Scene Recomposition by Learning-based ICP

Hamid Izadinia
University of Washington

Steven M. Seitz
University of Washington

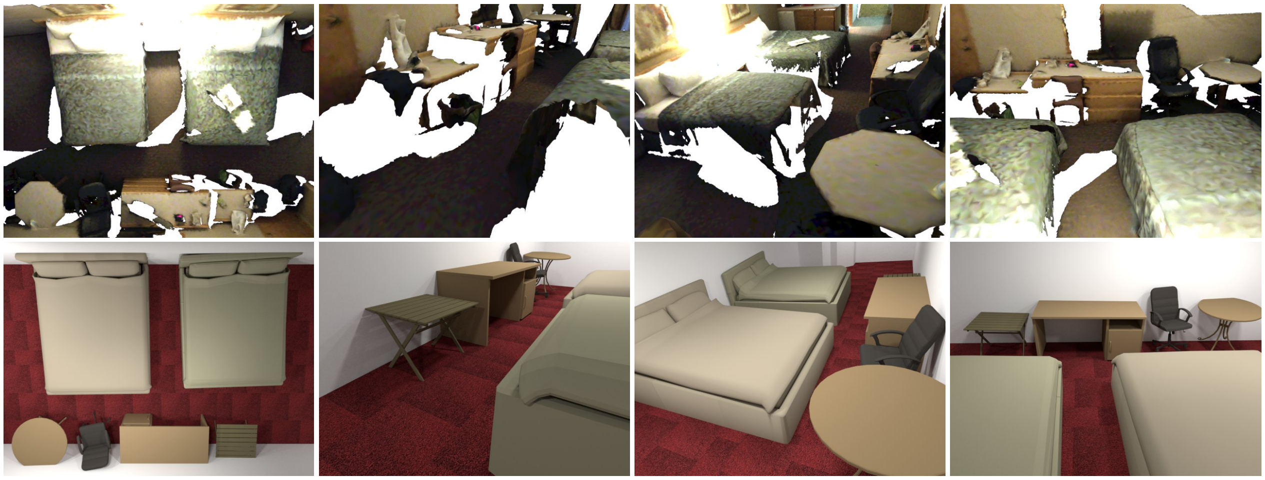

Figure 1. Given an RGBD sequence from a moving camera, we produce a 3D CAD ***recomposition*** of the scene. While a fused reconstruction (top) contains holes and noisy geometry, our recomposition (bottom) models the scene as a set of high quality 3D shapes from CAD databases.

Abstract

By moving a depth sensor around a room, we compute a 3D CAD model of the environment, capturing the room shape and contents such as chairs, desks, sofas, and tables. Rather than reconstructing geometry, we match, place, and align each object in the scene to thousands of CAD models of objects. In addition to the fully automatic system, the key technical contribution is a novel approach for aligning CAD models to 3D scans, based on deep reinforcement learning. This approach, which we call Learning-based ICP, outperforms prior ICP methods in the literature, by learning the best points to match and conditioning on object viewpoint. LICP learns to align using only synthetic data and does not require ground truth annotation of object pose or keypoint pair matching in real scene scans. While LICP is trained on synthetic data and without 3D real scene annotations, it outperforms both learned local deep feature matching and geometric based alignment methods in real scenes. The proposed method is evaluated on real scenes datasets of SceneNN [23] and ScanNet [14] as well as synthetic scenes of SUNCG [56]. High quality results are demonstrated on a range of real world scenes, with robustness to clutter, viewpoint, and occlusion.

1. Introduction

3D scene reconstruction is a fundamental challenge of computer vision. Most reconstruction techniques focus on estimating surface geometry, in the form of meshes, point-clouds, voxels, or other low-level representations. Suppose that you had access to a database of 3D models of every object in the world; then you could generate a scene model by identifying which objects are in which locations and placing them there. We call this variant of the reconstruction problem *scene recomposition*. While previously such an approach was not feasible at scale, the advent of large CAD repositories like *ShapeNet* [10] begins to make scene recomposition tractable for real-world scenes.

Scene recomposition has a number of advantages over scene reconstruction. First, whereas reconstruction methods often generate holes and capture only visible surfaces, recomposition yields more complete models, including back-facing and hidden geometry (see Figure. 1). Second, CAD models are clean, segmented, and hand-optimized, and thus better suited for applications like games, VR, robotics, etc. And third, recomposed models can be easily edited by moving objects around, replacing objects, and often come with semantic labels and annotated parts.

Recomposition is not a new idea, dating back to the first “blocks world” methods from the 1960s [44], with a model-based approach to more recent examples of SLAM++ [51] and IM2CAD [25]. We introduce the first fully automatic 3D scene recomposition that takes an RGBD sequence as input and produces a model of the scene composed of best-matched CAD models from *thousands* of 3D CAD models. In addition, we propose a novel learning-based ICP technique for aligning CAD models to scanned geometry.

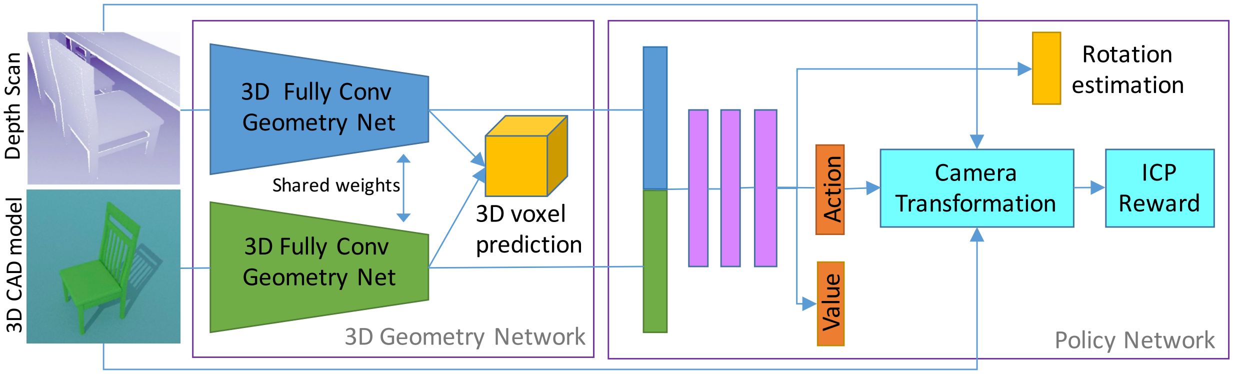

Figure 2. **LICP Network Architecture**: The input to our network consists of a scanned object paired with a reference CAD model (left) which are processed by the **geometry network** (middle). The geometry network is trained via a supervised loss to predict 3D voxel labels (yellow). The input representations are then concatenated to form the input to the **policy network** (right) which is trained via policy gradient to predict action distribution and value (orange) in order to maximize an ICP reward function. An auxiliary reward function (yellow) that estimates the rotation degree of the 3D CAD model with respect to the scanned shape is also incorporated.

Aligning 3D object models to depth scans is a classical problem in computer vision and geometry processing, and a staple of many practical applications spanning mapping, robotics, and visualization. The Iterative Closest Point algorithm (ICP) [7] works by alternating between finding the closest points between the model and the depth image (or other sensor data), solving for the best transformation that aligns the two point sets, and iterating until convergence. ICP and its variants can robustly converge when the model is initialized close to the solution, but suffer without good initialization or in the presence of significant occlusions and scene clutter. Matching discriminative local 3D features [27, 17, 61, 49, 50] is an alternative which relaxes the initialization requirements to be more robust, but is less effective for matching synthetic CAD models to real scenes, where 1) the models are simple and feature-poor, and 2) the shapes of the model and real object only approximately agree.

To address these problems, we cast the problem of aligning 3D CAD models to RGBD scans in a reinforcement learning framework which we call Learning-based ICP (LICP). LICP is trained entirely on synthetic scenes without requiring ground truth annotation of object pose alignment or keypoint pairs in real scenes. Despite this fact, our quantitative evaluations show that LICP outperforms prior methods in real scenes. We demonstrate the application of our approach for fully automatic scene recomposition of complex real environments populated with different types of furniture exhibiting a high degree of occlusion. Our recomposed scenes are comprised of best-matched CAD models from thousands of 3D CAD models in ShapeNet.

2. Related work

Inferring 3D object pose and scene recomposition relates to prior works in computer vision and graphics, as follows.
ICP: ICP was introduced by [13] and [7] and solves for the transformation between two point sets. Much research has been devoted to improving this method over the years, including [47, 13, 46]. Where prior methods focus on feature representation and optimization, we introduce a data-driven and learning-based approach.

3D shape alignment, 3D features and keypoint matching: An alternative to dense alignment via ICP is to detect robust features (aka *keypoints*) to facilitate shape alignment. [27] proposed *spin images* and used RANSAC for shape alignment. Other examples of geometric descriptors are Geometry Histograms [17], Signatures and Histograms [61], Feature Histograms [49] and many more available in Point Cloud Library [50]. However, keypoint methods can be sensitive to noise and do not always perform well, particularly for matching CAD models which are often piece-wise planar and feature-poor. Local features are not robust to symmetries (e.g., all chair legs may have the same features). Model-fitting approaches, also known as registration approaches, try to align an input with a training model but without using descriptors [7, 26, 65]. These approaches do not incorporate learning so that they do not benefit from large amount of data to gain robustness in keypoint detection and matching. Techniques like [22, 55, 19, 31, 39, 35, 29] estimate complete scene geometry by fitting instance-level 3D mesh models to the observed depth map. Compared to these methods, our model learns global models over CAD shapes to align poses.

A recent approach for CAD to scan alignment [4], requires *manual* annotation and curation of a large dataset of 3D keypoint correspondences between object CAD model and real scans. [4] uses the collected annotation data for learning correspondences between CAD models and scans. However, our proposed method only uses available synthetic data during training without needing annotated keypoint correspondences in both CAD and real scan domains. While not needing annotated data, our proposed method performs well in the real scene scenarios at the test time. Also to find correspondences at the test time, [4] uses the ground truth object set or a limited set of CAD models, whereas our method can find corresponding CAD models from an *unconstrained* set of objects.

Object level RGBD scene reconstruction: Like our approach, SLAM++ [51] performs room scale semantic object reconstruction using KinectFusion [40] followed by 3D shape recognition. Also, SLAM++ only uses a handful of 3D object models (vs. the thousands in ShapeNet), and does not incorporate a learning-based approach.

3D CAD scene model generation: Several prior works proposed methods of generating CAD-based room models using a variety of techniques. Example of these approaches are CAD from text descriptions [11], example based methods [16] or optimizing furniture arrangements in a space [68, 37]. Scene models can also be generated by matching 3D objects to a given image [52, 34], rendering a low fidelity synthesize model using RGBD images [21] or recomposing each scene by analyzing layout and furniture and jointly optimizing their placements [25].

Voxel prediction and shape completion: Single object shape completion and voxel category prediction has been studied by several authors [45, 60, 66]. In this paper, we utilize voxel category prediction as an auxiliary loss function to learn 3D representation, but the output of our model is a 3D CAD model with correct pose instead of a voxel grid. As such, we do shape completion, but compared to prior voxel-wise shape completion methods, our method produces CAD meshes with shape semantics.

Shape pose estimation: Single object 3D pose recognition from a photograph or depth image is also related to our work [3, 28, 51, 33, 24, 62, 5, 64]. However our approach differs since we learn the best points to match by conditioning on object viewpoint.

Deep feature learning and deep reinforcement learning: A number of researchers have used deep neural networks to learn 3D feature representations [56, 69]. Recently, deep Reinforcement Learning (RL) approaches have gained considerable attention due to their success in learning efficient policies to play games [38, 53] and obtaining promising performance in robotics [20, 2]. Part of the success of deep RL is its applicability in solving black-box non-differentiable optimization problems [59]. Our approach for selecting the correct camera transformation action based on score approximation is closely related to a class of RL techniques called policy gradients [6, 63]. In our method, we have a non-differentiable reward function based on ICP scores of two point clouds and we want to learn the policy that results in receiving maximum reward by using stochastic gradient decent and following a policy gradient update rule.

3. Proposed Method

We begin by describing our learning-based ICP (LICP) approach. Then we explain how to use LICP for recomposing a scene from an input point cloud. For scene recomposition, 3D object detection and 3D semantic segmentation are incorporated for extracting the object instances in the scene. Then, LICP is applied to match and align 3D object CAD models to segmented regions of scene geometry.

LICP seeks to estimate the transformation parameters of a scanned rigid object in natural real scenes. This is a challenging task due to inter-object occlusion, self-occlusion and clutter. We train a deep neural network that takes in a scanned shape (query) paired with a reference CAD model as input and learns to infer the transformation that should be applied to the reference CAD model to best align its point cloud with the query scan (Figure. 2). To learn such a model, we take advantage of the fact that we can apply any transformation on the reference CAD object and simulate a depth map (point cloud) of the transformed object using ray tracing. To this end, we generate a training set of 3D scans, each paired with a 3D object with known 6DoF parameters. We pose the learning problem in an RL framework where the task is to predict the best action that should be applied to the reference shape such that we can generate the query input scan. Each action encodes a possible 3D transformation that will be applied to the reference 3D shape. By applying each action, we produce a reward that reflects how much the transformed 3D shape matches the query shape.

3.1. Shape Alignment by Deep RL

We pose the problem of 3D pose estimation with respect to a reference shape in an RL framework. Suppose we have a reference shape X^r which is presented in a reference pose P^r. Using this reference shape, we want to learn to predict the 3D pose of any query 3D object scan X^q that is being cropped out of a complete scene scan. The 3D scan can contain a high amount of occlusion, complicating the alignment process. For representing 3D models, we use a voxel-based 3D feature representation function $\Phi(X)$ for both reference and query shapes. The goal of the RL agent is to select transformation actions to the query object which maximize the expected sum of future rewards. Our reward function, shows the matching score of the query shape with the reference shape if point-to-point local closest point alignment is performed (details in Section 3.2).

We consider a Markov Decision Process (MDP) defined by states $s \in \mathcal{S}$ and actions $a \in \mathcal{A}$. Each 3D rotational camera transformation is an action a that the RL agent can potentially apply to a 3D shape. We define each pair of query object scan and reference object scan captured with camera transformation ϱ as a state $s : (\Phi_\tau(X^q), \Phi_\varrho(X^r))$. Each camera transformation action a can transit the agent to a new state by capturing the 3D scan of the reference object X^r. We uniformly discretise the action space of each dimension of rotation degrees into a list of 32 bins where each bin corresponds to a rotation transformation with a fixed angle. Reducing the action space complexity by discretization accelerates learning and makes it more sample efficient.

3.2. ICP-based Rewards

Each training instance is composed of a 3D point cloud of a scanned query object $\Phi_\tau(X^q)$ captured with an unknown camera pose τ paired with a reference 3D object

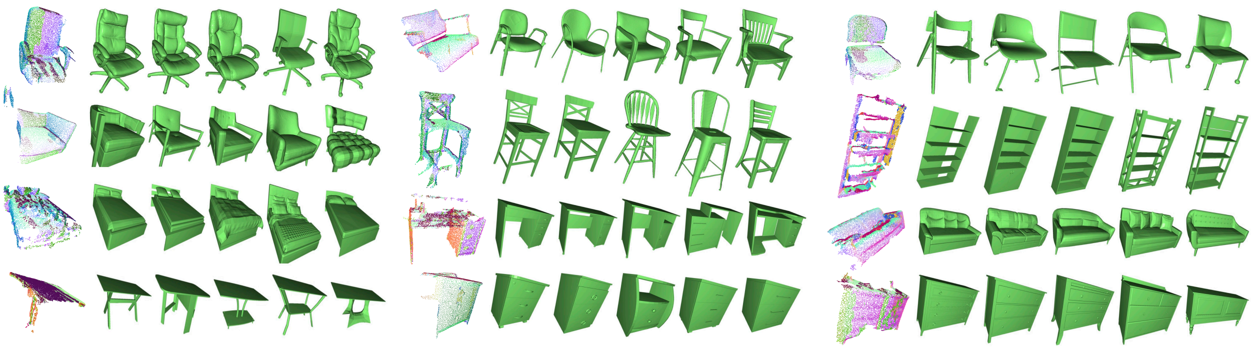

Figure 3. Top retrieved CAD models for each object instance segmentation as query. Point cloud query is color-coded with surface normal.

X^r. After choosing an action a, we apply the corresponding camera transformation a and render the transformed point cloud $\Phi_a(X^r)$ of the reference shape X^r. Our reward function takes in the point cloud of the query object $\Phi_\tau(X^q)$ and the point cloud of the reference object $\Phi_a(X^r)$ captured under camera transformation imposed by a and produces a score value which reflects how well the two of the point clouds can be matched. We leverage the ICP matching score as the feedback to compute the reward function f.

$$r(s,a) = f(\Phi_\tau(X^q), \Phi_a(X^r)) \tag{1}$$

3.3. Learning by REINFORCE

Our reward function is non-differentiable. To solve this black-box optimization problem we opt to use the REINFORCE learning rule [63] where our goal is to find a policy $\pi_\theta(a|s)$ with parameters θ which maximizes the expected sum of rewards: $J(\theta) = \mathbb{E}_{\rho_\theta \tau}[R_t]$, where $R_t = \sum_t \gamma^{t-1} r(s_t, a_t)$. This expectation is with respect to the distribution of rollout trajectories generated by the policy π_θ. The gradient of this objective with respect to the parameters θ can be computed by $\nabla_\theta J = \mathbb{E}_\theta[\sum_t \nabla_\theta \log \pi(s_t|a_t)(R_t - b_t)]$ where b_t is a baseline that does not depend on a_t of the future states and actions. Following a well-known approach, we choose the baseline to be $\mathbb{E}[R_t|s_t]$ and in practice we approximated it with the average value of rewards, updated over time.

To accelerate training, we augmented the loss function obtained from the REINFORCE learning rule with an auxiliary reward function that is particularly tailored for our task of shape pose estimation. This loss function encodes the error in estimating the rotation angles between the reference CAD model and the shape query scan and corresponds to sum of squared distance between the ground truth rotation and the regressed rotation. We use stochastic action sampling based on the probability produced by the current policy. We use dropout [57, 18] to incorporate stochastic action selection and standard epsilon-greedy strategy in RL [59] for providing exploration in learning.

3.4. LICP Network Architecture

Learning a complex shape representation from sparse rewards is very challenging and requires a large number of trials. Instead, we learn the shape representation using dense voxel category labels in a supervised approach, as follows. Freezing the learned shape representation network, we compute features of the 3D observation signal and use a separate network to learn the policy for finding the object poses.

3D Geometry Network: For 3D geometry feature representation, we use a 3D fully convolutional network that takes in 3D volumes as input and learns to produce per-voxel category labels in a supervised fashion, using softmax loss function over object categories. Each tower of our geometry network uses the 3D fully convolutional architecture of [56] which incorporates several 3D convolution layers.

Input volume generation: Our observation signal is in the form of 2D depth maps, which we convert to a volumetric grid of Truncated Distance Function (TDF) values. The TDF representation can encode both single depth and multiple depth images. Specifically, each voxel takes a value which indicates the distance between the center of that voxel to the nearest 3D surface. Following [69], these values are truncated, normalized and then inverted to be between 1 and 0, indicating on surface and far from surface, respectively.

Policy Network: Our policy is learned via a fully connected network consisting of three layers, each with 256 units followed by dropout and ReLU, using the policy learning and loss and reward function in Sections 3.2 and 3.3.

Training Details: We implement our model in TensorFlow [1] and use stochastic gradient descent with a learning rate of 0.001 and decay factor of 0.95. We train both 3D geometry and policy network over more than 1 million training samples in simulation.

3.5. Generate Training Data using Simulation

We generate synthetic training data using SUNCG scenes [56]. In each room, we move the camera at a person's height while looking at different objects in the scene. We generate a wide range of camera angles: yaw varies between $[-180, 180]$, pitch depends on the height of objects and varies between $[-90, 90]$ and roll randomly takes a value in $[-10, 10]$ degrees. To produce a variety of viewpoints, we jitter the camera with a small amount of noise. For each view, we capture the depth image and crop the

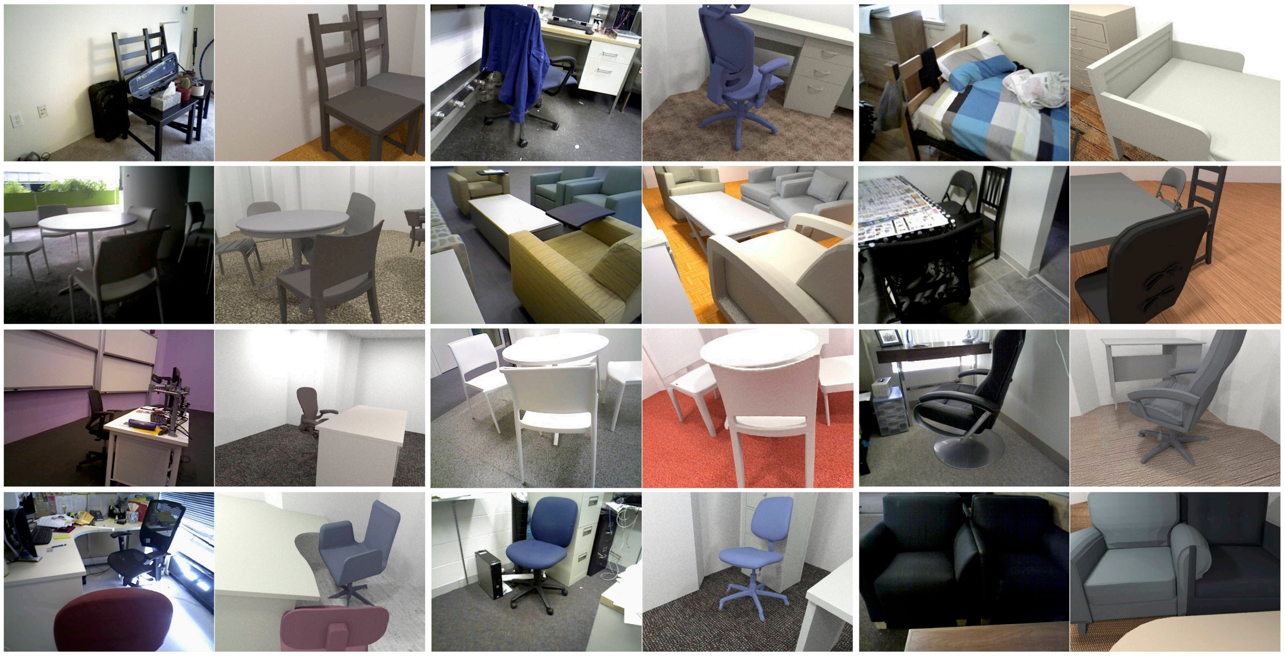

Figure 4. Qualitative examples of the recomposed CAD model of the scene. Each example shows a view of the camera in the scanned scene on *left* and recomposed CAD from the same view on *right*. Our method can successfully recompose cluttered scenes with lots of distractor objects (first row) and huge amount of occlusions in scenes populated with many furniture objects and in confined spaces (second and third Row). Less accurate CAD recomposition can occur due to ambiguous extent of scanned meshes with nearby objects (bottom row, right), or lack of discriminative shape features in different views (cabinet in bottom row, middle)

box around the object which also contains some parts of the other objects. We then pass the partial point cloud to the network as input. We rasterize the mesh of the 3D CAD model into a point cloud and use the produced point cloud as the reference input of the network. The truncated distance function of the point cloud is used as input to the network.

3.6. Scene Recomposition

Our scene recomposition pipeline takes in a point cloud which is produced from RGBD video of a real scene. We apply 3D object detection and semantic segmentation for extracting 3D object instances. Then, we use the output of our trained 3D geometry network (see Figure 2) for finding the nearest 3D CAD model in the set of CAD models and use it as reference 3D shape. Finally, we deploy LICP for aligning the 3D CAD model to object instance segmentation, as described in Section 3.1.

3D Object Detection: We use the two-step object detection regime [43, 12, 30] as follows. We train a category agnostic region proposal network which gives the objectness score for different 3D bounding boxes over the point cloud. We simultaneously train another network for classification of 3D bounding boxes for each of the object categories. Both networks share the feature extraction layers which are based on the VGG architecture [54]. We use cross entropy loss for both region proposal and classification networks. We also learn the deviation of the 3D boxes using regression loss in x and y dimensions and the z_l and z_h for the lower and higher extent of the object along the Z axis orthogonal to the ground plane. We rectified the point cloud in world coordinates by rotating the gravity direction and then making it axis aligned with the dominant X-Y orientation on the ground plane. To compute feature maps from the point clouds we use the orthographic projection of the point cloud representations and extract feature from planes in different heights following [12]. For training, we use rendered depth images from SUNCG [56] as explained in 3.5. We use the entire scene composed of multiple objects in the field of view for each camera pose. We set 0.5 as the threshold for intersection over union (IoU) of 3D detection boxes and use non-maximum suppression for removing low scoring 3D boxes which have high overlap with higher scoring detections. We find the translation and scale of the objects via 3D object detection and apply the inferred translation and scale to the CAD models.

3D Semantic Segmentation: Clean object instance segmentation is important for the alignment stage of our method. For instance, when a chair is next to a table the 3D bounding box of the chair may include some part of table and vice versa. In order to remove such distractors from the detection bounding box of each object detection we incorporate the semantic segmentation inferred on the point clouds. We take all points inside the 3D detection box and remove the points with semantic label of other object categories with overlapping detected bounding boxes. We also remove the points with "floor" and "wall" labels. We follow [41, 32, 42] for training semantic segmentation over the point cloud and learn a model for all object categories

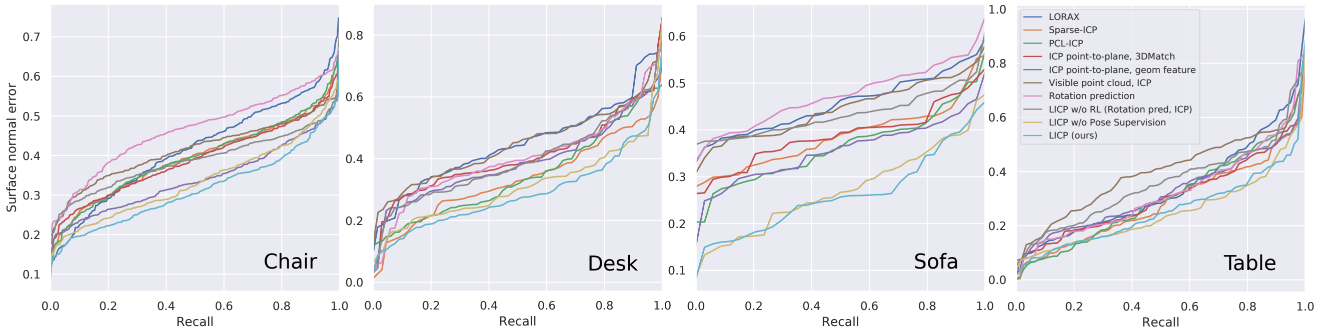

Figure 5. Comparison of proposed LICP method with local feature matching and alignment methods on real data (*lower values are better*). The legend is only shown on the right plot for better readability and the color of methods are the same for all plots.

as well as floor and wall classes.

Room Layout Estimation and Scene Visualization: We use the inferred wall points from the 3D point cloud segmentation to estimate the room layout. For each point on the ground plane (X, Y), we count the number of wall 3D points, aggregating over the Z axis. The locations on the ground plane with high frequency of wall voxels define the boundary of the room. We use the extent of the floor voxels wherever scan does not have wall in the boundary. Once all wall voxels on the ground plane are computed we run the concave hull algorithm to find the room boundary. We infer the location of the floor plane to be at the Z which has the highest frequency of floor voxels inferred via semantic segmentation of 3D points. The color of each object is estimated by medoid color of the point clouds belonging to the object instance segmentation. The floor texture is selected based on the feature similarity to a set of texture image.

4. Experiments

In our experiments we want to investigate: 1) How accurate is our learning-based ICP compared to non-learning previous approaches, 2) how does our method compare with keypoint matching approaches based on deep features, and 3) how can our model be applied in scene CAD model recomposition of unstructured and cluttered real world environments. To answer these questions, we evaluate the performance of our method both quantitatively and qualitatively. For real-world evaluation, we use the publicly available SceneNN [23] and ScanNet [14] datasets. SceneNN and ScanNet test sets contain scans of 95 and 312 scenes from different real world indoor spaces, respectively. These scene point clouds are scanned from various offices, bedrooms, living room, kitchen, etc., and exhibit a diverse collection of unstructured real world scenes populated with various furniture types, styles, and types of clutter from many distractor objects. These scenes are scanned with commodity depth cameras and we use the fused output.

4.1. Quantitative Evaluation

We evaluate the accuracy of our method for 6DoF pose estimation of furniture objects in both real and synthetic scenarios. We compare our results with prior works of [13, 48, 69, 15, 8, 70, 67, 9, 36]. For the evaluation criteria, we compute the alignment error between the scanned mesh and the CAD model with the predicted pose. To compute the alignment score, the closest point on the CAD model is found for each point in the input scan and the cosine distance between surface normals is computed. In the synthetic data experiment, we use the distance between points on reference CAD model and scan given that we have access to the ground truth mesh of the object in simulation.

Quantitative evaluation on real data: To evaluate the effectiveness of LICP for 6DoF object pose estimation, we incorporate the ground truth point cloud segments and object labels. We use the feature representation of our trained 3D geometry network for finding the nearest 3D CAD model from a database of 1550 CAD models from [56, 58] and use it as the reference CAD model. The quality of the object style match for retrieved CAD models is shown for several examples in Figure 3.

We compare LICP with local feature matching and variants of ICP from the literature. For local feature matching, we compare against the hand-designed geometric feature of FPFH [48], learned local deep feature by 3DMatch [69] and LORAX [15]. After matching the local features, we use RANSAC for coarse registration followed by point-to-plane ICP [13] for fine alignment of CAD model and input scan. For comparing against LORAX, we use the released code of [15] for super-point extraction and use local deep features learned in an unsupervised fashion from point clouds of synthetic object CAD models via GAN. We also compare with *Sparse ICP* [8] (a variant of ICP that is robust to input noise), and the *PCL* implementation of ICP. Figure 5 summarizes our quantitative comparison results. In the plots of Figure 5 *"ICP point-to-plane, geom feature"* refers to FPFH setting. As demonstrated in Figure 5, our method outperforms all aforementioned prior methods.

We also compare LICP with other baselines and variants of proposed LICP with different combinations of loss and reward function. **Rotation prediction** only uses object rotation estimation output of the learned network in Figure 2 and does not use our RL component. **Rotation pred., ICP point-to-plane** uses the rotation estimation output of the LICP network and applies ICP point-to-plane for finer ob-

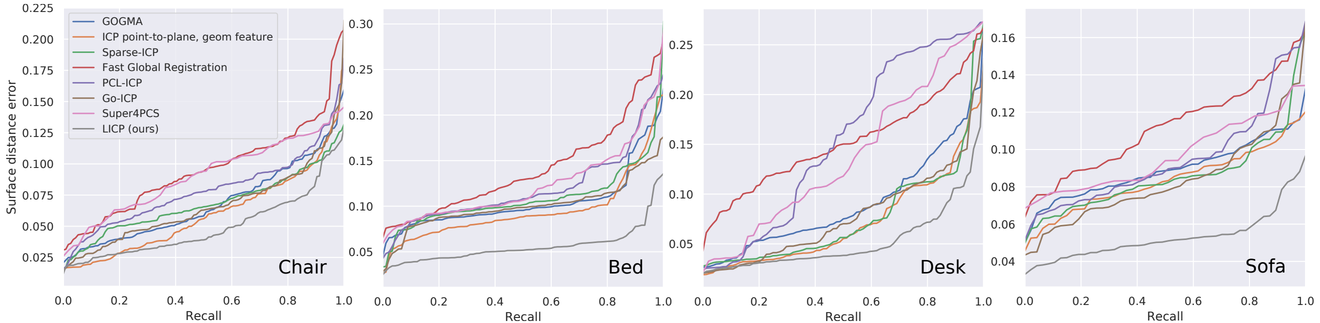

Figure 6. Comparison of proposed LICP method with robust and global alignment algorithms on synthetic data (lower values are better).

ject alignment. **Visible point cloud, ICP** only uses the visible points of the point cloud from predicted object pose for ICP alignment. **LICP w/o Pose Supervision** uses a policy network that is only trained with RL component and without strong object pose supervision of auxiliary loss. All of these variants have lower performance than our full LICP model that combines ICP-based reward and auxiliary loss for learning the policy network. Also the performance of LICP only with RL is close to LICP which suggests that LICP performance is mostly gained by RL learning rather than strong object pose supervision.

We do not have access to the ground truth CAD model of the shapes in the input scan and we use the surface normal error between recomposed CAD and input scan. We plot the surface normal error vs. recall for each category, which is the percentage of samples with surface normal error lower than each error value. Note that the smallest average ICP distance between the pair of scan and CAD model never goes to zero since the point cloud input pairs to the ICP method are sampled differently and are never identical.

Quantitative evaluation on synthetic data: We test on the SUNCG [56] test set where objects are placed in 3D scenes with realistic furniture arrangements. This experiment is performed on several input CAD models and input scans. The alignment error is the mean surface point distance in meters between the object surface in scan and the reference CAD model. In this experiment we test on synthetic scans where we have the ground truth surface of the scanned object. Therefore, we can compute the distance between the surface of the reference CAD and surface of the CAD in the scan. We compare LICP with robust and global alignment algorithms: Fast Global Registration [70], globally-optimal algorithm Go-ICP [67], GOGMA [9], Super4PCS [36] and Sparse ICP [8]. We also compare LICP against point-to-plane ICP [13] with FPFH geometric feature and PCL implementation of ICP. The results are summarized in Figure 6. Our LICP alignment outperforms other global and robust alignment methods by a large margin.

We also evaluate the robustness of LICP against large orientation differences between the object scan input and the reference CAD model and compare against Chen and Medioni ICP [13] in Figure 7. The reference CAD models are initialized with different orientations for each experiment. In Figure 7, the x-axis shows the initialization error while the y-axis shows the final alignment error after ICP is converged. While both methods reduce the alignment error, LICP obtains lower final error compared to [13].

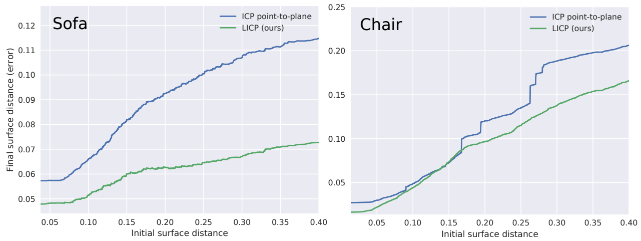

Figure 7. Evaluating the robustness of our proposed LICP method for aligning 3D CAD models with drastic orientation differences to the input scan using synthetic data.

4.2. Qualitative Evaluation

Real scene shape alignment: Figure 4 demonstrates several examples of scene CAD models recomposed (on right) from the depth scan of real scenes (on left) by applying our algorithm where best-matched CAD models and 6Dof object poses are estimated. The first row in Figure 4 shows several recomposed CAD scene models in the presence of a high amount of scene clutter. For example, the surface of the two chairs on the top left is filled with random objects, and the back cushion of the blue office-chair (first row, middle figure) is occluded with a shirt. While such arbitrary objects results in significant amount of noise in the depth scans, our method can estimate the 6DoF pose and object style reasonably well. Examples of the second row in Figure 4 show scenarios with significant occlusions as the result of a densely populated scene. As shown in the figure, our method handles such occlusions well and produces CAD scene models with accurate object pose and styles. Several failure cases are shown in the bottom row of Figure 4 where the estimated object poses are less accurate. For example, in the middle example of the forth row, the pose of the cabinet behind the blue chair is not estimated correctly due to the lack of strong discriminative shape features between the right face and the front face of the cabinet. Also the retrieved armchair style is not accurate in the left example of the forth row, as the extent of the armchair cannot accurately be obtained from the scanned point cloud

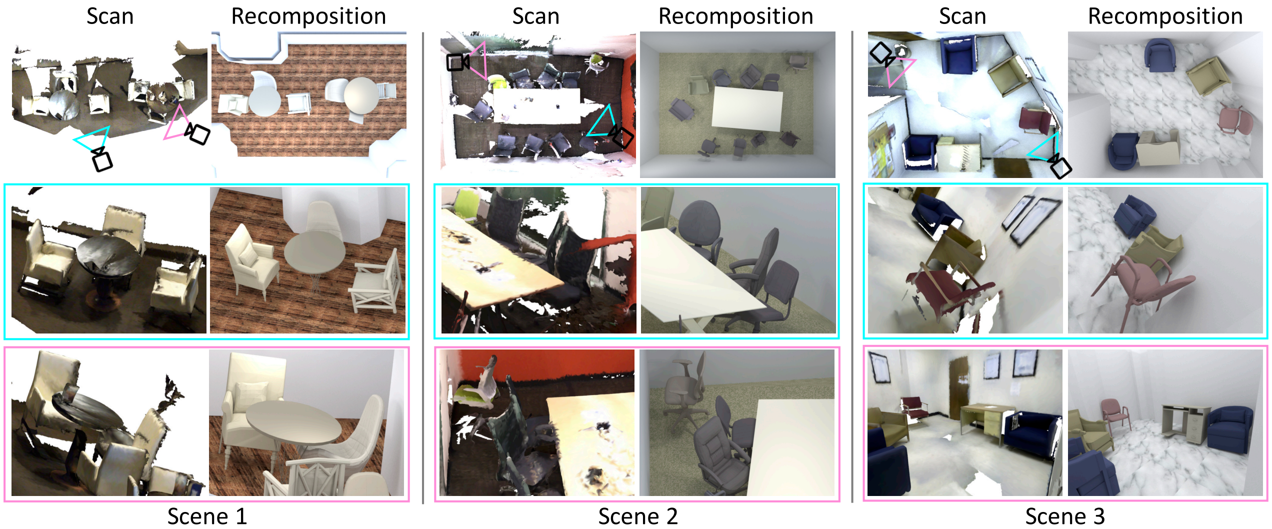

Figure 8. Scene *recomposition* using our proposed fully automatic method. Scene recomposition is shown for three different scenes. In each scene, the top row shows the top-down view of the scene; the middle and bottom rows demonstrate two close-up views of each scene. Camera location and pose is color coded on top-down view).

because of high level of occlusion with the nearby objects.

Real scene recomposition: We deploy our fully automatic scene recomposition method on real scenes, with results shown in Figure 8. For each scene, we render two different close-up camera viewpoints and the top-down view of the scene recomposed by our method and also show corresponding views from the scan. As shown in Figure 8, these scenes are densely populated with different furniture and the scene scans contain many holes. Despite many occlusions and holes, our method produces satisfying scene recompositions. Using TITAN Xp GPU, the computational time for a typical scene with an average complexity is approximately 6.5 seconds for 3D object detection and 9.5 seconds for 3D semantic segmentation. LICP 3D CAD alignment takes 1.22 seconds per object instance which includes 0.65 seconds for 3D Geometry Net, 0.008 seconds for Policy Net and 0.56 seconds for ICP Reward.

Surface point visualization during inference: LICP learns to assign different weights to surface points of the reference CAD model when queried with arbitrary posed object scans. The assigned weights for surface points in the reference CAD model are computed based on the visible surface points. The visible surface points are captured via ray tracing from the actions inferred, i.e., the camera transformation multiplied with the value estimated by the value function in our policy network. These weights reflect the contribution of each surface point in inferring the correct transformation action.

Figure 9 shows the surface point weights obtained for different objects when queried with scans from various viewpoints. The assigned weights are conditioned on the viewpoint of the query shape. When LICP is queried with a left-sided armchair, the visible surface points on the left side of the reference armchair gain higher weights and vice versa. Similarly, office chairs with different poses and occlusion patterns are provided. LICP assigns higher weights to the surface points that are not occluded and ignores the contribution of the occluded surface points. The bottom row of Figure 9 shows similar patterns in the produced weights for surface points of desk and L-shaped sofa instances.

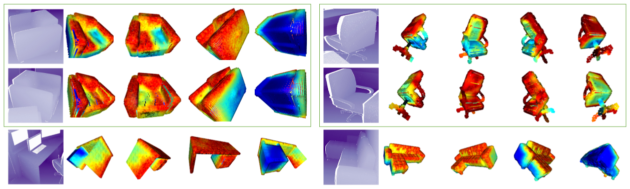

Figure 9. Visualization of the learned weights (*right*) for different samples and various query scan viewpoints (*left*). The learned weights are shown from four different views of the reference CAD model. Weight values are color-coded from low (*blue*) to high (*red*). The first two rows show that the surface points of the same reference CAD model are assigned with different weights depending on the query scan viewpoint.

5. Conclusion

In this paper, we compute 3D scene recompositions from a sequence of RGBD scans captured by a moving camera from a real scene. We present a learning based approach for shape alignment called Learning-based ICP (LICP). LICP combines deep 3D feature learning with reinforcement learning and is able to infer the 6DoF object transformation with respect to a reference shape. By leveraging large scale shape 3D databases and learning the transformation policy for various object poses, LICP becomes robust to scene clutter and partial occlusions. Our experimental results on diverse real world scans demonstrate high performance of our method compared to various baselines.

Acknowledgments

This work was supported in part by the University of Washington Animation Research Labs and Google.

References

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, 2016. 4

[2] M. Andrychowicz, D. Crow, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba. Hindsight experience replay. In *NIPS*, 2017. 3

[3] M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic. Seeing 3D chairs: exemplar part-based 2D-3D alignment using a large dataset of cad models. In *CVPR*, 2014. 3

[4] A. Avetisyan, M. Dahnert, A. Dai, M. Savva, A. X. Chang, and M. Niessner. Scan2cad: Learning cad model alignment in rgb-d scans. In *CVPR*, 2019. 2

[5] A. Bansal, B. Russell, and A. Gupta. Marr revisited: 2d-3d alignment via surface normal prediction. In *CVPR*, 2016. 3

[6] J. Baxter and P. L. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 2001. 3

[7] P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. In *Sensor Fusion IV: Control Paradigms and Data Structures*. International Society for Optics and Photonics, 1992. 2

[8] S. Bouaziz, A. Tagliasacchi, and M. Pauly. Sparse iterative closest point. In *Proceedings of the Eleventh Eurographics/ACMSIGGRAPH Symposium on Geometry Processing*. Eurographics Association, 2013. 6, 7

[9] D. Campbell and L. Petersson. Gogma: Globally-optimal gaussian mixture alignment. In *CVPR*, 2016. 6, 7

[10] 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. 1

[11] A. X. Chang, M. Savva, and C. D. Manning. Learning spatial knowledge for text to 3d scene generation. In *EMNLP*, 2014. 3

[12] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3d object detection network for autonomous driving. In *CVPR*, 2017. 5

[13] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992. 2, 6, 7

[14] 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. 1, 6

[15] G. Elbaz, T. Avraham, and A. Fischer. 3d point cloud registration for localization using a deep neural network auto-encoder. In *CVPR*, 2017. 6

[16] M. Fisher, D. Ritchie, M. Savva, T. Funkhouser, and P. Hanrahan. Example-based synthesis of 3d object arrangements. *TOG*, 2012. 3

[17] A. Frome, D. Huber, R. Kolluri, T. Bülow, and J. Malik. Recognizing objects in range data using regional point descriptors. In *ECCV*, 2004. 2

[18] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016. 4

[19] A. Geiger and C. Wang. Joint 3d object and layout inference from a single rgb-d image. In *German Conference on Pattern Recognition*. Springer, 2015. 2

[20] S. Gu, E. Holly, T. Lillicrap, and S. Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *ICRA*, 2017. 3

[21] R. Guo, C. Zou, and D. Hoiem. Predicting complete 3d models of indoor scenes. *arXiv preprint arXiv:1504.02437*, 2015. 3

[22] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Aligning 3d models to rgb-d images of cluttered scenes. In *CVPR*, 2015. 2

[23] 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 *International Conference on 3D Vision (3DV)*, 2016. 1, 6

[24] Q. Huang, H. Wang, and V. Koltun. Single-view reconstruction via joint analysis of image and shape collections. In *SIGGRAPH*, 2015. 3

[25] H. Izadinia, Q. Shan, and S. M. Seitz. Im2cad. In *CVPR*, 2017. 1, 3

[26] H. Jiang and J. Xiao. A linear approach to matching cuboids in rgbd images. In *CVPR*, 2013. 2

[27] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *TPAMI*, 1999. 2

[28] N. Kholgade, T. Simon, A. Efros, and Y. Sheikh. 3D object manipulation in a single photograph using stock 3d models. In *SIGGRAPH*, 2014. 3

[29] Y. M. Kim, N. J. Mitra, D.-M. Yan, and L. Guibas. Acquiring 3d indoor environments with variability and repetition. *TOG*, 2012. 2

[30] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. Waslander. Joint 3d proposal generation and object detection from view aggregation. *IROS*, 2018. 5

[31] K. Lai and D. Fox. Object recognition in 3d point clouds using web data and domain adaptation. *The International Journal of Robotics Research*, 2010. 2

[32] Y. Li, R. Bu, M. Sun, and B. Chen. Pointcnn: Convolution on $\mathcal{X}$-transformed points. In *NIPS*, 2018. 5

[33] J. J. Lim, A. Khosla, and A. Torralba. Fpm: Fine pose parts-based model with 3d cad models. In *ECCV*, 2014. 3

[34] Z. Liu, Y. Zhang, W. Wu, K. Liu, and Z. Sun. Model-driven indoor scenes modeling from a single image. In *Proceedings of the 41st Graphics Interface Conference*, 2015. 3

[35] O. Mattausch, D. Panozzo, C. Mura, O. Sorkine-Hornung, and R. Pajarola. Object detection and classification from large-scale cluttered indoor scans. In *Computer Graphics Forum*. Wiley Online Library, 2014. 2

[36] N. Mellado, D. Aiger, and N. J. Mitra. Super 4pcs: fast global pointcloud registration via smart indexing. In *Computer Graphics Forum*, volume 33, pages 205–215. Wiley Online Library, 2014. 6, 7

[37] P. Merrell, E. Schkufza, Z. Li, M. Agrawala, and V. Koltun. Interactive furniture layout using interior design guidelines. In *SIGGRAPH*, 2011. 3

[38] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 2015. 3

[39] L. Nan, K. Xie, and A. Sharf. A search-classify approach for cluttered indoor scene understanding. *TOG*, 2012. 2

[40] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, 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. 3

[41] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 5

[42] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, 2017. 5

[43] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 5

[44] L. G. Roberts. *Machine perception of three-dimensional solids*. PhD thesis, 1963. 1

[45] J. Rock, T. Gupta, J. Thorsen, J. Gwak, D. Shin, and D. Hoiem. Completing 3d object shape from one depth image. In *CVPR*, 2015. 3

[46] S. Rusinkiewicz. A symmetric objective function for ICP. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 38(4), July 2019. 2

[47] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pages 145–152. IEEE, 2001. 2

[48] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *ICRA*. IEEE, 2009. 6

[49] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz. Aligning point cloud views using persistent feature histograms. In *IROS*. IEEE, 2008. 2

[50] R. B. Rusu and S. Cousins. 3d is here: Point cloud library (pcl). In *ICRA*, 2011. 2

[51] R. Salas-Moreno, R. Newcombe, H. Strasdat, P. Kelly, and A. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *CVPR*, 2013. 1, 3

[52] S. Satkin, M. Rashid, J. Lin, and M. Hebert. 3dnn: 3d nearest neighbor. *IJCV*, 2015. 3

[53] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 2016. 3

[54] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[55] S. Song and J. Xiao. Sliding shapes for 3d object detection in depth images. In *ECCV*. Springer, 2014. 2

[56] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser. Semantic scene completion from a single depth image. In *CVPR*, 2017. 1, 3, 4, 5, 6, 7

[57] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 2014. 4

[58] 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 *CVPR*, 2018. 6

[59] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018. 3, 4

[60] D. Thanh Nguyen, B.-S. Hua, K. Tran, Q.-H. Pham, and S.-K. Yeung. A field model for repairing 3d shapes. In *CVPR*, 2016. 3

[61] F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. In *ECCV*. Springer, 2010. 2

[62] S. Tulsiani and J. Malik. Viewpoints and keypoints. In *CVPR*, 2015. 3

[63] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Reinforcement Learning*. Springer, 1992. 3, 4

[64] J. Wu, T. Xue, J. J. Lim, Y. Tian, J. B. Tenenbaum, A. Torralba, and W. T. Freeman. Single image 3d interpreter network. In *ECCV*, 2016. 3

[65] K. Wu and M. D. Levine. Recovering parametric geons from multiview range data. In *CVPR*, 1994. 2

[66] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015. 3

[67] J. Yang, H. Li, D. Campbell, and Y. Jia. Go-icp: A globally optimal solution to 3d icp point-set registration. *TPAMI*, 2016. 6, 7

[68] L.-F. Yu, S.-K. Yeung, C.-K. Tang, D. Terzopoulos, T. F. Chan, and S. J. Osher. Make it home: automatic optimization of furniture arrangement. In *SIGGRAPH*, 2011. 3

[69] 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. 3, 4, 6

[70] Q.-Y. Zhou, J. Park, and V. Koltun. Fast global registration. In *ECCV*. Springer, 2016. 6, 7