions are not well-suited for direct use with many applications, as the geometric reconstructions remain incomplete (e.g., due to occlusions and sensor limitations), are often noisy or oversmoothed, and often consume a large memory footprint due to high density of triangles or points used to represent a surface at high resolution. There still remains a notable gap between these reconstructions and artist-modeled 3D content, which are clean, complete, and lightweight [16].

Inspired by these attributes of artist-created 3D content, we aim to construct a CAD-based scene representation of an input RGB-D scan, with objects represented by individual CAD models and scene layout represented by lightweight meshes. In contrast to previous approaches which have individually tackled the tasks of CAD model alignment [22, 2, 3] and of layout estimation [28, 25, 6], we observe that object arrangement is typically tightly correlated with the scene layout. We thus propose to jointly optimize for CAD model alignment and scene layout to produce a globally-consistent CAD-based representation of the scene.

From an input RGB-D scan along with a CAD model pool, we align CAD models to the scanned scene by establishing dense correspondences. To estimate the scene layout, we characterize the layout into planar elements, and propose a hierarchical layout prediction by first detecting corner locations, then predicting scene edges, and from sets of edges potentially presenting a layout plane, predicting the final layout. We then propose a graph neural network architecture for optimizing the relationships between objects and layout by predicting object-object relative poses as well as object-layout support relationships. This optimization guides both object and layout arrangement to be consistent with each other. Our approach is fully-convolutional and trained end-to-end, generating a CAD-based scene representation of a scan in a single forward pass.

In summary, we present the following contributions:

- We formulate a lightweight heuristic-free 3D layout prediction algorithm that hierarchically predicts corners, edges and then planes in an end-to-end fashion consisting of only $\approx 1M$ trainable parameters generating satisfactory layouts without the need for extensive heuristics.
- We present a scene graph network that learns relationships between objects and scene layout, enabling globally consistent CAD model alignments and results in a significant increase in prediction performance in both synthetic as well as real-world datasets.
- We introduce a new richly-annotated real-world scene layout dataset consisting of 1151 CAD shells and wireframes on top of the ScanNet RGB-D dataset, allowing large-scale data-driven training for layout estimation.

## 2 Related Work

*CAD model alignment* Aligning an expert-generated 3D model or a 3D template to 3D scan data has been studied widely due to its wide range of applications, for instance motion capture [4], 3D object detection and localization [12, 13, 39], and scene registration [35]. Our aim is to leverage large-scale datasets of CAD models to reconstruct a lightweight, semantically-informed, high-quality CAD representation of an RGB-D scan of a scene. Several approaches have been developed to retrieve and align CAD models from a shape database and align them in real time to a scan during the 3D scanning process [19, 22], although their use of handcrafted features for geometric scan-to-CAD matching limit robustness.

Zeng et al. [40] developed a learned feature extractor using a siamese network design for geometric feature matching, which can be employed for scan-to-CAD feature matching, though this remains difficult due to the domain gap between synthetic CAD models and real-world scans. Avetisyan et al. [2] proposed a scan-to-CAD retrieval and alignment approach leveraging learned features to detect objects in a 3D scan and establish correspondences across the domain gap of scan and CAD. They later built upon this work to develop a fully end-to-end trainable approach for this CAD alignment task [3]. For such approaches, each object is considered independently, whereas our approach exploits contextual information from object-object and object-layout to produce globally consistent CAD model alignment and layout estimation.

Other approaches retrieve and align CAD models to RGB images [23, 38, 33]; our work instead focuses on geometric alignment of CAD models and layout.

*Graph neural networks and relational inference in 3D.* Recent developments in graph inference and graph neural networks have shown significant promise for inference on 3D data. Recently, various approaches have viewed 3D meshes as graphs in order find correspondences between 3D shapes [5], deform a template mesh to fit an image observation of a shape [36], or generate a mesh model of an object [10], among other applications. Learning on graphs has also shown promise for estimating higher-level relational information in scenes, as a scene graph. 3D-RelNet [21] predicts 3D shapes and poses from single RGB images and establish pairwise pose constraints between objects to improve overall prediction quality. Our approach is similarly inspired to establish relationships between objects; we additionally employ relationships between objects and structural components (i.e., walls, floors, and ceilings), which considerably inform object arrangement. Armeni et al. [1] propose a unified hierarchical structure that hosts building, room, and object relationships into one 3D scene graph. They leverage this graph structure to generate scene graphs from 2D images. Our approach focuses on leveraging relational information to reconstruct imperfect scans with a CAD-based representation for each object and layout element.

*Layout estimation.* Various layout estimation approaches have been developed to infer structural information from RGB and RGB-D data. Scan2BIM [28] generates building information models (BIM) from 3D scans by detecting planes and finding

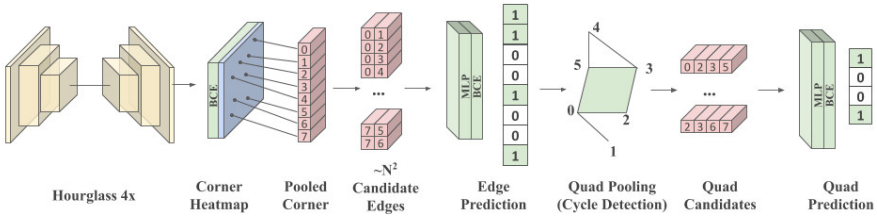



**Fig. 2.** Layout estimation as planar quad structures. Layout components are characterized as planar elements which are detected hierarchically. From an input scene, corners of these layout elements are predicted in heatmap fashion leveraging non-maximum suppression. From these predicted corners, edges are then predicted for each possible pair of corners as a binary classification task. From the predicted edge candidates, valid quads of four connected edges are considered as candidate layout elements, with a binary classification used to produce the final layout prediction.

plausible intersections to produce room-level segmentation of floors, ceilings and walls under Manhattan-style constraints. PlaneRCNN [24] and PlaneNet [26] propose deep neural network architectures to detect planes from RGB images and estimate their 3D parameters. FloorNet [25] estimates a 2D Manhattan-style floorplan representation for an input RGB-D scan using a point-based neural network architecture. Floor-SP [6] relaxes the Manhattan constraints with an integer programming formulation, and produces more robust floorplan estimation. In contrast to these layout estimation approaches, our focus lies in leveraging global scene relations between objects as well as structural elements in order to produce a CAD-based representation of the scene.

*Single view 3D reconstruction.* Holistic 3D Scene Parsing [18] parses a single RGB image and reconstruct a holistic 3D arrangements of CAD models jointly optimizing for 3D object detection, scene layout and hidden human context. Zou et al. [41] infers a complete interpretation of the scene from a single RGBD frame where objects and scene layout are predicted in data-driven fashion. In contrast to single view reconstruction, our approach aims towards holistic scene understanding that can operate on large-scale 3D scenes while consuming only a few seconds of runtime at test time.

## 3 SceneCAD: Joint Object Alignment and Layout Estimation

The input scan is represented as a sparse 3D voxel grid of the occupied surface geometry carrying fused RGB data. The scan is first encoded by a series of sparse 3D convolutional layers [8] to produce a feature volume $F'$. The sparse output $F'$ is then densified into a dense 3D feature grid $F \in \mathbb{R}^{N_f \times N_x \times N_y \times N_z}$ where $N_F$ is the number of channels in the feature and $N_x, N_y$, and $N_z$ are the resolution

of the feature along x, y, and z axis respectively. Note that the encoder serves as backbone for proceeding modules. Hence, $F$ is the input to the CAD alignment module as well as the layout estimation module.

Based on $F$, we detect objects along with their bounding box in the object detection module and layout planes in the layout detection module. We then establish our relational inference by formulating a message-passing graph neural network on the predicted objects and layout planes, where each node represents an object or layout plane, with losses on edge relationships representing relative poses and support. Finally, we predict a set of retrieved CAD models along with their 9-DoF poses (3 translation, 3 rotation, and 3 scale) for every detected object.

The message-passing graph neural network helps to inform objects of both relations between other objects as well as with the scene layout, e.g., certain types of furniture such as beds and chairs are typically directly supported by a floor, chairs near a table often face the table. This joint optimization thus helps to enable globally consistent CAD model alignment in the final output.

### 3.1 Layout Prediction

The indoor scene of interest in our problem consists of planar or quadrilateral components such as walls, floors, and ceilings. However, some of these planar elements create complex geometry such as bars, beams, or other structures that effectively make template-matching approach to find the room layout challenging. Thus, we propose a bottom-up approach that predicts corners, edges, and planar elements sequentially to predict the room layout. Our layout prediction pipeline is structured hierarchically: first predicting the corner locations, then predicting edges between the corners, and finally extracting quads from the predicted edges. We visualize the overview of the pipeline on Figure 2.

Corners are predicted by a convolutional network that decodes $F$ to its original dimension by predicting a heatmap; i.e. a voxel-wise score that indicates a *cornerness* likeliness. The loss for this predicted heatmap is a voxel-wise binary cross-entropy classification loss in conjunction with a softmax and a negative log-likelihood over the entire voxel grid where the problem is formulated as a spatial multi-class problem. This is structured as an encoder-decoder, where the bottleneck lies at a spatial reduction of $4\times$. Note that we make predictions for corners which have not been observed in the input scan (e.g., due to occlusions, c.f.). See supplemental material for a visual illustration of the layout prediction pipeline. From the output corner heatmap, we apply a non-maximum suppression to filter out weak responses, and define the final corner predictions as a set of xyz coordinates $\mathcal{V} = \{\mathbf{v}_i\}_i$, $\mathbf{v}_i = [x_i, y_i, z_i]$.

We the predict the layout edges from the predicted corners $\mathcal{V}$. We construct the candidate set of edges by taking all pair-wise combinations of corners $\mathbf{e}_{ij} = (\mathbf{v}_i, \mathbf{v}_j)$ for all $i \in [1, ..., |\mathcal{V}|]$ and $j \in [1, ..., i-1]$. We denote all edges as $\mathcal{E} = \{\mathbf{e}_{ij}\}_{ij}$. From the pool of candidate edges we predict a set of edges that belongs to the scene structure using a graph neural network. Specifically, for each potential edge $\mathbf{e}_{ij} = (\mathbf{v}_i, \mathbf{v}_j)$, we extract corresponding features from the vertex prediction

convolutional network, $F[\mathbf{v}_i], F[\mathbf{v}_j]$ where $F[\cdot]$ denotes the feature vector at the specified $x, y, z$ coordinate. We concatenate these features along with the normalized coordinates to form an input feature vector for each edge $\mathbf{f}_{\mathbf{e}_{ij}} = [F[\mathbf{v}_i], F[\mathbf{v}_j], \mathbf{N}(\mathbf{v}_i), \mathbf{N}(\mathbf{v}_j)]$. For each edge we construct two feature descriptors with alternating order of corner features $\mathbf{f}_{\mathbf{e}_{ji}}$ to mitigate the effect of order dependency. We feed these concatenated features into a graph network, which we train with edge-wise binary cross entropy loss against ground truth edges. As the vertex predictions have uncertainty, we label edges with predicted vertices within a certain radius from the ground truth layout vertices to be positives. This edge prediction limits the set of candidate layout quads which would otherwise be $O\left(\binom{|\mathcal{V}|}{4}\right)$.

From these predicted edges, we then compute the set of candidate layout quads as the set of planar, valid 4-cycles within these edges $\mathbf{q}_{ijkl} = \{\mathbf{e}_{ij}, \mathbf{e}_{jk}, \mathbf{e}_{kl}, \mathbf{e}_{li}\}$. To detect valid cycles, we use the depth-first-search cycle detection algorithm We predict the final set of layout quads as either positive or negative where the positive predictions constitute the scene layout, decomposed as quads. The feature descriptor for a candidate quad is constructed by concatenating the features from $F$ corresponding to the corner locations of its vertices and normalized corner locations, $\mathbf{q}_{ijkl} = [F[\mathbf{v}_i], F[\mathbf{v}_j], F[\mathbf{v}_k], F[\mathbf{v}_l], \mathbf{N}(\mathbf{v}_i), \mathbf{N}(\mathbf{v}_j), \mathbf{N}(\mathbf{v}_k), \mathbf{N}(\mathbf{v}_l)]$. Similar to the edge features, every quad feature descriptor is 4-way permuted $\mathbf{q}_{jkli}, \mathbf{q}_{klij}$, and $\mathbf{q}_{lijk}$ in order to mitigate order-dependency. This feature is input to an MLP followed by a binary cross entropy loss. From these predicted quads, we recover the scene layouts without heuristic post-processing.

### 3.2 CAD Model Alignment

Along with the room layout, we aim to find and align light-weight CAD models to objects in the scanned scene. To this end, we propose a CAD model alignment pipeline that detects objects, retrieves CAD models, and finds transformations that aligns the CAD model to the scanned scene. First, we use a single-shot anchor-based object detector to identify objects [17], using the features from the backbone we extracted ($\mathbf{F}$) from the previous stage. We then filter the predicted anchors with non-maximum suppression following the standard single-shot object detection pipeline [27]. Given this set of object bounding boxes $\mathcal{B}$, we extract $N_d \times N_d \times N_d$ feature volume $F_o$ for all $o \in [1, ..., |\mathcal{B}|]$ from the feature map $F$ around the object anchor $a_o$. We use this feature volume for CAD model retrieval and alignment. A corresponding CAD model is retrieved by calculating an object descriptor of length 512 and searching the nearest neighbor CAD model from an shared embedding space. This shared embedding space is established by minimizing the distance between descriptors of scanned objects and their CAD counterpart with an L1 loss during training.

Finally, given the nearest CAD model for all object anchors, we find dense correspondences between the CAD model and the feature volume $F_o$. Dense correspondences are trained through an explicit voxel-wise L1 regression loss. We use Procrutes [15] to estimate a rotation matrix and an L1 distance loss with

respect to the groundtruth rotation matrix to further enhance correspondence quality. Note that the Procrutes method yields a transformation matrix through the Singular Value Decomposition which is differentiable, allowing for end-to-end training.

### 3.3 Learning Object and Layout Relationships

From our layout prediction and CAD model alignment, we obtain a set of layout quads and aligned CAD models, both obtained independently from the same backbone features. However, this can result in globally inconsistent arrangements; for instance, objects passing through the ground floor, or shelves misaligned with walls. We thus propose to learn the object-layout as well as object-object relationships as a proxy loss used to guide the CAD model alignments and layout quads into a globally consistent arrangement.

We construct this relationship learning as a graph problem, where the set of objects and layout quads form the nodes of the graph. Edges are constructed between every object-object node-pair and every object-quad node-pair, forming a graph on which we formulate a message-passing graph neural network.

Each node of the graph is characterized by a feature vector of length 128. For objects this feature vector is obtained by pooling the object feature volume to $8^3$ resolution, followed by linearization. For layout quads, this feature vector is constructed by concatenating the features from $F$ or the associated corner locations, upon which an MLP is applied to obtain a 128-dimensional vector.

Figure 3 shows an overview of our message-passing network. Messages are passed from nodes to edges for a graph $G = (V, E)$, with nodes $v_i \in V$ and edges $e_{j,k} = (v_j, v_k) \in E$. We define the message passing similar to [14, 20, 10, 37]:

$$v \rightarrow e : \mathbf{h}_{i,j}^{t+1} = f_e(\text{concat}(\mathbf{h}_i^t, \mathbf{h}_j^t - \mathbf{h}_i^t))$$

where $\mathbf{h}_i^t$ is the feature corresponding to vertex $v_i$ at message passing step $t$, $\mathbf{h}_{i,j}^t$ is the feature corresponding edge $e_{i,j}$ at step $t$, and $f_e$ represents an MLP. That is, edges features are computed as the concatenation of its constituent vertices.

We then take these output edge features from the message passing and perform a classification of various relationships using a cross entropy loss. We describe the relationships as follows, which we chose as they do not require extra manual annotation effort given existing ground truth CAD alignments and scene layout; see Section 4.2 for more detail regarding extraction of ground truth object and layout relationships. For object-layout relationships, we formulate a 3-class classification task for support relations, predicting *horizontal support*, *vertical support*, or no support. Only one relationship per object-layout pair is allowed. For object-object relationships, we predict the angular difference between the front-facing vectors of the respective objects, in order to recognize common relative arrangements of objects (e.g., chairs often face tables). This is trained with a 6-class cross entropy loss where the angular deviation up to 180◦ is discretized into 6 bins.

Here, the relationship prediction adds a proxy loss to the model in Figure 2 which inter-correlates object and layout alignments, implicitly guiding the CAD model alignment and layout quad estimation to become more globally consistent.

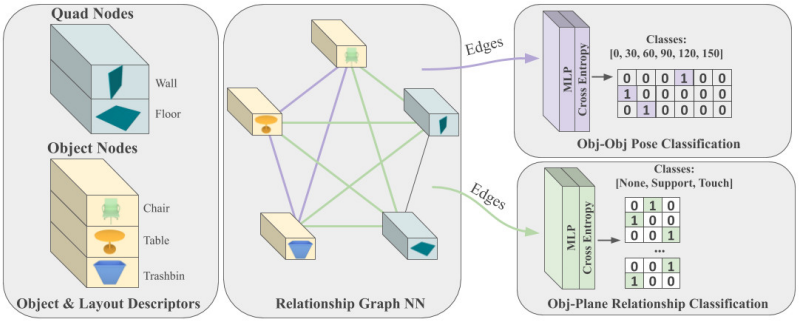



**Fig. 3.** Object and layout relational prediction. We establish a message-passing neural network in order to predict object-object and object-layout relations. The inputs are feature descriptors of detected objects and quads pooled to the same size, and the output is relationship classification between objects and layout elements, as well as pose relations between objects. Note this relational inference is fully differentiable, enabling end-to-end prediction.

## 4 Object+Layout Dataset

To train and evaluate our method, we introduce a new dataset of 1151 CAD layout annotations to the real-world RGB-D scans of the ScanNet dataset [9]. These layout annotations, in addition to the CAD annotations of Scan2CAD [2] to ScanNet scenes, inform our method and evaluation on real-world scan data.

In order to obtain these room layout annotations, we use a semi-automated annotation process. We then automatically extract the object-object and object-layout relations.

### 4.1 Extraction of Scene Layouts

We performed a semi-automatic layout annotation for ScanNet scene data. First, large planar surfaces are detected using RANSAC on the reconstructed scans. We then employ a manual refinement step to modify potential errors in the automatic extraction. The surface extraction is preceded by a semantic instance segmentation to obtain wall, floor, ceiling, window, door, etc. instances. RANSAC is then applied to extract 3D planes from each instance. Planes that fall below a threshold will be merged or connected. All planes are projected onto the floor plane and through a set of various heuristics the most plausible intersection

points are selected to ultimately become corner points for the final layout. The room height is either estimated by the maximum height of the detected wall instances or is spanned by the ceiling.

Following the proposals given by RANSAC, we then manually verified which proposals were plausible. This step is relatively quick ($\approx$ 2min per scene) and indicated that the RANSAC produced 1151 plausible initial layouts. These layouts were then refined through a manual annotation process. We developed a Blender[4]-based tool was introduced for the layout refinement, allowing annotators to edit/merge/delete corner junctions as well as add or modify edges and planes. All automatically generated layouts were verified and refined by two student annotators ($\approx 15min$ per scene). An illustration of layouts annotation samples on ScanNet can be found in the supplemental.

### 4.2 Extraction of Object and Layout Relationships

To support learning global scene relationships, we extract object and layout relations to supervised our message-passing approach to learning relationships. We opt to learn relations which can be automatically extracted from given CAD model and layout annotations.

We extract object-object and object-layout relationships. For the object-object case, we compute the angular difference between the front-facing vectors of each object where symmetrical properties are ignored; in practice, we compute this on-the-fly during the training process.

Relationships between objects and layout elements are established by support:

- A vertical **support** relationship between a layout element and an object is valid if the bottom side of the bounding box of the object within close proximity to and close to parallel to the layout element.
- A horizontal **touch** relationship is valid if the left, right, front or back side of the bounding box of the object is within close proximity to and close to parallel to the layout element.

These relations are extracted through an exhaustive search. That is, each pair of object-layout is checked for vertical support or horizontal touch. To estimate proximity of objects, we expand the bounding box of the objects by $\tau_p$, and expand the sides of the bounding boxes of the layout elements by $\tau_p$. We then consider the object and layout element to be in close proximity if their expanded bounding boxes overlap. We use $\tau_p = 0.2$ meters for all experiments.

### 4.3 Synthetic Data

We additionally evaluate our approach on synthetic data, where CAD object and layout ground truth are provided in the construction of the synthetic 3D scenes. We use synthetic scenes from the SUNCG dataset [32]. SUNCG contains models of indoor building environments including CAD models and room layouts. Layout

[4] https://www.blender.org

| | bathtub | bookshelf | cabinet | chair | display | other | sofa | table | trashbin | class avg. | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| FPFH (Rusu et al. [31]) | 0.00 | 1.92 | 0.00 | 10.00 | 0.00 | 5.41 | 2.04 | 1.75 | 2.00 | 2.57 | 4.45 |
| SHOT (Tombari et al. [34]) | 0.00 | 1.43 | 1.16 | 7.08 | 0.59 | 3.57 | 1.47 | 0.44 | 0.75 | 1.83 | 3.14 |
| Li et al. [22] | 0.85 | 0.95 | 1.17 | 14.08 | 0.59 | 6.25 | 2.95 | 1.32 | 1.50 | 3.30 | 6.03 |
| 3DMatch (Zeng et al. [40]) | 0.00 | 5.67 | 2.86 | 21.25 | 2.41 | 10.91 | 6.98 | 3.62 | 4.65 | 6.48 | 10.29 |
| Scan2CAD (Avetisyan et al. [2]) | 36.20 | 36.40 | 34.00 | 44.26 | 17.89 | **70.63** | 30.66 | 30.11 | 20.60 | 35.64 | 31.68 |
| End2End (Avetisyan et al. [3]) | 38.89 | 41.46 | 51.52 | 73.04 | 26.53 | 26.83 | 76.92 | **48.15** | 18.18 | 44.61 | 50.72 |
| Ours (dense) | 33.33 | 39.39 | **58.62** | 70.76 | 28.57 | 33.72 | 50.00 | 34.55 | 23.73 | 41.41 | 51.05 |
| Ours (dense) + obj-obj | 44.44 | **54.55** | 49.15 | 68.05 | 37.50 | 36.05 | 61.11 | 42.01 | 27.12 | 46.66 | 52.97 |
| Ours (dense) + layout | **54.55** | 47.37 | 38.33 | 71.11 | 32.88 | 28.05 | 62.86 | 37.91 | **32.26** | 45.04 | 52.06 |
| Ours (dense) full | 39.39 | 42.11 | 48.33 | 74.32 | 42.47 | 36.59 | 62.86 | 36.26 | 30.65 | 45.89 | 54.33 |
| Ours (sparse) | 42.42 | 39.47 | 51.67 | 77.28 | 45.21 | 28.05 | 77.14 | 37.91 | 25.81 | 47.22 | 55.77 |
| Ours (sparse) + obj-obj | 42.42 | 44.74 | 50.00 | 77.53 | 43.84 | 30.49 | 74.29 | 39.56 | **32.26** | 48.35 | 56.70 |
| Ours (sparse) + layout | 45.45 | 42.11 | 48.33 | 78.27 | 42.47 | 31.71 | 77.14 | 37.36 | 27.42 | 47.81 | 56.29 |
| Ours (sparse) full | 42.42 | 36.84 | 58.33 | **81.23** | **50.68** | 40.24 | **82.86** | 45.60 | **32.26** | **52.27** | **61.24** |

**Table 1.** CAD alignment evaluation on ScanNet Scan2CAD data [9, 2]. Our final method (last row), incorporating contextual information from both object-object relationships and object-layout relationships, outperforms the baseline by a notable margin of 10.52%.

components are given and hence extraction into planar quads can be performed automatically. To generate the input partial scans, we virtually scan the scenes to produce input scans similar to real-world scenarios, following previous approaches to generate synthetic partial scan data [17].

Object and layout relational information was extracted following the same procedure for ScanNet data.

## 5 Results

### 5.1 CAD Alignment Performance

We evaluate our method on synthetic SUNCG [32] scans as well as real-world ScanNet [9] scans in Tables 3 and 1, respectively. We follow the CAD alignment evaluation metric proposed by [2], which measures alignment accuracy where an alignment is considered successful if it falls within 20cm, 20°, and 20% scale of the ground truth. On both SUNCG and ScanNet scans we compare to several state-of-the-art handcrafted geometric feature matching approaches [31, 34, 22] and learned approaches [40, 2, 3]. We additionally show qualitative comparisons in Figures 6 and 5.2. On synthetic scan data we outperform the strongest baseline by 16.58%, and improve by 10.52% on real scan data. This demonstrates the benefit of leveraging global information regarding object and layout relations in improving object alignments.

We also perform an ablation study on the various design choices and impact of relation information. We evaluate a dense convolutional backbone for our network architecture (*dense*) in contrast to our final sparse convolutional backbone leveraging the sparse convolutions proposed by [8]. We additionally show that the object-to-object relational inference (*obj-obj*) as well as layout estimation (*layout*) improve upon no relational inference, and our full method incorporating both object and layout relational inference, the most contextual information, yields the best performance.

### 5.2 Layout Prediction

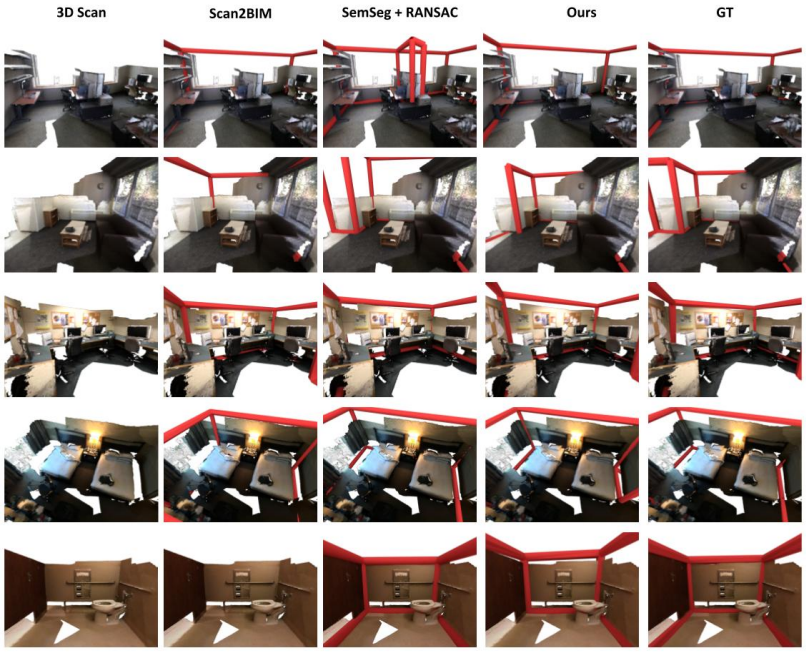



**Fig. 4.** Qualitative comparison of our layout estimation on the ScanNet dataset [9]. Layout elements are highlighted with their wireframes. Our method provides a very lightweight, learned approach ($\approx 1M$ trainable parameters) for layout estimation.

For the final quad prediction we achieve a F1-score of 37.9% on ScanNet and 69.6% on SUNCG. Corners are considered as successfully detected if the predicted corner is within a radius of $40cm$ from the ground truth corner. Edges are considered as correctly predicted if they connect the same corners as the ground truth edges. Similarly, correctly predicted quads are spanned by the same 4 corners as the associated ground truth quad. We aim to achieve a high recall for corners and edges due to our hierarchical prediction. We achieve robust results on both datasets, although ScanNet is notably more difficult as many scenes can miss views of entire layout components (e.g., missing ceilings).

## 6 Limitations

While the focus of this work was to show improved scene understanding through joint prediction of objects **and** layouts, we believe there is potential for further achievements. For instance, our layout prediction method is bound to predict

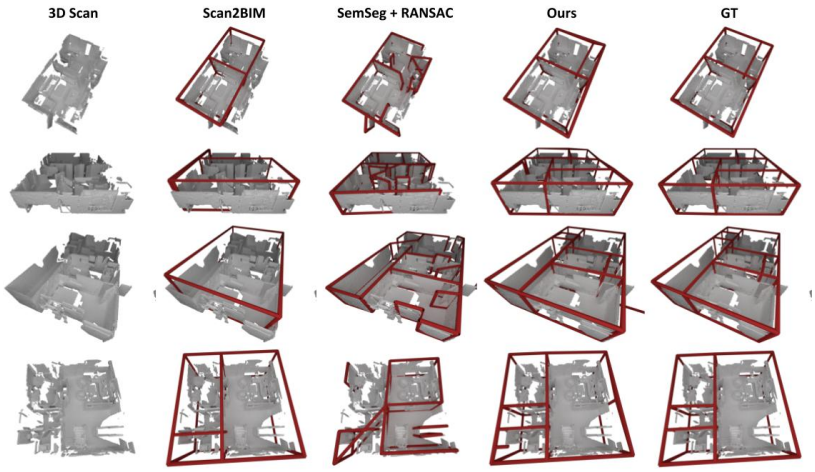



**Fig. 5.** Layout estimation on SUNCG [32] scans. Layout elements are highlighted with their wireframes. Our method excels with its simplicity, especially for very large and complex scenes where heuristics to determine intersections tend to struggle.

| # voxels | 18K | 42K | 71K |
|---|---|---|---|
| Scene extent | $2.6m^2 \times 2.4m^2$ | $3.2m^2 \times 3.5m^2$ | $7.5m^2 \times 6.2m^2$ |
| # objects | 1 | 5 | 26 |
| Timing | **1.9s** | **2.0s** | **2.60s** |

**Table 2.** Runtime (seconds) of our approach on different test scenes categorized into small, medium and large.

| | bed | cabinet | chair | desk | dresser | other | shelves | sofa | table | class avg. | avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| SHOT (Tombari et al. [34]) | 13.43 | 3.23 | 10.18 | 2.78 | 0.00 | 0.00 | 1.75 | 3.61 | 11.93 | 5.21 | 6.30 |
| FPFH (Rusu et al. [31]) | 38.81 | 3.23 | 7.64 | 11.11 | 3.85 | 13.21 | 0.00 | 21.69 | 11.93 | 12.39 | 9.94 |
| Scan2CAD (Avetisyan et al. [2]) | 52.24 | 17.97 | 36.00 | 30.56 | 3.85 | 20.75 | 7.89 | 40.96 | 43.12 | 28.15 | 29.23 |
| End2End (Avetisyan et al. [3]) | 71.64 | 32.72 | 48.73 | 27.78 | 38.46 | 37.74 | 14.04 | 67.47 | 45.87 | 42.72 | 41.83 |
| Ours (dense) | 63.89 | 35.16 | 56.82 | 39.02 | 30.00 | 38.85 | 29.17 | **76.67** | 31.03 | 44.51 | 44.48 |
| Ours (dense) + obj-obj | 77.78 | 36.26 | 53.03 | 41.46 | 40.00 | 47.48 | 20.83 | **76.67** | 25.86 | 46.60 | 46.41 |
| Ours (dense) + layout | 75.00 | 37.04 | 60.68 | 37.14 | 38.89 | 45.53 | **33.33** | 72.41 | 32.08 | 48.01 | 48.33 |
| Ours (dense) full | **81.25** | 40.00 | 51.92 | 45.45 | 41.18 | 49.17 | 31.58 | 75.86 | **46.00** | 51.38 | 50.41 |
| Ours (sparse) | 54.29 | 42.55 | 66.67 | 48.57 | **44.44** | 57.60 | 27.27 | 57.89 | 36.84 | 48.46 | 52.31 |
| Ours (sparse) + obj-obj | 74.29 | 40.43 | 70.09 | **65.71** | 27.78 | **60.80** | 27.27 | 55.26 | 38.60 | 51.14 | 55.27 |
| Ours (sparse) + layout | 65.71 | 42.55 | **77.78** | 54.29 | 38.89 | **60.80** | 22.73 | 57.89 | 45.61 | 51.81 | 57.12 |
| Ours (sparse) full | 71.43 | **43.62** | **77.78** | 54.29 | 38.89 | **60.80** | 22.73 | 68.42 | 45.61 | **53.73** | **58.41** |

**Table 3.** CAD alignment accuracy on SUNCG [32] scans. Our final method (last row) goes beyond considering only objects and jointly estimates room layout and object and layout relationships, resulting in significantly improved performance.

quad planes only and hence more sophisticated methods could be used for

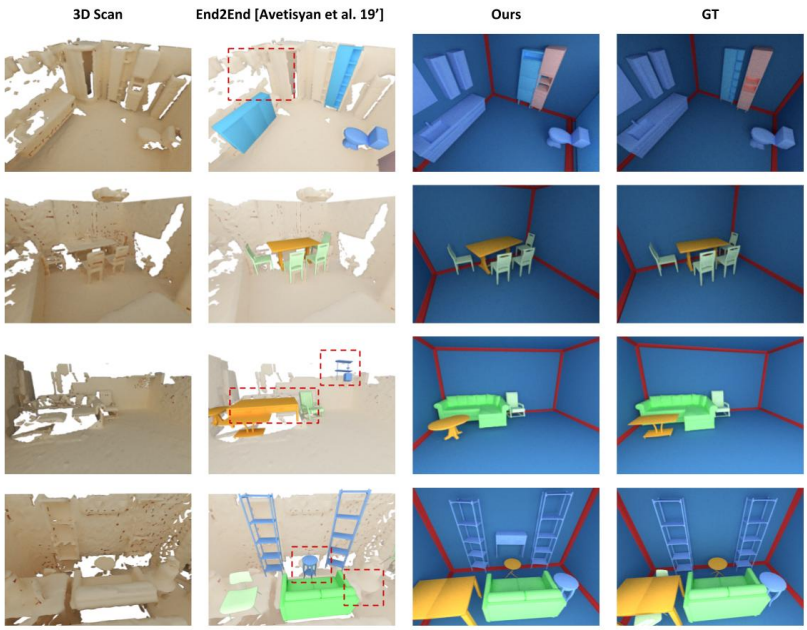



**Fig. 6.** Qualitative CAD alignment and layout estimation results on SUNCG [32] scans. Our joint estimation approach produces more globally consistent CAD alignments and generates additionally room layout applicable for VR/AR applications.

more accurate layout estimation. Also, we used a very lightweight graph neural network for message passing. One could use a more sophisticated method for more accurate relationship prediction and a richer set of relationships that may contain functionality relationships, spatial relationships or room semantic relationships.

## 7 Conclusion

In this work we formulated a method to digitize 3D scans that goes beyond the focus of objects in the scene. We propose a novel method that estimates the layout of the scene by sequentially predicting corners, then edges and finally quads in a fully differentiable way. The estimated layout is used in conjunction with an object detector to predict contact relationships between objects and the layout and ultimately to predict a CAD arrangement of the scene. We can show that objects and the surrounding (scene layout) go hand in hand and are a crucial factor towards full scene digitization and scene understanding. Objects in the scene are often not arbitrarily arranged, for instance often cabinets are leaned at walls or a table is surrounded by chairs in a dining room, hence we leverage the inherent coupling between objects and layout structure in the learning process. Our approach improves global CAD alignment accuracy by

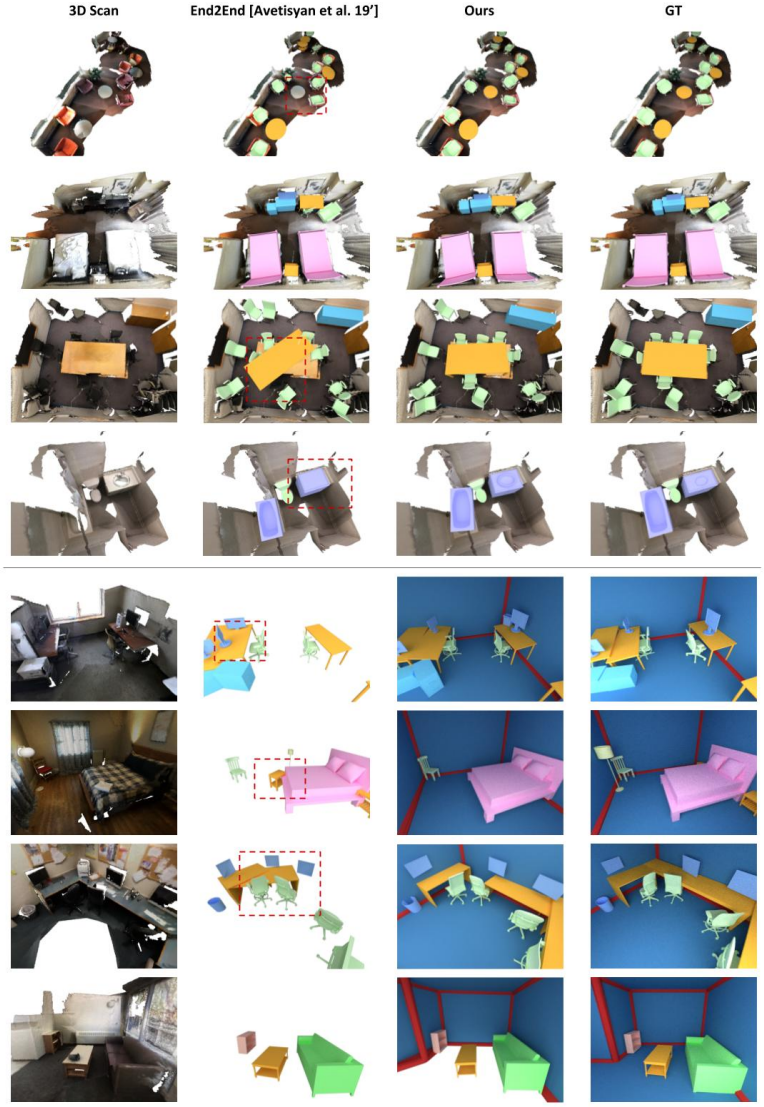



**Fig. 7.** Qualitative CAD alignment and layout estimation results on ScanNet [9] scans (zoomed in views on the bottom). Our approach incorporating object and layout relationships produces globally consistent alignments along with the room layout.

learning those patterns on both real and synthetic scans. We hope that we can encourage further research towards this avenue, and see as next immediate steps for future work the necessity of texturing digitized shapes in order to enhance the immersive experience in VR environments.

## References


1. Armeni, I., He, Z.Y., Gwak, J., Zamir, A.R., Fischer, M., Malik, J., Savarese, S.: 3d scene graph: A structure for unified semantics, 3d space, and camera (2019)
2. Avetisyan, A., Dahnert, M., Dai, A., Savva, M., Chang, A.X., Nießner, M.: Scan2cad: Learning cad model alignment in rgb-d scans. In: Proc. Computer Vision and Pattern Recognition (CVPR), IEEE (2019)
3. Avetisyan, A., Dai, A., Nießner, M.: End-to-end cad model retrieval and 9dof alignment in 3d scans. arXiv preprint arXiv:1906.04201 (2019)
4. Bogo, F., Kanazawa, A., Lassner, C., Gehler, P., Romero, J., Black, M.J.: Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In: European Conference on Computer Vision. pp. 561–578. Springer (2016)
5. Bronstein, M.M., Bruna, J., LeCun, Y., Szlam, A., Vandergheynst, P.: Geometric deep learning: going beyond euclidean data. IEEE Signal Processing Magazine **34**(4), 18–42 (2017)
6. Chen, J., Liu, C., Wu, J., Furukawa, Y.: Floor-sp: Inverse cad for floorplans by sequential room-wise shortest path. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2661–2670 (2019)
7. Choi, S., Zhou, Q.Y., Koltun, V.: Robust reconstruction of indoor scenes. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5556–5565. IEEE (2015)
8. Choy, C., Gwak, J., Savarese, S.: 4d spatio-temporal convnets: Minkowski convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3075–3084 (2019)
9. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proc. Computer Vision and Pattern Recognition (CVPR), IEEE (2017)
10. Dai, A., Nießner, M.: Scan2mesh: From unstructured range scans to 3d meshes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5574–5583 (2019)
11. Dai, A., Nießner, M., Zollhöfer, M., Izadi, S., Theobalt, C.: Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. ACM Transactions on Graphics (TOG) **36**(3), 24 (2017)
12. Drost, B., Ilic, S.: 3d object detection and localization using multimodal point pair features. In: 2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission. pp. 9–16. IEEE (2012)
13. Engelmann, F., Stückler, J., Leibe, B.: Joint object pose estimation and shape reconstruction in urban street scenes using 3d shape priors. In: German Conference on Pattern Recognition. pp. 219–230. Springer (2016)
14. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. arXiv preprint arXiv:1704.01212 (2017)
15. Goodall, C.: Procrustes methods in the statistical analysis of shape. Journal of the Royal Statistical Society: Series B (Methodological) **53**(2), 285–321 (1991)
16. Gupta, S., Arbeláez, P.A., Girshick, R.B., Malik, J.: Aligning 3d models to rgb-d images of cluttered scenes. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 4731–4740 (2015)
17. Hou, J., Dai, A., Nießner, M.: 3d-sis: 3d semantic instance segmentation of rgb-d scans. In: Proc. Computer Vision and Pattern Recognition (CVPR), IEEE (2019)
18. Huang, S., Qi, S., Zhu, Y., Xiao, Y., Xu, Y., Zhu, S.C.: Holistic 3d scene parsing and reconstruction from a single rgb image (2018)

19. Kim, Y.M., Mitra, N.J., Huang, Q., Guibas, L.: Guided real-time scanning of indoor objects. In: Computer Graphics Forum. vol. 32, pp. 177–186. Wiley Online Library (2013)
20. Kipf, T., Fetaya, E., Wang, K.C., Welling, M., Zemel, R.: Neural relational inference for interacting systems. arXiv preprint arXiv:1802.04687 (2018)
21. Kulkarni, N., Misra, I., Tulsiani, S., Gupta, A.: 3d-relnet: Joint object and relational network for 3d prediction (2019)
22. Li, Y., Dai, A., Guibas, L., Nießner, M.: Database-assisted object retrieval for real-time 3D reconstruction. In: Computer Graphics Forum. vol. 34, pp. 435–446. Wiley Online Library (2015)
23. Lim, J.J., Pirsiavash, H., Torralba, A.: Parsing ikea objects: Fine pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2992–2999 (2013)
24. Liu, C., Kim, K., Gu, J., Furukawa, Y., Kautz, J.: Planercnn: 3d plane detection and reconstruction from a single image (2018)
25. Liu, C., Wu, J., Furukawa, Y.: Floornet: A unified framework for floorplan reconstruction from 3d scans. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 201–217 (2018)
26. Liu, C., Yang, J., Ceylan, D., Yumer, E., Furukawa, Y.: Planenet: Piece-wise planar reconstruction from a single rgb image. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (Jun 2018). https://doi.org/10.1109/cvpr.2018.00273, http://dx.doi.org/10.1109/CVPR.2018.00273
27. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European conference on computer vision. pp. 21–37. Springer (2016)
28. Murali, S., Speciale, P., Oswald, M.R., Pollefeys, M.: Indoor scan2bim: Building information models of house interiors. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 6126–6133. IEEE (2017)
29. Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohi, P., Shotton, J., Hodges, S., Fitzgibbon, A.: Kinectfusion: Real-time dense surface mapping and tracking. In: Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on. pp. 127–136. IEEE (2011)
30. Nießner, M., Zollhöfer, M., Izadi, S., Stamminger, M.: Real-time 3d reconstruction at scale using voxel hashing. ACM Transactions on Graphics (TOG) (2013)
31. Rusu, R.B., Blodow, N., Beetz, M.: Fast point feature histograms (fpfh) for 3d registration. In: Robotics and Automation, 2009. ICRA'09. IEEE International Conference on. pp. 3212–3217. Citeseer (2009)
32. Song, S., Yu, F., Zeng, A., Chang, A.X., Savva, M., Funkhouser, T.: Semantic scene completion from a single depth image (2017)
33. Sun, X., Wu, J., Zhang, X., Zhang, Z., Zhang, C., Xue, T., Tenenbaum, J.B., Freeman, W.T.: Pix3d: Dataset and methods for single-image 3d shape modeling. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (Jun 2018). https://doi.org/10.1109/cvpr.2018.00314, http://dx.doi.org/10.1109/CVPR.2018.00314
34. Tombari, F., Salti, S., Di Stefano, L.: Unique signatures of histograms for local surface description. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) Computer Vision – ECCV 2010. pp. 356–369. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)

35. Wald, J., Avetisyan, A., Navab, N., Tombari, F., Niessner, M.: Rio: 3d object instance re-localization in changing indoor environments. In: The IEEE International Conference on Computer Vision (ICCV) (October 2019)
36. Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., Jiang, Y.G.: Pixel2mesh: Generating 3d mesh models from single rgb images. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 52–67 (2018)
37. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics **38**(5), 1–12 (Oct 2019). https://doi.org/10.1145/3326362, http://dx.doi.org/10.1145/3326362
38. Xiang, Y., Kim, W., Chen, W., Ji, J., Choy, C.B., Su, H., Mottaghi, R., Guibas, L.J., Savarese, S.: Objectnet3d: A large scale database for 3d object recognition. In: ECCV (2016)
39. Zakharov, S., Shugurov, I., Ilic, S.: Dpod: 6d pose object detector and refiner. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1941–1950 (2019)
40. Zeng, A., Song, S., Nießner, M., Fisher, M., Xiao, J., Funkhouser, T.: 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In: Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on. pp. 199–208. IEEE (2017)
41. Zou, C., Guo, R., Li, Z., Hoiem, D.: Complete 3d scene parsing from an rgbd image (2017)

# Acronyms

| | |
|---|---|
| AR | Augmented Reality. |
| BRDF | Bidirectional Reflectance Distribution Function. |
| CAD | Computer Aided Design. |
| CNN | Convolutional Neural Network. |
| CRF | Conditional Random Field. |
| DL | Deep Learning. |
| DoF | Degrees of Freedom. |
| FLOPS | Floating Point Operations per Second. |
| GUI | Graphical User Interface. |
| ICP | Iterative Closest Point. |
| IOU | Intersection over Union. |
| IR | Infrared. |
| LM | Levenberg–Marquardt. |
| MLP | Multilayer Perceptron. |
| NMS | Non-maximum Suppression. |
| NN | Neural Network. |
| OCR | Optical Character Recognition. |
| PFH | Point Feature Histograms. |
| PnP | Perspective-n-Point. |
| PPF | Point-Pair Features. |
| RANSAC | Random Sample Consensus. |
| ROS | Robot Operating System. |
| SDF | Signed Distance Function. |

| | |
|---|---|
| SIFT | Scale Invariant Feature Transform. |
| SIS | Semantic Instance Segmentation. |
| SLAM | Simultaneous Localization and Mapping. |
| SSCNN | Submanifold Sparse Convolutional Neural Network. |
| SVD | Singular Value Decomposition. |
| TOF | Time-of-Flight. |
| TSDF | Truncated Signed Distance Function. |
| URDF | Universal Robot Description Format. |
| VR | Virtual Reality. |

# List of Tables

# List of Figures